1998

Rtl Power Estimation of Sequential Circuits

Sridhar Muthrasanallur
Portland State University

Recommended Citation
Muthrasanallur, Sridhar, "Rtl Power Estimation of Sequential Circuits" (1998). *Dissertations and Theses.* Paper 6434.
https://doi.org/10.15760/etd.3579

THESIS APPROVAL

The abstract and thesis of Sridhar Muthrasanallur for the Master of Science in Electrical Engineering were presented May 27, 1998, and accepted by the thesis committee and the department.

COMMITTEE APPROVALS: [redacted]

Dr. W. Robert Daasch, Chair

[redacted]

Dr. Douglas V. Hall

[redacted]

Dr. Timothy R. Anderson
Representative of the Office of Graduate Studies

DEPARTMENT APPROVAL: [redacted]

Dr. Lee W. Casperson, Chair
Electrical and Computer Engineering

ABSTRACT

An abstract of the thesis of Sridhar Muthrasanallur for the Master of Science in Electrical Engineering presented May 27, 1998.

Title: RTL Power Estimation of Sequential Circuits

Power consumption has become a major concern in the electronic industry in recent years because of the increased demand for portable electronic devices. Part of the problem in power conscious design is accurate power estimation. Power estimation at low-levels of design abstraction is slow since the units of low-levels of design abstraction are transistors or gates. But designers need reliable power estimates early in the design process. Therefore designers need to have tools for fast and accurate power estimation at higher levels of design abstraction such as the Register Transfer Level (RTL).

This thesis introduces a new method for RTL power estimation of CMOS sequential circuits. This method tries to estimate the average power of a sequential circuit through the combination of a low-effort synthesis of the RTL description of the sequential circuit and the parameters readily available from the RTL description of the circuit like the sum-of-product count and literal count. The quantitative and qualitative aspects of the new model are studied with MCNC91 benchmark circuits and a large set of randomly generated circuits. Quantitative power estimation with the new model is seen to be very difficult because of the highly irregular surfaces of the functions that are being modeled in an effort to understand how a synthesis tool changes the power of a circuit during optimization. A qualitative measure is then proposed for the performance of a synthesis tool in preserving the qualitative ordering of power values of different implementations of a sequential circuit. An inference about such a performance of the synthesis tool would help the designer make informed decisions about the choice of implementation of a sequential circuit from a set of broad alternatives.

RTL POWER ESTIMATION OF SEQUENTIAL CIRCUITS

by

SRIDHAR MUTHRASANALLUR

A thesis submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
in
ELECTRICAL ENGINEERING

Portland State University
1998

Acknowledgements

First and foremost, I would like to thank my advisor, Prof. Daasch, who was instrumental in the development of this thesis. His invaluable guidance at each stage of this thesis was very vital. He is a researcher I hold in high respect. I thank Prof. Anderson and Prof. Hall for their very valuable feedback which helped me improve this document and also for their valuable suggestions for future directions.

I'm more than thankful to my friends, Sandeep, Easwar, Ryan and Vikranth, who cheerfully and painstakingly proof read my document. Their excellent feedback was highly important for the fast turn around time in preparing this document. And, my parents. Eventhough they are far away, their support throughout this thesis was nothing but great.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The growing demand for portable electronic devices has led to an increased emphasis on power consumption within the semiconductor industry. Consumers demand smaller devices at lower prices and increased speed. This demand for smaller size and higher speed has resulted in extremely dense designs. Functionality which was once achieved with several chips is now being integrated into a single chip. This has led to increased thermal concerns and an acute concern over how to sustain and improve battery life. Overheating and battery life are major concerns in portable consumer products. A design that generates too much heat or shortens battery life can delay or sometimes obstruct the successful introduction of a new product. The best and most cost effective solution is for designs to consume less power and this requires a methodology for practical power analysis. Designers are now encouraged to consider the impact of their decisions not only on speed and area, but also on power, throughout the design process. CAD (computer aided design) tools are aiding them in solving many of the problems associated with low power design.

1.1 Power Estimation - The Variety of Solutions

Power estimation of a CMOS VLSI chip is possible at different levels in the design hierarchy - transistor, gate, RTL (architectural), behavioral - each offering its own advantages and disadvantages. As you move up the design hierarchy, from

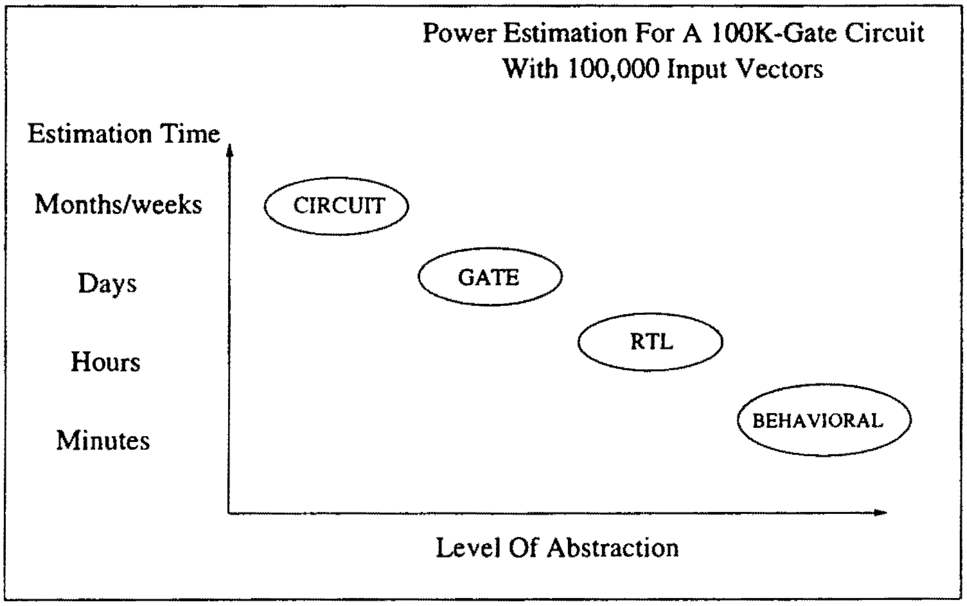


Figure 1.1: Power Estimation Time vs. Level of Hierarchy

transistor to behavioral, you gain in speed but lose in accuracy. Figure 1.1 shows the time needed for power estimation versus the level of abstraction [Nur97]. At RT and behavioral levels, accurate quantitative power measurements become a hard problem. Qualitative measures that help designers in choosing the right kind of circuits to implement from among the available design choices, are useful at these early stages of the design process.

At each of these levels, designers require a full spectrum of analysis choices : estimation, optimization and simulation. Although research has produced some promising methods [Landman94] to help designers create reduced-power systems and chips, a full power analysis solution still presents a significant development challenge. While most agree that design changes at the earlier stages of the design process (architecture or RTL) offer the highest potential gain, a proven general methodology is not yet available. Available solutions are very limited in their scope. At the other end of the spectrum, there is transistor-level analysis, which enables the highest level of accuracy, but is simply not an option for most designers, because of the simulation time involved with transistors. Gate-level power analysis is faster by several orders of magnitude than transistor-level analysis while still providing the accuracy necessary to produce low power designs. All details about transistors are abstracted away in gate-level analysis which makes them much faster than transistor-level analysis. While RTL estimation is typically used to provide early feedback to the designer to make decisions among design choices, gate/transistor level analysis is used for formal verification of a design's power consumption.

1.2 Sources of Power Consumption

Power consumption in CMOS circuits can be divided into two types: static and dynamic, also known as DC and AC power.

1.2.1 Static Power

Static power represents the power dissipated when a gate is in a steady state. Although the static power is only a small component of the total power, it is becoming important for portable designs where static power dissipation during idle phases becomes a significant part of the overall power dissipation. The sources of static power dissipation in a digital CMOS device are

- leakage currents
- static currents.

Leakage currents consist of reverse-bias diode leakage at the transistor drains, and sub-threshold leakage through the channel of an off device. This is shown with arrow 2 in Figure 1.2. Leakage power is process technology dependent and typically in the microamp range. While this is typically a small fraction of the total power consumption, it could be significant for a system application which spends much of its time in standby operation, since this power is always being dissipated even when no switching is occurring. Static current represents the current which is

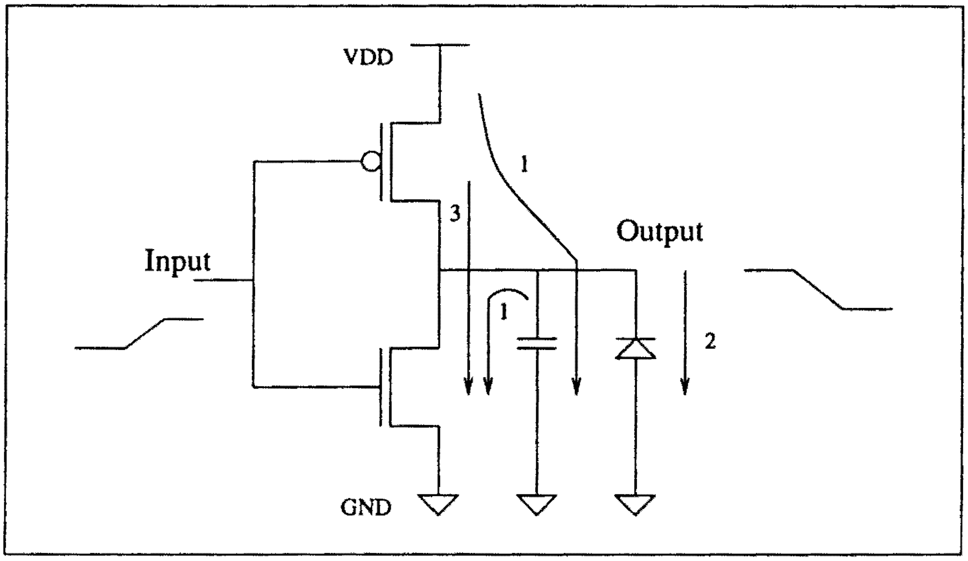


Figure 1.2: Sources Of Power Dissipation In CMOS Circuits

continuously drawn from the supply in certain kinds of circuits, when the circuit is in a certain state. For example, in a pseudo-NMOS inverter [Weste93], the pull-up transistor is on all the time and whenever the pull-down transistor is also on, current flows from supply to ground, continuously, contributing to static power.

1.2.2 Dynamic Power

Dynamic power represents the power dissipated when a gate is switching. The sources of dynamic power consumption are

- Short circuit current
- Capacitive load switching current

The short circuit current is the current flowing directly from power supply positive to ground when both NMOS and PMOS are conducting at the same time during an output transition. This is shown with the arrow labeled 3 in Figure 1.2. The capacitive load switching current is the current flowing in charging and discharging

the output load capacitance, during an output transition. This is shown with arrows labeled 1 in Figure 1.2.
The total power of a CMOS gate is then,

$$Totalpower = Dynamic\ power + Static\ power$$

In modern CMOS circuits, dynamic power is the dominant component of the total power. This will change in the future as we move towards very low supply voltages, and static leakage power could become a significant portion of the overall power dissipation [Chandrakasan96]. Veendrick [Veendrick84] showed that with proper gate sizing, the short circuit component of the dynamic power will be less than 15% of the total power dissipation of the circuit. Hence the capacitive load switching power becomes the dominant component of power dissipation in digital CMOS ICs. The average capacitive load switching power of a digital CMOS gate can be given as

$$P_{av} = \frac{1}{2}.C_L.V_{dd}^2.F_{clk}.\alpha \tag{1.1}$$

where,

C_L is the load capacitance at the output of a gate
V_{dd} is the power supply voltage
F_{clk} is the clock frequency of the circuit
α is the activity factor

The activity factor α is the probability with which the output of the gate changes state during a clock cycle. The term αF_{clk} is then the frequency at which the gate output switches and is referred to as *transition density* of the node.

1.3 Factors Affecting Power Dissipation

From the power equation Eqn. 1.1, it is clear that the power dissipation of a CMOS circuit depends on the load capacitance, the power supply voltage and the clock frequency. The load capacitance is fixed by the circuit structure. The supply voltage is constant for a design and so is the clock frequency. The activity factor, α, depends on

1. Switching activity of circuit inputs
2. Spatial and temporal correlations among the circuit inputs
3. Logic function of the circuit

The first factor is related to the pattern dependency of the power as discussed in [Najm1'95]. The higher the input switching activity, the higher the power. This value can vary for different circuits and different data representations. For finite state machines, this varies from 0.08 to 0.18 [Rabaey96]. For video signals the most significant bits have switching activities of 0.1 whereas the least significant bits have 0.5. The commonly used Uniform White Noise (UWN) has a switching activity range of 0.4-0.5. The second factor takes care of the temporal correlations and spatial dependencies of the circuit inputs. For example, if two inputs to a circuit cannot be one at the same time, then they are said to be spatially correlated. If there are dependencies for a signal in the time domain, then there is said to be a temporal correlation for that signal (e.g. a signal is a 1 only if its previous value is 0). Feedback in a finite state machine (FSM) creates temporally and spatially correlated signals. These correlations at the inputs of a circuit determine what switching happens at the output, as not all switching combinations might then be possible [Pedram96]. The logic function of the gate naturally plays a part in the activity of the output. For example, the activity at the output of a 2-input XOR is 1/2, if all possible combinations of input transitions (equally likely) are considered (i.e. 16). While the activity at the output of a two input AND gate is 3/8, if all possible combinations of the input transitions are considered.

1.4 Combinational Circuit Power Estimation

Pure combinational circuits are also called ***datapath*** circuits in the IC jargon. They are identified by a lack of a memory element in the circuit. Power estimation techniques for these kinds of circuits can be divided into low-level (gate/transistor) estimation techniques and RTL (architectural) estimation techniques.

1.4.1 Low-level Power Estimation

Low-level techniques use a gate or a transistor as the basic design unit. These techniques provide very good accuracy because they use accurate transistor or gate models. And because they work with a fixed circuit structure, they account for all signal correlations within the circuit. Transistor-level analysis is much more flexible than gate-level analysis in that it can handle various device models and design styles. But it is slow and so cannot scale to higher levels of integration. Gate-level analysis is less flexible than transistor-level analysis but much faster. The low-level power estimation techniques can be divided into:

1. Simulation-based techniques
2. Analytical techniques

Simulation-based techniques simulate the circuit with a sample set of input vectors and then report the power resulting from the simulation. An important design question is the number of input vectors to be used for the simulation. Techniques like Monte-Carlo [Najm92] [Xakellis94] solve this problem by an appropriate choice of input vectors, based on statistical theory. IRSIM [Horowitz89], PowerMill [Deng94] (transistor-level) and QUICK POWER [Nguyen97] (gate-level) are examples of this kind of simulator.

Analytical techniques propagate the circuit input signal probabilities (which is the probability of the node being in '1' state) from the primary inputs of the circuit to the primary outputs of the circuit, without any simulation. Since they do not use any simulation, they are quite fast. Once the switching activity of each node in the circuit is known, the average power dissipated by each node can be calculated using Eqn. 1.1. Addition of all node powers yields the average circuit power. These techniques use either a zero delay model or a finite delay model [Ghosh92] [Najm94] for the gates. The finite delay model has the ability to capture glitches and the zero delay model does not have that capability. The assumption made while propagating the activity values is that the input values at two consecutive clock cycles are temporally independent. With this assumption, the switching activity of a circuit node can be expressed as $\alpha = 2p(1-p)$, where p is the probability of the signal being a '1'. There are other proposed methods which improve this basic model by including spatial and/or temporal correlations [Roy94] [Marce94] in the input signals in the calculation of switching activity.

1.4.2 RTL Power Estimation

While low-level estimation techniques use a gate or a transistor as the basic unit of design, RTL estimation techniques for the *datapath* work with *reusable micro-architectural blocks* (like adders, multipliers, memory) as the basic unit of design. RTL techniques are also sometimes referred to as architectural techniques. Since the granularity of an RTL description is larger, the power analysis at this level is faster, but this speed of estimation comes at the expense of accuracy. The combinational power estimation techniques at this level can be divided into two groups: predictive and descriptive. These two techniques differ in that the predictive techniques do not require a pre-characterization step as part of the power estimation process whereas the descriptive techniques do.

Predictive techniques proposed thus far depend on

- Complexity-based models or
- Entropy-based Models

Complexity-based models return quick, but inaccurate values for power by characterizing the complexity of the design in terms of the gate-equivalent count [Glaser91] [Svensson94]. Gate equivalent count is the average number of reference gates needed to build a particular functional block. These models require only information about the gate-equivalent count and technology parameters. The main drawback of these models, is that the predicted power is independent of the circuit input activity, which is not true in practice.

Entropy-based techniques [Nemani97] [Marce96] introduce the concept of Entropy, which is a measure of the amount of information available in a signal. The entropy-based power model proposed by Najm et. al. in [Nemani97], is

$$Power = Average\ Entropy.Area$$

The entropy term is an approximation for the average switching activity of the circuit and the area term is an approximation for the total switched capacitance of the circuit. The basic assumption made while deriving the above model is that, the capacitance of the circuit is uniformly distributed over the circuit. Temporal and spatial correlations among the input signals are ignored. The average entropy of the circuit is calculated using the approximation that the entropy of the circuit nodes decreases quadratically with circuit depth. The area of the circuit is estimated using the concept of "average cube complexity". Average cube complexity is the average number of literals in the prime implicants of the boolean function implemented by a circuit. The relative erroirs of this model are more than 100% in some cases.

Descriptive techniques have a pre-characterization step in the power estimation process and build an RTL power library, which means that the basic RTL blocks used in the design (e.g. multiplier, multiplexer, ALU) are pre-characterized for their power dissipation. The circuit functionality and structure are known apriori for these RTL blocks and a characterization process of these RTL blocks would yield the RTL power library. Fixed activity descriptive techniques are based on the assumption of fixed input activity for the functional blocks. Example of fixed activity models is PFA [Chau91], which uses uniform white noise activity for the circuit inputs. Activity sensitive models take into account the effect of circuit input activity on power. Examples are ESP [Sato95] and dual bit type method (DBT) [Rabaey94] [Rabaey95]. The power model in all these techniques is basically an equation which gives the relation between circuit input activity, complexity of the circuit and the power. For example, the data-path power model for DBT, which works with 2's complement representation of data, is

$$P_{av} = (N_u C_u + N_s C_s) V_{dd}^2 f \tag{1.2}$$

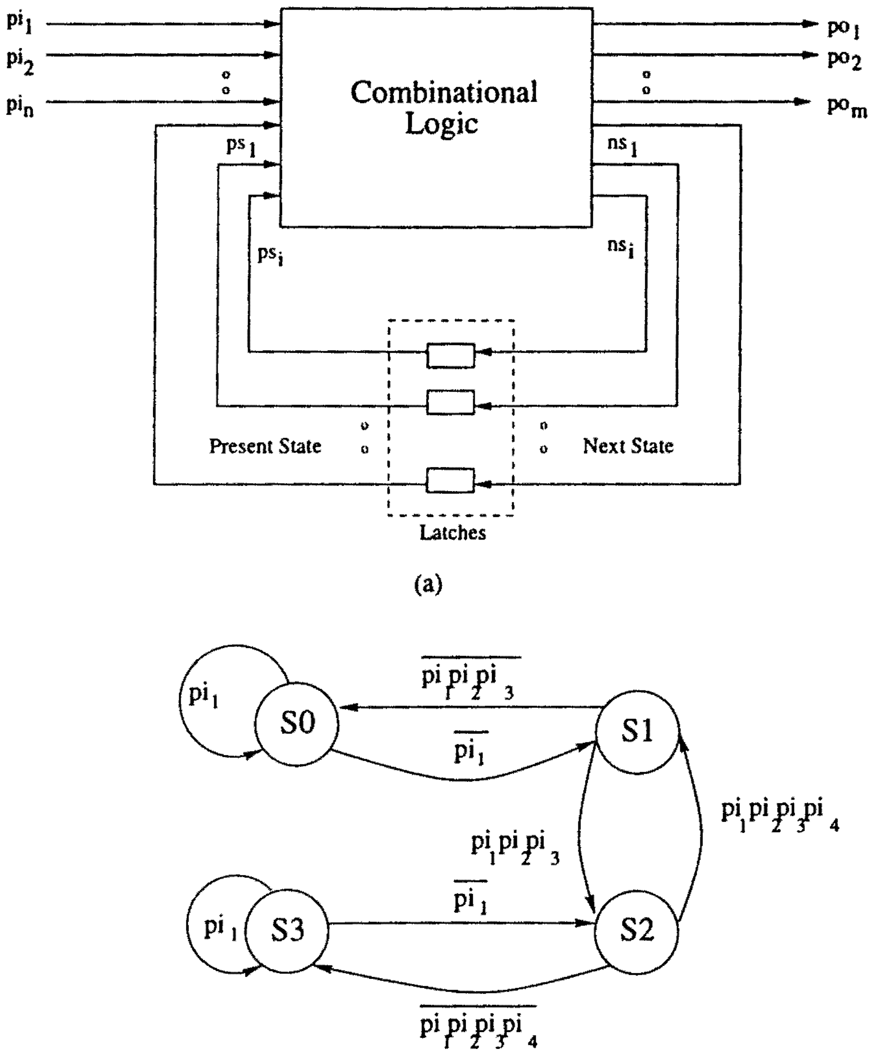
N_u and N_s are the number of sign and data bits in 2's complement representation of the data. C_u and C_s are the extracted coefficients in the characterization process. The activity factors are embedded in these coefficients. [Gupta97] approaches activity sensitive power modeling by building a power table indexed by the primary input activity and the primary output activity. The characterization process of the circuit yields this table. [Nur97] and [Wu97] independently propose an RTL power estimation procedure, where the power dissipated by an RTL block is captured as an equation in terms of the circuit input activity. They account for limited spatial and temporal correlations at the circuit inputs.

All the methods described so far are targeted towards power estimation of pure combinational logic blocks (datapath logic). To complete the RTL power picture of an IC, power estimation methods need to be developed for the control logic (finite state machines) also. These control circuits have a memory element in them and the combinational techniques described above cannot be directly applied for their power estimation. *Control logic* has the same meaning as *sequential circuit*, *finite state machine* and *random logic* and these will be used interchangeably throughout this thesis. The next chapter discusses the methods proposed so far for power estimation of control logic (sequential circuits). Chapter 3 discusses the new method proposed in this thesis for RTL power estimation of control logic. Chapter 4 presents the results of evaluation of the new power model. Future work and conclusions follow in the last two chapters.

Chapter 2

Previous Work In Sequential Circuit Power Estimation

All sequential circuit power estimation techniques assume the popular and well-structured design style of a synchronous sequential circuit, shown in Figure 2.1(a). The circuit contains a single clock driving a bank of edge-triggered flip-flops. On the rising edge of the clock, the flip-flops transfer the values at their inputs to their outputs. The signals pi_1, pi_2, ..., pi_n are the primary inputs to the sequential logic circuit and the signals po_1, po_2, ..., po_m are the primary outputs of the sequential logic circuit and the signals ps_1, ps_2, ..., ps_i are the present-state lines. Signals ns_1, ns_2, ..., ns_i are the next state lines. The primary inputs and the present state lines determine the next state lines and the primary outputs and this circuit implements a finite state machine (FSM). During any clock cycle the finite state machine is in a known state, determined by the values of the present state lines. A symbolic representation of a finite state machine is shown in Figure 2.1(b). Such a representation is called a state transition graph (STG). A state transition graph enumerates all the different states in a state machine and the primary input combinations that take the state machine from one state to the other. The transitions from one state to the other are indicated by directed arrows in the STG and the primary input combinations that cause a transition to occur are shown beside the transition. For example, in the STG in Figure 2.1(b), the state machine transitions from state S1 to S2 when the primary input combination $\{pi_1, pi_2, pi_3\}$ is $\{111\}$.



(a)

(b)

Figure 2.1: Model of a Finite State Machine and its State Transition Graph Representation

Sequential circuit power estimation techniques differ from combinational circuit power estimation techniques in that the sequential circuit power estimation techniques need to account for the presence of feedback (through the state lines) in the circuit. There is no feedback in pure combinational circuits. This presence of feedback in the circuit makes sequential circuit power estimation a hard problem. It is hard on two counts:

1. The presence of feedback in an FSM introduces spatial and temporal correlations among the state inputs to the combinational logic block. These correlations are very important to power estimation [Kozhaya97] and any sequential circuit power estimation technique should model these correlations properly for accurate power estimation. Not all methods proposed in literature model these correlations for want of faster estimation and so loose some accuracy in the estimation.

2. The test vectors applied to the sequential circuit during power estimation should not take the state machine into parts of the state space where it does not belong i.e., into modes of operation that are unrealistic and may never be exercised in practise. For example, the transition from state S1 to S2 in the STG in Figure 2.1(b) might rarely happen in a typical application of the finite state machine, whereas with randomly generated inputs to the finite state machine, this transition might happen more often. These modes of operation of an FSM are application specific. The solution to this problem is to choose vectors directly from the target application and hence to exercise the state machine as it would be in the real world. But these sets of vectors tend to be very large (sometimes in millions) and it is unrealistic to simulate the FSM for all of these vectors. Vector compaction techniques come in to play here.

With this background, let us review the previous work done in sequential circuit power estimation. Power estimation methods for sequential circuits have been proposed at both the RTL and gate-level of abstraction. RTL techniques rely on building a *power model* for finite state machine power dissipation, either through a characterization process or through theoretical models for circuit activity. Gate-level techniques estimate power either through simulation of the circuit or through analytical techniques for propagating the circuit input switching probabilities to the output of the circuit. Power estimation at the RT-level has the advantage that if the circuit input statistics change, only the power model needs to be reapplied with the changed input statistics, to reevaluate the power. This is a trivial process. At gate-level, the designer needs to go through the whole process of power estimation every time the input statistics change. This might include simulation of the circuit or re-propagating the circuit input probabilities to the output. This is a time consuming process.

2.1 Gate-level Sequential Circuit Power Estimation

Power estimation techniques at the gate-level of abstraction, use gates (e.g. Nand, Nor, Xor) as the basic design unit. The RTL description of the design is synthesized into gates (unique to each technology-library) and the resulting gate netlist is the input to the gate-level power estimation techniques. Techniques at this level are either simulation-based or analytical.

2.1.1 Simulation-Based Techniques

Simulation-based techniques, in general, apply a limited set of vectors to the circuit under consideration and observe the resulting power dissipation. The flow chart in Figure 2.2 gives an overall view of one kind of simulation-based technique called Monte-Carlo. The vectors applied to the circuit can be either randomly generated or could be picked from a set of typical vectors from an application (if available). The power value at the end of the sample phase is noted and used to decide whether to stop the process or to apply another vector. The decision is made based on the mean(η_T) and standard deviation(s_T) of the power values observed at

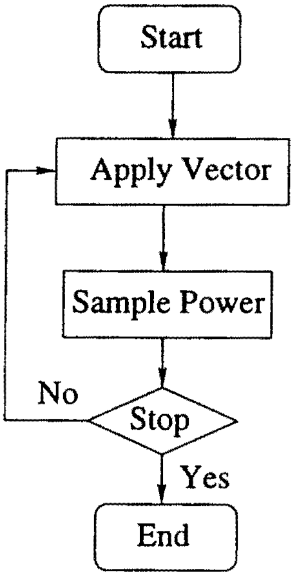


Figure 2.2: Monte-Carlo Power Estimation Method

the end of successive iterations. For a given confidence level $(1 - \alpha)$, the circuit is simulated till,

$$\frac{t_{\alpha/2} s_T}{\eta_T \sqrt{N}} < \varepsilon \tag{2.1}$$

where N is the number of the simulation run
ε is the percentage error tolerable in the measured power
$t_{\alpha/2}$ is obtained from the t distribution [Papoulis84] at N-1 degrees of freedom

The circuit power is found as the average value of the power values obtained during the successive sample phases.

Najm et. al. proposed a Monte-Carlo simulation-based technique [Najm2'95] which takes advantage of the existing combinational circuit power estimation techniques, to compute the average power of a sequential circuit. They break the FSM into a combinational logic block and a set of latches and independently estimate their power. To do so, the state line statistics at the input of the combinational logic block of the FSM must be known. Their technique, then, consists in applying a number of *randomly* generated input vectors to the circuit and collecting statistics (i.e. signal probabilities) on the latch outputs using fast zero-delay logic simulation or using functional simulation of a structural RTL description. High-level simulation can be done very fast, so one can afford to simulate a large of number cycles. Once these statistics are available, it is then possible to use any of the existing combinational circuit techniques, discussed in the previous chapter, to compute the total power.

The state line probability estimation starts with putting the FSM in some known initial state X0. With the assumption that state of the machine becomes independent of its initial state as time → ∞, we can write for the signal probability of a state signal ps_i:

$$\lim_{k\to\infty} P(ps_i|X0) = \lim_{k\to\infty} P(ps_i(k) = 1|X(0) = X0) = \lim_{k\to\infty} P(ps_i(k) = 1) = P(ps_i) \tag{2.2}$$

The index k represents time. The term $P(ps_i|X0)$ represents the signal probability of the state signal ps_i, given that the initial state of the state machine is X0. So the method consists of estimating $P(ps_i|X0)$ for increasing values of k until convergence according to the above equation is achieved. The method is actually implemented by doing repeated simulation runs of the circuit, starting from some initial state X0, and randomly generated input vectors (consistent with the statistics of the primary inputs of the state machine) and each of these will result in a logic waveform $ps_i{}^j(k)$, k=0,1, 2.... where j indicates the run number. If we average the results at every time k, we obtain an estimate of the probability at that time as follows:

$$ps_i{}^N(k) = \frac{1}{N}\sum_{j=1}^{N} ps_i{}^j(k) \tag{2.3}$$

From the law of large numbers, it follows that:

$$\lim_{N\to\infty} ps_i{}^N(k) = P(ps_i|X0) \tag{2.4}$$

The value of N for a user-specified error-tolerance and confidence level can be found from estimation of proportions as $N \geq max(N_1^2, N_2^2, N_3^2)$, where

$$N_1 = \frac{z_{\alpha/2}2}{2\varepsilon} \quad N_2 = \frac{z_{\alpha/2}\sqrt{2\varepsilon + 0.1} + \sqrt{(\varepsilon + 0.1)z_{\alpha/2}{}^2 + 3\varepsilon}}{2\varepsilon}$$

$$N_3 = \frac{\sqrt{63} + z_{\alpha/2}}{2\sqrt{\varepsilon}}$$

So with N known, N parallel simulations of the circuit are started and simulated for increasing values of k till $P_k(ps_i|X0) = P(ps_i)$, as per Eqn. 2.2. To answer the question of what value of k is large enough to achieve this convergence, Najm starts two sets of simulation runs of the circuit with different initial states X0 and X1 and finds out $P(ps_i|X0)$ and $P(ps_i|X1)$. When both the difference and average remain within a window of $\pm\varepsilon$ for three consecutive time instances, convergence is declared. The disadvantage with this approach is that the latch and combinational logic block are decoupled during power analysis and hence the spatial and temporal correlations at the state inputs of the combinational block are ignored. This leads to inaccuracies. Unfortunately, no results for actual power measurements are reported in the paper. Only the efficiency of the method in computing the state line statistics for uniform white noise (UWN) inputs is reported.

An improved Monte-Carlo method proposed in [Saxena97] consists of simulating the finite state machine as one whole block, thereby automatically accounting for the spatial and temporal correlations among the state bits of the FSM. Here again, the assumption is made that the FSM becomes independent of the initial state as the circuit is simulated for an increasing amount of time. The power estimation procedure starts by doing two sets of N simulations each. The two sets have different initial states for the state machine. All simulations within a set have the same initial state and the input vectors for each of these simulation runs are chosen independent of each other. For each simulation run in a set, j= 1,2,....N and for each clock cycle k=1,2,... we can compute the energy consumed per cycle and the power up to time K:

$$P_K{}^j = \frac{1}{KT_c}\sum_{k=1}^{K} e^j(k) \tag{2.5}$$

where T_c is the clock cycle time.
$e^j(k)$ is the energy in cycle k in the simulation run j.

For a given K, the set $(P_K{}^1, P_K{}^2, ..., P_K{}^N)$ will form a random sample of the random variable P_K, and from law of large numbers, the average of these values will tend to $E(P_K|X0)$.

$$\mu_N(K) = \frac{P_K{}^1 + P_K{}^2 + P_K{}^3 + ... + P_k{}^N}{N} \approx E[P_K|X0] \tag{2.6}$$

Najm fixes N to be 50 and to improve the convergence of the average value of the 50 samples to $E(P_K|X0)$, he increases the value of K, because

$$\lim_{k\to\infty} Var[P_K|X0] = 0 \tag{2.7}$$

If the value of K is large enough such that

$$\sigma_N(K) \leq \frac{\varepsilon\mu_N(K)\sqrt{N}}{z_{\alpha/2}} \tag{2.8}$$

is satisfied, then the condition in Eqn. 2.6 is achieved. We can then start monitoring the two power values resulting from the two sets of runs to determine if their $\mu_N(K)$ have converged to $E(P_K)$. When this convergence is achieved, the simulation is stopped. Because of the assumption that the machine becomes independent of the initial state as time tends to infinity, the two values of power must converge to the same value as simulation is done longer and longer. Convergence is declared when the average and the difference of the two values are within a window of $\pm\varepsilon$, in three consecutive clock cycles. The advantage with this approach is that the correlations at the state inputs of the finite state machine are automatically considered as the latch and combinational logic block are not decoupled. Errors of less than 5% at reasonable computation time have been reported.

The modified Monte-Carlo method [Roy96] removes the assumption made in [Najm2'95] [Saxena97], of the state-machine becoming independent of the initial state as the machine is exercised over a long period of time. This could prove important in some machines because of the existence of near-closed (NC) sets, which can bias the sampled power value and cause the simulation to terminate improperly. A Near-closed set is a set of states such that once the state machine enters that set of states, the probability that the state machine will leave that set of states is very low, or vice-versa. These NC sets are a problem, because if during simulation the initial state of the state machine is chosen to be in one of these sets, then the sampled power value during the simulation will be reflective of only those states inside that set. This is because with the set of vectors applied during the simulation, the state machine might have never gone from one NC set to the other. This method builds over the existing Monte-Carlo techniques. The only difference is that, instead of simulating the circuit with some random initial state, this method carefully selects the initial state, after identifying the presence of near-closed sets. For example, if there are two near-closed sets in a state machine, with probabilities $P(G_1)$ and $P(G_2)$ ($P(G_1)+P(G_2)=1$), then in each of N simulations of the circuit, the initial state is chosen from G1 with probability $P(G_1)$ and from G_2 with probability $P(G_2)$. So the normalized power would be given as

$$Power = (Power|G_1)P(G_1) + (Power|G_2)P(G_2) \tag{2.9}$$

The increased accuracy that this method brings, comes at the cost of extra computation to determine first the presence of near-closed sets and if present, their probabilities. Average relative errors of less than 3% have been reported with this method.

Yuan and Kang [Kang97] propose a statistical procedure to overcome the problem in all statistical mean estimation techniques of requiring independent and identically distributed power data i.e., a random sample of mutually independent power data. The sequential procedure they recommend determines an appropriate *independence interval*, separated by which two sample power data in a power simulation can be treated as mutually independent. A distribution-independent stopping criterion is applied to choose an appropriate convergent sample size for the power data. Very accurate results have been reported for this method (less than 1.5% error), for input activity of 0.5.

All the simulation-based techniques discussed so far require simulation vectors to be either randomly generated or to be picked directly from a typical application of the state machine. Sometimes these typical vector sets are very large (in the order of millions). It is unrealistic to simulate the machine for all these vectors. Techniques have been proposed for vector compaction while preserving the correlations present in the vector set, both temporal and spatial. They either use data structures [Pedram97] [Marce97] to capture the correlations of a fixed order in the input or use statistical techniques [Kozhaya97] to select a limited set of "blocks of vectors" from the large input set.

2.1.2 Analytical Techniques

Analytical techniques for sequential circuit power estimation, in general, are based on solving a system a linear or non-linear equations, for the state line probabilities. All these techniques assume that the FSM is Markov [Papoulis84], while calculating the state line probabilities, so that its future is independent of its past once its present state is known. Once the state line probabilities are known, and with primary input probabilities available from the user, the activity of the internal circuit nodes is calculated by propagating these input signal probabilities. The input signal/state probabilities are propagated to the internal circuit nodes under the assumption of input spatial independence. Average power of the circuit is then calculated from the basic power equation. The approach in [Hachtel] attempts a direct solution of the Chapman-Kolmogorov equations [Papoulis84]. While it is accurate, it remains computationally quite expensive and the largest test cases presented contain less than 30 latches. The method in [Devadas94] solves a system of non-linear equations to get the present-state probabilities. Given the probabilities

$p_1, p_2, ..., p_n$ of the primary inputs of the state-machine and the probabilities of the present state lines $P_{ps} = (ps_1, ps_2, ..., ps_n)$ and assuming that the present state lines are independent, one can compute the next state probabilities as $F(P_{ps})$. The function F() is non-linear and is determined by the combinational function implemented by the state machine. In general, if the next state probabilities form a vector P_{ns}, then $P_{ns} \neq F(P_{ps})$, because the present state lines are not independent. But the method in [Devadas94] makes the assumption that $P_{ns} \approx F(P_{ps})$. Since $P_{ns} = P_{ps}$, due to feedback, the system solves for the system P = F(P). One of the problems with this approach is that of independence assumption for the state lines, which is not true in practice. [Devadas94] also tries to correct this problem by accounting for m-wise correlations between state bits when computing their probabilities. Inspite of all these shortcomings, good results have been reported with this method (errors less than 5%).

2.2 RT-level Sequential Circuit Power Estimation

Register-Transfer Level power estimation of sequential circuits is better than gate-level power estimation in two ways:

- Power estimation at the Register-Transfer level is available earlier in the design cycle. In the absence of RTL power estimation, synthesis of the RTL design to gates and power estimation with gate-level power tools are necessary steps in the design process.

- Power estimation at this level is faster. This is because at the RT-level we work with boolean equations rather than circuits, which means that the granularity is larger.

The structural RTL representation of a VLSI IC is shown in Figure 2.3 [Nur97]. At the RT-level, an FSM has no fixed circuit structure yet i.e., the RTL description of

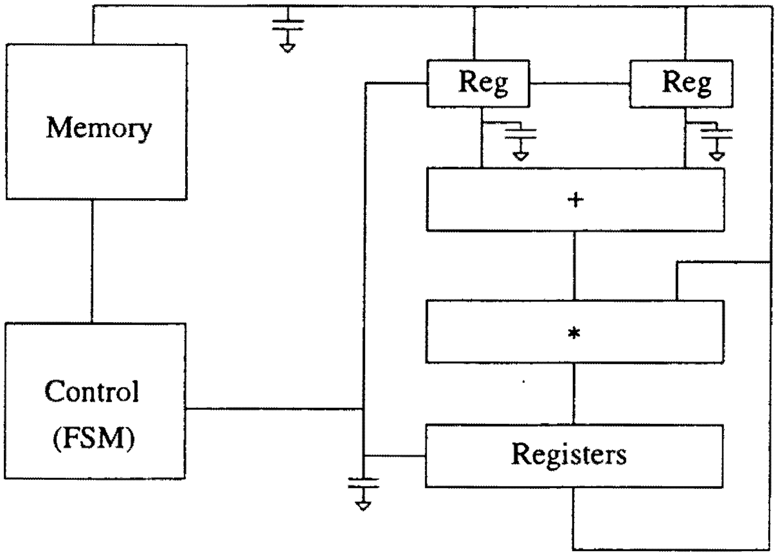


Figure 2.3: Structural RTL Representation of an IC

the FSM has not yet been synthesized into gates. This makes power estimation of an FSM at RT-level a very hard problem. Each design has its own unique control path, which means that there are, to most parts, no standard control path designs. This is unlike the datapath elements (micro-architectural blocks) like the adder, multiplexer etc., which are commonly used in most designs. The functionality and circuits for these standard datapath elements are known apriori, which in effect, fixes the physical capacitance (or power) of these elements. Characterizing these known circuits for power would then allow for RTL power estimation of the datapath. But the power dissipated by a finite state machine depends on the specification of the finite state machine, which is not known until run time. So power prediction at this level has to be purely based on parameters available at this level, like the number minterms in the boolean function representing the state machine, the number of literals in the sum of product terms of the boolean function etc.. All proposed techniques for sequential circuit power estimation at RT-level fall into one of two categories:

- Predictive techniques

- Descriptive techniques

All of these techniques break the FSM into a combinational block and a set of latches. The techniques then estimate the power of the combinational block. The latch power, which could be calculated from a knowledge of the latch output activity obtained from a high level simulation, is then added to the combinational logic power to yield the total power. Since these techniques decouple the latch and the combinational logic block, the correlation information at state inputs of the combinational logic block are lost.

2.2.1 Predictive Techniques

These techniques estimate power without the need for a pre-characterization step. All predictive techniques found in literature for FSM power estimation at RT-level use the concept of entropy, an information theoretic measure, to estimate circuit activity. The entropy of a random boolean variable X is defined as

$$H(X) = p\ log\frac{1}{p} + 1 - p\ log\frac{1}{1-p} \tag{2.10}$$

where p represents the signal probability. In the entropy-based power model presented in [Nemani97] [Nemani96], the authors observe that the power is proportional to the product of physical capacitance and activity. They then use area as a measure of physical capacitance and entropy as a measure of activity. In using entropy as an approximation for activity, the input signals are assumed to be temporally independent.

$$P_{av} \propto Capacitance \times Activity \propto H \sum_{i} C_i$$

A model is then derived for the average entropy (H) as a function of input and output entropies, under the assumption that the entropy inside the circuit decreases quadratically with circuit depth. The final expression obtained for the average entropy is shown in Eqn. 2.11, where n is the number of primary inputs, m is the number of primary outputs, H is the total entropy, H_i is the total input entropy and H_o is the total output entropy.

$$H \approx \frac{2}{3} \frac{(H_i + 2H_o)}{n+m} \tag{2.11}$$

Nemani and Najm also present a model for the area complexity of a boolean function based on the concept of average cube complexity. Average cube complexity is the average literal count of the prime implicants of the function. The area model is

$$A(f) = 2^{C(f)} k(H) \tag{2.12}$$

where $C(f)$ is the average cube complexity and $k(H)$ is a proportionality constant that depends on the entropy H. The relative errors of the above models have been more than 100% [Nemani97] in some cases. The average entropy proposed in [Nemani96], underestimated the circuit activity in some examples. This is understood because the circuit activity depends on the functionality of the circuit as well as the type of data being processed. For example, the proposed model will return the same result for two different functions that have identical average entropy. Additionally the average entropy may be predicted to be the same for two different input streams applied to the circuit. Marculescu et. al. [Marce96] have made an effort to estimate switching activity based on both the entropy and the distribution of nodes in the circuit.

All the predictive techniques discussed above have some practical limitations. First of all, they ignore glitch power because they assume zero-delay model of operation. Temporal correlations at the inputs are ignored because of the assumption of independence of signal values in successive clock cycles. And finally, the circuit capacitance is assumed to be uniformly distributed over the circuit, which might not hold in practise.

2.2.2 Descriptive Techniques

These techniques require pre-characterization of existing finite state machines of varying complexities and construction of an RTL power model equation. Activity based control model (ABC) proposed by Landman [Landman96] builds a model for an FSM implemented in standard cells as:

$$P_{av} = (C_i t_i N_i N_M + C_o t_o N_o N_m) {V_{dd}}^2 f \tag{2.13}$$

The first product term measures the input plane complexity of the FSM i.e., the amount of decoding that needs to be done at the input of the FSM. Similarly the second term measures the output plane complexity. t_i, t_o are the transition probabilities of the inputs and outputs i.e., the fraction of the input and output bits that toggle per cycle. N_i is the sum of the number of state bits and primary inputs. N_o is the sum of the number of state bits and primary outputs. N_M is the number of minterms in the logic minimized control table of the finite state machine. C_i and C_o are the capacitive coefficients estimated during the characterization phase. The characterization phase consists of actually measuring the switched capacitance (a proxy for power) for FSM (in control table format) implementations of varying complexities and input and output activities. Since it is impractical to characterize all possible FSM implementations, *randomly generated* control tables are used instead. The observed switched capacitance values are then used to find the capacitive coefficients that give the best fit to the measured data. All terms except N_M are available from a high level description/simulation of the state machine. N_M is obtained by optimizing the circuit using a logic optimizer like espresso and then using the number of min-terms in the minimized boolean table as an estimate of N_M. Max error of 29% has been reported [Landman96] with this approach. Unfortunately, results of model performance are shown for just two circuits. The disadvantage with this approach is the requirement of a logic optimization step to determine N_M. State line correlations are not accounted for since this technique decouples the combinational block from the latches.

2.3 Chapter Summary

This chapter summarized the techniques proposed so far for sequential circuit power estimation. Each method has its own advantages and disadvantages. Only the gate-level Monte-Carlo simulation-based techniques take care of the correlations among the state bits of a sequential circuit. None of the RTL methods account for the correlations among the state bits. Gate-level circuits require the synthesis of the FSM RTL description to a set of technology-dependent library-gates. None of the RTL methods need this step, though the ABC model requires a single run of a logic-minimization tool like espresso. Errors in excess of 100% have been reported for some of the RTL techniques. Gate-level techniques are much more accurate with errors less than 5%. The power model studied in this thesis takes a path in between the existing RTL and gate-level techniques in an attempt to balance speed and accuracy. This will be discussed in the next chapter. Note that none of the techniques found in literature attempt a *qualitative* judgement about choice of finite state machine implementation (like one-hot Vs. horizontal-encoded FSM) from a set of broad alternatives. They all attempt only a *quantitative* measurement of the average power of an FSM. The new method studied in this thesis affords both a quantitative measurement of the average power of an FSM and also a qualitative comparison among different finite state machine implementations. This new method is studied in the next chapter.

Chapter 3

New RTL Power Estimation Method For Sequential Circuits

Chapters I and II talked about the general problem of power estimation and the methods proposed so far for power estimation of sequential circuits. The picture that emerges from these two chapters is that power estimation at the RT-Level is much faster than at the lower levels, but at a reduced accuracy. The goal of this chapter is to introduce the new method proposed in this thesis for RTL power estimation of sequential circuits. The proposed RTL power estimation technique differs from the existing techniques in two ways:

- It requires a *low-effort synthesis* step as part of the RTL power estimation process. Low-effort typically means synthesis without optimization.

- The model developed is *synthesis-tool-specific.*

Previous methods were either not synthesis-tool-specific [Nemani97] [Nemani96] [Marce96] or did not require a low-effort synthesis step [Landman96] or both [Nemani97] [Nemani96] [Marce96]. These techniques are fast in power estimation, as is required at the RT-level, but are poor in accuracy. Errors in excess of 100% are reported for some of these techniques, as discussed in the previous chapter. The new method studied in this thesis aims to strike a balance between accuracy and speed of estimation. Also, it includes both qualitative and quantitative measures for power estimation. The quantitative measure attempts to give quantitative results for the average power dissipation of finite state machines. The qualitative aspect of it tries to give enough information to the designer to make informed decisions about choice of finite state machine implementation from a set of broad alternatives.

Pure RTL estimation techniques [Nemani97] [Marce96] [Landman96] are fast, but are inaccurate. Where as, pure gate-level [Najm2'95] [Saxena97] [Roy96] [Devadas94] estimation techniques have very good estimation accuracy, but are slow. The new method attempts to improve the accuracy without trading-off "too much" of estimation time through a "quick" gate-level analysis of the circuit. A "quick" gate-level power analysis of the circuit could be done through probabilistic [Devadas94] [Marce94] or empirical [Shravan98] techniques for propagation of the circuit input activity values to the internal nodes of the circuit and calculating the power dissipated at each node in the circuit. Addition of all these power values yields the overall circuit power. The gate-level analysis takes place on an unoptimized circuit, which is got through a "quick" low-effort synthesis of the RTL description. *Low-effort* synthesis means a synthesis without any design constraints imposed or a synthesis without any synthesis optimization steps, which is much faster than a synthesis step with optimizations included. This time gain is crucial to make the new technique work at RT-level speeds. Examples of time gain achieved for different benchmark circuits will be presented in the next chapter. A secondary factor affecting the speed of this technique, is the speed of the gate-level power analysis itself. The faster the analysis, the better. The requirement of a synthesis step as part of the power estimation process binds the model developed to the synthesis tool used to develop the model. The implication of this requirement is that the model has to be recomputed if the synthesis tool changes. But, the proposed method is generic in its applicability towards power estimation from any given RTL description. Though the new model has a direct application in power estimation of finite state machines, it could as well be used to predict the power of an unsynthesized RTL-block (also called micro-architectural block) like an adder, multiplexer etc., which are combinational.

Term	Meaning
SOP	The number of product terms in the sum-of-product expressions of the primary outputs of the FSM.
LIT	The number of literals in the sum-of-product expressions of the primary outputs of the FSM.
DC	The number of product terms in the sum-of-product expressions of the don't-care set of the output functions of the FSM.
DCLIT	The number of literals in the sum-of-product expressions of the don't-care set of the output functions of the FSM.
PI	Number of primary inputs to the finite state machine
PO	Number of primary outputs from the finite state machine
STATES	Number of states in the finite state machine
P_{opt} $Power_{opt}$	Power of the optimized implementation of the FSM
P_{unopt} $Power_{unopt}$	Power of the unoptimized implementation of the FSM
SW_CAP	Switched Capacitance of the FSM
K_x	Power-model coefficients

Table 3.1: Notation

Quantitative power estimation with the new method has two main phases, built upon a *power model*:

- Power Characterization Phase
- Power Estimation Phase

These two phases are discussed in the next few sections. Qualitative measures with the new model would require the same phases mentioned above but without actually building the *power model*. Table 3.1 summarizes the notation used throughout this thesis. The next section discusses the motivation for the new power model and section 3.2 explains the new power model in detail. Sections 3.3 and 3.4 discuss the power characterization and power analysis phases with the new power model. Discussion in the rest of the chapter centers around finding a quantitative measure for the average power. As required, changes that are needed in the different phases to reflect qualitative predictions about power will be made.

3.1 Motivation For A New Power Model

Finite state machine power estimation at the RT-Level can be approached in two ways. One approach would be to assume nothing about the structure of the circuit synthesized from the RTL description of the finite state machine, except to build theoretical models for the distribution of activity within a circuit, as is done in [Nemani96] [Nemani97]. These are *pure-RTL* techniques. This approach has two main disadvantages. The first and foremost being the loss of the ability to model correlations (both temporal and spatial) among the internal signals of the circuit. This is because the circuit structure dictates what correlations exist among the different signals inside the circuit. The resulting inaccuracies could result in errors as high as 100% [Nemani97]. The second disadvantage with all proposed techniques which follow this approach is that, they decouple the latch and the combinational logic of the FSM and work with either the average value of the input signal activity or the average value of the input entropy. This results in the loss of information about the spatial and temporal correlations that exist among the inputs (primary inputs and state bits) to the combinational logic block of the state machine, leading to additional inaccuracies in estimation.

The second approach to RTL power estimation of sequential circuits needs some information about the circuit structure of the synthesized RTL description. Because it needs structural information about the circuit, techniques which follow this approach are not pure-RTL techniques, but rather a combination of RTL and gate-level techniques. Previously done research which follows this approach [Marce96], is analytical in nature and is based on the notion of estimating the average switching activity of the circuit from the input and output entropies (informational energy) of the circuit and an implementation dependent information scaling factor. It does not account for the correlations that exist among the input signals to the finite state machine and also among the signals internal to the circuit. Unfortunately, results are reported in [Marce96] for only RTL combinational-blocks and none for FSMs. The method investigated in this thesis follows the second approach of using some structural information about the finite state machine. The proposed method is motivated by the need for fast power estimation of random logic and finite state machines, at acceptable levels of error. Existing methods are either not sufficiently accurate [Nemani97] or not fully proven [Landman96] [Marce95] to be of commercial value. Looking back at [Nemani97] [Landman96] [Marce95], it is clear that the primary source of error in RT-Level power estimation is the lack of gate-level circuit implementation of the RTL description. So, if it is possible to get a gate-level implementation of the RTL description "quickly", we could extract accurate values of power, through either gate-level simulation or gate-level probabilistic or empirical techniques, at close to RT-level speeds. This method would be accurate because it is based on gate-level power analysis and it would be fast because it needs only a synthesis step with no optimization. But, in practise we are more interested in the power of the optimized implementation of the circuit and it is also a known fact that synthesis with optimization produces very different results for the circuit structure, and hence power, compared to synthesis without optimization. So, a necessary step in the new power estimation method, would then be to characterize the change in power between the optimized and unoptimized implementations of a given circuit.

3.2 Power Model

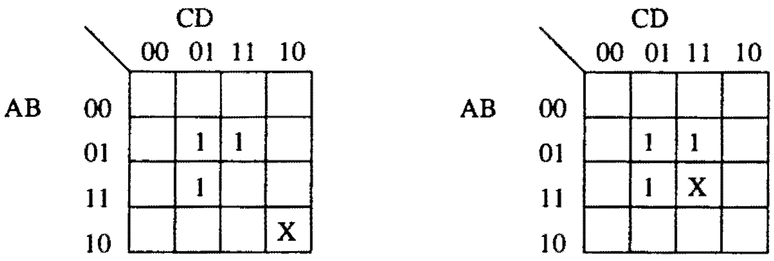
As discussed in the previous section, the new method estimates the power of an unoptimized implementation of the FSM, through a gate-level power analysis and then from a knowledge of how the optimizer changes the power of a circuit during optimization, applies a scaling factor to that power, to yield a final power value.

The effect of the optimizer on power is what is captured in the *power model.*

The parameters available from an RTL description, which are indicative of the complexity of the boolean circuit are listed in Table 3.1. There are parameters other than the ones listed in the table, which could also be obtained from an RTL description. An example would be the average literal count in the prime-implicants of a boolean function. But, these need more processing of the RTL description and are not listed and used in the new model. [Nemani97] uses the average value of literal count in the prime implicants of a function to estimate the gate count and hence the power of the optimized implementation of the function. The idea behind the new power model is to capture the effect of the optimizer (i.e. the synthesis tool) on the power of the unoptimized circuit, as a function of these top-level variables (parameters), i.e.

$$\frac{Power_{opt}}{Power_{unopt}} = f(PI, PO, LIT, SOP, DC, DCLIT, STATES) \qquad (3.1)$$

It is known that all of these top-level variables have some influence on how the optimizer optimizes the circuit. But, the degree of dependence of the optimization on each of these top-level variables, individually, and their interaction effects are not clear. No previous work exists, which models these inter-relationships. As an example of these complex relationships, two circuits with the same number of number of ON(1)-minterms and DC(don't care)-minterms can result in entirely different optimizations (an ON-minterm is a product-term corresponding a '1' entry in the Karnough map and a don't-care-minterm is a product-term corresponding an 'X' entry in the Karnough map). For the two functions shown in Figure 3.1, which have the same number of 1 and don't-care-minterms, optimization results in different expressions for the output function and hence the structure of the circuit. It requires the higher sophistication of a synthesis tool to model all these effects accurately. This is a computationally expensive process and is not amenable to



AB \ CD	00	01	11	10
00				
01		1	1	
11		1		
10				X

$OP = B\bar{C}D + \bar{A}BD$

AB \ CD	00	01	11	10
00				
01		1	1	
11		1	X	
10				

$OP = DB$

Figure 3.1: Illustration of Different Synthesis Results for Circuits with same number of ON-minterms and DC-minterms

RT-level speeds. Also, it is a question of how each optimizer optimizes the circuit. Every optimizer applies its own set of local optimization techniques. Frequently, optimizers use heuristics in their optimization steps. Every optimizer uses a different heuristic. Modeling all optimizer details would then preclude the new technique from being a generic technique. The question in this thesis is whether the optimizer's behavior is predictable from an analysis of interaction of all the top-level variables. If for example the power-ratio is linear in all top-level variables, then the function in Eqn. 3.1 can take on a form like:

$$\begin{aligned} \frac{Power_{opt}}{Power_{unopt}} &= K_1PI + K_2PO + K_3LITERALS + K_4SOP \\ &\quad + K_5STATES + K_6DC + K_7DCLIT + K_8 \qquad (3.2) \end{aligned}$$

Henceforth, we will call the ratio $Power_{opt}/Power_{unopt}$ as the power-ratio. The coefficients K_1 to K_7 represent the slopes associated with the different terms in the equation. The constant term K_8 represents an average value of the power-ratio, in a least-squares sense. The values of these coefficients are estimated through a characterization process, and are specific to a set of cell-library and synthesis and layout tools used. Although these coefficients are in general evaluated for power-ratio values, they would not change if they are instead evaluated for the switched capacitance ratio values, as long as the time period over which the power values are calculated is the same for both the optimized and the unoptimized circuits. Under this constraint of equal time periods, modeling the switched capacitance is the same as modeling power. The methodology followed for finding exactly what the function in Eqn. 3.1 looks like, is discussed in the next chapter. Note that the function in Eqn. 3.1 does not explicitly include an activity term, although power dissipation in a CMOS circuit is a function of the signal activity. This gap is bridged by plotting the power-ratio as a function of circuit input activity for several different circuits and then adding a correction factor to Eqn. 3.1, if the power-ratio is found to vary with activity. This would be a necessary step in the model characterization process.

3.3 Power Characterization

The power characterization process is an *empirical study* of existing finite state machines of varying complexities and consists of extracting the technology-specific power model coefficients. The idea here is to actually measure the difference in power dissipated between the optimized and unoptimized implementations of circuits of varying complexities, for a fixed input activity. The observations are then used to find a set of model coefficients that give the best fit to the measured data. Eqn. 3.1 can be written in matrix form as

$$PR = TK \tag{3.3}$$

PR = Power-Ratio Matrix

T = Parameter Matrix

K = Coefficient Matrix

Eqn. 3.3 in expanded form is shown in Figure 3.2, for a power-model with only linear terms of the top-level variables. The parameter matrix is derived from the RTL descriptions of circuits studied during the characterization phase. Details about the number and nature of circuits studied in the characterization phase (and hence the number of rows in each of the matrices) appear in the next chapter.

$$\begin{array}{l} \text{FSM 1} \rightarrow \\ \text{FSM 2} \rightarrow \\ \\ \text{FSM M} \rightarrow \end{array} \underbrace{\begin{bmatrix} PI_1 & PO_1 & SOP_1 & LIT_1 & DC_1 & DCLIT_1 & STATES_1 & 1 \\ PI_2 & PO_2 & SOP_2 & LIT_2 & DC_2 & DCLIT_2 & STATES_2 & 1 \\ \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ PI_M & PO_M & SOP_M & LIT_M & DC_M & DCLIT_M & STATES_M & 1 \end{bmatrix}}_{\text{Parameter Matrix}} \underbrace{\begin{bmatrix} K_1 \\ K_2 \\ \bullet \\ K_8 \end{bmatrix}}_{\text{Coefficient Matrix}} = \underbrace{\begin{bmatrix} \text{Power-Ratio}_1 \\ \text{Power-Ratio}_2 \\ \bullet \\ \text{Power-Ratio}_M \end{bmatrix}}_{\text{Power-Ratio Matrix}}$$

Figure 3.2: Power-Ratio Matrix

The power-ratio matrix is formed from the power simulation of these circuits with a low-level simulator. The coefficient matrix is estimated by the model coefficient extractor, which is a tool like Matlab. The flow of the power characterization phase is shown in Figure 3.3. The ovals in this figure represent either an input to the flow or an output from the flow. The rectangles represent the functional parts of the flow.

There are five distinct sub-phases in the power characterization phase.

1. High-level circuit synthesis of the RTL description, with and without optimization
2. Input vector generation
3. low-level power estimation
4. Model coefficient extraction
5. Analysis of sensitivity of power-ratio to input activity

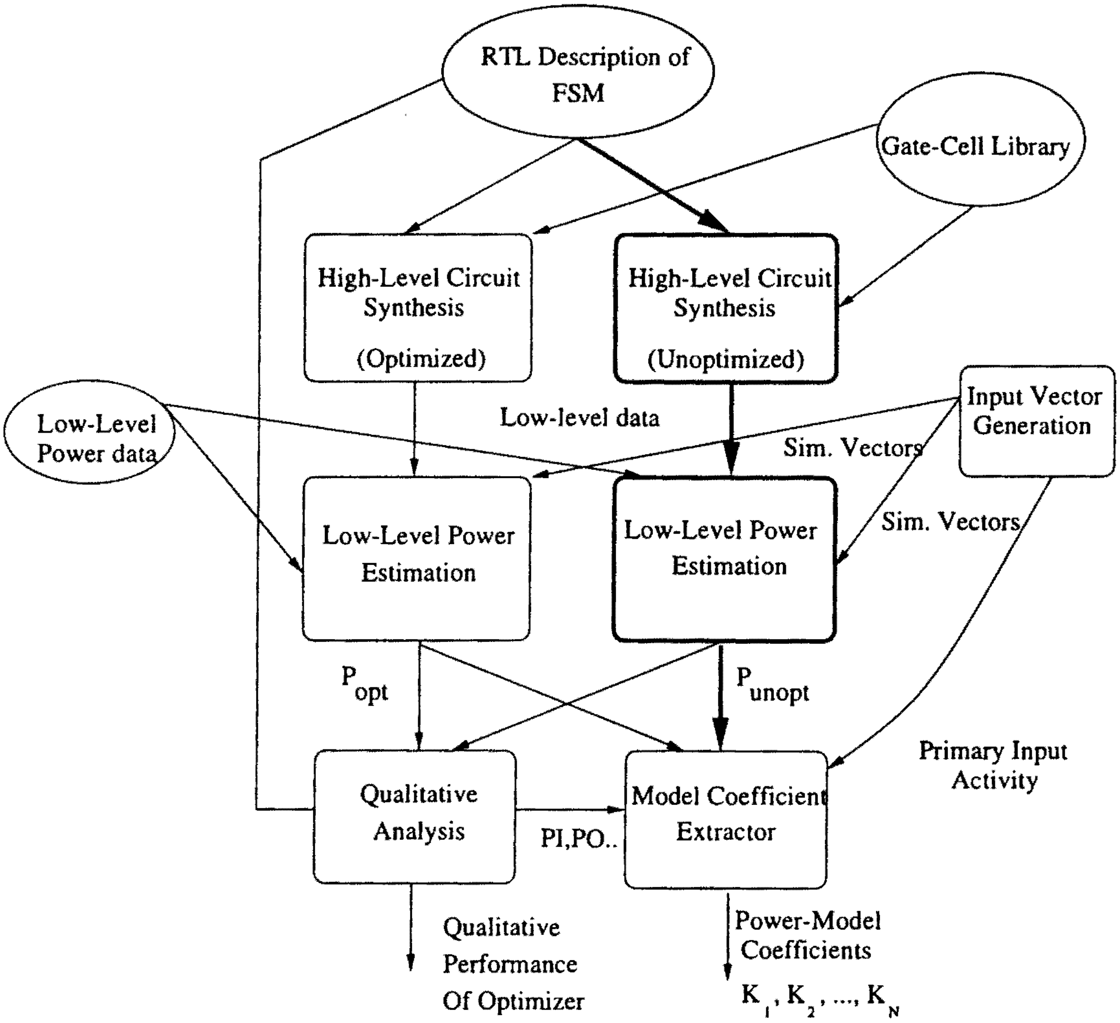


Figure 3.3: Power Model Characterization

3.3.1 High-level Circuit Synthesis

This step involves converting the RTL description of a Finite-State Machine into a low-level (gate or transistor) circuit description. This step can be subdivided into two separate steps:

- Logic synthesis of the RTL description of the finite state machine using a synthesis tool like SIS, to yield a gate-level description of the finite state machine

- Optionally, converting the gate netlist from the previous step into a transistor-level netlist, by placing and routing the gate netlist, using a tool like MAGIC. This step would only be required if the low-level power analysis is done at the transistor level, which would not be preferred for speed considerations.

The two inputs to the synthesis tool are the RTL description of the FSM and a gate-cell library. The layout and route tool takes the transistor-level description of the cells and the gate netlist from the logic synthesis tool, as its inputs. Two separate runs of the high-level circuit synthesis have to be done, one with logic optimization done during the logic synthesis step and another without logic optimization done during the logic synthesis step. These two runs will yield two functionally-equivalent circuits, which have different circuit structures and power values. The ratio of the power values from these two runs of the synthesis tool generates the power-ratio matrix. The reader must be aware here that though two separate runs of circuit synthesis are required during the power characterization phase, only one run of the circuit synthesis (unoptimized run) is needed during the power analysis phase, when the designer actually uses the model developed in the characterization phase. This flow is shown with dark arrows/boxes in Figure 3.3. More details of the power analysis phase follow in the next section. There are several different optimizations possible during the logic synthesis step. Optimization could be done for minimum area of the resulting circuit or optimization could be done for minimum delay of the resulting circuit, or it could be done for a combination of the two. Each of these optimizations could potentially result in different values for the model coefficients. The choice of the optimization made for study in this thesis is discussed in the next chapter.

3.3.2 Input Vector Generation

This step generates the input vectors, of given statistics, for use by a low-level power simulator like QUICK POWER (gate-level) or IRSIM (transistor-level). Questions about the number of vectors, time of simulation for each vector and circuit input switching statistics used for the experiments will be addressed in the next chapter. It would suffice to say now that a supplementary program generates these vectors. The inputs to this program are the number of vectors to be generated, the number of bits in a vector and the input vector statistics.

3.3.3 Low-level Power Estimation

This step involves estimating the power dissipated by the optimized and unoptimized implementations of the FSMs under study, using a low-level power estimation tool like QUICK POWER. The power estimation tool takes the two circuits generated from the high-level circuit synthesis step as its input, along with the vectors generated by the vector generation program. Power estimation could either be simulation-based, as with QUICK POWER or it could be based on any gate-level probabilistic or empirical techniques. The presence of a power-characterized cell-library along with a method for efficient propagation of the circuit input transition probabilities to the internal nodes of the circuit, would make the latter choice faster than the simulation-based techniques. Note that the model coefficients would directly depend on the low-level power analysis tool used and this creates one more binding on the power model. In simulation-based power estimation, a couple of precautions have to be taken during the simulation runs. The power simulation starts with putting the state machine in some random initial state. First, care has to be taken to make sure that the choice of initial state for the state machine does not the bias the measured power value. This could be done by choosing a vector sequence long enough to make the state machine independent of the initial state. Secondly, the simulation runs have to account for the presence of near closed sets in the state machine, if there are any, as per the discussion in chapter 2. So, instead of just doing one simulation run with some random initial state, repeated runs with different random initial states have to be done.

3.3.4 Model Coefficient Extraction

The penultimate step in the characterization process is the model coefficient extraction. This step extracts values of the model coefficients K_1,K_2.... K_N. The model coefficients are extracted by solving the linear system shown in Figure 3.2, using a technique like Singular Valued Decomposition (SVD). The power-ratio matrix is obtained at the end of the low-level power estimation step. The parameter matrix is obtained by processing the RTL description of the FSM, through a supplementary program.

3.3.5 Analysis of Power-Ratio Sensitivity to Input Activity

The final step in the characterization process would be to plot the sensitivity of the power-ratio of an FSM, as a function of the primary input activity. This could be done by evaluating the power-ratio for various values of the primary input activity and plotting them against the activity values. A complete range of input statistics would yield a robust model. These plots must be repeated for a number of circuits of varying sizes. These plots would determine whether a correction factor is needed to be applied to Eqn. 3.1, to account for any effect that the input activity might have on the power-ratio.

For making *qualitative* inferences about the "power performance" of the optimizer, the only change required in the power characterization phase described above is that the model coefficient extraction and sensitivity analysis steps are not required. No *power model* is actually built i.e., no coefficients are extracted. Instead, we look for qualitative trends and measures regarding the performance of the optimizer in changing the power of an unoptimized circuit. This is shown with the *Qualitative Analysis* box in Figure 3.3.

3.4 Power Analysis

Given the power model from the power characterization phase, the power analysis process estimates the power of the optimized implementation of a given finite state machine. The flow-chart in Figure 3.4 gives the power analysis process. The power analysis of a FSM consists of four steps:

1. High-level Circuit Synthesis of the FSM with no optimization
2. Low-level power analysis of the unoptimized circuit
3. Power-Ratio calculation based on the RTL description of the FSM and the power model
4. Application of the power-ratio from step 3 to the power value from step 2, to predict the power of the optimized circuit.

3.4.1 High-level Circuit Synthesis Without Optimization

Power analysis starts with an unoptimized circuit synthesis of the RTL description of the FSM under consideration, to yield a low-level description of the design. The cell-library used during synthesis must be the same as the one used during the model characterization phase. The resulting structural description of the design will become the input for low-level power analysis.

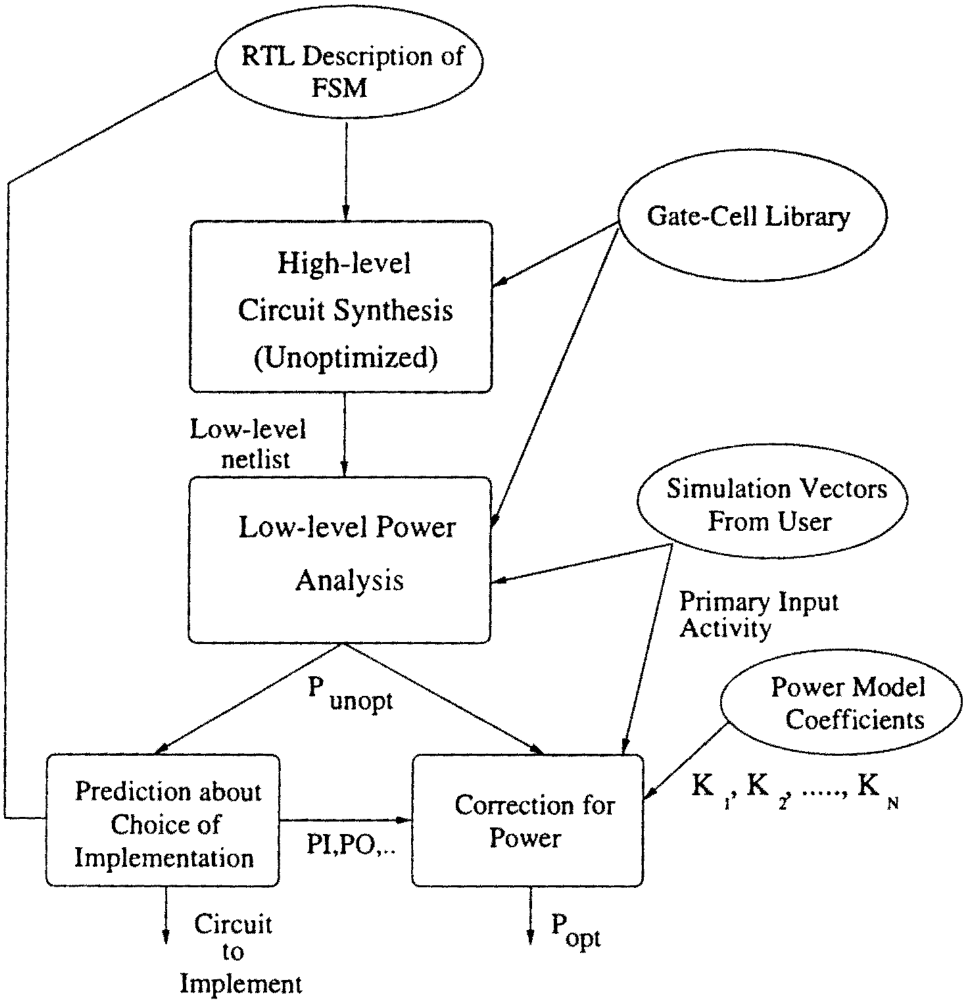


Figure 3.4: Power Analysis With New Power Model

3.4.2 Low-level Power Analysis of the Unoptimized Circuit

The low-level circuit description resulting from the high-level circuit synthesis step becomes input to the power estimation tool. The other input to the power analysis tool is the input vectors from the user. There are two different choices available for the generation of input vectors. The user could specify just the input statistics of the finite state machine and then vectors could be *randomly* generated conforming to those statistics. Or, the vectors could come from a real-world application of the finite state machine. The later would be preferred wherever possible. Low-level power analysis is possible at both the gate-level and transistor-level. Gate-level power analysis would be preferred for speed considerations. There are two different options for gate-level power analysis. It could be either simulation-based, using a gate-level simulation tool like QUICK POWER, or it could be probabilistic or empirical. The method followed here must be same as the one used during the power characterization phase. The use of empirical or probabilistic techniques would require a power-characterized cell-library. Power-characterized means the cell-library has been characterized for cell power dissipation for a wide range of input activities. A power-characterized cell-library, along with a fast and efficient technique for propagating the input signal probabilities to the internal nodes of the synthesized circuit, would give the capability to calculate the power dissipated by each cell in the circuit. Addition of the power dissipated by each cell would then yield the average circuit power. Obviously, the latter method is faster than the simulation-based method.

3.4.3 Power-Ratio Extractor

This phase consists of extracting the value of the power-ratio for the FSM under analysis. The two inputs to this stage are the power model coefficients from the power characterization phase (K) and the top-level parameters of the FSM from the RTL description of the FSM(T). The equation PR=T.K yields the required power scaling factor.

3.4.4 Correction Factor for Power

The final step is to multiply the unoptimized power obtained from step 2 with the value of power-ratio obtained from step 3, to yield the estimated value of the power of the optimized circuit.

For making *qualitative* judgements about choice of finite state machine implementation from a set of broad alternatives, the designer needs to go through only up to the low-level power estimation (on an unoptimized circuit) step, in the power analysis phase described above. Judgements about which design alternative to pick for the final implementation follow from the unoptimized power values of the different design alternatives and from qualitative inferences made in the power characterization phase. This is shown with the *prediction about choice of implementation* box in Figure 3.4.

3.5 Chapter Summary

This chapter presented the motivation behind the new method for RTL power estimation of sequential circuits and the quantitative and qualitative aspects of the new method. *Quantitative* measures of the average power dissipated by an FSM are made by first doing a quick gate-level power-analysis of an unoptimized implementation of the FSM and then applying a correction factor to it, to yield the average power of the optimized implementation of the FSM. The correction factor for the power comes out of the power model. The power-model is built out of parameters directly available from the RTL description of the finite state machine and the coefficients of the model are estimated during the power characterization phase by studying finite state machine implementations of varying complexities. *Qualitative* judgements about choice of finite state machine implementation from a set of broad alternatives are made by first estimating the power of the unoptimized implementations of the different alternatives and then using the knowledge gained in the characterization phase about the qualitative performance of the optimizer in reducing the power of the unoptimized circuits. No power model is actually built for qualitative analysis with the new method.

The next chapter discusses the evaluation of the model proposed in this chapter.

Chapter 4

Power Model Evaluation

The previous chapter explained the new power model in detail and the power analysis process with the new model. This chapter presents experimental results with the new power model. First, the experimental setup used for evaluating the new power model will be presented and then the actual experimental results for both the qualitative and quantitative measurements with the new model will be presented. The power model was studied using a large set of randomly generated boolean circuits and several MCNC91 benchmark circuits. Note that throughout the experiments with all the benchmark circuits and randomly generated circuits, the model coefficients were evaluated, not for the power values of the circuits, but for the switched capacitance values of the circuits. This would not make any difference, because the clock period over which the power value is calculated is the same during both the model characterization and evaluation phases. In short, modeling the energy, instead of the power is carried out. Again, for the same reason, the words *power* and *switched capacitance* will be used interchangeably, throughout this chapter. Also, throughout this chapter, the words *synthesis tool* and *optimizer* are used interchangeably. Wherever needed, deviations made from the setup proposed in chapter 3 for the power characterization and power analysis phases will be marked and explained.

4.1 Experimental Setup

This section discusses all the details that went into performing the experiments. Specifically, the tools used for the experiments, the technology-library used for the synthesis, the custom/benchmark circuits studied and other supplementary programs used for several important functions will be discussed. Experimental setup is first described with respect to finding a quantitative measure for power. Then the setup for the qualitative aspect of power prediction will be described.

4.1.1 Power Characterization Implementation

The experimental setup for the model characterization phase is shown in Figure 4.1

4.1.1.1 Logic Synthesis with SIS

The synthesis of the finite state machine was done with SIS [Sis], a public-domain synthesis tool from UC-Berkeley. The FSM descriptions of all MCNC91 circuits are fed into the synthesis tool in the *kiss* format, a special format for specifying finite state machines. This format has the advantage of specifying symbolic codes for the states in the state machine, thus allowing the synthesis tool to apply state minimization techniques and to choose optimal state assignment for the state machine. Other formats like BLIF come with state assignment done for the state machine and hence do not allow the synthesis tool to do state minimization and state assignment for the state machine. The kiss format, thus has the advantage of conveying the same information that any state machine description written in a Hardware Description Language (Verilog HDL or VHDL) would convey. An example finite state machine in the kiss format is shown in Appendix C. The cell-library used in synthesis was built out of standard gates (inverter, nand, nor, dff), developed locally in PSU using the MOSIS 1.2μ SCMOS technology. For the randomly generated circuits, the standard espresso format was used.

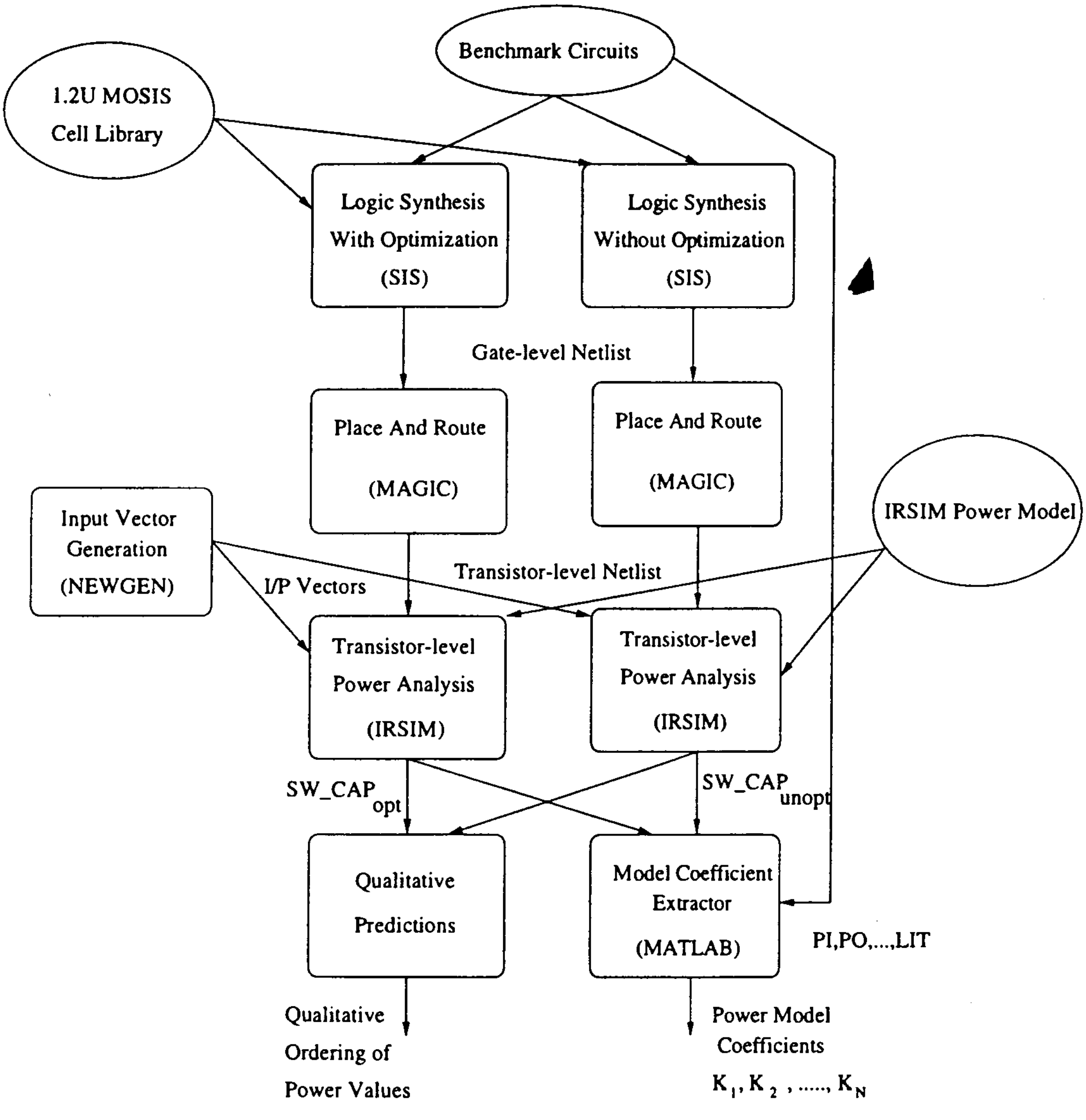

Benchmark Circuits
1.2U MOSIS Cell Library
Logic Synthesis With Optimization (SIS)
Logic Synthesis Without Optimization (SIS)
Gate-level Netlist
Place And Route (MAGIC)
Place And Route (MAGIC)
Input Vector Generation (NEWGEN)
I/P Vectors
Transistor-level Netlist
IRSIM Power Model
Transistor-level Power Analysis (IRSIM)
Transistor-level Power Analysis (IRSIM)
SW_CAP opt
SW_CAP unopt
Qualitative Predictions
Model Coefficient Extractor (MATLAB)
PI,PO,...,LIT
Qualitative Ordering of Power Values
Power Model Coefficients
K1, K2, ..., KN

Figure 4.1: Implementation of Power Model Characterization

Low-effort or unoptimized synthesis was done in SIS with the following steps:

- State assignment for the state machine
- Mapping the state assigned state machine to the cell-library for minimum area. The mapping process in SIS is done against at least one design constraint. Minimum area mapping was done for this step, because it is the fastest.

Synthesis with optimization was done as follows:

- State minimization of the finite state machine
- State assignment of the finite state machine
- Technology independent logic minimization
- Mapping of the logic-minimized circuit to the gate-cell library, for minimum delay. The choice of synthesis for minimum delay was made to account for the worst-case power dissipation of the circuit.

For the randomly generated boolean circuits, the synthesis steps are the same as for the benchmark set, but without any state minimization and assignment.

4.1.1.2 Place and Route with MAGIC

The synthesized circuit from SIS was placed and routed with MAGIC [Magic]. Supplementary scripts automate the process of converting the SIS gate-netlist description to a MAGIC layout description and then invoking the MAGIC channel-router to route the necessary connections. These scripts run in batch mode and hence are quite fast. The output from this stage is the transistor-level netlist. This step is required because throughout the experiments, all power analysis are done at the transistor-level, as against the gate-level. This is because of the ease of integration of IRSIM (a switch-level circuit simulator) with the other tools used in the design flow and the lack of a working tool-set which employs gate-level probabilistic or empirical techniques for power estimation.

4.1.1.3 Input Vector Generation with NEWGEN

The C program, NEWGEN, generates vectors for the power analysis, to specified input statistics. The inputs to the program are the number of pins in the circuit, the pin transition probability values, and the number of vectors to generate. During the characterization phase, UWN input activity was used, as this is very commonly used and found in literature for reporting results and a good starting point for extending the analysis to a complete range of input statistics. The number of vectors needed for reliable power estimates is calculated as per Eqn. 2.1 in chapter 2. [Saxena91] reports maximum numbers of 2000 vectors and average numbers of 500 vectors to achieve convergence as per Eqn. 2.1, for a wide variety of benchmark circuits, at 5% estimation error and 95% confidence level. Based on this observation, the number of vectors was fixed at 4000.

4.1.1.4 Transistor-level Power Analysis with IRSIM

The placed and routed circuit was analyzed for switched capacitance using IRSIM [Horowitz89], a switch-level circuit simulator from UC-Berkeley. Supplementary scripts generate the necessary control files for the IRSIM runs. The vectors for the IRSIM runs come from the NEWGEN program. The clock period for simulation was fixed to be 2000ns, a period long enough for the circuit to settle down fully, and hence account for all activity within the circuit for any applied input vector. The initial state of the state machine was chosen to be all zeros for the state bits, a rather arbitrary choice. This choice of initial state for the state machine was experimentally verified to not affect the final value of switched capacitance significantly, since the number of vectors applied (4000) is large enough to make the machine independent of the initial state. Errors between runs with four different initial states for different circuits were found to be within 2%. Near closed sets is not a problem with the set of benchmark circuits (MCNC91) and the primary input statistics (UWN) chosen for the experiments. This follows directly from the observations in [Roy96] about the MCNC91 benchmark FSM circuits.

4.1.1.5 Model Coefficient Extraction with MATLAB

Power model coefficients were extracted using MATLAB's [Matlab] least square solution facility. A small awk program was written which extracts the top-level parameters (SOP, Literals, etc..) from the pla-format description of the finite state machine and the switched capacitance values from the IRSIM output files, to form the matrices of Eqn. 3.3.

4.1.1.6 FSMs and Randomly Generated Circuits

41 MCNC91 FSM circuits were used for characterization and evaluation of the power model. These are small to medium-sized circuits available in the kiss format. The list of the benchmark circuits used and their details are given in the next section. Apart from these benchmark circuits, a big set - 249 circuits - of randomly generated boolean circuits were studied. These circuits were generated with a program called RANDGEN [Gilliam91]. These circuits were used only in building and evaluating the power model for *quantitative* prediction of average power. A wrapper script built around the RANDGEN program generates random boolean circuits with a specified number of inputs and outputs (less than 17), and with a given percentage ON-set of the boolean space of each output function. This program outputs the circuit in the espresso format, which is directly readable into SIS. Details about the random circuits used in the experiments and their availability are given in Appendix D.

4.1.1.7 Qualitative Predictions

The experimental setup for qualitative analysis is the same as described so far except that the model coefficient extraction phase is absent. Experiments were conducted on 32 MCNC91 benchmark circuits to study the qualitative power performance of the optimizer, which could be useful in making decisions about choice of finite state machine implementation from a set of broad alternatives. The idea here is to see if, given two choices for the implementation of a finite state machine, any qualitative predictions can be made about the power of the optimized implementations of these two circuits, by just looking at the power of the unoptimized implementations of these circuits. For example, if the unoptimized average power of implementation1 is greater than the unoptimized average power of implementation2, then is the optimized average power of implementation1, greater than that of implementation2 also? Henceforth, we will call this observation a *qualitative ordering* of the average power or switched capacitance. If it were true that the optimizer consistently preserved the *qualitative ordering* of the power values of the circuits, then the designer needs to estimate only the power of the unoptimized implementations of the design alternatives under consideration and pick the one which has a lower value of power for the final implementation. The alternatives chosen for study in this thesis are one-hot encoded finite state machine and horizontal encoded finite state machine. Results of qualitative analysis are presented later in this chapter.

4.1.2 Power Model Performance Evaluation Setup

The power model (for quantitative measurements of power) was evaluated as shown in Figure 4.2. The RTL description of the finite state machine under study is taken through two runs of SIS, using the same gate library that was used during the power characterization phase. One run does not use any logic optimization and the other run optimizes the circuit for delay. The internal SIS steps during each of these runs are exactly the same as in the characterization phase. The gate-level netlists obtained from the two runs of SIS are placed and routed with the layout-tool, MAGIC, to yield a transistor-level netlist of the circuit. The transistor-level netlist is then simulated with IRSIM, to yield the power of the optimized and unoptimized implementations of the circuit. Power analysis is done at the transistor-level rather than at the gate-level, for reasons mentioned earlier. Simulation vectors come from the input vector generation program called, NEWGEN. These are a random set of vectors, generated to a given input statistic. The power-model from the power

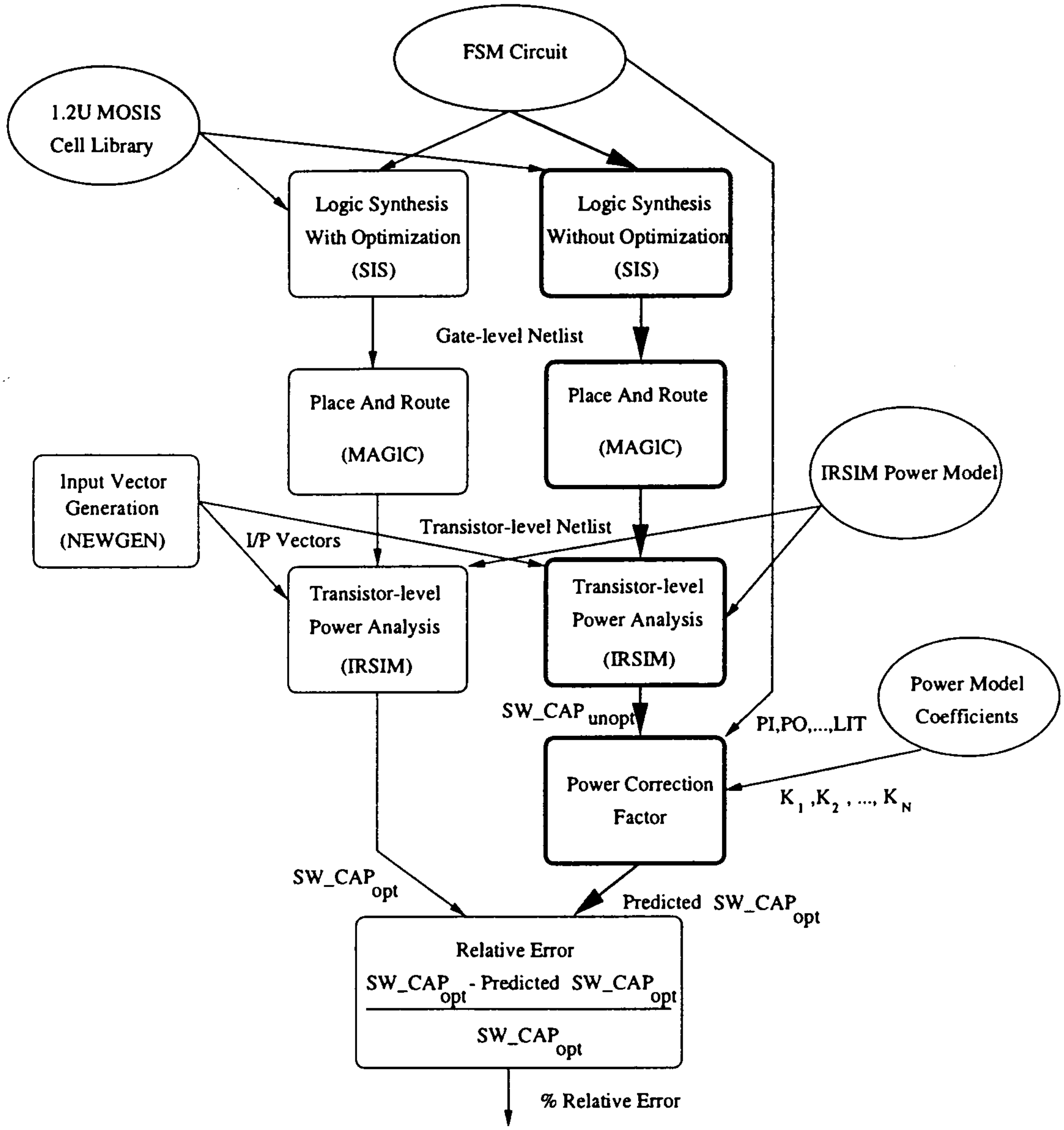


Figure 4.2: Power Model Performance Evaluation

characterization phase, along with the RTL description of the finite state machine are then used to apply a correction factor to the unoptimized circuit's power to give the predicted value for the optimized circuit's power. The predicted value of the optimized circuit's power and the IRSIM-measured power of the optimized circuit are compared to give the relative error of estimation. The flow highlighted with dark arrows/boxes in Figure 4.2, gives the implementation of the power analysis phase with the new power model. This is typical of the path taken by a designer on the field to predict the power of a given finite state machine.

4.2 Results

This section presents the results of the experiments done for model evaluation. This section is divided into two parts. The first part contains the experiment results for quantitative power analysis and the second section contains the experiment results for qualitative power analysis. Experiments for quantitative power analysis were done with a set of 41 MCNC91 benchmark circuits and a big set of 249 randomly generated boolean circuits. The reasons for using this twin sets of circuits will become clear at the end of the next section. Experiments for qualitative power analysis were performed with 32 MCNC91 circuits.

4.2.1 Quantitative Power Analysis

4.2.1.1 MCNC91 Benchmark Set Results

A total of 41 MCNC91 benchmark circuits were used for characterization and evaluation of the proposed power-model. These are small to medium sized circuits available in the kiss format. The functionality of these circuits is not known. Table 4.1 gives the list of the benchmark circuits and their characteristics. "Opt" column in the table gives the parameter values with optimized SIS run and the "Unopt" column gives the parameter values with unoptimized SIS run. For example, the "Unopt" column under "States" indicates the number of states in the unminimized

Circuit	PI	PO	States		LIT		SOP	DC	DCLIT	SW_CAP	(pf)
			Unopt	Opt	Unopt	Opt				Opt	Unopt
bbara	4	2	10	7	50	126	34	12	24	16647	22409
bbsse	7	7	16	13	130	326	61	41	157	57545	56329
bbtas	2	2	6	6	24	38	15	5	10	14375	12847
beecount	3	4	7	4	23	87	26	35	168	11907	21380
cse	7	7	16	16	205	653	119	74	483	55568	90146
dk14	3	5	7	7	93	362	84	8	24	51653	83800
dk15	3	5	4	4	78	194	45	0	0	39753	44961
dk16	2	3	27	27	237	626	120	16	56	143555	150761
dk17	2	3	8	8	61	137	38	0	0	36813	36469
dk27	1	2	7	7	24	37	15	5	15	23666	20126
dk512	1	3	15	15	57	99	28	7	28	38707	38369
ex1	9	19	20	18	227	1132	200	288	1224	92778	162242
ex2	2	2	19	14	76	270	64	90	296	50148	76202
ex3	2	2	10	5	25	85	24	32	82	22558	32225
ex4	6	9	14	14	72	114	33	26	52	42778	39279
ex5	2	2	9	4	14	89	28	44	140	10905	24205
ex6	5	8	8	8	90	356	84	11	55	45324	75938
ex7	2	2	10	4	23	111	39	46	140	22062	36970
keyb	7	2	19	19	209	593	114	172	798	62719	100809
kirkman	12	6	17	17	213	665	128	120	403	89019	114828
lion	2	1	4	4	14	19	9	4	16	10317	8481
lion9	2	1	9	4	16	23	12	45	145	9356	15095
mark1	5	16	16	12	70	269	72	67	133	35512	54605
mc	3	5	4	4	26	67	22	0	0	12832	13720
opus	5	6	11	9	81	307	67	20	60	27327	41352
pma	8	8	24	24	215	567	132	520	2665	124365	113701
s1	8	6	20	20	197	1047	193	44	132	95639	181269
s208	11	2	18	18	114	187	45	56	168	28966	30440
s27	4	1	6	5	35	63	20	4	8	21118	19533
s386	7	7	13	13	114	334	60	22	66	43474	61425
s420	19	2	18	18	90	165	36	56	168	25674	27903
s510	19	7	47	47	304	1009	207	91	416	114741	113447
s820	18	19	25	24	322	1015	171	96	360	114225	213296
s832	18	19	25	24	359	850	130	96	360	108325	170802
sand	11	9	32	32	577	2677	374	130	974	214890	358603
shiftreg	1	1	8	8	9	18	8	0	0	15596	13410
sse	7	7	16	13	130	326	61	41	157	56743	55909
tav	4	4	4	4	28	35	11	0	0	21635	18543
tma	7	6	20	18	174	291	84	396	1529	82858	93904
train11	2	1	11	4	11	40	18	58	208	9338	20308
train4	2	1	4	4	17	25	9	6	19	11628	10544

Table 4.1: Characteristics of MCNC91 Benchmark FSM Circuits

(unoptimized) state machine and the "Opt" column indicates the number of states in the state-minimized state machine (optimized). The circuits considered cover a range of primary inputs from 1 to 19 and primary outputs from 1 to 19. The number of states in the state machine range from 4 to 47. The switched capacitance values in the table are the values from the IRSIM runs. In the absence of vectors from a typical application of these state machines, randomly generated vectors were used during simulation with UWN statistics. A total of 4000 vectors were used for the runs with a clock period of 2000ns. Refer back to section 4.1.1.3 for details on reliable power estimates at 4000 vectors. Table 4.2 gives the run times (measured as CPU time) obtained for the high-level logic synthesis of some of the benchmark circuits, with and without optimization. There run times were measured on a Sun SPARC-5 station with 64MB memory.

Before proceeding to estimate the power model coefficients, the function in Eqn. 3.1 has to be known. To make a good guess at what the function might look like, any hidden regular surfaces that might be there in the N+1 dimensional space (where N is the number of top-level variables) of the switched capacitance ratio function must be revealed. These regular surfaces could then be captured mathematically by an appropriate equation. These plots could also tell us about the non-dependency of the switched capacitance ratio on any of the top-level variables, thus serving as a method for variable reduction in the model. To this end, 3-D surface plots of pairs of top-level variables against the switched capacitance ratio were done using the gnuplot utility. A few of these plots are shown in the Figures 4.3 to 4.6. A more complete set of plots is given in Appendix A.

Circuit	PI	PO	States	CPU Time Opt.	(sec.) Unopt.	Speedup
dk27	1	2	7	4	1	4
ex7	2	2	10	4	1	4
s27	4	1	6	7	1	7
pma	8	8	24	126	5	25
s386	7	7	13	32	3	11
s1	8	6	20	90	7	13
s510	19	7	47	145	8	18
sse	7	7	16	45	3	15
s208	11	2	18	26	2	13
sand	11	9	32	838	16	52
s420	19	2	18	19	2	10
s820	18	19	25	195	8	24
s832	18	19	25	165	7	24

Table 4.2: Run Times for High-level Logic Synthesis of 13 MCNC91 FSM Circuits Using SIS

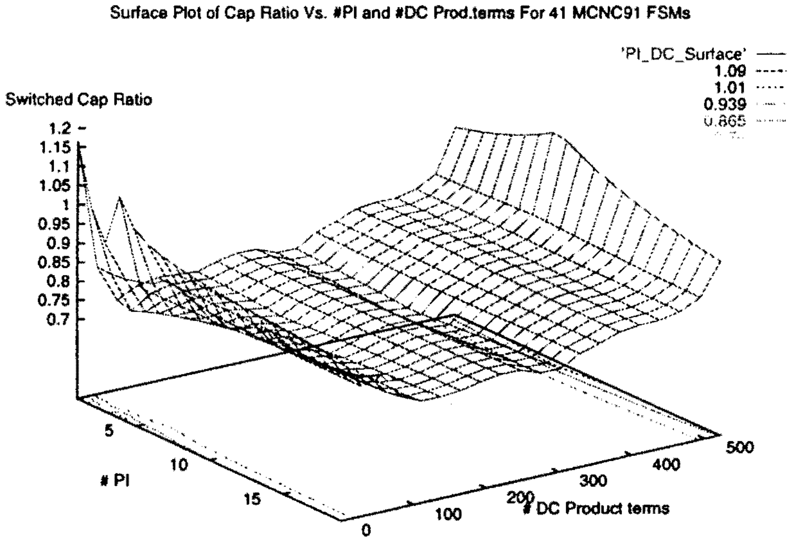


Figure 4.3: Surface Plot of PI and DC for MCNC91 Circuits

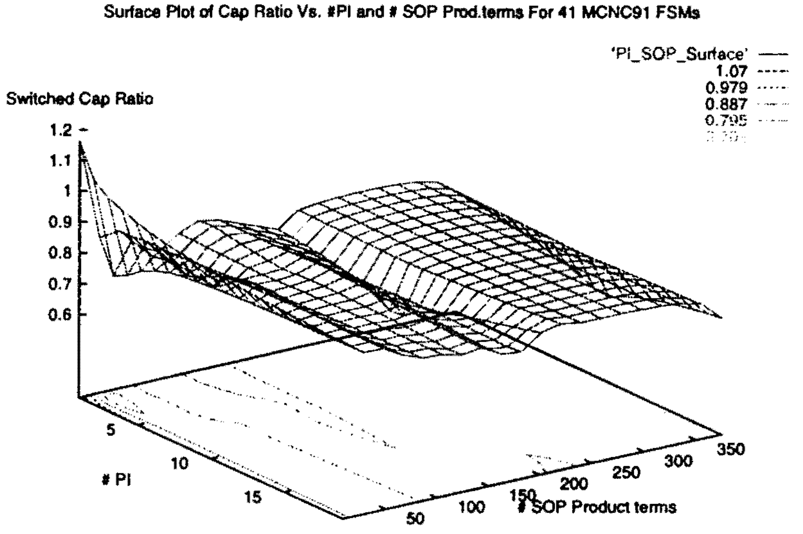


Figure 4.4: Surface Plot of PI and SOP for MCNC91 Circuits

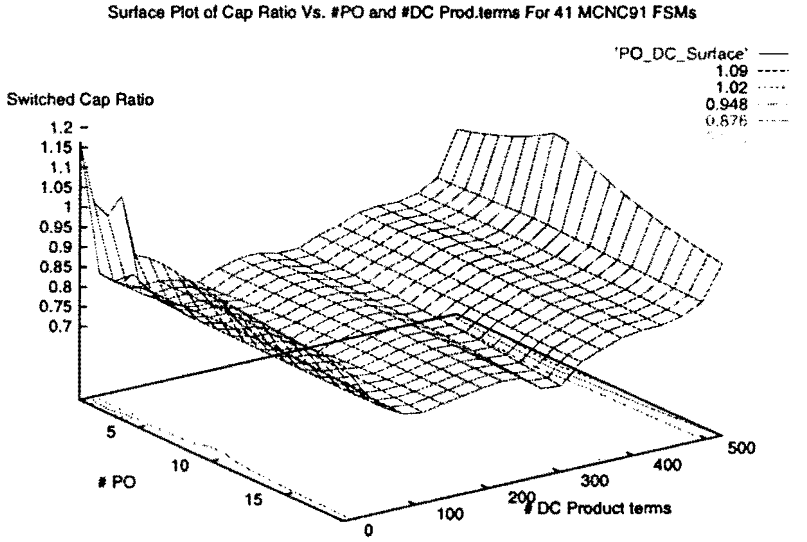


Figure 4.5: Surface Plot of PO and DC for MCNC91 Circuits

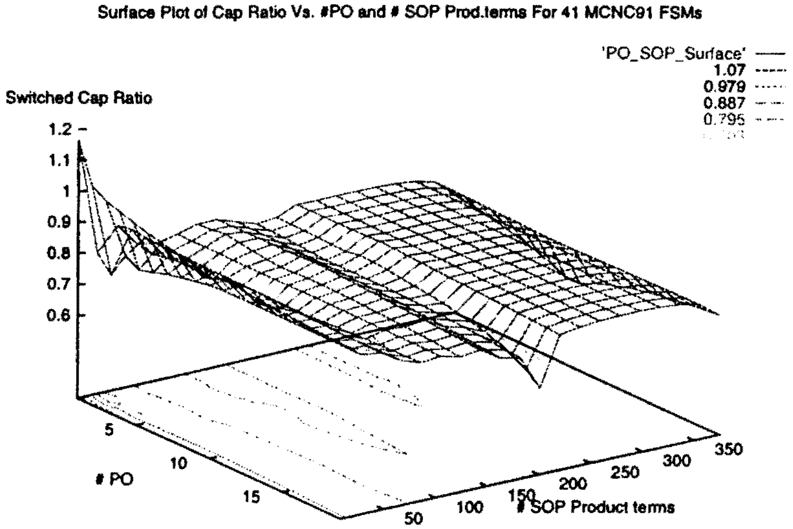


Figure 4.6: Surface Plot of PO and SOP for MCNC91 Circuits

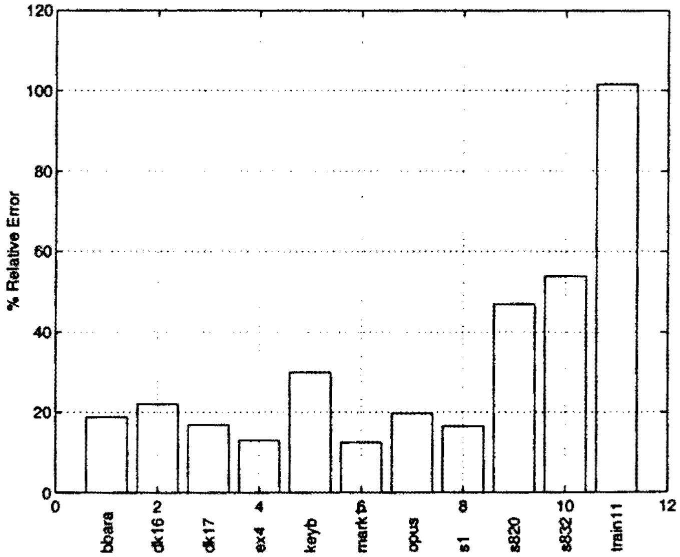


Figure 4.7: Relative Error in Switched Capacitance Predicted by the Power Model for 11 MCNC91 Circuits

Statistic	Value
Mean Relative Error	32%
Standard Deviation of Relative Error	38%
Maximum Relative Error	102%
Coefficients (K_1-K_8)	902e-03 to -6.6e-03

Table 4.3: Results of Least Squares Solution in Predicting Switched Capacitance Ratio for 11 MCNC91 Circuits

The surfaces in these plots reveal no regular pattern, except for the flat portions in the surfaces which represent a lack of detail in the MCNC91 data set. This implies that there is not a mathematical expression that could be made out of the top-level variables to fit these surfaces, to yield results at acceptable error levels. In fact, numerical values for the least-squares fit of the data in Table 4.1 to the equation in 3.2, show a maximum error of 102%. MATLAB's direct matrix solution capability was used for the least-squares solution. These results are summarized in Table 4.3. A total of 30 circuits were used in the characterization phase and a different set of 11 circuits were used in the evaluation phase, and their relative errors are plotted in Figure 4.7. The circuit train11 has the maximum error at 102%. This could be explained by the large reduction in the number of states in the state machine (from 11 to 4) during optimization. This being a small circuit (2 PI and 1 PO), this reduction in the number of states - hence flip-flops - makes a big difference in the final value of switched capacitance and hence relative error. The same is not the case with s820, s832 circuits which show errors close to 56%, even when there is no reduction in the number of states. This clearly points to the inability of the linear model to adequately capture the reduction in switched capacitance. The percentage of circuits with relative errors greater than 30% is 27%. From a research point of view, it needs to be verified that the high values of relative error are not because of the lack of detail in these surfaces, or in other words, not because of the scattered set of datapoints of the benchmark circuits chosen. This issue is addressed in the next section where we study a large set of randomly-generated circuits, which are generated with controlled values for different parameters.

4.2.1.2 Random-Circuit-Set Results

Randomly generated circuits were used to study the effect of the optimizer on the power of an unoptimized boolean circuit, for the following reasons. We can have more datapoints for analysis and hence capture more details about the power-ratio surface. Using randomly generated circuits would let us have more control over the values of some of the top-level parameters during the experiments. These parameters are the number of primary outputs and the number of primary inputs (and to a certain extent the literal count of the sum-of-product terms of the output functions). Keeping one of these controllable parameters a constant during the circuit generation and varying the last parameter would let us see the effects of the trends in the varying parameter on the power-ratio. As an example of control we have in generating circuits for the experiments, the MCNC91 set has only two circuits with 8 primary inputs while the random set has 25 circuits with 8 primary inputs. These 25 random circuits have a range of primary outputs from 3 to 16 and literal count in the range from 17 to 283. The primary output count in the two MCNC91 circuits are 6 and 8 and the literal count values are 567 and 1047. So the random set has more variety of circuits for a given primary input, and hence would make the surface plots involving the primary input count as one of the variables, more accurate.

The random circuits were generated by the program RANDGEN. These random circuits have the following characteristics. The number of primary inputs ranges from 5 to 15 and the primary outputs from 3 to 16. These ranges are about the same as the MCNC91 benchmark set. The literal count for the random circuits ranges from 16 to 923. MCNC91 circuits had literal counts from 18 to 2677, but much more dispersed in that range than the random set. One difference between the MCNC91 benchmark set and the randomly-generated set is that the random-set of circuits are purely combinational, as against the presence of states in the MCNC91 set. But they implicitly include the notion of the number of states by including the state bit lines among the primary inputs to the circuit. This would entail that rather than explicitly using the number of states as a top-level variable in the power-ratio function of Eqn. 3.1, the number of state-bits is used as a top-level variable. The implication of this deficiency on the design flow is that there are no state minimization and state assignment steps in the logic synthesis phase of the power characterization process. This deficiency in the random circuits is not of any consequence for the purpose of the experiments, because they are only being used to reveal as much information about the optimizer as is possible, in its behavior towards affecting the power dissipation of different boolean functions, which should be the same whether it is from a finite state machine or a combinational logic block.

More details about the random circuits and their availability are available in Appendix D. For all the IRSIM runs with the random circuits, UWN activity was used for the inputs. A total of 4000 vectors were used, each with a simulation period of 2000ns. The 3-D surface plots for switched capacitance ratio against a pair of top-level variables are shown in Figures 4.8 to 4.11, and more are available in Appendix B. One would expect to see surfaces like these for the benchmark set also, if enough datapoints are available. These surfaces again reveal no regular pattern for us to build any function out of the top-level variables for the switched capacitance ratio. These plots are more detailed than those of the MCNC91 set, but are no different in revealing any regular surfaces in the switched-capacitance ratio function. Table 4.4 gives the results of the least-squares solution for a linear-function of the top-level variables to the switched capacitance ratio, for these random circuits. A total of 198 circuits were used for evaluating the model coefficients. Model coefficients were tested with 51 circuits. The relative errors for these evaluation circuits are shown in Figure 4.12. The average relative error is 15%. The maximum relative error is 83%. 50 of the 51 circuits (2%) have errors less than or equal to 50%. Only 12% of the circuits have relative errors greater than 30%. These results are improved over

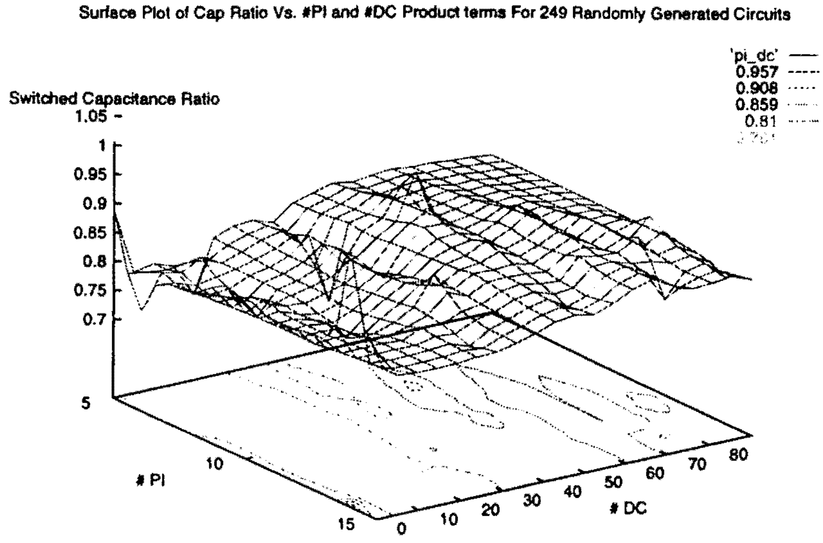


Figure 4.8: Surface Plot of PI and DC for Random Boolean Circuits

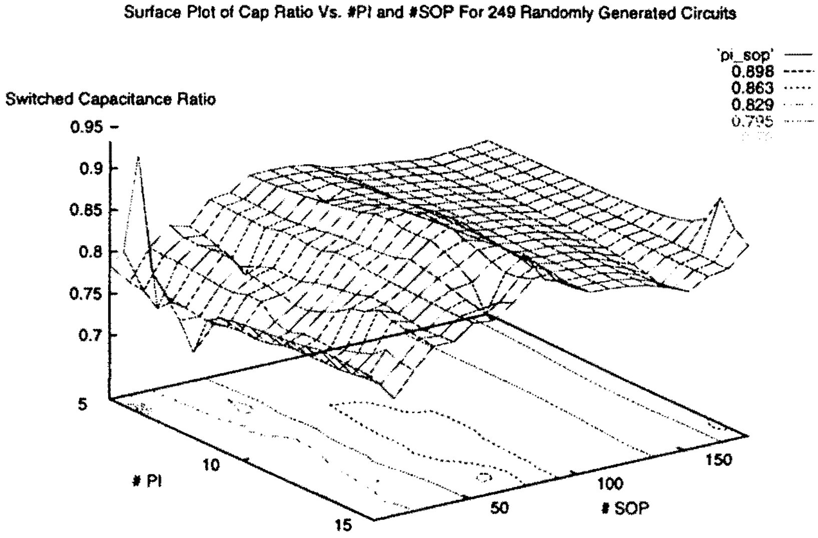


Figure 4.9: Surface Plot of PI and SOP for Random Boolean Circuits

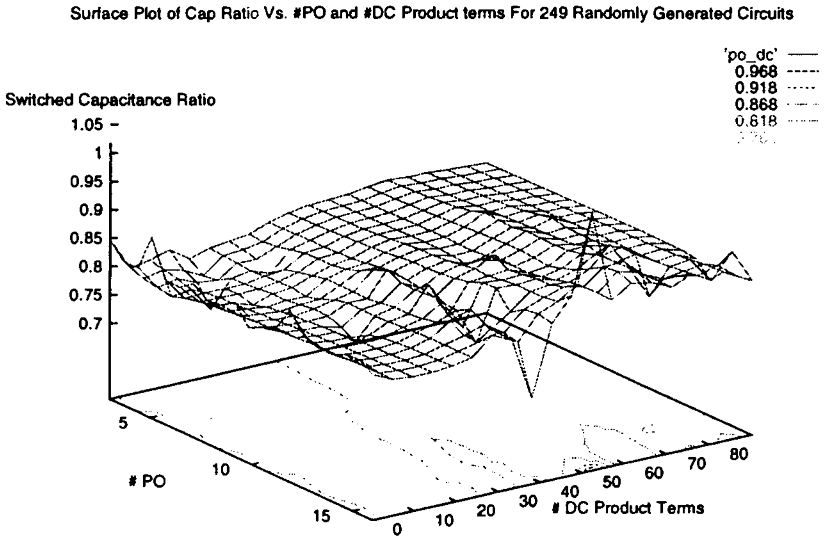


Figure 4.10: Surface Plot of PO and DC for Random Boolean Circuits

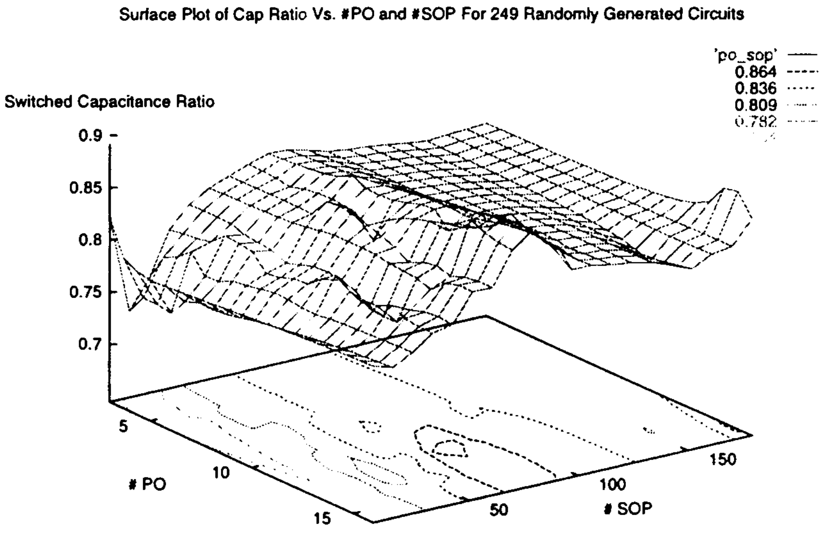


Figure 4.11: Surface Plot of PO and SOP for Random Boolean Circuits

Statistic	Value
Mean Relative Error	15%
Standard Deviation in Relative Error	21%
Maximum Relative Error	83%
Coefficients (K_1-K_8)	-1.6e-03 to 538e-03

Table 4.4: Results Of Least Squares Solution in Predicting Switched Capacitance Ratio for 51 Randomly Generated Circuits

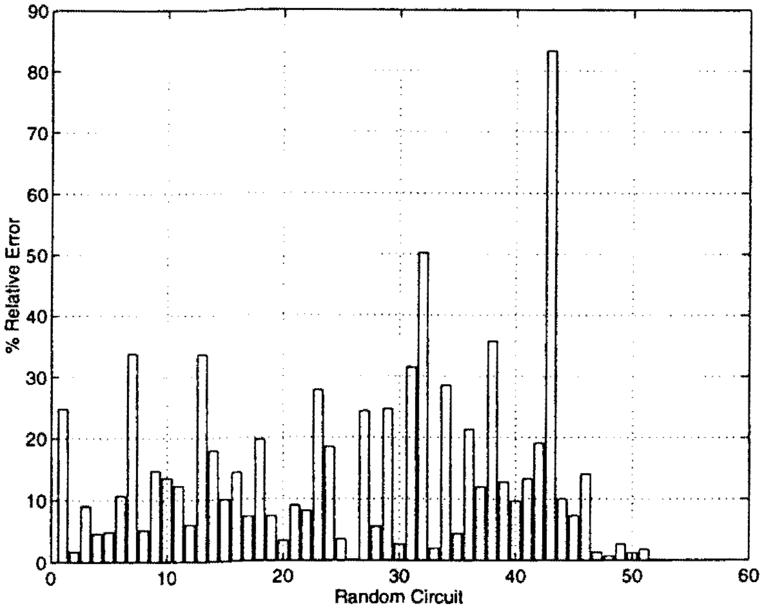


Figure 4.12: Relative Error in Switched Capacitance Predicted by the Power Model for 51 Randomly Generated Boolean Circuits

the MCNC91 benchmark set results.

The results discussed so far for the randomly generated circuits and the MCNC91 circuits are in-line with the existing RTL techniques for FSM power estimation. [Nemani97] reports a maximum relative error of more than 100% (the report of results in this paper is not very clear to provide an exact number for the relative error) and [Landman96] reports a maximum relative error of 29%, but for just two circuits. The method explored in this thesis shows a maximum relative error of 102%.

4.2.1.3 Statistical Confirmation of Independence

The results of independence between the top-level variables and the switched capacitance ratio, shown in the previous section for both the MCNC91 benchmark set and the randomly generated circuit set, were statistically confirmed through the *Spearman-R rank correlation test* [Gibbons92]. This test has the ability to detect correlations between two series of data, both positive and negative, even though the exact nature of the correlation itself is not known through this test, other than being either positive or negative. Since this is a statistical test, it is always done to a confidence level, meaning, with a specified accuracy with which the presence of a correlation can be detected. These tests were done by carefully selecting a few top-level variables and then testing them for correlation with the switched capacitance ratio. The various variables chosen for the Spearman-R test and the results of this test are listed in Table 4.5 and are described next. The naming convention followed in Table 4.5 is as follows:

1. Ratio of two variables is indicated with a "per" term in the name. For example, lit_per_sop indicates the ratio (LIT/SOP).
2. A product term between two variables is indicated with a "_" between the names. For example, the variable pi_sop indicates the product (PI.SOP).
3. DNR (Do Not Reject H0) in the table indicates that the presence of correlation in the data set is not detectable.

Literals(lit) and sum-of-product(sop) terms are directly indicative of the complexity of the boolean function. "pi_sop" term, which is the product of the number of primary inputs and number of sum-of-product terms, is indicative of the complexity of the input ON-plane ('1' plane) of the circuit i.e., the amount of decoding that needs to be done at the input of the circuit. Similarly, the "po_sop" term indicates the output ON-plane complexity of the circuit. In the same fashion, "pi_dc" and "po_dc" are indicative of the input and output complexities of the don't-care-plane of the boolean functions. "lit_per_sop", the ratio of number of literals and the sum-of-product terms, is indicative of the size of the ON-set of an output function. The higher the value of this ratio, the lesser the coverage of the ON-set of the function and fewer are the 1s in the boolean space.

The Spearman-R test tests for independence between two series of observations collected in pairs, to a certain level of significance. Its a distribution-free test, meaning that the test does not depend on the distribution of the underlying population of the observations. The test starts with formulating the null hypothesis, which is "no-correlation between the two sets of observations". The test then sets out to disprove the null hypothesis, at a certain level of significance. The level of significance is the probability of rejecting the null-hypothesis (i.e. no-correlation) when its true, and this is the error level tolerable while doing the test. The test statistic for this test is

$$R = \frac{n(n^2-1) - 6\sum_{i=1}^{n} d^2}{n(n^2-1)}$$

where, n is the number of observations, d is the difference in the ranks of each observation pair, when the pairs are arranged in the order of increasing magnitude in each series.

If the value of R lies in what is called the critical-region (explained next), then we reject the null-hypothesis and conclude that the two streams of observations are correlated. If the value of R lies outside the critical-region, then we do not reject the null-hypothesis. Critical-region is the region where the value of R> $R_{critical}$ or $R < -R_{critical}$. The value of $R_{critical}$ at a significance level of 0.05 is 0.336. The

Variable	Z	R	Do-Not Reject /Reject H0
lit	1.26909	-0.217647	DNR
sop	1.53369	0.263025	DNR
pi_sop	1.18579	0.203361	DNR
po_sop	1.11229	0.190756	DNR
lit_per_sop	0.390363	0.0669468	DNR
pi_dc	-0.316864	-0.0543417	DNR
po_dc	-0.409963	-0.0703081	DNR

Table 4.5: Results of Spearman Rank Correlation Test For Test of Correlation Between Top-level Variables and Switched Capacitance Ratio

test procedure is given below:

Null Hypothesis : H0 = The two pairs of observations are uncorrelated

Alternate Hypothesis : H1 = The two pairs of observations are correlated

```
If $R \geq 0.336$ or $R \leq -0.336$
    Reject H0
else
    Do Not Reject H0
```

A total of 35 circuits were randomly picked for the test from the set of 249 random circuits. As shown in Table 4.5, all the tests came out not detecting any correlation between the switched capacitance ratio and the top-level variable under consideration. Though no correlation is detectable between the switched capacitance ratio and the *top-level variables as defined*, a more sophisticated RTL methodology which possibly involves more computation than is done in this thesis, could turn out better results. This could be an area for future research.

4.2.2 Qualitative Power Analysis

In the absence of discernible patterns in the surface plots for switched capacitance ratio for both the MCNC91 circuits and the large set of randomly generated circuits, and a statistical confirmation of the inference of no correlation between the top-level variables and the switched capacitance ratio, a good numerical prediction of the power of a finite state machine, looking at just the RTL variables from the finite state machine description, is very difficult. This, in fact, is true for power prediction from the RTL description of a combinational logic block also, because the method explored so far is generic in its approach to analyzing the effect of the optimizer on the power of both finite state machines and combinational logic. An example of an RTL combinational block could be an unsynthesized micro-architectural block like a multiplexer, comparator, adder etc.. Though a quantitative prediction of the switched capacitance is not possible by this method, some general observations about the distribution of switched capacitance ratio and qualitative observations which could effect decisions about the choice of finite state machine implementation from a set of design alternatives can be made.

The distribution of the switched capacitance ratio values for the MCNC91 circuits and the randomly generated circuits is given in Figure 4.13 and Figure 4.14. Figure 4.14 shows a normal distribution (at 0.05 significance level) for the switched capacitance ratio with a mean of 0.822 (with error in estimate of ±0.019 at a sample size of 249) and standard deviation of 0.1547 (with error in estimate of approximately ±0.013 at a sample size of 249). The average value of switched-capacitance ratio for the MCNC91 circuits is 0.8219 with a standard deviation of 0.2336. Even though the average value of the switched capacitance ratio for the MCNC91 circuits is about the same as that for the random boolean circuits (this could be a coincidence), the optimizer's behavior towards power-ratio (or switched capacitance ratio) is not even qualitatively predictable for the benchmark circuits, from the perspective of power. This is because of the distribution of the switched capacitance ratio for these circuits as shown in Figure 4.13. But an analysis of a set

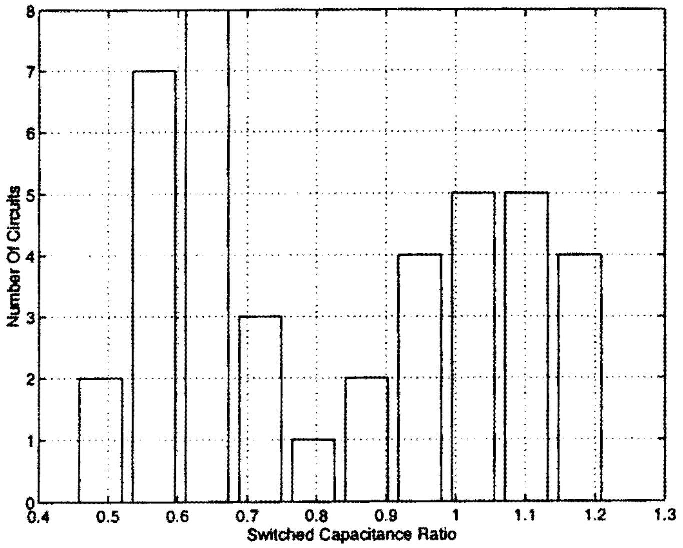


Figure 4.13: Distribution Of Switched Capacitance Ratio for 41 MCNC91 Benchmark Circuits

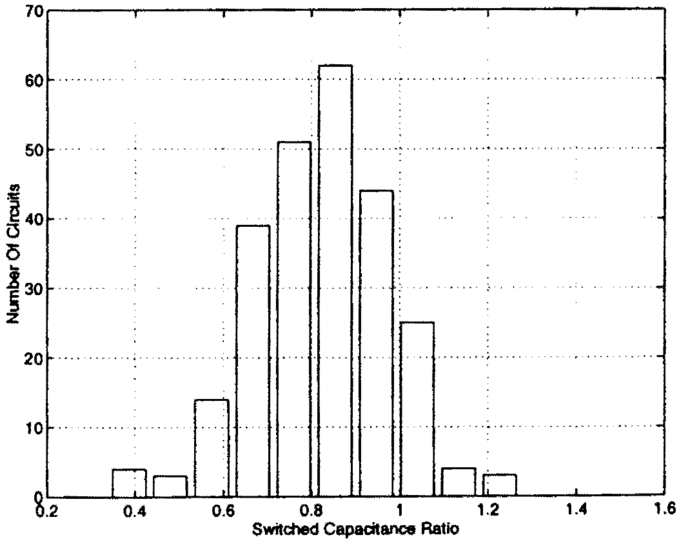


Figure 4.14: Distribution Of Switched Capacitance Ratio for 249 Randomly Generated Boolean Circuits

of 32 benchmark circuits reveals a qualitative trend about the optimizer's behavior towards preserving the qualitative ordering of power values. The idea, as described previously, is to see if given two choices for the implementation of a finite state machine, any qualitative predictions can be made about the power of the optimized implementations of these two circuits, by just looking at the power of the unoptimized implementations of these circuits. For example, is the optimizer consistent in preserving the qualitative ordering of the power values of two implementations? A prediction about the qualitative ordering of the optimized power values of two FSM implementations which are under design consideration, is purely statistical in nature and no RTL parameters are involved in the prediction. The user needs to just estimate the power of the unoptimized implementations of the two finite state machine alternatives under consideration. Since estimating the power of the unoptimized implementation of an FSM is much faster than that of the optimized implementation, any data that could point to this trend would let the user make informed decisions about design choices, *early* in the design cycle. Analysis for this trend was done with a set of 32 MCNC91 circuits, which were each implemented in two different ways. The two implementations for the finite state machines were chosen to be:

1. Horizontal encoding for the FSM
2. One-hot encoding for the FSM

These two methods of FSM implementation are about the most popular. In a horizontally encoded FSM, state encoding is done to minimize the number of flip-flops needed in the final implementation of the FSM. In one-hot encoding, no effort is made to minimize the number of flip-flops. Each state in the state machine is assigned a flip-flop. So there are as many flip-flops as there are states. As an example, if a state machine has 10 states, a horizontally encoded machine will consume 4 flip-flops, where as a one-hot encoded machine will consume 10 flip-flops. One-hot is generally preferred over horizontal encoding when speed of the circuit is a major design concern. From the perspective of the optimizer's power performance, these two implementations can lead to very different results. This is because the boolean functions for these two implementations can be quite different. In one-hot implementation, only one bit worth of information needs to be decoded to determine the current state of the state machine, irrespective of the number of states in the state machine, whereas, in horizontal encoding, the number of bits to be decoded is logarithmic in the number of states. This leads to different boolean equations for the output functions, even though they are functionally equivalent.

Of the 32 MCNC91 circuits which were each implemented in two different ways (one-hot and horizontal encodings), 28 preserved the qualitative ordering of the switched capacitance values, through the optimization step. Plots of the switched capacitance values for these 32 circuits are shown in Figures 4.15 and 4.16. The following observations follow from these graphs. No significant evidence of one implementation performing consistently better than the other (in power) was found. Neither are any reported in literature. The optimizer's effect on qualitative ordering is not predictable if the unoptimized switched capacitance values of the two implementations are close to each other. In all the four cases (dk14,dk15,ex4,pma) where the qualitative ordering was not preserved, the unoptimized switched capacitance values of the two implementations were within 10% of each other. The optimizer tends to go either way in these situations. So, a designer when making design choices has to be aware of this characteristic. One other situation when the designer might expect the qualitative ordering to be disturbed is when there is a potential for a large decrease in the number of states in the state machine on optimization. This might considerably reduce the power of the one-hot circuit in the optimized implementation, as is seen in the benchmarks train11 and beecount in Figures 4.15 and 4.16. In the benchmark circuits where the characteristics noted above are absent, the optimizer consistently preserved the qualitative ordering of the power values. So the choice of circuit to implement must involve consideration of all the factors noted above. Note that the plots in Figures 4.15 and 4.16 are for the particular optimizer (SIS) used in this thesis. They need to be redone for other optimizers and their behavior verified. Furthermore, more number of finite state machines could be implemented with horizontal and one-hot encodings, than is done here to build a better confidence about the optimizer's qualitative behavior. Limitations with the number of sequential circuits available in the benchmark set prevents us from doing that in this thesis.

In conclusion, given two choices of horizontal and one-hot encodings for implementation of a finite state machine, the one with a lower value of switched capacitance in the unoptimized implementation of the circuits, also ends up having a lower value of switched capacitance in the final optimized implementation of the circuits, except in situations explained above. Given this data and a designer's knowledge of the finite state machine under consideration, the designer can then make decisions about which design alternative to pick.

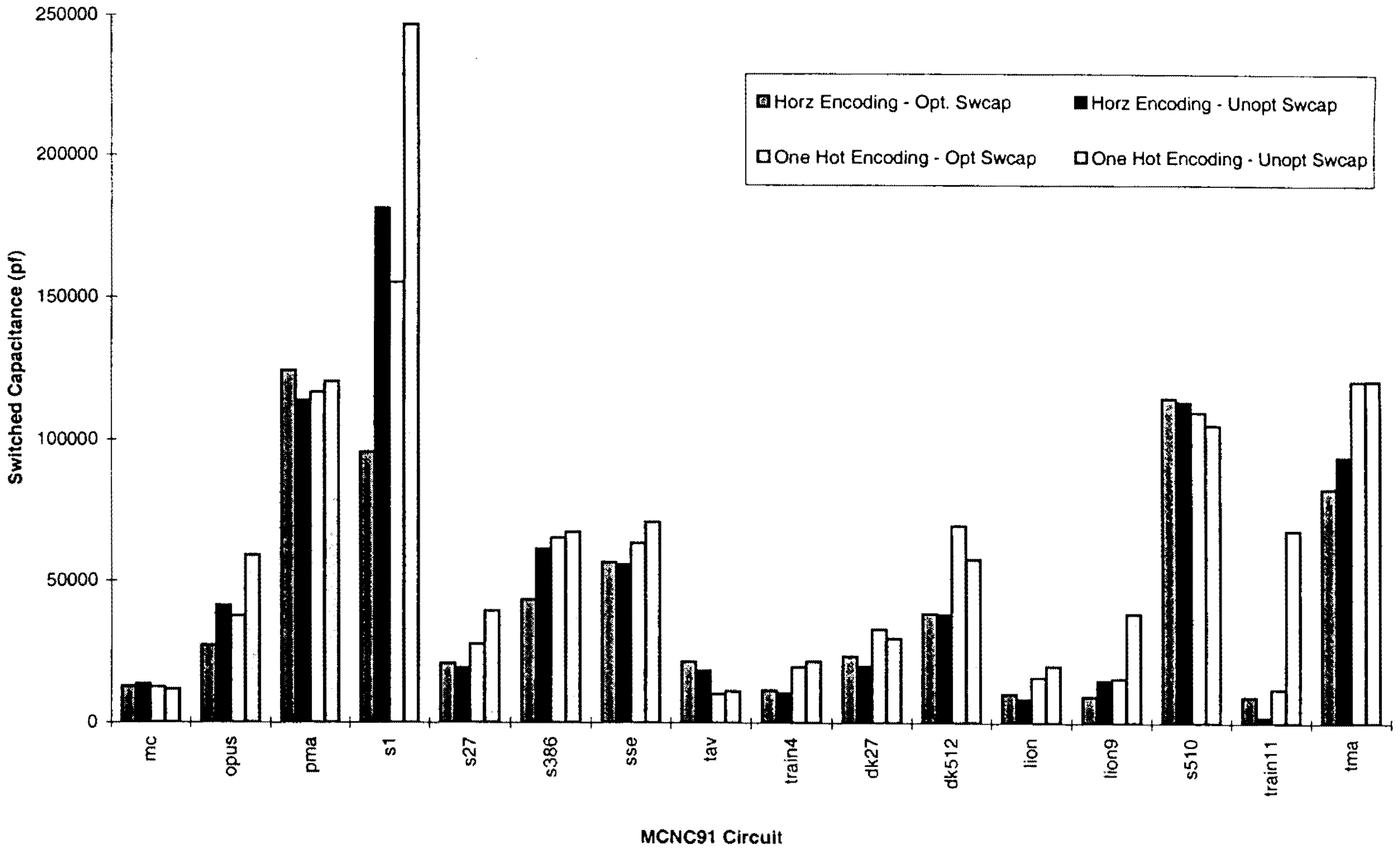


Figure 4.15: Switched Capacitance Values of 32 MCNC91 Benchmark Circuits Implemented with One-Hot and Horizontal Encodings - (a)

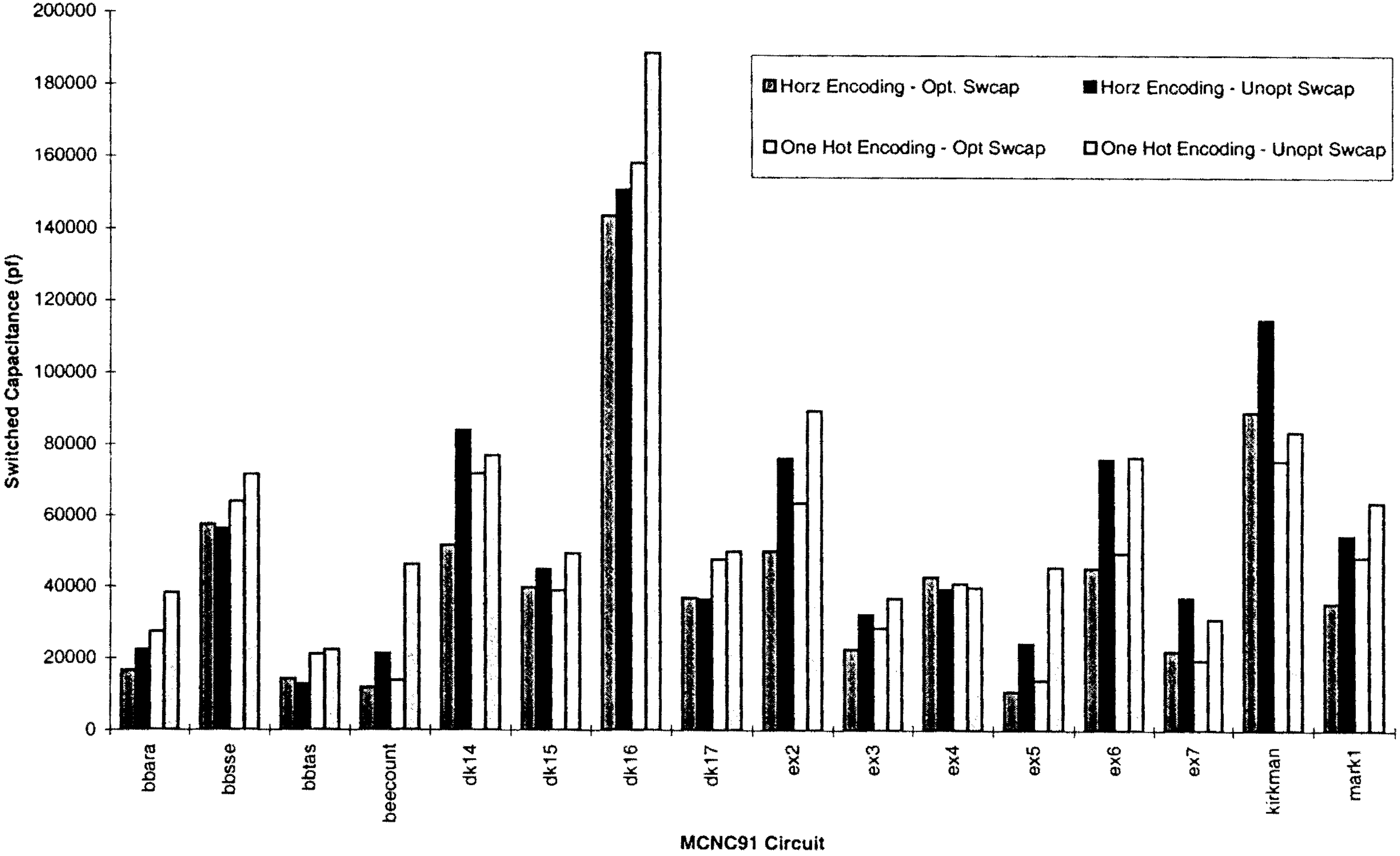


Figure 4.16: Switched Capacitance Values of 32 MCNC91 Benchamark Circuits Implemented with One-Hot and Horizontal Encodings - (b)

4.3 Chapter Summary

This chapter discussed in detail the results of the power model evaluation with a set of MCNC91 benchmark circuits and a large set of randomly generated circuits. Experiments with both these sets of circuits reveal that a numerical prediction of the worst case power dissipation of finite state machines with the new model, looking at just the RTL description of the state machine is very difficult. Maximum relative Errors are 102% for the MCNC91 set and 83% for the randomly generated set. But a qualitative prediction about the choice of finite state machine implementation from a set of available design alternatives is possible. Analysis of a set of benchmark circuits presents evidence to say that, given two choices of one-hot and horizontal encodings for finite state machine implementation, the one with a lower value of switched capacitance in the unoptimized implementation of the circuit, would also end up having a lower value of switched capacitance in the optimized implementation of the circuit, except in situations explained earlier. A designer can then utilize this observation along with his knowledge of the finite state machine design to make choices about the design alternatives.

Chapter 5

Future Work

This thesis proposed and evaluated a new method for RTL power estimation of sequential circuits. This chapter notes some possible future directions for exploration with the new model.

Relative errors reported in the previous chapter for the benchmark circuits and the randomly generated circuits show that the proportion of circuits with relative errors more than 30% is quite less. A sorting scheme that is capable of separating these circuits with high relative errors from the rest, based on the information available from the RTL parameters, would be useful. Such a sorting scheme could possibly lead into a technique where these two sets of circuits have different power models, which work well within their respective sets. This is one opening for future research. Analysis of the structure of finite state machines at the optimized and unoptimized stages and characterizing the behavior of the optimizer based on the structural changes incurred in a circuit through the optimization step, is another possibility. Care needs to be taken to not end-up in a scheme which is highly optimizer-specific. The logic synthesis process with optimization was done for minimum delay of the resulting circuit, to account for the worst case power dissipation of the circuit. Quantitative RTL measurements of this worst case power dissipation was not possible in this thesis. Another possible alternative in optimization is for minimum area of the resulting circuit. Minimum area optimization *usually* results in the worst case delay for the circuit and hence a lower-bound on power dissipation of a circuit. There is no apriori reason to believe that quantitative measurements of this lower bound on power would also be not possible with the new model. Exploration of a more sophisticated modeling technique, but one which would not slow down the RTL power estimation process too much, is another area open for research. This technique could possibly capture in more detail any global optimization techniques that most optimizers use in common, which was not achieved in this thesis. Research done previously which follows more sophisticated modeling techniques still show errors in excess of 100%. Though these techniques show acceptable errors for a number of cases, identifying cases where the model gives high values for errors is still a challenge. The method proposed in this thesis already has a gate-level power estimation step in its power analysis flow. So any new method explored should preferably be fast. Predictions about qualitative ordering of power values could be improved with more circuits implemented with one-hot and horizontal encoding and building a better confidence about the optimizer's performance, than achieved in this thesis. Limitations with the number of sequential circuits available in the benchmark set prevents us from doing that in this thesis.

Chapter 6

Conclusions

This thesis introduced the problem of finite state machine power estimation and investigated the feasibility of a new model for finite state machine power estimation.

Finite state machine power estimation is a hard problem because of the presence of feedback in the circuit through the state lines. This feedback introduces correlations among the state inputs to the combinational logic block of the state machine. Existing gate level techniques for FSM power estimation, some of which account for this correlation in the state lines, are accurate but slow. Existing RTL techniques are fast, but inaccurate. None of them account for the correlation in the state lines. A proven RTL technique for FSM power estimation is not yet available. The motivation for the new model studied in this thesis is the requirement for fast FSM power estimation at reasonable levels of accuracy, a characteristic not found in literature so far. It adopts a path in between the existing gate-level and RTL techniques. Quantitative measurements with the new model were done by first estimating the power of the unoptimized implementation of an FSM and then applying a correction factor to that power to give the power of the optimized implementation of the FSM. The correction factor comes out of the *power model*, whose coefficients are computed during the model characterization phase. The model characterization phase studies the power dissipated by the optimized and unoptimized implementations of finite state machines of varying complexities and then models the change in power between the optimized and unoptimized implementations of these FSMs as a function of the variables available from the RTL description of the FSMs. These variables are the number of sum-of-product terms in the output functions of the FSM, literal count in the output functions of the FSM etc.. Numerical results with this approach show a maximum relative error of 102%. This is in-line with the existing techniques for RTL power estimation of finite state machines. Qualitative analysis with the new model shows that qualitative comparisons that could help designers choose the right kind of circuits to implement from among the available design choices, are possible. Studies done on a set of MCNC91 benchmark circuits suggest that, between one-hot and horizontal encodings for finite state machine implementation, the relative "pecking order" of the switched capacitance values of these implementations is preserved through the optimization process. That is, given a finite state machine implemented with horizontal and one-hot encodings, the implementation with a lower value of switched capacitance in the unoptimized circuit also turns out to have the lower value in the optimized circuit, except in a few cases which were explained earlier in chapter 4. Given this observation about the optimizer's behavior towards preserving the qualitative ordering of the power values of these two different implementations, and the designer's knowledge about the finite state machine, he/she can then make choices about design alternatives.

This thesis has clearly demonstrated the infeasibility of a global quantitative modeling of an optimizer's performance towards changing the worst case power dissipation of a finite state machine, given no more information than is available readily from the RTL description of the finite state machine. It has made a valuable contribution in that regard.

Bibliography

[Chandrakasan96] A. P. Chandrakasan, R. W. Brodersen, *Low Power Digital CMOS Design*, chapters 2,3, Kluwer Academic Publishers, 1996.

[Chau91] S. R. Powell, P. M. Chau, *A Model for Estimating Power Dissipation in a Class of DSP VLSI Chips*, IEEE Trans. on Circuits and Systems, Vol. 38, no. 6, June, 1991.

[Deng94] C. Deng, *Power Analysis for CMOS/BiCMOS Circuits*, Proceedings of the 1994 International Workshop on Low Power Design, pp. 3-8, April, 1994.

[Devadas94] S. Devadas, C. Tsui, J. Monteiro, M. Pedram, A. Despain, B. Lin, *Exact and Approximate Methods for Calculating Signal and Transition Probabilities in FSMs*, ACM/IEEE 31st Design Automation Conf., CA, pp. 18-23, June, 1994.

[Ghosh92] A. Ghosh, S. Devadas, K. Keutzer, J. White, *Estimation of Average Switching Activity in Combinational and Sequential Circuits*, 29th ACM/IEEE Design Automation Conf., CA, pp. 253-259, June, 1992.

[Gibbons92] J. Gibbons, S. Chakraborti, *Nonparametric Statistical Inference*, Marcel Dekker Inc., New York, 1992.

[Gilliam91] Paul Gilliam, *A Practical Algorithm for the Minimization of Kronecker Reedmuller Expansions*, M.S. Thesis, Portland State University, 1991.

[Glaser91] K. D. Muller-Glaser, K. Kirsch, K. Neusinger, *Estimating Essential Design Characteristics to Support Project Planning for ASIC Design Management*, ICCAD91, Los alamitos, CA, pp. 148-151, Nov, 1991.

[Gupta97] S. Gupta, F. Najm, *Power Macromodeling for High Level Power Estimation*, 34th Design Automation Conference, Anaheim, CA, 1997.

[Hachtel] G. Hachtel, E. Macii, A. Pardo, F. Somenzi, *Symbolic Algorithms to Calculate Steady-State Probabilities of a Finite State Machine*, Proc. of the European Conference on Design Automation, pp. 214-218, Paris, France.

[Horowitz89] M. Horowitz, A. Salz, *IRSIM: An incremental MOS Switch-Level Simulator*, Proceedings of the 26th Design Automation Conference, pp. 173-178, 1989.

[Kang97] S. Kang, L. Yuan, *A Sequential Procedure for Average Power Analysis of Sequential Circuits*, ECE Dept., UIUC, 1997.

[Kozhaya97] J. Kozhaya, F. N. Najm, *Accurate Power Estimation of Large Sequential Circuits*, IEEE International Conference on Computer Aided Design, Nov, 1997.

[Landman94] P. Landman, *Low Power Architectural Design Methodologies*, Ph.D. Thesis, UC Berkeley, 1994.

[Landman96] P. Landman, J. M. Rabaey, *Activity-Sensitive Architectural Power Analysis for the Control Path*, IEEE Transactions on CAD, Vol. 15, No. 6, pp. 571-587, June, 1996.

[Magic] John Ousterhout, *MAGIC Tutorial #1-#4*, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley.

[Marce94] R. Marculescu, D. Marculescu, M. Pedram, *Switching Activity Analysis Considering Spatiotemporal Correlations*, Proc. International Conf. Computer-Aided Design, 1994.

[Marce96] D. Marculescu, R. Marculescu, M. Pedram, *Information Theoretic Measures for Power Analysis*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 15, No. 6, June, 1996.

[Marce97] M. Marculescu, D. Marculescu, M. Pedram, *Composite Sequence Compaction for Finite-State Machines Using Block Entropy and High-Order Markov Models*, Proc. Symp. on Low Power Electronics and Design, pp. 190-195, August, 1997.

[Matlab] *On-line Tutorial on PSU Unix Systems*

[Najm92] Farid Najm, Richard Burch, Ping Yang, Timothy Trick, *McPOWER: A Monte Carlo Approach to Power Estimation*, IEEE/ACM Int. Conf. Computer-Aided Design, CA, pp. 90-97, Nov, 1992.

[Najm94] Farid N. Najm, *A survey of Power Estimation Techniques in VLSI Circuits*, IEEE Transactions on VLSI Systems, pp. 446-455, December, 1994.

[Najm1'95] F. N. Najm, *Feedback, Correlation, and Delay Concerns in the Power Estimation of VLSI Circuits*, 32nd Design Automation Conference, pp. 612-617, 1995.

[Najm2'95] F. N. Najm, *Power Estimation in Sequential Circuits*, ACM/IEEE Design Automation Conference, 1995.

[Nemani96] M. Nemani, F. N. Najm, *Towards a High-Level Power Estimation Capability*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 15, No. 6, June, 1996

[Nemani97] F. N. Najm, M. Nemani, *High-level area and power estimation for VLSI circuits*, IEEE/ACM International Conference on Computer-Aided Design, pp. 114-119, Nov, 1997.

[Nguyen97] D. V. Nguyen, *An ASIC Power Analysis System for Digital CMOS Design*, M.S. Thesis, Portland State University, 1997.

[Nur97] N. Karsilayan, *Complete-Range Activity-Based RTL Power Estimation*, M.S. Thesis, Portland State University, 1997.

[Papoulis84] A. Papoulis, *Probability, Random Variables and Stochastic Processes*, 2nd Ed., New York: McGraw-Hill, 1984.

[Pedram96] M. Pedram, *Power Minimization in IC Design: Principles and Applications*, ACM Transactions on Design Automation of Electronic Systems, Voll. 1, no. 1, pp. 1-20, January, 1996.

[Pedram97] R. Marculescu, D. Marculescu, M. Pedram, *Vector Compaction Using Dynamic Markov Models*, IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences, Vol. E80-A, No. 10, October, 1997.

[Rabaey94] J. M. Rabaey, P. E. Landman, *Black box Capacitance Models for Architectural Power Analysis*, Proceedings of the 1994 International Workshop on Low-Power Design, Napa Valley, CA, April, 1994.

[Rabaey95] J. M. Rabaey, P. E. Landman, *Architectural Power Analysis: The Dual Bit Type Method*, IEEE Trans. on VLSI Systems, Vol. 3, no. 2, June, 1995.

[Rabaey96] J. M. Rabaey, M. Pedram, *Low Power Design Methodologies*, Kluwer Academic Publishers, 1996.

[Roy94] K. Roy, T. Chou, S. Prasad, *Estimation of Circuit Activity Considering Signal Correlations and Simultaneous Switching*, IEEE Intl. Conf. on Computer-Aided-Design, pp. 300-303, 1994.

[Roy96] K. Roy, T. Chou, *Accurate Power Estimation of CMOS Sequential Circuits*, IEEE Trans. on VLSI Systems, Vol. 4, No. 3, September, 1996.

[Sato95] T. Sato, Y. Ootaguro, M. Nagamatsu, H. Tago, *Evaluation of Architecture-Level Power Estimation for CMOS RISC processors*, IEEE International Symposium on Low Power Electronics and Design, San Jose, CA, 1995.

[Saxena97] V. Saxena, F. N. Najm, I. Hajj, *Monte-Carlo Approach for Power Estimation in Sequential Circuits*, European Design and Test Conference, Paris, France, March, 1997.

[Shravan98] R. Saravanan, *Empirical Estimation of Circuit Switching Activity*, On going M.S. Thesis, Portland State University, 1998.

[Sis] A. Vincentelli, *SIS: A System for Sequential Circuit Synthesis*, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, 1992.

[Svensson94] D. Liu, C. Svensson, *Power Consumption Estimation in CMOS VLSI Chips*, IEEE Journal of Solid-State Circuits, Vol. 29, no.6, June, 1994.

[Veendrick84] H. J. M. Veendrick, *Short-Circuit Dissipation of Static CMOS Circuitry and its Impact on the Design of Buffer Circuits*, IEEE Journal of Solid State Circuits, pp. 468-473, August 1984.

[Weste93] N. H. E. Weste, K. Eshraghian, *Principles of CMOS VLSI Design*, chapter 2, Addison Wesley, 1993.

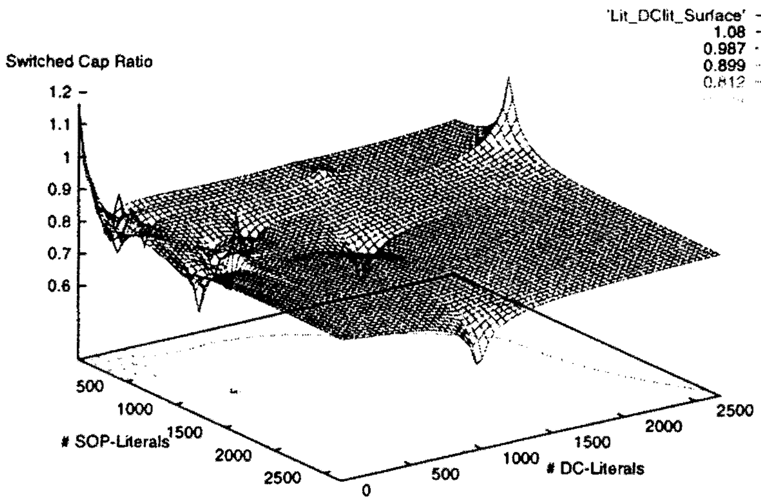
[Wu97] Q. Wu, Q. Qiu, M. Pedram, C. Ding, *Cycle-Accurate Macro-Models for RT-level Power Analysis*, IEEE Intl. Symposium on Low Power Electronics and Design, 1997.

[Xakellis94] Michael G. Xakellis, Farid N. Najm, *Statistical Estimation of the Switching Activity in Digital Circuits*, 31st ACM/IEEE Design Automation Conference, San Diego, CA, pp. 728-733, June, 1994.

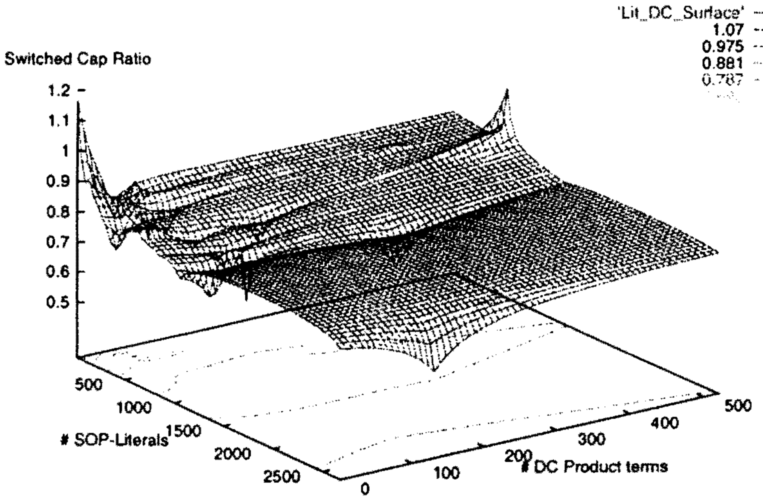
Appendix A

Surface Plots of Switched Capacitance Ratio For MCNC91 Circuits

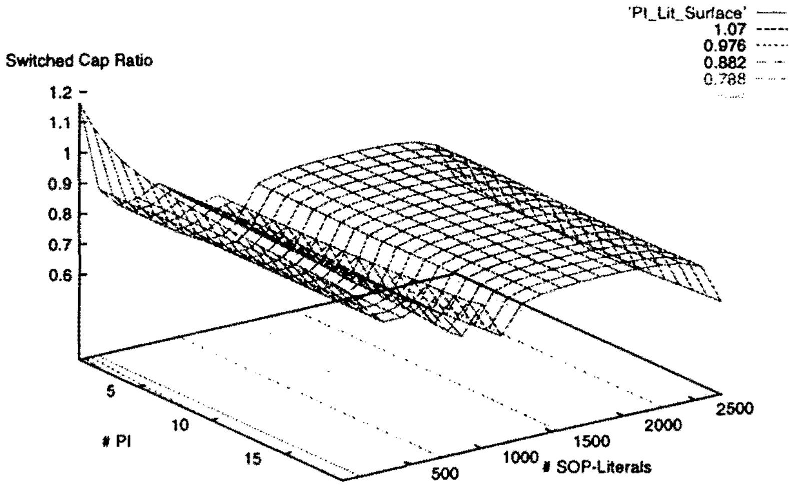
Surface Plot of Cap Ratio Vs. #SOP Literals and #DC Literals For 41 MCNC91 FSMs



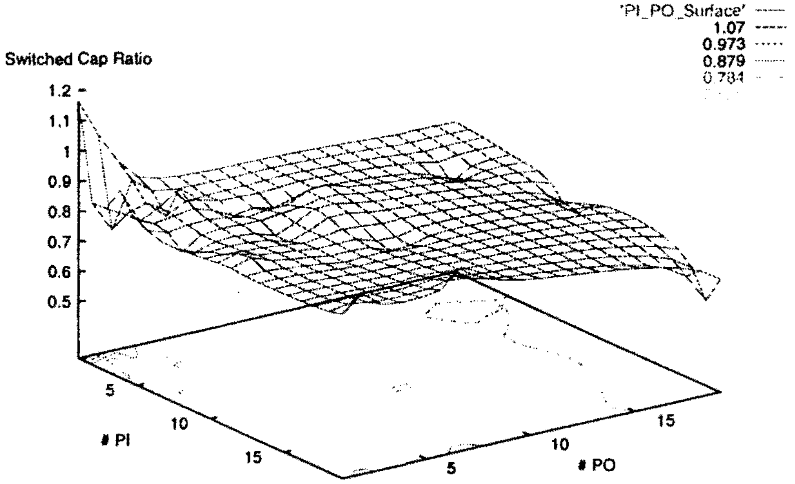
Surface Plot of Cap Ratio Vs. #SOP Literals and #DC Prod.terms For 41 MCNC91 FSMs



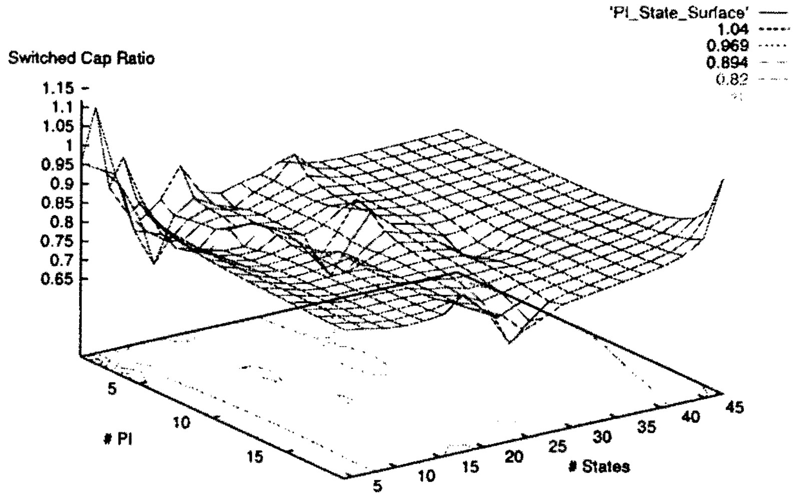
Surface Plot of Cap Ratio Vs. #PI and #SOP Literals For 41 MCNC91 FSMs



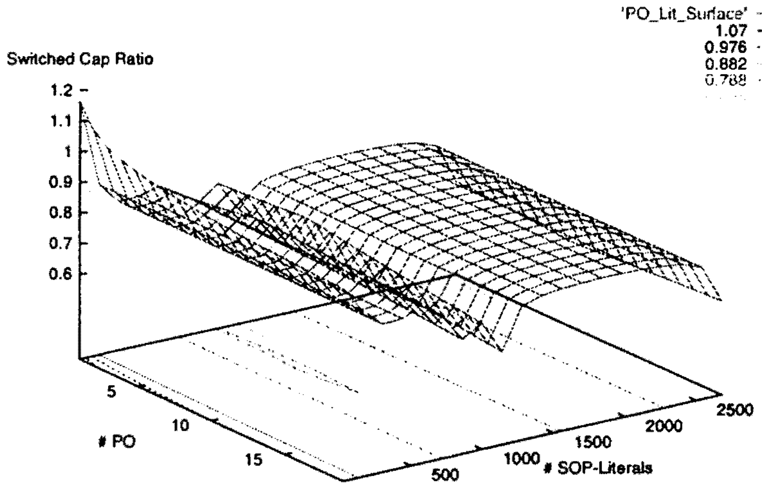
Surface Plot of Cap Ratio Vs. #PI and #PO For 41 MCNC91 FSMs

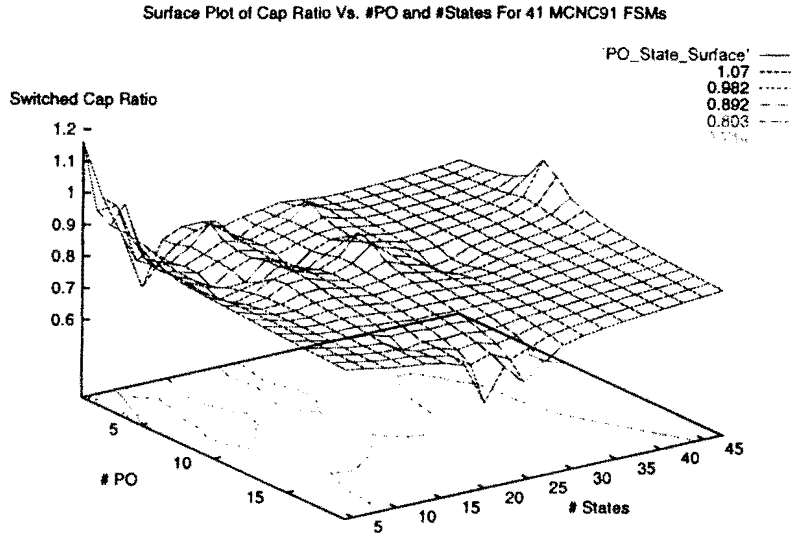


Surface Plot of Cap Ratio Vs. #PI and #States For 41 MCNC91 FSMs

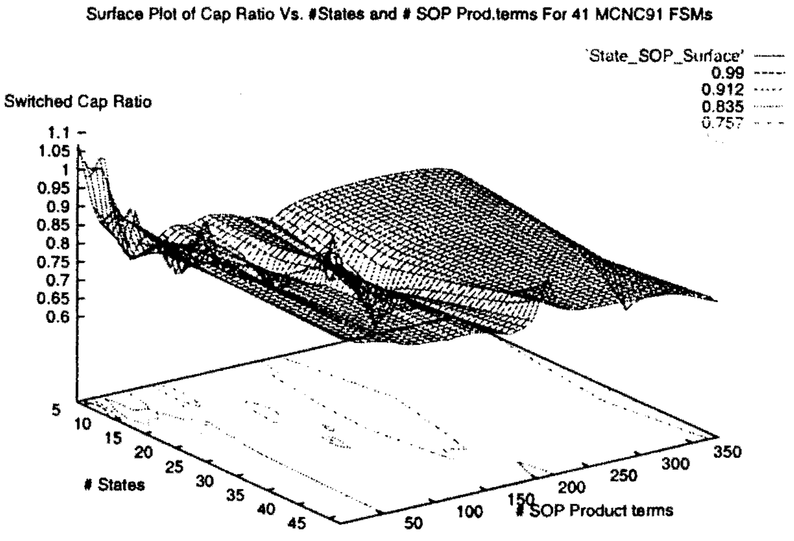


Surface Plot of Cap Ratio Vs. #PO and Literal Count For 41 MCNC91 FSMs



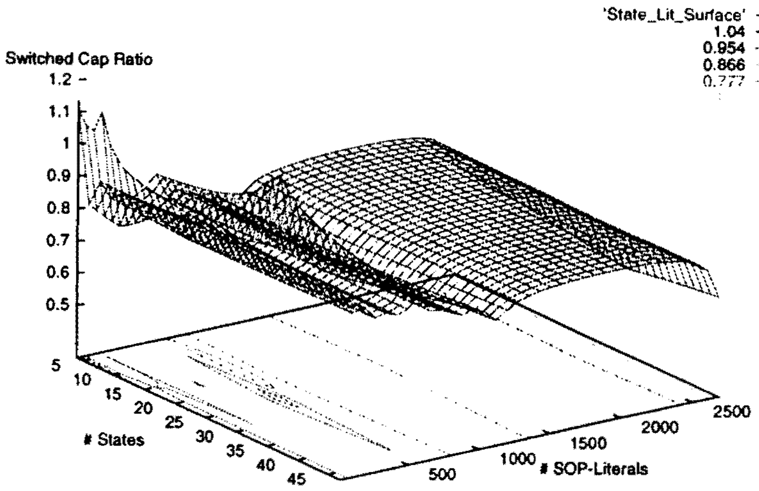


Surface Plot of Cap Ratio Vs. #PO and #States For 41 MCNC91 FSMs

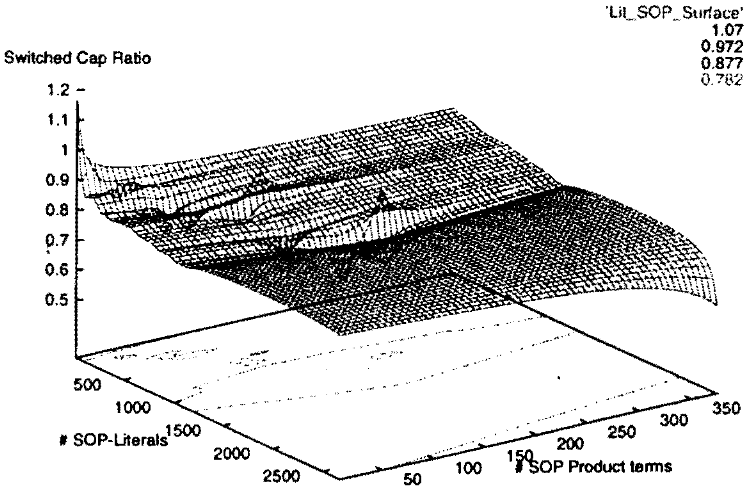


Surface Plot of Cap Ratio Vs. #States and # SOP Prod.terms For 41 MCNC91 FSMs

Surface Plot of Cap Ratio Vs. #States and #SOP Literals For 41 MCNC91 FSMs



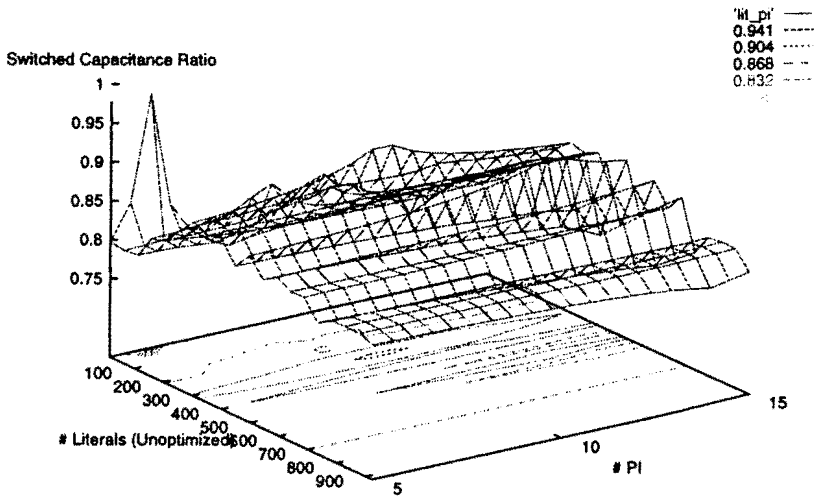
Surface Plot of Cap Ratio Vs. #SOP Literals and # SOP Prod.terms For 41 MCNC91 FSMs



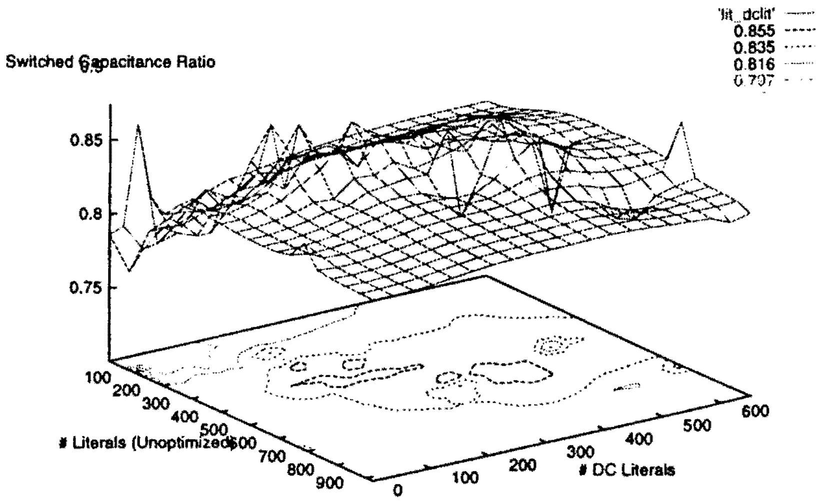
Appendix B

Surface Plots of Switched Capacitance Ratio For Random Circuits

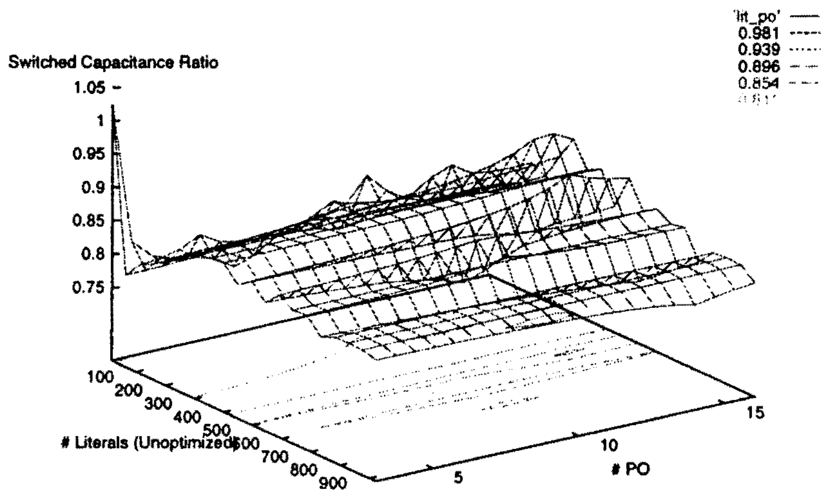
Surface Plot of Cap Ratio Vs. #PI and Literal Count For 249 Randomly Generated Circuits



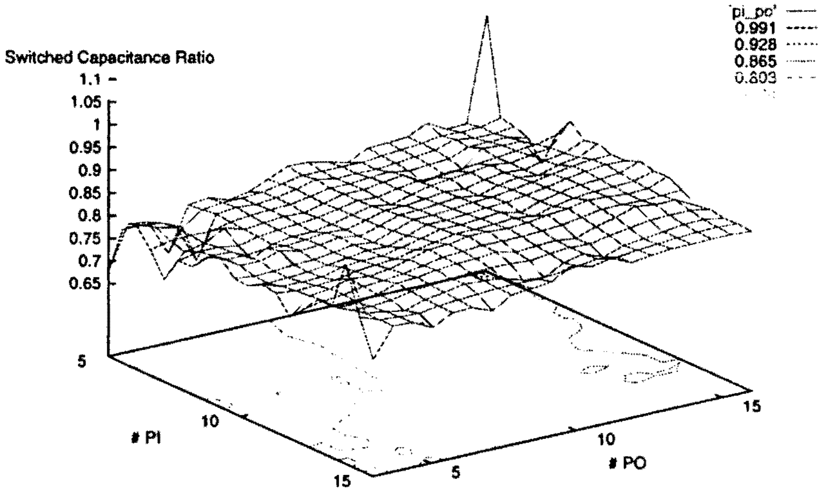
Surface Plot of Cap Ratio Vs. #DC Literals and Literal Count For 249 Randomly Generated Circuits

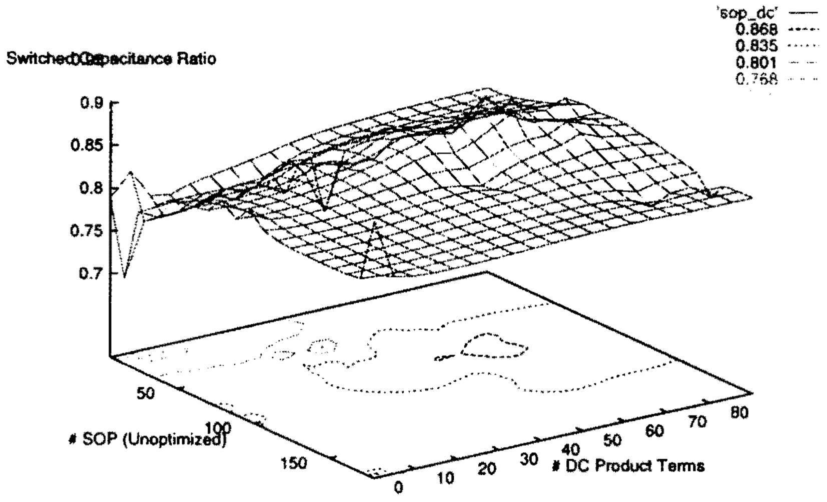


Surface Plot of Cap Ratio Vs. #PO and Literal Count For 249 Randomly Generated Circuits

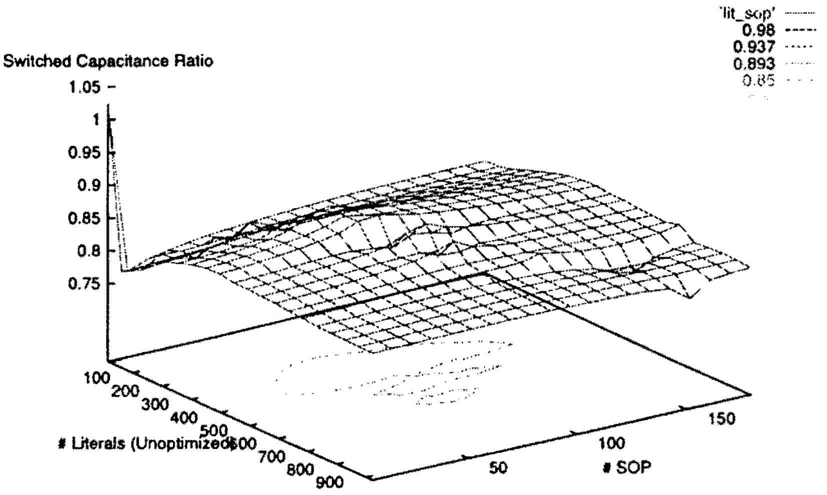


Surface Plot of Cap Ratio Vs. #PI and #PO For 249 Randomly Generated Circuits





Surface Plot of Cap Ratio Vs. #SOP and #DC Product terms For 249 Randomly Generated Circuits



Surface Plot of Cap Ratio Vs. #SOP and Literal Count For 249 Randomly Generated Circuits

Appendix C

FSM Description in Kiss Format

```
.i 5
.o 6
.p 22
.s 10
--1--   * init0   110000
--1-- init0  init0   110000
--0-- init0  init1   110000
--00- init1  init1   110000
--01- init1  init2   110001
--0-- init2  init4   110100
--01- init4  init4   110100
--00- init4  IOwait  000000
0000- IOwait  IOwait  000000
1000- IOwait  init1   110000
01000 IOwait  read0   101000
11000 IOwait  write0  100010
01001 IOwait  RMACK   100000
11001 IOwait  WMACK   100000
--01- IOwait  init2   110001
--0-0 RMACK  RMACK   100000
--0-1 RMACK  read0   101000
--0-0 WMACK  WMACK   100000
--0-1 WMACK  write0  100010
--0-- read0  read1   101001
--0-- read1  IOwait  000000
--0-- write0  IOwait  000000
```

Appendix D

Random Boolean Circuits

The characteristics of the random boolean circuits are summarized in the table below. They are available for download at:
ftp://ee.pdx.edu/~daasch/randFSM.tar

Variable	Range
Primary Inputs	5 to 15
Primary Outputs	3 to 16
Literals	16 to 923
Sum of Products	6 to 179
DC Sum of Products	0 to 90
DC Literals	0 to 650
Switched Capacitance Ratio	0.34 to 1.27