June 1998

UILU-ENG-98-2215
DAC 65

University of Illinois at Urbana-Champaign

Accurate Power Estimation for Large Sequential Circuits

Joseph Nicolas Kozhaya

Coordinated Science Laboratory
1308 West Main Street, Urbana, IL 61801

UNCLASSIFIED

URITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188
REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited
DECLASSIFICATION/DOWNGRADING SCHEDULE	
ERFORMING ORGANIZATION REPORT NUMBER(S) UILU-ENG-98-2215 DAC-65	5. MONITORING ORGANIZATION REPORT NUMBER(S)

NAME OF PERFORMING ORGANIZATION Coordinated Science Lab University of Illinois	6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION SRC
ADDRESS (City, State, and ZIP Code) 1308 W Main St Urbana, IL 61801		7b. ADDRESS (City, State, and ZIP Code) PO 12053 Research Triangle Park, NC 27709
NAME OF FUNDING/SPONSORING ORGANIZATION RC	8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER 96-DP-109

ADDRESS (City, State, and ZIP Code) O 12053 esearch Triangle Park, NC 27709	10. SOURCE OF FUNDING NUMBERS			
	PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

TITLE (Include Security Classification)

ccurate Power Estimation for Large Sequential Circuits

PERSONAL AUTHOR(S)

ozhaya, Joseph Nicolas

. TYPE OF REPORT Technical	13b. TIME COVERED FROM ______ TO ______	14. DATE OF REPORT (Year, Month, Day) 1998 June 25	15. PAGE COUNT 39

SUPPLEMENTARY NOTATION

COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Power estimation; sequential circuit; random sampling; power vector set

. ABSTRACT (Continue on reverse if necessary and identify by block number)

The average power dissipation of a logic circuit depends on the input vector set. For sequential circuits, performing power estimation based on randomly generated vectors (based on some assumed or measured signal activity and probability values) can lead to power estimates that are completely wrong. This is because the real vector set (as opposed to a randomly generated vector set) may contain specific vector sequences that put the circuit in specific operational modes or subspaces of its large state space and, in different operational modes, the circuit may dissipate quite different values of power. Essentially, simulation with unrealistic vectors can take the circuit to parts of the state space where it was never meant to operate, and once this happens, the estimated power dissipation can be completely different from what the circuit would dissipate under realistic legal operation. This is a common problem in most existing power estimation methods. However, simulating real vector sets for purposes of power estimation can be very expensive because the vector set (in order to be representative of the real power) can be huge. To solve this problem, a power estimation approach is presented in which blocks of consecutive vectors are selected at random from a user-supplied realistic input vector set and the circuit is simulated for each block starting from an unknown state. This leads to two (upper and lower) bounds on the desired power value which can be quite tight (under 10% difference between the two in many cases). As a result, the power dissipation is obtained by simulating only a fraction of the potentially very large vector set.

. DISTRIBUTION/AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL

ACCURATE POWER ESTIMATION FOR LARGE SEQUENTIAL CIRCUITS

BY

JOSEPH NICOLAS KOZHAYA

B.E., American University of Beirut, 1996

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1998

Urbana, Illinois

ABSTRACT

The average power dissipation of a logic circuit depends on the input vector set. For sequential circuits, performing power estimation based on randomly generated vectors (based on some assumed or measured signal activity and probability values) can lead to power estimates that are completely wrong. This is because the real vector set (as opposed to a randomly generated vector set) may contain specific vector sequences that put the circuit in specific operational modes or subspaces of its large state space and, in different operational modes, the circuit may dissipate quite different values of power. Essentially, simulation with unrealistic vectors can take the circuit to parts of the state space where it was never meant to operate, and once this happens, the estimated power dissipation can be completely different from what the circuit would dissipate under realistic legal operation. This is a common problem in most existing power estimation methods. However, simulating real vector sets for purposes of power estimation can be very expensive because the vector set (in order to be representative of the real power) can be huge. To solve this problem, a power estimation approach is presented in which blocks of consecutive vectors are selected at random from a user-supplied realistic input vector set and the circuit is simulated for each block starting from an unknown state. This leads to two (upper and lower) bounds on the desired power value which can be quite tight (under 10% difference between the two in many cases). As a result, the power dissipation is obtained by simulating only a fraction of the potentially very large vector set.

ACKNOWLEDGMENTS

I thank my academic advisor, Dr. Farid Najm, for the guidance he gave me and the time he devoted to this work. Finally, I thank my parents for their love and support throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Maximizing circuit speed and minimizing chip area used to be the only major concerns of VLSI designers. In recent years, power consumption of integrated circuits (ICs) has proved to be just as important of a concern. Thus, VLSI designs nowadays emerge as a trade-off between three goals: minimum area, maximum speed, and minimum power dissipation.

The decrease in feature size and the corresponding increase in chip density have made power dissipation a major factor in the reliability of the chip. Excessive power dissipation causes overheating and raises the device operating temperature. This may lead to soft errors as well as permanent damage. Thus elaborate and costly packaging as well as system level cooling are required. In other words, high power consumption results in higher risk and consequently higher cost.

In addition to the reliability issue, low power designs play an important role in promoting portable applications such as cellular phones, pagers, personal digital assistants (PDAs), and laptop computers. The success of portable applications depends greatly on the availability of these applications anywhere anytime. The longer an application is available, the more desirable it is. However, the operating time of an application is limited by the battery life, i.e., the time for which the equipment can be used before having to replace or recharge the battery. To extend the operating time of a portable application, either the battery used should have a longer life or the chip used should consume less power. Because not much improvement has been cited in battery design, low power chips tend to be the better alternative.

It is clear from the above discussion that low power designs are essential to cope with the technological progress. The first step in designing for power optimization is power estimation, which is the subject of this thesis. Because most useful circuits are sequential, our work targets the average power estimation problem for large sequential circuits.

1.1 Sources of Power Dissipation in CMOS Circuits

In estimating the average power dissipation of a CMOS VLSI chip, it suffices to estimate the average supply current since the supply voltage is approximately constant. Four components of the supply current can be identified [1]:

- Capacitive current: This is the current responsible for charging/discharging the node capacitances during a logic transition. This current is the dominant source of power dissipation in CMOS circuits. The power dissipated because of this current is given by:

  $$P = \frac{1}{2} C_L V_{dd}^2 D$$

  where C_L is the physical load capacitance, V_{dd} is the supply voltage, and D is the transition density, that is, the average number of logic transitions per second.

- Short circuit current: In a CMOS circuit, during a logic transition, both the nMOS and the pMOS transistors conduct simultaneously for a short while. During that time, a short circuit is established between the power bus and ground. The current that flows in this short circuit is referred to as the short circuit current.

- Standby current: This current is continuously drawn from the power supply even in steady state. This kind of current is significant in design styles like pseudo-nMOS. Actually, such a current doesn't exist in fully complementary CMOS designs.

- Leakage current: This current further subdivides into two categories: diode leakage and subthreshold leakage. In CMOS circuits, all source and drain diffusions constitute reverse-biased diodes in which a diode leakage current flows. However, this current is typically small and often neglected. The subthreshold leakage current, on the other hand, is more prominent. It is due to the inversion charge that exists when the MOS transistor is supposed to be off.

1.2 Previous Power Estimation Techniques

Before presenting previous work done on power estimation, we should note that by power estimation we refer to the problem of *average* power estimation. This is different from the estimation of the worst case instantaneous power. Chip reliability and equipment lifetime are directly related to the average power.

Several approaches have been proposed for power estimation [2], especially at the gate level. The most straightforward approach is simply to perform circuit simulation, monitor the power supply current waveform, and then compute its average. These simulation-based techniques can be quite accurate; however, they are strongly input pattern dependent. That is, simulating the circuit for different inputs results in quite different power values. Furthermore, these techniques are too slow to be useful for large circuits.

To improve on simulation-based techniques, probabilistic techniques were proposed. All probabilistic techniques require the user to supply some probability measure, such as signal probability, transition probability, or transition density [2], at the primary inputs. Then these probability measures are propagated through the circuit using specialized probabilistic models for circuit blocks. The first probabilistic approach, which was proposed in [3], assumed a zero-delay model as well as temporal independence. Further improvements on that approach, which incorporated a nonzero delay model and accounted for temporal correlations, were achieved by the *probabilistic simulation* method proposed in [4]. Another probabilistic approach, which introduced the transition density measure of circuit activity was proposed in [5] and [6]. However, all the above probabilistic methods assume spatial independence and this can lead to significant inaccuracy. In other words, these techniques assume that the inputs to a particular gate are mutually independent which is not true in general. One common example is reconvergent fanout, which leads to correlation between inputs driving a gate. A probabilistic approach that accounts for spatial correlations was presented in [7]. This approach uses Binary Decision Diagrams (BDDs) [8] to account for internal node correlations, temporal correlations, and toggle power. Toggle power, also known as glitch power, is accounted for by considering a nonzero delay model. Although the last approach accounted for spatial correlations, it proved to be computationally expensive, especially for circuits where toggle power is dominant. Another probabilistic approach that accounts for pairwise correlations at the input was proposed in [9]. However, this approach doesn't account for all possible input signal correlations that may exist in a sequential circuit and hence, a significant error is still involved in the approximation. Thus, it is clear that the spatial independence assumption is essential to maintain the speed advantage of probabilistic techniques. Consequently, the usefulness of such techniques is limited by how much loss in accuracy can be tolerated.

To avoid the severe accuracy/speed trade-off imposed by probabilistic methods, statistical techniques based on Monte Carlo simulation were proposed in [10], [11], and [12]. The idea is quite simple: simulate the circuit repeatedly and study the information produced to generate a reliable power estimate. Eventually the power monitored will converge to the average power. Two issues of concern are input vector generation and convergence checking. The input vectors are generated randomly according to user-specified statistics. As for convergence, it is decided based on statistical mean estimation techniques [13]. Using statistical techniques, correlations arising within the circuit are automatically taken care of since *actual* simulations are performed. A further advantage of statistical techniques is that they can be implemented using pre-existing simulators and can be integrated into the design environment with minimal effort.

The above techniques do not apply directly to sequential circuits. Other approaches, [14], [15], [16], [17], [18], and [19], have been proposed to handle sequential circuits where [14], [15], [16], and [17] use probabilistic methods while [18] and [19] use statistical methods. All probabilistic techniques handling sequential circuits make the following assumptions. The first assumption is that the sequential circuit implements a finite state machine (FSM) with a connected state space. The second assumption is that the FSM is Markov [20] (so that its future doesn't depend on its past once its present state is specified). These methods try to find the signal and transition probabilities at the present state lines of the FSM. Then they use any of the above mentioned combinational techniques, with some approximation, to compute the power.

However, the major problem with all these techniques that handle sequential circuits is that the input vectors (or probabilities) are random and unrealistic. Thus, unrealistic input probabilities (or vectors) may take the circuit into parts of its state space where it was not meant to operate under typical operation. In that case, the obtained power values may be completely wrong. As a solution to this problem, we present in this thesis a *block sampling* technique with an all-X initial state (all state bits unknown) that samples a (potentially very long) *real* (or *typical*) vector set to provide accurate power estimates for large sequential circuits.

1.3 Thesis Overview

The rest of the thesis is organized as follows. In Chapter 2, we discuss the central problem in power estimation (input-pattern dependence), present a formulation of the problem, and show how the bounds on the power are to be estimated. Then in Chapter 3, we present the implementation details of our proposed approach. Results giving empirical evidence of the validity of the approach and illustrating its advantages are provided in Chapter 4. Finally, conclusions and final comments are presented in Chapter 5.

CHAPTER 2

PROPOSED APPROACH

Given a low-level (typically, gate-level) design representation, many may consider the power estimation to be a "solved problem." We will argue that this is not the case at all. At least two open problems remain: (1) *Accurate* and *fast* estimation of the average power dissipated by individual gates, typically inside of an optimization loop, and (2) *Accurate* and *fast* estimation of the total average power dissipation in *large* sequential circuits. The words "accurate" and "fast" are emphasized in both cases to indicate that existing techniques are either inaccurate and fast or accurate and slow. The fact that the first problem is not yet solved has recently been clearly illustrated in [21]. In this paper, we will argue and demonstrate that the second problem is also still open, and we offer a new method that provides accurate and fast estimation of the total average power of large sequential circuits.

2.1 Input Pattern Dependence

We have mentioned before that the central problem in power estimation is that power is input-pattern dependent. Thus, the "average power dissipation" of a given circuit is not well-defined until a specific vector set is chosen. For combinational circuits, this may not be very critical, because different vector sets may dissipate approximately the same power, provided they have approximately equal values of *switching activity*. Thus, using a set of randomly generated vectors (with the right statistics) may be appropriate for combinational circuits. However, this does not hold for sequential circuits, because a real vector set (as opposed to a randomly generated, artificial vector set) may contain specific vector sequences that put the circuit in specific operational modes or subspaces of its large state space and, in different operational modes, the circuit may dissipate quite different values of power. All one has to do is think of all the many different operational modes of a large microprocessor. Thus, for sequential circuits, the power may be *critically* dependent on the specific vector sequences that occur during typical operation.

Most previous techniques that we referred to in Chapter 1 suffer from a major drawback. They run the risk of taking the finite state machine (FSM) implemented by the sequential circuit into parts of its state space where it was not meant to operate. When this happens, there is no guarantee that the estimated power has any relation to what the circuit will actually dissipate under typical operation.

To illustrate this problem, we have considered a number of sequential circuits and constructed two sets of input vectors for each. Both sets of vectors have the same switching activity and signal probability for each input node. However, in one vector set, the input signals were generated at random, without any correlation between them, and in the other nonzero correlations were considered, both in space (between pairs of bits in the same vector) and in time (between pairs of consecutive vectors). The intention is that these correlations would mimic to some degree the relationships that typically exist between signals, such as signals resulting from decoded instructions or general control signals. Note, however, that these correlations are only the simplest kinds of correlation relations, because they do not model the temporal correlations that can exist in vector streams over *several* clock cycles. In fact, it is not hard to see that a sequential circuit may receive certain input vectors only in certain modes of operation that are activated by some control signals, so that it is possible for signal values to be correlated across a large number of clock cycles. We emphasize this point to indicate that proposed approaches that use correlation coefficients [9] may be able to handle pairwise correlations between bits in a vector or between consecutive vectors, but can not handle the variety of other input signal relations that can exist in sequential circuits. Even with just these simple correlations applied, we found that big differences are possible in the resulting power

Table 2.1 Differences in power values (in mW) due to the presence of correlation in the vector set.

Ckt	#inputs	#latches	#gates	Power (uncorr)	Power (corr)	Error(%)
s27	4	3	13	0.1405	0.1212	15.9
s298	3	14	119	0.3152	0.2615	20.5
s344	9	15	160	0.3865	0.3115	24.1
s349	9	15	161	0.3924	0.3181	23.4
s382	3	21	158	0.2021	0.1651	22.4
s386	7	6	159	0.4122	0.3591	14.8
s400	3	21	164	0.1988	0.1597	24.5
s444	3	21	181	0.2085	0.1692	23.3
s526	3	21	193	0.3319	0.2561	29.6
s526n	3	21	194	0.3375	0.2545	32.6
s641	35	19	379	0.3232	0.2982	8.4
s713	35	19	393	0.3240	0.3034	6.8
s820	18	5	289	0.4912	0.3998	22.9
s832	18	5	287	0.4679	0.3921	19.3
s1196	14	18	529	1.4719	1.7009	−13.5
s1238	14	18	508	1.5742	1.8381	−14.4
s1423	17	74	657	0.3461	0.2308	50.0
s1488	8	6	653	0.3729	0.1796	107.7
s1494	8	6	641	0.3648	0.1657	120.2
s35932	35	1728	16065	13.8721	12.2942	12.8

values, as shown in Table 2.1. Power (uncorr) refers to power due to uncorrelated vector set while Power (corr) refers to power due to correlated vector set. The error is measured as the difference between the two power values, relative to the power due to the correlated set. A positive error means that the power of the uncorrelated set is higher than that of the correlated set. Note that the power in both cases (uncorrelated and correlated inputs) was measured using the same simulator, so that the errors are due only to the presence of the additional correlations in one vector set but not in the other.

It would seem, therefore, that the only truly accurate power estimation method for sequential circuits is to simulate the circuit for a specific *realistic* and *typical* vector set. We refer to such a vector set as the *power vector set.* If one has a power vector set which is short enough to simulate in its entirety, then this would certainly be the method of choice. However, in practice it is very hard (almost impossible) to specify a power vector set which is both short enough for simulation and long enough to cover all the interesting operational modes of a large sequential circuit. Microprocessor designers will usually agree that millions of vectors may be needed in order to satisfactorily exercise their large designs.

In order to overcome this impasse, we propose a method of power estimation that takes a (potentially very long) power vector set and provides an estimate of the total power by simulating only a fraction of the vector set. The vectors to be simulated are selected by repeatedly choosing blocks of consecutive vectors at random, until certain accuracy criteria are met. We call this a *block-sampling* approach. From the repeated simulations of the blocks, we collect statistics on the *mean upper bound* and *mean lower bound* for the power per block. These means are the means over *all* possible vector blocks in the power vector set. However, using standard Monte-Carlo mean estimation techniques, the two means can be estimated with user-specified accuracy and confidence without having to simulate all blocks. The net effect is that only a fraction of the total vector set is simulated and accurate tight bounds on the total power are estimated, yielding a viable accurate power measure.

2.2 Problem Formulation

Let $u_1, u_2, \ldots, u_m$ be the primary input nodes of a sequential logic circuit, as shown in Figure 2.1, and let $x_1, x_2, \ldots, x_n$ be the *present state* lines. For simplicity of presentation, we have assumed that the circuit contains a single clock that drives a bank of edge-triggered flip-flops. On the falling edge of the clock, the flip-flops transfer the values at their inputs to their outputs. The inputs $u_i(k)$ and the *present state* values $x_i(k)$ determine the *next state* values $x_i(k+1)$ and the circuit outputs, where k denotes the clock cycle, so that the circuit implements a *finite state machine* (FSM).

Suppose also that a power vector set is provided that consists of the input vectors $U(1), U(2), \ldots, U(M)$, where $U(k) = [u_1(k)\ u_2(k)\ \ldots\ u_m(k)]$ is the input vector applied

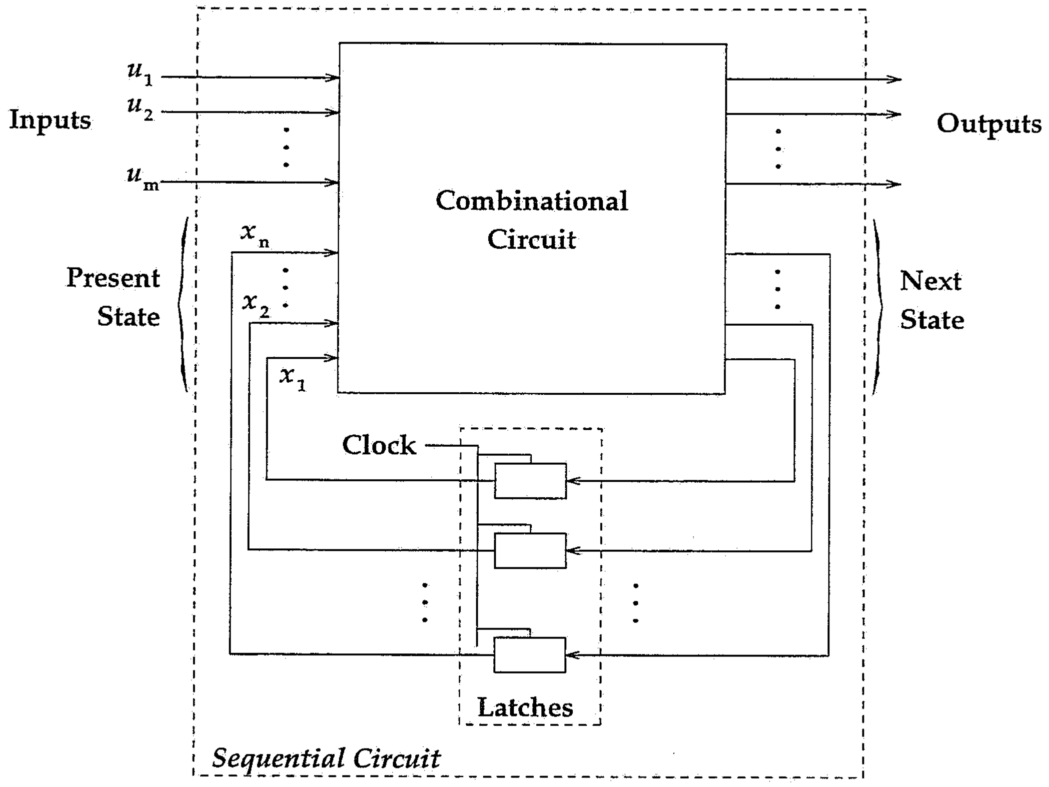

Figure 2.1 An FSM model of a sequential logic circuit.

during cycle k, and M is the total number of vectors. We assume that the initial state vector $X(1)$ is well-defined, so that there exists a well-defined resulting sequence of state vectors $X(1), X(2), \ldots, X(M)$, where $X(k) = [x_1(k)\ x_2(k)\ \ldots\ x_m(k)]$. The initial state need not be known, it only needs to be well-defined, i.e., not arbitrary or variable, in order for the power (due to this vector set) to be well-defined.

The total energy dissipated in the circuit in the kth cycle, denoted $e(k)$, is a function of $X(k-1)$, $X(k)$, $U(k-1)$, and $U(k)$. For $e(1)$, because $X(0)$ and $U(0)$ are not defined, we arbitrarily define $e(1) = 0$. Over a block of K consecutive input vectors, starting at cycle i, the average power dissipated is:

$$P_K(i) = \frac{1}{KT} \sum_{k=i}^{i+K-1} e(k) \tag{2.1}$$

where T is the clock period, and the desired total power dissipation P (over the whole vector set) is given by:

$$P = \frac{1}{MT}\sum_{k=1}^{M} e(k) = \frac{1}{M}\sum_{i=-K+2}^{M} P_K(i) \tag{2.2}$$

Note that for the last $K-1$ blocks, for which $i+K-1 > M$, one should use $e(k) = 0$ in (2.1) for all $k = M+1, \ldots, M+K-1$. The same applies to the first $K-1$ blocks, $e(k) = 0$ for $k = -K+2, \ldots, 0$. This is required in order for the average power per block to be equal to the average power per cycle, leading to (2.2). If we now consider a probability experiment in which a block of vectors is chosen at random from the power vector set so that all blocks are equi-probable, then the average power per block becomes a random variable, denoted $\mathbf{P_K}$, which takes values in the set $\{P_K(-K+2), P_K(-K+3), \ldots, P_K(M)\}$. We will use bold font to denote random quantities. From (2.2), it becomes clear that the total power is the following *mean* or *expected value*:

$$P = E[\mathbf{P_K}] \tag{2.3}$$

where $E[\cdot]$ denotes the expected value operator. If $P_K^u(i)$ and $P_K^l(i)$ are upper and lower bounds on $P_K(i)$, respectively, then we can also talk about the random variables $\mathbf{P_K^u}$ (random upper-bound value) and $\mathbf{P_K^l}$ (random lower-bound value), and it also becomes clear that:

$$E[\mathbf{P_K^l}] \leq P \leq E[\mathbf{P_K^u}] \tag{2.4}$$

In the next chapter, we propose a practical method for estimating the two bounds in (2.4).

CHAPTER 3

IMPLEMENTATION DETAILS

If it were possible to obtain sample values of $P_K(i)$ for a sufficient number of values i, it would then be possible to estimate P based on (2.3) to any desired accuracy (with some specified confidence) using traditional statistical methods of mean estimation. However, since the FSM state at the start of a block is unknown, this cannot be done. Instead, our approach is based on Equation (2.4) and involves using mean-estimation techniques to find two bounds on the unknown power value.

Briefly stated, we make N random choices for the block start index i (let these constitute a set of indices I) from which we compute by simulation N sample values of each of the random variables $\mathbf{P_K^u}$ and $\mathbf{P_K^l}$. We then compute the two means:

$$E[\mathbf{P_K^u}] \approx \frac{1}{N}\sum_{i\in I} P_K^u(i) \qquad \text{and} \qquad E[\mathbf{P_K^l}] \approx \frac{1}{N}\sum_{i\in I} P_K^l(i) \tag{3.1}$$

which we can use as bounds on the desired power value P, based on (2.4). It remains to describe how to perform the simulation in order to obtain $P_K^u(i)$ and $P_K^l(i)$, and how we choose values for K and N. These topics are covered below.

3.1 Block Simulation

The simulation of a block of vectors is complicated by the fact that the state of the FSM at the beginning of that block is not known. Any wrong choice made for the state at that time can have the effect that the simulation of this block takes the FSM into states that never occur in practice. Therefore, we set the FSM to an all-X state (all state bits are in the unknown state) and perform three-valued gate-level simulation, with the values (0, 1, X). During the simulation of the block, we compute two bounds on the power due to that block. The upper (lower) bound is found by assuming that every signal transition containing an X value actually occurs with the X replaced by either a 0 or a 1, whichever leads to the larger (smaller) power dissipation for that transition. For instance, if the output of a gate makes an X $\rightarrow$ 1 transition, then it is assumed to be a 0 $\rightarrow$ 1 transition for purposes of computing the upper bound and a 1 $\rightarrow$ 1 transition for purposes of computing the lower bound. For purposes of continuing the simulation, the transition is kept as X $\rightarrow$ 1. Likewise, when the output of a gate makes an X $\rightarrow$ X transition, it is assumed to be a 0 $\rightarrow$ 1 (or 1 $\rightarrow$ 0) transition for purposes of computing the upper bound and a 0 $\rightarrow$ 0 (or 1 $\rightarrow$ 1) transition for purposes of computing the lower bound. For continuing the simulation, it is kept as an X $\rightarrow$ X transition. In this way, the true (unknown) signals in the circuit are guaranteed to be sub-sets of the simulated signals, and the true power for that block is guaranteed to be between the two resulting bounds $P_K^u(i)$ and $P_K^l(i)$.

The reason that this method can be useful in practice is that in many cases, many of the X values become definite 0 or 1 values during three-valued simulation (more on this in Chapter 4). In fact, we have found that sufficiently many X values become known that the two bounds resulting from the simulation of one vector block can be very close, close enough to constitute a viable measure of power.

We should point out that any type of simulation model may be used–the measured power will be as accurate as the simulation model. Because we are computing the total power of the circuit (and not the powers of individual gates), we find that a logic simulator with a good timing model is sufficient. In our implementation, every gate has a scalable delay value, depending on the output loading capacitance due to its drain capacitance and the MOSFET gate capacitance of the logic gates on the fanout branches. Our simulator is event-driven, so that the estimated power includes the power due to glitches. Let $n_k^l(i)$ and $n_k^u(i)$ be lower and upper bounds on the number of logic transitions made by node i in clock cycle k, respectively. These are computed during the simulation by simply considering that signal transitions involving an X value can be interpreted in two ways as explained above, with one way representing more actual transitions and another way representing less. Thus, for example, upon observing a $0 \rightarrow X$ transition at node i, we would increment $n_k^u(i)$ (due to the $0 \rightarrow 1$ possibility) and not increment $n_k^l(i)$ (due to the $0 \rightarrow 0$ possibility). From this, the total energy dissipated in clock cycle k is bounded by $e^l(k) \leq e(k) \leq e^u(k)$, where the energy bounds are computed as follows:

$$e^l(k) = \frac{1}{2}\sum_i V_{dd}^2 C_i n_k^l(i) \tag{3.2}$$

and

$$e^u(k) = \frac{1}{2}\sum_i V_{dd}^2 C_i n_k^u(i) \tag{3.3}$$

respectively, where C_i is the node capacitance and the summations are taken over all gate/latch output nodes in the circuit. The reason for the 1/2 coefficient is that, on average, half the transitions will be low-to-high and the other half will be high-to-low. As pointed out above, one does not have to use this particular power model (3.2–3.3), and any number of more accurate power modeling approaches can be used. All that is required is that the energy bounds $e^l(k)$ and $e^u(k)$ be computable during the simulation. In this work, this model was deemed sufficiently accurate in order to illustrate the feasibility of the approach. From this, the block upper/lower bound values $P_K^u(i)$ and $P_K^l(i)$ are computed in a way similar to Equation (2.1), as follows:

$$P_K^l(i) = \frac{1}{KT}\sum_{k=i}^{i+K-1} e^l(k) \tag{3.4}$$

$$P_K^u(i) = \frac{1}{KT}\sum_{k=i}^{i+K-1} e^u(k) \tag{3.5}$$

3.2 Choice of Block Size, K

The choice of block size, K, can affect the tightness of the bounds in (2.4). This is because the larger the block is, the more probable it is that more X values will be converted to definite 0 or 1 values during the simulation. On the other hand, K should not be too large because beyond some point there will typically be very little or no reduction in the number of X values. In the development leading to Equation (2.2), it was made clear that K should be a constant (the same for all blocks). However, during the simulation of a block of vectors, if it is found that $P_K(i)$ has already been estimated with sufficient accuracy before all K vectors have been simulated, there would clearly be no reason to continue the simulation of that block. Therefore, we use a dynamic scheme in which we may simulate only $K' < K$ vectors from a block, with K being the *hard limit* on the number of simulation vectors per block. In our implementation, this hard limit was chosen empirically, by looking at a large number of simulations, and we found that a value of $K = 500$ is appropriate. Typical plots are shown in Figures 3.1 and 3.2. In practice, say for a microprocessor design, the value of K would probably have to depend on the instruction set and on the number of instructions that may be required to constitute meaningful processing tasks.

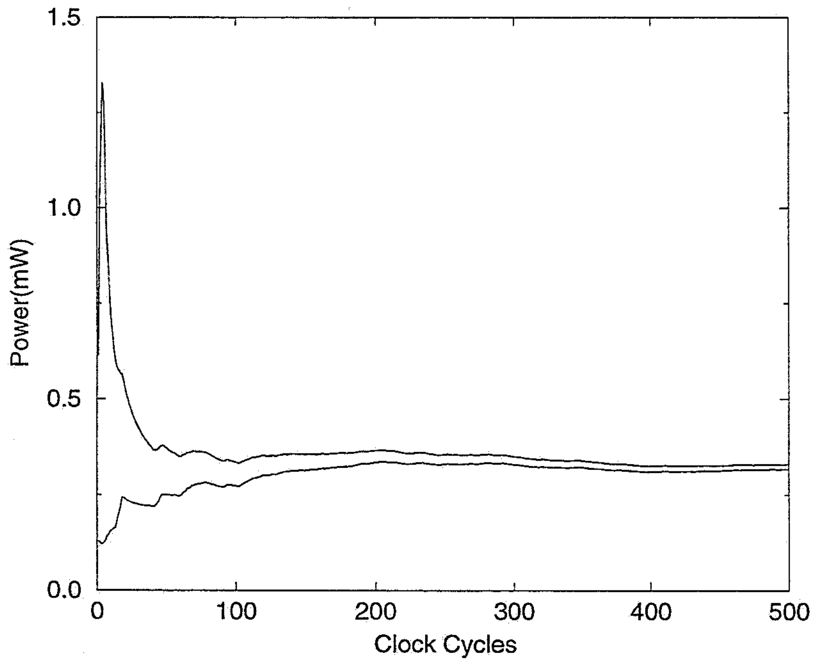

Figure 3.1 Upper and lower power bounds under a correlated vector set for circuit s1423 (74 latches and 657 gates).

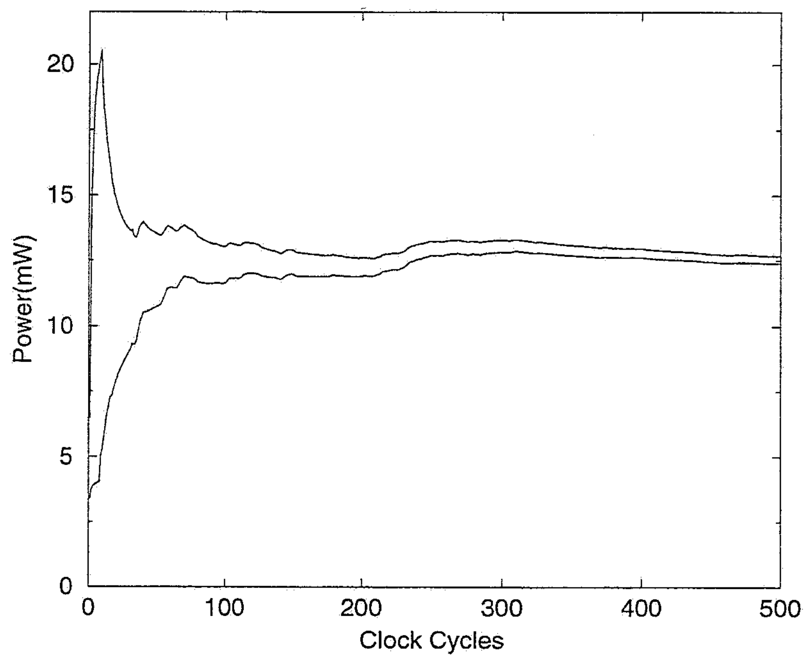

Figure 3.2 Upper and lower power bounds under a correlated vector set for circuit s35932 (1728 latches and 16,065 gates).

As for the choice of K', we use the following heuristic approach to determine when to stop the simulation within a block. Basically, we continue the simulation until either a user-specified tightness criteria is achieved or no further improvement can be obtained. For every simulation cycle, we compute the latest value of the slope of the power waveform–this is a waveform which is obtained by taking the average at every time point of the two power bound waveforms. When this slope becomes small (relative to a built-in threshold), then we check the *tightness* of the bounds.

The tightness is defined as the difference between the two block power bounds divided by their average. It can be shown that if the bounds on the power of every block satisfy a given tightness specification, then the two bounds on the total power in Equation (2.4) also satisfy the same tightness. If the tightness is less than a user-specified value, then we stop the simulation of that block. Otherwise, if the tightness is not small enough but remains constant (up to a certain tolerance) for a number (in our implementation,

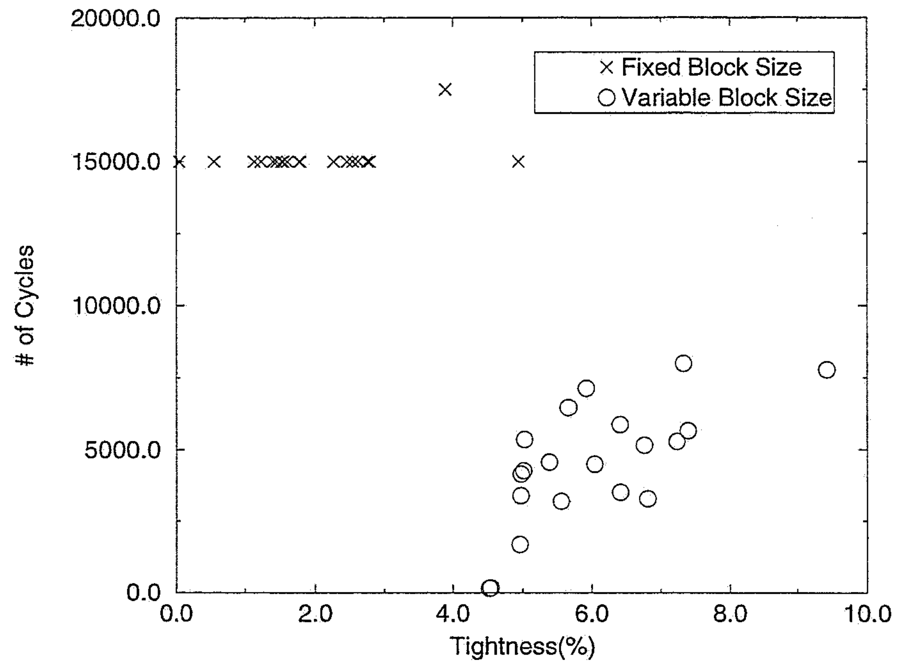

Figure 3.3 Comparison between the fixed and variable block size approaches.

10) of consecutive clock cycles, then we also stop the simulation because no further improvement can be achieved on the obtained results. Figure 3.3 shows a plot of the number of vectors simulated versus the tightness achieved for some benchmark circuits comparing the fixed block size method to the variable block size method. By fixed block size method, we mean having a constant block size for all blocks, i.e., each block consists of K cycles where K is a constant. By variable block size method, we mean implementing the heuristic apporach presented above, i.e. each block consists of K' cycles where K' varies from one block to the other.

It is clear from Figure 3.3 that the tightness obtained by the fixed block size method is better than that obtained by the variable block size method. However, the total number of vectors simulated by the variable block size method is much less while still resulting in acceptable tightness (relative to a user-specified criteria). Another advantage of the variable block size approach is that the user has the power to trade off tightness for speed based on the specific application. The higher the tightness threshold is, the more are the simulation cycles required and vice versa. In the limit, the user can achieve the same tightness as that obtained by the fixed block size method simply by setting the tightness threshold to zero.

3.3 Choice of Sample Size, N

The choice of sample size, N, affects the quality of the approximations in (3.1). It should be clear that the larger N is, the better the approximation, but how much should N be for a certain desired error tolerance? This is the classical problem of mean-estimation in statistics. We will briefly review the mean-estimation procedure with reference to an arbitrary random variable $\mathbf{x}$ whose mean $E[\mathbf{x}]$ is to be estimated from N sample values $x_1, \ldots, x_N$, using:

$$E[\mathbf{x}] \approx \mu_N = \frac{x_1 + \cdots + x_N}{N} \tag{3.6}$$

which is what is done in (3.1).

In order for the results below to be true, the values $x_1, \ldots, x_n$ should be observed values of *independent, identically distributed (iid)* random variables. To guarantee this, in our work, the start of a block is chosen completely at random every time, independently of all prior block positions, using a uniform random number generator that gives a value between $-K+2$ and M. With this, the value μ_N is a sample of a random variable, called the *sample mean* [22], whose mean is equal to $E[\mathbf{x}]$ and whose variance is equal to σ^2/N, where σ^2 is the variance of $\mathbf{x}$. If the sample values $x_1, \ldots, x_N$ are taken from a *normal* population having the mean $E[\mathbf{x}]$ and the variance σ^2, then $t = \frac{\mu_N - E[\mathbf{x}]}{\sigma/\sqrt{N}}$ is the value of a random variable having the t-distribution with $\nu = N-1$ degrees of freedom [13]. Consequently, with $(1-\alpha)$ confidence, it follows that [13]

$$-t_{\alpha/2} \leq \frac{\mu_N - E[\mathbf{x}]}{s_N/\sqrt{N}} \leq t_{\alpha/2} \tag{3.7}$$

where $0 < \alpha < 1$ and where $t_{\alpha/2}$ is defined so that the area to its right under the t-distribution curve is equal to $\alpha/2$. The value of $t_{\alpha/2}$ for a given α can be easily found using standard statistical tables. As for s_N, it is defined as the standard deviation of the observed N data values $x_1, \ldots, x_N$, measured as follows:

$$s_N^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_N)^2 \tag{3.8}$$

Therefore, with confidence $(1-\alpha)$, we have:

$$\frac{|\mu_N - E[\mathbf{x}]|}{\mu_N} \leq \frac{t_{\alpha/2}s_N}{\mu_N\sqrt{N}} \tag{3.9}$$

If ϵ_1 is a small positive number, and if N is large enough to achieve:

$$\frac{s_N}{\mu_N\sqrt{N}} \leq \frac{\epsilon_1}{t_{\alpha/2}} \tag{3.10}$$

then ϵ_1 places an upper bound on the relative error of the sample, with $(1-\alpha)$ confidence:

$$\frac{|\mu_N - E[\mathbf{x}]|}{\mu_N} \leq \frac{t_{\alpha/2}s_N}{\mu_N\sqrt{N}} \leq \epsilon_1 \tag{3.11}$$

This may also be expressed as the relative deviation from the mean $E[\mathbf{x}]$:

$$\frac{|\mu_N - E[\mathbf{x}]|}{E[\mathbf{x}]} \leq \frac{\epsilon_1}{1-\epsilon_1} = \epsilon \tag{3.12}$$

Here, $\epsilon > 0$ is defined as the user-specified *error tolerance*, and α (or $1-\alpha$) is the user-specified *confidence*. Thus Equation (3.10) provides a stopping criterion that determines when to stop sampling in order to yield the accuracy specified in (3.12) with confidence $(1-\alpha)$. Notice that the required number of samples N is not known a priori, but is determined only when (3.10) is first met.

We should note here that it is not always the case that the sample values, $x_1, \ldots, x_N$, come from a normal population. In order to account for the case when the distribution of the random variable $\mathbf{x}$ is not normal, we make the following observation. Based on the *Central Limit Theorem* [13], the distribution of the sample mean approaches the *normal distribution* for large N. The minimum number of samples, N, to satisfy near-normality is typically about 30 [13]. Thus, if we define an $\mathcal{N}$*-sample* as $y_k = \frac{x_{30(k-1)+1}+\cdots+x_{30k}}{30}$, where $x_{30(k-1)+1}, \ldots, x_{30k}$ are samples of a non-normal distribution, then y_k is a sample of a random variable $\mathbf{y_k}$ whose distribution is near-normal. Consequently, one option in handling non-normal populations is to take the sample mean of 30 *iid* samples of the non-normal population, and consider it as one $\mathcal{N}$-sample. Then, repeat the process to obtain as many $\mathcal{N}$-samples as needed. This way, it is guaranteed that the obtained $\mathcal{N}$-samples are samples of a near-normal distribution. Hence, we can apply the previous technique of stopping the simulation when the user-specified accuracy and error-criteria are satisfied with some approximation. Note here that a minimum of 60 samples of the non-normal distribution are needed for convergence. This is because a minimum of 2 $\mathcal{N}$-samples (or sample means) are needed to satisfy the accuracy and error criteria specified in (3.12).

To avoid the minimum requirement of 60 samples, the following approximation proves useful. For large sample sizes (N larger than 30 or so), one may approximate $\sigma/\sqrt{N}$ by $s_N/\sqrt{N}$ [13] where σ^2 is the variance of $\mathbf{x}$ and s_N is the standard deviation of the observed N data values $x_1, \ldots, x_N$, measured as given in (3.8). Because the distribution of the sample mean μ_N approaches the *normal* distribution for large N (30 or more), it follows that with $(1 - \alpha)$ confidence [13]

$$-z_{\alpha/2} \leq \frac{\mu_N - E[\mathbf{x}]}{\sigma/\sqrt{N}} \leq z_{\alpha/2} \tag{3.13}$$

where $0 < \alpha < 1$ and where $z_{\alpha/2}$ is defined so that the area to its right under the standard normal distribution curve is equal to $\alpha/2$. The value of $z_{\alpha/2}$ for a given α can be easily found using standard statistical tables. In other words, we simply take N samples from the given distribution (whether normal or not). Then we simply check if (3.13) satisfied, replacing $\sigma/\sqrt{N}$ by $s_N/\sqrt{N}$. Hence,

$$-z_{\alpha/2} \leq \frac{\mu_N - E[\mathbf{x}]}{s_N/\sqrt{N}} \leq z_{\alpha/2} \tag{3.14}$$

Therefore, with confidence $(1 - \alpha)$, we have:

$$\frac{|\mu_N - E[\mathbf{x}]|}{\mu_N} \leq \frac{z_{\alpha/2} s_N}{\mu_N \sqrt{N}} \tag{3.15}$$

If ϵ_1 is a small positive number, and if N is large enough to achieve:

$$\frac{s_N}{\mu_N \sqrt{N}} \leq \frac{\epsilon_1}{z_{\alpha/2}} \tag{3.16}$$

then ϵ_1 places an upper bound on the relative error of the sample, with $(1-\alpha)$ confidence:

$$\frac{|\mu_N - E[\mathrm{x}]|}{\mu_N} \leq \frac{z_{\alpha/2} s_N}{\mu_N \sqrt{N}} \leq \epsilon_1 \tag{3.17}$$

This may also be expressed as the relative deviation from the mean $E[\mathrm{x}]$:

$$\frac{|\mu_N - E[\mathrm{x}]|}{E[\mathrm{x}]} \leq \frac{\epsilon_1}{1-\epsilon_1} = \epsilon \tag{3.18}$$

Here, $\epsilon > 0$ is defined as the user-specified *error tolerance*, and α (or $1-\alpha$) is the user-specified *confidence.*

We implemented both techniques (the one based on the t-distribution, and the approximate method based on replacing $\sigma/\sqrt{N}$ by $s_N/\sqrt{N}$) and compared the results shown in Table 4.4 (Chapter 4). The results show that replacing $\sigma/\sqrt{N}$ by $s_N/\sqrt{N}$ is indeed a valid approximation because the error is minimal while the speed advantage is quite significant.

The above discussion is applicable for estimating both the mean upper and lower bounds on the power dissipation, $E[\mathbf{P}^{\mathbf{u}}_{\mathbf{K}}]$ and $E[\mathbf{P}^{\mathbf{l}}_{\mathbf{K}}]$. Because the number of iterations required to estimate $E[\mathbf{P}^{\mathbf{u}}_{\mathbf{K}}]$ may be different from that required for $E[\mathbf{P}^{\mathbf{l}}_{\mathbf{K}}]$, we continue the sampling until the condition in Equation (3.12) has been met for both means.

CHAPTER 4

RESULTS

The technique proposed above has been implemented and tested on a number of sequential benchmark circuits. All the results to be presented were performed with 5% error-tolerance ($\epsilon = 0.05$) and 95% confidence ($\alpha = 0.05$). All the circuits were derived from the ISCAS-89 benchmark circuits [23], after mapping them to a gate library with delay and capacitance values typical of 0.5 μ CMOS technology. An important issue for our technique is the initializability of circuits. A circuit is said to be functionally initializable if, once implemented, it can always be initialized to a definite state. On the other hand, a circuit is said to be logically initializable if, started from an all-X state (unknown initial state), it can be driven into a definite state using three-valued logic simulation. According to [24], almost all circuits that are functionally initializable are also logically initializable. Because practical circuits will always be functionally initializable, we have restricted our results to the subset of the ISCAS-89 circuits that are known to be logically initializable. Other than this, no special considerations were used in picking the circuits below. Only two circuits were shown in [24] to be functionally but not logically initializable and, on these two circuits, our method does not work very well. While more circuits may need to be tested, this may mean that the method is best suited to circuits that are known to be logically initializable.

Because no input vector sets are available for these benchmarks, we have tried to mimic the correlations that exist in real vector sets by generating a long correlated vector set consisting of 100,000 vectors. The correlation coefficients were changed arbitrarily every 10 vectors. Thus, the statistical properties of the vectors vary widely depending

Table 4.1 Performance under correlated input vectors while using the fixed block size approach. Execution time was measured on a SUN Sparc 5.

Ckt	LB (mW)	UB (mW)	Power (mW)	Tight (%)	#cycles	Time (sec)
s27	0.095782	0.096317	0.096050	0.56	15000	10.82
s298	0.259092	0.263725	0.261408	1.77	15000	61.35
s344	0.267126	0.271950	0.269538	1.79	15000	81.28
s349	0.265840	0.272470	0.269155	2.46	15000	78.19
s382	0.164030	0.166580	0.165305	1.54	15000	73.61
s386	0.267279	0.270303	0.268791	1.13	15000	115.48
s400	0.166057	0.170336	0.168197	2.54	15000	76.54
s444	0.168225	0.172092	0.170158	2.27	15000	80.44
s526	0.255717	0.262518	0.259117	2.62	15000	91.45
s526n	0.252428	0.259562	0.255995	2.79	15000	91.83
s641	0.292337	0.296596	0.294467	1.45	15000	263.63
s713	0.296606	0.300797	0.298702	1.40	15000	300.58
s820	0.281707	0.285168	0.283437	1.22	15000	243.13
s832	0.280086	0.284371	0.282228	1.52	15000	249.61
s1196	1.142727	1.143376	1.143052	0.06	15000	440.03
s1238	1.226477	1.227087	1.226782	0.05	15000	474.97
s1423	0.322120	0.334891	0.328505	3.89	17500	445.13
s1488	0.469119	0.482246	0.475683	2.76	15000	458.96
s1494	0.433075	0.455039	0.444057	4.95	15000	457.70
s35932	12.590623	12.794483	12.692553	1.61	15000	17673.51

on where they are in the 100,000 vector stream. For each circuit, we first estimated the power due to the whole 100,000 vectors by simulation, and then used our block sampling approach to estimate the power, with 5% error-tolerance ($\epsilon = 0.05$) and 95% confidence ($\alpha = 0.05$).

In the first set of experiments, with results shown in Table 4.1, we explored the tightness of the power bounds and the speed of convergence. For some details of these circuits (gate count, etc.), the reader is referred to Table 2.1 in Chapter 2.

Table 4.1 shows the results when the vector set was generated taking into account temporal as well as spatial correlations. Furthermore, these results are obtained by applying the fixed block size method. The table lists the power upper and lower bounds in mW, and the average of the two under the "Power" column. The tightness of the

Table 4.2 Performance under correlated input vectors while using the variable block size approach with a tightness threshold of 5%. Execution time was measured on a SUN Sparc 5.

Ckt	LB (mW)	UB (mW)	Power (mW)	Tight (%)	#cycles	Time (sec)
s27	0.094267	0.099063	0.096665	4.96	1689	3.45
s298	0.251627	0.267296	0.259461	6.04	4499	20.56
s344	0.264320	0.277944	0.271132	5.02	5354	30.33
s349	0.263815	0.279198	0.271506	5.67	6454	34.62
s382	0.164924	0.176549	0.170737	6.81	3287	18.37
s386	0.261433	0.274770	0.268102	4.97	3408	29.94
s400	0.165993	0.176987	0.171490	6.41	5873	31.77
s444	0.161148	0.172421	0.166785	6.76	5154	29.31
s526	0.256781	0.276077	0.266429	7.24	5283	34.97
s526n	0.245221	0.264067	0.254644	7.40	5655	37.13
s641	0.289601	0.304496	0.297049	5.01	4280	82.27
s713	0.293731	0.308735	0.301233	4.98	4162	92.12
s820	0.281246	0.297332	0.289289	5.56	3209	60.99
s832	0.274477	0.292660	0.283568	6.41	3521	67.23
s1196	1.138855	1.191813	1.165334	4.54	181	10.61
s1238	1.185894	1.240754	1.213324	4.52	164	10.44
s1423	0.325843	0.350668	0.338256	7.34	7991	215.63
s1488	0.465915	0.494339	0.480127	5.92	7123	232.17
s1494	0.432316	0.475055	0.453685	9.42	7773	246.95
s35932	12.151903	12.824729	12.488316	5.39	4573	5487.64

bounds was measured as the difference between them divided by their average, expressed as a percentage. The values illustrate that the bounds can be quite tight in most cases. The table also lists the number of cycles (i.e., vectors) that were required for convergence and the CPU time required. In many cases, the number of cycles was equal to the minimum allowable (minimum of 30 blocks, at 500 vectors per block) for the fixed block size approach as explained in Chapter 3. In some cases, however, more cycles were required. This illustrates the importance of having a convergence check (a stopping criterion). It would not be sufficient, for instance, to simulate all circuits for the same number of vectors. It is also notable that the required number of cycles is not necessarily larger for larger designs.

Table 4.3 Error between simulation of all 100,000 correlated vectors and using the Block Sampling (BS) scheme (applied with variable block size).

Ckt	Power (AV)	Power (BS)	Error (%)	Compaction
s27	0.0958	0.0967	−0.89	0.01689
s298	0.2615	0.2595	0.79	0.04499
s344	0.2657	0.2711	−2.03	0.05354
s349	0.2690	0.2715	−0.94	0.06454
s382	0.1666	0.1707	−2.47	0.03287
s386	0.2721	0.2681	1.46	0.03408
s400	0.1667	0.1715	−2.90	0.05873
s444	0.1682	0.1668	0.81	0.05154
s526	0.2573	0.2664	−3.55	0.05283
s526n	0.2549	0.2546	0.09	0.05655
s641	0.2940	0.2970	−1.03	0.04280
s713	0.2990	0.3012	−0.76	0.04162
s820	0.2833	0.2893	−2.10	0.03209
s832	0.2804	0.2836	−1.12	0.03521
s1196	1.1525	1.1653	−1.12	0.00181
s1238	1.2315	1.2133	1.48	0.00164
s1423	0.3296	0.3383	−2.62	0.07991
s1488	0.4613	0.4801	−4.08	0.07123
s1494	0.4448	0.4537	−2.00	0.07773
s35932	12.7332	12.4883	1.92	0.04573

Then we verified that the variable block size approach provides better results than the fixed block size approach in terms of speed while maintaining an acceptable tightness level. The results are shown in Table 4.2.

Then, the power estimated by our block sampling approach (average of the two bounds) was compared to that computed by simulation of the whole 100,000 vector set. The results are shown in Table 4.3 where the power values are expressed in mW. In this case, the variable block size approach was used. It is clear that the errors are very small and that all are below the specified 5% error tolerance.

Table 4.3 also includes a column named "Compaction." This is the ratio of the total number of vectors simulated by the block sampling method to the total number of vectors

(100,000) in the power vector set. In many cases, it turns out to be enough to simulate only around 5% of the total vector set. In some cases, this ratio is larger, up to 8% in some cases. Thus, the net effect is that the power is estimated by simulating only a small fraction of the total vector set. This feature is essential for simulation of large sequential circuits. Effectively, an *implicit compaction* of the vector set has been achieved. The adjective "implicit" denotes the fact that this was done on the fly, during the simulation, rather than up-front. We feel that this is the only correct way of performing compaction, mainly because, as observed in relation to Tables 4.1 and 4.2, the number of vectors required for convergence depends very much on the special characteristics of the circuit and is not determined simply by signal statistics or by circuit size. In fact, as was pointed out above, sometimes smaller circuits will require more cycles to converge.

Looking at the last column of Table 4.3, some readers may conclude that perhaps simply simulating the first 3% (or whatever the fraction may be, according to the compaction ratio) of the vectors in the long vector set, in the order in which they occur, may be enough. This is not correct because the first section of the vector set may be biased for some reason–it may, for instance, have much lower switching activity than the rest of the vector set. The random choice of the blocks from anywhere in the vector set is essential in order to guarantee that the result is representative of all the various modes of operation in the long vector set. Granted, the random sampling does not explore *all* the vectors, but it does explore enough of them, and in the right way, in order to provide the desired result with the specified accuracy and confidence.

One further set of experiments was performed to verify that approximating $\sigma/\sqrt{N}$ by $s_N/\sqrt{N}$ is valid when $N \geq 30$. We simulated the circuits using the exact approach in which we consider every 30 samples as one $\mathcal{N}$*-sample* and then check the convergence criteria on the obtained $\mathcal{N}$-samples. The results are shown in Table 4.4 which shows the lower bound, upper bound, and average power, all in mW. It also shows the tightness and the number of cycles required for convergence, as well as the error caused by the approximation. Notice that the error is less than 3% in most cases while achieving a considerable speed advantage. The reason is that by using the approximation, we

Table 4.4 Results obtained by using the exact approach as well as the error resulting from using the approximation. Power values (LB, UB, and Power) are all given in mW.

Ckt	LB	UB	Power	Tight (%)	#cycles	Time (sec)	Error (%)
s27	0.0929	0.0976	0.0952	4.96	3296	4.41	−1.49
s298	0.2541	0.2713	0.2627	6.53	10584	48.89	1.23
s344	0.2636	0.2776	0.2706	5.17	11759	63.80	−0.19
s349	0.2618	0.2772	0.2695	5.72	14537	78.20	−0.74
s382	0.1634	0.1736	0.1685	6.03	7752	41.43	−1.32
s386	0.2626	0.2761	0.2693	4.98	7712	67.40	0.46
s400	0.1645	0.1747	0.1696	6.00	11134	61.88	−1.13
s444	0.1651	0.1759	0.1705	6.34	10332	60.24	2.18
s526	0.2575	0.2784	0.2679	7.79	11294	78.54	0.56
s526n	0.2517	0.2737	0.2627	8.35	17403	118.48	3.07
s641	0.2871	0.3019	0.2945	5.01	8012	154.98	−0.87
s713	0.2930	0.3081	0.3006	5.02	8746	181.69	−0.21
s820	0.2817	0.2984	0.2900	5.78	6330	119.04	0.26
s832	0.2751	0.2941	0.2846	6.68	9169	172.78	0.37
s1196	1.1226	1.1754	1.1490	4.60	390	17.84	−1.42
s1238	1.1919	1.2482	1.2200	4.61	403	18.54	0.55
s1423	0.3222	0.345946	0.3341	7.10	16586	444.89	−1.25
s1488	0.4528	0.4875	0.4702	7.38	15106	497.51	−2.12
s1494	0.4329	0.4792	0.4560	10.15	17696	580.43	0.51
s35932	12.2416	12.9049	12.5733	5.27	10541	12828.20	0.68

get rid of the minimum 60 samples requirement enforced by the exact method. Thus fewer samples, and consequently fewer cycles, are required for convergence while still maintaining acceptable accuracy.

CHAPTER 5

CONCLUSIONS

We have proposed a simulation-based method for estimating the power dissipation of sequential circuits. The method works by sampling blocks of consecutive vectors from a user-supplied (potentially very long) power vector set and simulating them. Because the state of the circuit at the beginning of each block is unknown, we put the circuit in an all-X state and simulate it for one block using three-valued logic simulation. The simulator includes delay information, so that it does capture glitching activity. During the simulation of each block, we compute an upper bound and a lower bound on the power due to that block, which we found can be very tight (close together). The blocks are selected at random from anywhere in the power vector set, and the process is repeated until, using statistical methods of mean estimation, the algorithm determines that the upper and lower bounds on the power have been determined with sufficient (user-specified) accuracy and confidence.

The major advantage of the method is that the state of the sequential circuit is always guaranteed to be valid–the FSM never goes outside its valid state space. Thus, the estimated power corresponds to realistic typical circuit operation. The problem with all previous power estimation methods is that they can take the machine to illegal states, so that the power estimated may have no relation to the power under realistic conditions. Another advantage of the method is that only a fraction of the vectors in the (potentially huge) power vector set need to be simulated. For the 100,000 vector sets that we considered, this fraction varied between 5% and 8%.

REFERENCES

[1] M. Pedram, "Power minimization in IC design: principles and applications," *ACM Transactions on Design Automation of Electronic Systems*, vol. 1, pp. 3–56, January 1996.

[2] F. Najm, "A survey of power estimation techniques in VLSI circuits," *IEEE Transactions on VLSI Systems*, vol. 2, no. 4, pp. 446–455, December 1994.

[3] M. A. Cirit, "Estimating dynamic power consumption of CMOS circuits," in *IEEE International Conference on Computer-Aided Design*, 1987, pp. 534–537.

[4] F. Najm, R. Burch, P. Yang, and I. Hajj, "Probabilistic simulation for reliability analysis of CMOS VLSI circuits," *IEEE Transactions on Computer-Aided Design*, vol. 9, no. 4, pp. 439–450, April 1990 (Errata in July 1990).

[5] F. Najm, "Transition density: a new measure of activity in digital circuits," *IEEE Transactions on Computer-Aided Design*, vol. 12, no. 2, pp. 310–323, February 1993.

[6] F. Najm, "Low-pass filter for computing the transition density in digital circuits," *IEEE Transactions on Computer-Aided Design*, vol. 13, no. 9, pp. 1123–1131, September 1994.

[7] A. Ghosh, S. Devadas, K. Keutzer, and J. White, "Estimation of average switching activity in combinational and sequential circuits," in *ACM/IEEE Design Automation Conference*, 1992, pp. 253–259.

[8] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," in *IEEE Transactions on Computer-Aided Design*, pp. 677–691, August 1986.

[9] R. Marculescu, D. Marculescu, and M. Pedram, "Efficient power estimation for highly correlated input streams," in *32nd Design Automation Conference*, 1995, pp. 628–634.

[10] C. M. Huizer, "Power dissipation analysis of CMOS VLSI circuits by means of switch-level simulation," in *IEEE European Solid State Circuits Conference*, 1990, pp. 61–64.

[11] R. Burch, F. Najm, P. Yang, and T. Trick, "A Monte Carlo approach for power estimation," *IEEE Transactions on VLSI systems*, vol. 1, no. 1, pp. 63–71, March 1993.

[12] M. Xakellis and F. Najm, "Statistical estimation of the switching activity in digital circuits," in *ACM/IEEE Design Automation Conference*, 1994, pp. 728–733.

[13] I. R. Miller, J. E. Freund, and R. Johnson, *Probability and Statistics for Engineers*, 4th edition. Englewood Cliffs, NJ: Prentice-Hall Inc., 1990, pp. 210–211.

[14] A. A. Ismaeel and M. A. Breuer, "The probability of error detection in sequential circuits using random test vectors," *Journal of Electronic Testing*, vol. 1, pp. 245–256, January 1991.

[15] G. D. Hachtel, E. Macii, A. Pardo, and F. Somenzi, "Probabilistic analysis of large finite state machines," in *31st ACM/IEEE Design Automation Conference*, 1994, pp. 270–275.

[16] J. Monteiro and S. Devadas, "A methodology for efficient estimation of switching activity in sequential logic circuits," in *ACM/IEEE 31st Design Automation Conference*, 1994, pp. 12–17.

[17] C-Y Tsui, M. Pedram, and A. M. Despain, "Exact and approximate methods for calculating signal and transition probabilities in FSMs," in *ACM/IEEE 31st Design Automation Conference*, 1994, pp. 18–23.

[18] F. Najm, S. Goel, and I. Hajj, "Power estimation in sequential circuits," in *32nd ACM/IEEE Design Automation Conference*, 1995, pp. 635–640.

[19] T-L. Chou and K. Roy, "Statistical estimation of sequential circuit activity," in *IEEE/ACM International Conference on Computer-Aided Design*, 1995, pp. 34–37.

[20] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, 2nd edition. New York, NY: McGraw-Hill Book Co., 1984.

[21] D. Brand and C. Visweswariah, "Inaccuracies in power estimation during logic synthesis," in *IEEE/ACM International Conference on Computer-Aided Design*, 1996, pp. 388–394.

[22] M. H. DeGroot, *Probability and Statistics*, 2nd edition. Reading, MA: Addison-Wesley, 1986.

[23] F. Brglez, D. Bryan, and K. Koźmiński, "Combinational profiles of sequential benchmark circuits," in *IEEE International Symposium on Circuits and Systems*, 1989, pp. 1929–1934.

[24] J. Wehbeh, D. G. Saab, "On the Initialization of Sequential Circuits," in *IEEE International Test Conference*, 1994, pp. 233–239.