12-1-1996

Accurate and Efficient Technique to Calculate Sensitivities of' Power to Primary Inputs

Zhanping Chen
Purdue University School of Electrical and Computer Engineering

Kaushik Roy
Purdue University School of Electrical and Computer Engineering

Tanli Chou
Intel Corp.

ACCURATE AND EFFICIENT TECHNIQUE TO CALCULATE SENSITIVITIES OF POWER TO PRIMARY INPUTS

ZHANPING CHEN
KAUSHIK ROY
TANLI CHOU

TR-ECE 96-20
DECEMBER 1996

SCHOOL OF ELECTRICAL
AND COMPUTER ENGINEERING
PURDUE UNIVERSITY
WEST LAFAYETTE, INDIANA 47907-1285

Accurate and Efficient Technique to Calculate Sensitivities of Power to Primary Inputs

Zhanping Chen, and Kaushik Roy
Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285

Tanli Chou
Intel Corp.
Beaverton, OR 97006

Contact person: Kaushik Roy
Ph: 317-494-2361
Fax: 317-494-3371
e-mail: kaushik@ecn.purdue.edu

Accurate and Efficient Technique to Calculate Sensitivities of Power to Primary Inputs

Abstract

Recently, several power estimation techniques have been proposed. Some of them can be used to accurately estimate power dissipation provided that the exact signal probability (probability of a signal being logic ONE) and signal activity (probability of signal switching) of primary inputs are known. In general, very accurate specification of primary input signal probability and activity may not be available. This in turn may result in uncertainties in average power estimation. In this paper we present a novel and efficient technique to estimate the sensitivity of average power dissipation to input signals using a symbolic estimation technique. Results for benchmark circuits show that power sensitivities can vary widely for different primary inputs of a circuit. Hence, in order to accurately estimate average power dissipation, the sensitive inputs of a circuit have to be specified accurately. We have also developed a Monte-Carlo based technique to estimate power sensitivity which also acts as a figure of merit for the symbolic technique.

1 Introduction

With the growing use of portable and wireless electronic systems, it has become important to consider power dissipation for low power circuit design [1]. Deep submicron processes are pushing higher levels of integration. This results in even higher power density. For example, Motorola's PowerPC consumes 8.5w, Intel's Pentium consumes 16w, and DEC's Alpha chip consumes 30w [10]. In order to determine the bottleneck for power and to synthesis circuits satisfying the power requirements, accurate estimation of power dissipation is needed.

For a CMOS digital circuit, there are three sources of power dissipation: switching current, short-circuit current, and leakage current. Among the three sources of power dissipation, switching current which charges or discharges load capacitances of logic gates contributes to a majority of the power dissipation. Therefore, accurate estimation of power dissipation for CMOS circuits requires accurate estimation of signal switching activity at the internal and the output nodes.

Recently, several probabilistic and statistical techniques have been proposed to estimate power dissipation [2, 3, 4, 5, 6, 7, 8]. In the probabilistic approach, the signals are usually modeled as stochastic processes where each signal has a signal probability and an activity. Signal probability is defined as the probability of a signal being logic ONE while signal activity is defined as the probability of signal switching. A single symbolic simulation run determines the internal node signal probability and activity by propagating the input probability information through the circuit. The statistical approach, on the other hand, simulates the circuit with a limited number of randomly generated vectors conforming to the given signal probability and activity of the primary inputs. The number of simulations for combinational circuits are determined by user-specified parameters such as confidence levels and errors that can be tolerated. No matter which approach one uses, the spatio-temporal correlations among signals have to be considered to increase the accuracy of the estimate;. In [2], Najm defined a new measure of circuit activity called the transition density. He also presented an algorithm to propagate the density from the primary inputs to the internal nodes. However, the algorithm does not consider simultaneous switching of multiple primary inputs. Hence, activity of a node is overestimated. In [4], Ghosl et al. presented a method that deals with simultaneous switching of multiple primary inputs but neglected temporal correlation between primary inputs. In [3], Chou et al., and in [7], Tsui et al. presented exact methods to calculate signal probability and activity.

The above methods can be used to accurately estimate the power dissipation of a circuit provided that the exact probability and activity of each primary input is known. However, accurate signal probability or activity values for the primary inputs often may not be available. Rather, one may have ranges over which the probability and the activity values may vary. I-Ience, there can be uncertainties in the specification of primary input probabilities and activities. Since power dissipation depends on the input signal distribution, uncertainties in primary input specification can cause uncertainties in power dissipation estimation. This leads to the question: *how much effect does an uncertain primary input specijication have on total power dissipation?* Intuitively, the activity at a node can be very sensitive to the activity of some primary inputs of the circuit. As a result, a minor activity variation of those primary inputs can lead to large variation in estimated activity, which causes power dissipation to deviate widely. Therefore, for power conscious designs, either the sensitive inputs should be accurately specified or design techniques should consider such sensitivities during synthesis.

In this paper, we first define a measure of power sensitivity to primary inputs and then present novel algorithms to estimate such sensitivities in large CMOS circuits. The rest of the paper is organized as Follows. Section 2 introduces the formal definition of power sensitivity to primary input signal probability and activity. A symbolic technique to compute power sensitivity is discussed in section 3. Section 4 describes a Monte Carlo based technique to estimate sensitivities and can be used as a figure of merit for the symbolic technique. Section 5 presents the implementation details and experimental results on ISCAS and MCNC benchmark circuits. Finally, conclusions are given in Section 6.

2 Definition of Power Sensitivity

2.1 Signal Probability and Activity

Given a logic signal $x(t)$ and a random variable τ, the companion process of $x(t)$ is defined as $\mathbf{x}(t) = x(t + \tau)$, where τ is uniformly distributed over $\Re$ (the set of real numbers). The bold font is used to represent a stochastic process. The primary inputs of a circuit are modeled as mutually independent companion processes of logic signals. It has been shown [2] that the probability of a companion process of a logic signal $x(t)$ assuming the logic value ONE at any given time t ($\lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} \mathbf{x}(t)dt$) approaches a constant and is called the equilibrium probability. This is denoted by $P(x)$. The signal probability is defined as the number of clock cycles in which the signal is steady state ONE divided by the total clock number of cycles. Note that steady state signals are only considered in signal probability estimation and any spurious transitions are ignored. Najm [2] has also shown that the activity $A(x)$ (average number of switching events per unit time) (defined as $\lim_{T\to\infty} \frac{n_x(T)}{T}$) is equal to the expected value of $\frac{n_x(T)}{T}$ (mean-ergodic). The variable n_x is the number of switching for $x(t)$ in the time interval $(-T/2, T/2]$.

If we assume that all primary inputs to the circuits under consideration switch only at the leading edge of the clock and that the circuits are delay-free, we can define normalized activity. Normalized activity, denoted by $a(x)$, is defined as $A(x)/f$. $A(x)$ and f are the activity at node x and clock frequency, respectively.

2.2 Power Dissipation in CMOS Logic Circuits

In CMOS circuits, power dissipation is mainly due to three sources – switching current, short-circuit current, and leakage current. Switching power is by far the most dominant. Thus the average power for a CMOS circuit can be approximated by

$$Power_{avg} = \frac{1}{2}V_{dd}^2 \sum_{j \in allnodes} C_j\, A_j \tag{1}$$

where V_{dd} is the supply voltage, A_j is the activity at node j, and C_j is the node capacitance. For each node of a CMOS circuit, A_j is proportional to the normalized activity a_j. The node capacitance C_j is approximately proportional to the fanout at node j. So we can define the *normalized power dissipation measure* Φ *as:*

$$\Phi = \sum_{j \in allnodes} f_{anout}(j)\, a_j \tag{2}$$

where $f_{anout}(j)$ is the number of fanouts at node j. It should be noted that for a CMOS circuit Φ is proportional to the average power dissipation .

2.3 Power Sensitivity

The techniques to estimate power dissipation include probabilistic and statistical methods [10]. The problem involved in using such techniques is that designers require accurate probabilities and activities at the primary inputs. In general, it is very difficult to obtain the exact *primary input distribution.* Both primary input probability variation and activity variation can result in uncertainties in the estimation of power dissipation for a CMOS circuit. To measure this effect we define *power sensitivity to primary input activity,* denoted by S_{a_i}, as

$$S_{a_i} = \lim_{\Delta a_i \to 0} \frac{\Delta Power_{avg}}{\Delta a_i} \tag{3}$$

where a_i is the activity of the i^{th} primary input.

Similarly, we define *power sensitivity to primary input probability,* denoted by S_{P_i}, as

$$S_{P_i} = \lim_{\Delta P_i \to 0} \frac{\Delta Power_{avg}}{\Delta P_i} \tag{4}$$

where P_i is the probability of the i^{th} primary input.

If we can. express the activity and probability of each internal and primary output node in terms of the activity and probability of the primary input nodes, then we can express S_{a_i} and S_{P_i} as follows.

$$S_{a_i} = \frac{\partial Power_{avg}}{\partial a_i} \tag{5}$$

$$S_{P_i} = \frac{\partial Power_{avg}}{\partial P_i} \tag{6}$$

Since $Power_{avg}$ is proportional to the *normalized power dissipation measure* Φ, we can define *normalized power sensitivity to primary input activity* ζ_{a_i} and *normalized power sensitivity to primary input probability* ζ_{P_i} in terms of Φ as follows:

$$\zeta_{a_i} = \frac{\partial \Phi}{\partial a_i} = \sum_{j \in allnodes} f_{anin}(j)\, \frac{da_j}{\partial a_i} \tag{7}$$

while the power sensitivity ζ_{P_i} is equal to

$$\zeta_{P_i} = \frac{\partial \Phi}{\partial P_i} = \sum_{j \in allnodes} f_{anin}(j) \frac{\partial a_j}{\partial P_i} \tag{8}$$

where a_j is the activity of each internal node and primary output and a_i is the activity of each primary input. We define $\partial a_j / \partial a_i$ as *activity sensitivity to primary input activity,* and $\partial a_j / \partial P_i$ as *activity sensitivity to primary input probability.*

3 Symbolic Method to Compute Power Sensitivity

The basic idea of the symbolic method for computing power sensitivity is to express the probability and activity of each internal node in terms of the probability and activity of primary inputs so that spatial correlation between internal nodes can be handled. First, we will describe how to compute internal signal probabilities and activities. Next, we will show mathematically how to compute power sensitivities. We will then propose a circuit partitioning scheme for large circuits. Finally, we will present an algorithm for sensitivity computation.

3.1 Signal Probability Calculation

A Boolean function f, representing an internal or an output node of a logic circuit, can always be written as a canonical sum of products of primary inputs which are assumed to be independent. A method to compute signal probability has been introduced in [11]. Boolean conjunction (AND) f of two inputs I_1 and I_2 (primary inputs or internal nodes) has probability $P(f) = P(I_1 I_2)$. Boolean disjunction (OR) f has the probability $P(f) = P(I_1) + P(I_2) - P(I_1 I_2)$. Starting from the primary inputs of a circuit, we can obtain the probability expression for each internal node and primary output in terms of primary inputs after exponent suppression [11, 12] of the conjunction probability terms $P(I_1 I_2)$. Therefore, signal probability $P(f)$ can be expressed as a sum of *primary input signal probability product terms* $\sum_{i=1}^{p} \alpha_i (\prod_{k=1}^{n} P^{m_{i,k}}(x_k))$, where n is the number of the independent inputs to the circuit,, and α_i is some integer. The exponent $m_{i,k}$ is either 1 or 0. The sum has p product terms. The following example shows this procedure.

Example *1 Given $y = x_1 x_2 + x_1 x_3$, where x_i, $i = 1, 2, 3$, are mutually independent. Then $P(y)$ can be determined as follows:*

$$P(y) = P(x_1 x_2) + P(x_1 x_3) - P(x_1 x_2) P(x_1 x_3)$$

But we know that

$$P(x_1 x_2) = P(x_1) P(x_2)$$

$$P(x_1 x_3) = P(x_1) P(x_3)$$

$P(x_1)$ will be suppressed to power one instead of two. Therefore,

$$P(y) = P(x_1) P(x_2) + P(x_1) P(x_3) - P(x_1) P(x_2) P(x_3)$$

For convenience, in this paper $\bar{P}(x_i)$ is defined as $P(\bar{x}_i)$ and equals to $1 - P(x_i)$. Therefore, $P(f)$ can be expressed as $\sum_{i=1}^{p} \alpha_i (\prod_{k=1}^{n} P^{m_{i,k}}(x_k)\bar{P}^{l_{i,k}}(x_k))$, where $m_{i,k}$ and $l_{i,k}$ are either 1 or 0 but both cannot be 1 simultaneously. Since

$$P(x_k)\bar{P}(x_k) = P(x_k)(1 - P(x_k)) = P(x_k) - P^2(x_k)$$

and equals 0 after exponent suppression, the product term $P^{m_{i,k}}(x_k)\bar{P}^{l_{i,k}}(x_k)$ will be eliminated from $P(f)$ if $m_{i,k} = l_{i,k} = 1$. Thus in the previous example,

$$P(y) = P(x_1)P(x_2)\overline{P}(x_3) + P(x_1)P(x_3)$$

3.2 Activity Calculation

Considering temporal correlation of signals between time instants t and $t - T$, where T is the clock period, $x_j(t-T)x_j(t) = 1$ and $\bar{x}_j(t-T)\bar{x}_j(t) = 1$ signify that x_j does not switch at time t. Thus x_j remains ONE or ZERO, respectively. Similarly, $x_j(t-T)\bar{x}_j(t) = 1$ and $\bar{x}_j(t-T)x_j(t) = 1$ signify that x_j has a transition from ONE to ZERO and from ZERO to ONE respectively. Therefore, the following equations hold,

$$P(x_j(t-T)x_j(t) + \bar{x}_j(t-T)\bar{x}_j(t)) = P(x_j(t-T)x_j(t)) + P(\bar{x}_j(t-T)\bar{x}_j(t)) = 1 - a(x_j)$$

$$P(x_j(t-T)\bar{x}_j(t) + \bar{x}_j(t-T)x_j(t)) = P(x_j(t-T)\bar{x}_j(t)) + P(\bar{x}_j(t-T)x_j(t)) = a(x_j)$$

$$P(x_j(t-T)) = P(x_j(t-T)x_j(t)) + P(x_j(t-T)\bar{x}_j(t)) \tag{9}$$

$$P(x_j(t)) = P(x_j(t-T)x_j(t)) + P(\bar{x}_j(t-T)x_j(t)) \tag{10}$$

However, since $P(x_j(t))$ is a constant with respect to time,

$$P(x_j(t-T)) = P(x_j(t)) = P(x_j) \tag{11}$$

$$P(\bar{x}_j(t-T)) = P(\bar{x}_j(t)) = P(\bar{x}_j) = 1 - P(x_j) \tag{12}$$

Therefore, equating equations 9 and 10 we obtain

$$P(x_j(t-T)\bar{x}_j(t)) = P(\bar{x}_j(t-T)x_j(t)) = \frac{a(x_j)}{2} \tag{13}$$

In fact, one can notice that each transition from ONE to ZERO will be followed by a transition from ZERO to ONE. Since $x_j(t-T)x_j(t) + \bar{x}_j(t-T)x_j(t) = x_j(t)$, it follows that $P(x_j) = P(x_j(t)) = P(x_j(t-T)x_j(t) + \bar{x}_j(t-T)x_j(t)) = P(x_j(t-T)x_j(t)) + P(\bar{x}_j(t-T)x_j(t))$ Hence,

$$P(x_j(t-T)x_j(t)) = P(x_j) - \frac{a(x_j)}{2} \tag{14}$$

and similarly

$$P(\bar{x}_j(t-T)\bar{x}_j(t)) = 1 - P(x_j) - \frac{a(x_j)}{2} \tag{15}$$

Therefore, we can get the activity of node y as follows

$$a(y) = 2(P(y) - P(y(t-T)y(t))) \tag{16}$$

To compute the activity $a(y)$ from the pre-computed signal probability $P(y)$ we must first calculate $P(y(t-T)y(t))$ which can be calculated as follows. From section *3.1,* we know that $P(y)$ can be expressed as a *sum of probability products*, $\sum_{i=1}^{p} \alpha_i (\prod_{k=1}^{n} P^{m_{i,k}}(x_k)\bar{P}^{l_{i,k}}(x_k))$, where n is the number of the independent inputs to the circuit, and a_i is some integer. Let

$$U(y(t)) = \sum_{j=1}^{p} \alpha_j (\prod_{k=1}^{n} P^{m_{j,k}}(x_k(t))\bar{P}^{l_{j,k}}(x_k(t))). \text{ Therefore,} \tag{17}$$

$$U(y(t-T)) = \sum_{i=1}^{p} \alpha_i (\prod_{k=1}^{n} P^{m_{i,k}}(x_k(t-T))\bar{P}^{l_{i,k}}(x_k(t-T))). \tag{18}$$

$U(y(t))$ and $U(y(t-T))$ represent the sums of probability products of $y(t)$ and $y(t-T)$. Hence, we have,

$$\begin{aligned} U(y(t-T))U(y(t)) &= (\sum_{i=1}^{p} \alpha_i (\prod_{k=1}^{n} P^{m_{i,k}}(x_k(t-T))\bar{P}^{l_{i,k}}(x_k(t-T)))) \\ &\quad (\sum_{j=1}^{p} \alpha_j (\prod_{k=1}^{n} P^{m_{j,k}}(x_k(t))\bar{P}^{l_{j,k}}(x_k(t)))) \\ &= \sum_{1 \le i,j \le p} \alpha_i \alpha_j (\prod_{k=1}^{n} P^{m_{i,k}}(x_k(t-T))\bar{P}^{l_{i,k}}(x_k(t-T)) \\ &\quad P^{m_{j,k}}(x_k(t))\bar{P}^{l_{j,k}}(x_k(t))). \end{aligned} \tag{19}$$

In *[3]* it has been shown that if
$P(x_k(t-T))P(x_k(t))$ is replaced with $P(x_k(t-T)x_k(t))$,
$P(x_k(t-T))\bar{P}(x_k(t))$ with $P(x_k(t-T)\bar{x}_k(t))$,
$\bar{P}(x_k(t-T))P(x_k(t))$ with $P(\bar{x}_k(t-T)x_k(t))$, and
$\bar{P}(x_k(t-T))\bar{P}(x_k(t))$ with $P(\bar{x}_k(t-T)\bar{x}_k(t))$
then equation *19* becomes

$$U(y(t-T))U(y(t)) = \sum_{1 \le i,j \le p} \alpha_i \alpha_j (\prod_{k=1}^{n} P(x_k^{m_{i,k}}(t-T)\bar{x}_k^{l_{i,k}}(t-T))P(x_k^{m_{j,k}}(t)\bar{x}_k^{l_{j,k}}(t))) \tag{20}$$

and is equal to $P(y(t-T)y(t))$

By substituting equations 9, *10, 11, 12,* and *13* into equation **20** and then substituting equation *16* into equation *16,* we can obtain the exact expression for signal activity. The following example illustrates the procedure to express the activity of a node in terms of the probabilities and activities of primary inputs.

Example 2 Let $y = x_1 + x_2$. *Assume* x_1 *and* x_2 *are independent inputs. Therefore,* $P(y) = P(x_1) + \overline{P}(x_1)P(x_2)$, *where* $\overline{P}(x_1) = P(\overline{x_1}) = 1 - P(x_1)$. *Calculate* $a(y)$.

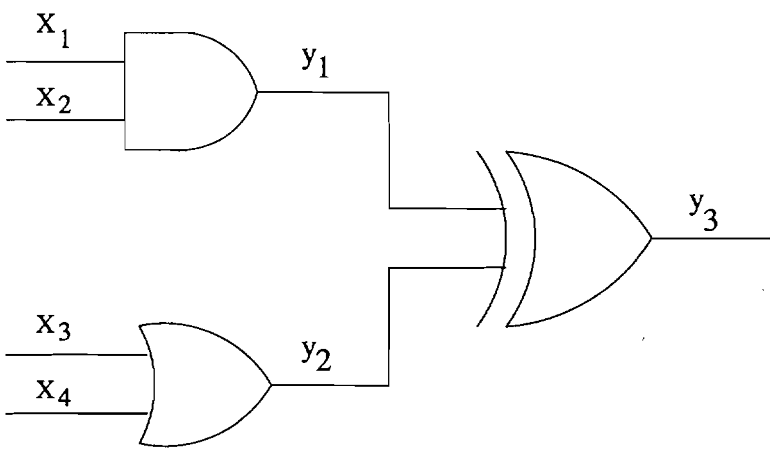

Figure 1: A circuit for example

$$
\begin{aligned}
U(y(t-T))U(y(t)) &= (P(x_1(t-T)) + \overline{P}(x_1(t-T))P(x_2(t-T))) \\
&\quad (P(x_1(t)) + \overline{P}(x_1(t))P(x_2(t))) \\
&= P(x_1(t-T))P(x_1(t)) + (P(x_1(t-T))\overline{P}(x_1(t))P(x_2(t)) + \\
&\quad \overline{P}(x_1(t-T))P(x_1(t))P(x_2(t-T)) + \\
&\quad \overline{P}(x_1(t-T))\ \overline{P}(x_1(t))P(x_2(t-T))P(x_2(t)))
\end{aligned}
$$

$$
\begin{aligned}
P(y(t-T)y(t)) &= P(x_1(t-T)x_1(t)) + (P(x_1(t-T)\bar{x}_1(t))P(x_2(t)) + \\
&\quad P(\bar{x}_1(t-T)x_1(t))P(x_2(t-T)) + \\
&\quad P(\bar{x}_1(t-T))\bar{x}_1(t))P(x_2(t-T)x_2(t))) \\
&= (P(x_1) - \frac{1}{2}a(x_1)) + \frac{1}{2}a(x_1)P(x_2) + \frac{1}{2}a(x_1)P(x_2) \\
&\quad +(1-P(x_1) - \frac{1}{2}a(x_1))(P(x_2) - \frac{1}{2}a(x_2)).
\end{aligned}
$$

After rearranging and simplifying the terms, we have

$$
a(y) = (1-P(x_1))a(x_2) + (1-P(x_2))a(x_1) - \frac{1}{2}a(x_1)a(x_2)
$$

3.3 Power Sensitivity Calculation

Using the method outlined in the previous section, we can exactly express activity of a node in terms of the probability and activity of primary inputs. By using equations 7 and 8, we can obtain the power sensitivity to primary input probability and activity.

Let us consider an example to see how this symbolic method works. An example circuit is shown in Figure 1. In this circuit, node y_1 and y_2 are spatially correlated.

First, we write the function of each internal node and primary output in terms of primary inputs.

$$
y_1 = x_1x_2
$$

$$
\begin{aligned}
y_2 &= x_2 + x_3 \\
y_3 &= y_1 \oplus y_2
\end{aligned}
$$

Second, we express the probability of each internal node and primary output as a canonical sum of products of primary inputs to handle correlation between node y_1 and y_2 after exponent suppression.

$$
\begin{aligned}
P(y_1) &= P(x_1)P(x_2) \\
P(y_2) &= P(x_2) + P(x_3) - P(x_2)P(x_3) \\
P(y_3) &= (1 - P(x_1))P(x_2) + (1 - P(x_2))P(x_3)
\end{aligned}
$$

Third, by using $a(y) = 2(P(y) - P(y(t-T)y(t)))$ we can obtain an exact expression of the activity of each internal node and primary output.

$$
\begin{aligned}
a(y_1) &= P(x_1)a(x_2) + P(x_2)a(x_1) - \frac{1}{2}a(x_1)a(x_2) \\
a(y_2) &= (1 - P(x_2))a(x_3) + (1 - P(x_3))a(x_2) - \frac{1}{2}a(x_3)a(x_2) \\
a(y_3) &= P(x_2)a(x_1) + (1 - P(x_1) - P(x_3) + 2P(x_1)P(x_3))a(x_2) + \\
&\quad (1 - P(x_2))a(x_3) - \frac{1}{2}(a(x_1) + a(x_3))a(x_2)
\end{aligned}
$$

Fourth, by differentiating the activity equation of each internal node and primary output, a_j, we can get the activity sensitivity, $\partial a_j/\partial a_i$, to each primary input activity, a;.

$$
\begin{aligned}
\frac{\partial a(y_1)}{\partial a(x_1)} &= P(x_2) - \frac{1}{2}a(x_2) \\
\frac{\partial a(y_1)}{\partial a(x_2)} &= P(x_1) - \frac{1}{2}a(x_1) \\
\frac{\partial a(y_1)}{\partial a(x_3)} &= 0 \\
\frac{\partial a(y_2)}{\partial a(x_1)} &= 0 \\
\frac{\partial a(y_2)}{\partial a(x_2)} &= 1 - P(x_3) - \frac{1}{2}a(x_3) \\
\frac{\partial a(y_2)}{\partial a(x_3)} &= 1 - P(x_2) - \frac{1}{2}a(x_2) \\
\frac{\partial a(y_3)}{\partial a(x_1)} &= P(x_2) - \frac{1}{2}a(x_2) \\
\frac{\partial a(y_3)}{\partial a(x_2)} &= 1 - P(x_1) - P(x_3) + 2P(x_1)P(x_3) - \frac{1}{2}(a(x_1) + a(x_3)) \\
\frac{\partial a(y_3)}{\partial a(x_3)} &= 1 - P(x_2) - \frac{1}{2}a(x_2)
\end{aligned}
$$

For each primary input, $\partial a(y_j)/\partial a(x_i)$ is given by the following equations

$$
\frac{\partial a(y_j)}{\partial a(x_i)} = \begin{cases} 0 & : \; j \neq i \\ 1 & : \; j = i \end{cases}
$$

Finally, using equation 7 we can obtain power sensitivity to each primary input activity.

$$\zeta_{a_{x_1}} = f_{anout}(x_1)\frac{\partial a_{x_1}}{\partial a_{x_1}} + f_{anout}(y_1)\frac{\partial a_{y_1}}{\partial a_{x_1}} + f_{anout}(y_2)\frac{\partial a_{y_2}}{\partial a_{x_1}} + f_{anout}(y_3)\frac{\partial a_{y_3}}{\partial a_{x_1}}$$
$$\zeta_{a_{x_2}} = f_{anout}(x_2)\frac{\partial a_{x_2}}{\partial a_{x_2}} + f_{anout}(y_1)\frac{\partial a_{y_1}}{\partial a_{x_2}} + f_{anout}(y_2)\frac{\partial a_{y_2}}{\partial a_{x_2}} + f_{anout}(y_3)\frac{\partial a_{y_3}}{\partial a_{x_2}}$$
$$\zeta_{a_{x_3}} = f_{anout}(x_3)\frac{\partial a_{x_3}}{\partial a_{x_3}} + f_{anout}(y_1)\frac{\partial a_{y_1}}{\partial a_{x_3}} + f_{anout}(y_2)\frac{\partial a_{y_2}}{\partial a_{x_3}} + f_{anout}(y_3)\frac{\partial a_{y_3}}{\partial a_{x_3}}$$

3.4 Circuit Partitioning Algorithm

Accurate calculation of power sensitivity depends on whether we can accurately express the probability and activity of each internal node and primary output in terms of independent inputs. However, the size of symbolic probability expression and activity expression grow exponentially with respect to the number of independent inputs. Consequently, we resort to circuit partitioning to trade-off accuracy versus computation time.

We have developed a two-level partitioning algorithm. The first level partitioning is achieved as follows. Starting from one primary output y, we first find the Minimum Set of Topologically Independent *Inputs* of node y (*MSTII(y)*). *MSTII(y)* is defined as the Set of Topologically Independent Inputs *(STII)* with the minimum number of elements. **A** set of nodes Ψ is a Set of Topologically Independent Inputs (STII) to a node y (*STII(y)*) if and only if Ψ is topologically independent and topologically determines y. **A** set of nodes Ψ is topologically independent if and only if any two nodes x_1 and x_2 in Ψ do not have a common ancestor. A set of nodes Ψ topologically determines a node y and only if y does not have an ancestor x, such that $x \notin \Psi$ and x is neither a descendant nor an ancestor of any node in Ψ. Then we determine whether the probability and activity of each intermediate node in *MSTII(y)* can be calculated. We recursively repeat the same procedure to all intermediate nodes. In this way, the first level partitioning can be achieved without losing any accuracy.

Let us consider the circuit of Figure 2. Each node represents a logic function. The primary inputs x_1, x_2, ..., x_{12} are assumed to be independent. The sets { x_1,x_2,...,x_{12} }, { x_1, x_2, x_3, x_4, x_5, w_3, x_8, z_1, x_{11}, x_{12} }, and {y_1,x_5,w_3,x_8,z_1,x_{11},x_{12}} are *STII(y)*'s. It can be proved that {y_1,x_5,w_3,x_8,z_1,x_{11},x_{12}} is the *MSTII(y)*. Therefore, the first level partitioning algorithm'starts from primary output y to find the *MSTII(y)*. However, the probabilities and activities of the intermediate nodes y_1, w_3, and z_1 have to be determined before those of node y can be computed. The *MSTII(y_1)* is { x_1, x_2, x_3, x_4 }. By recursively computing the signal probabilities for y_1, w_3, and z_1, the first level partitioning can be finished.

The cardinality of the MSTII to some nodes may be larger than we can handle in terms of computation time. The purpose of the second level partitioning is to limit the cardinality of the MSTII. When the cardinality of the MSTII exceeds the preset threshold values, the cost of each fanin of y is computed. The cost of each fanin y_i is defined as $\sum_{j=1,i\neq j}^{k} |\Lambda(y_i) \cap \Lambda(y_j)|$, where k is the number of fanins of y and $\Lambda(y_i)$ and $\Lambda(y_j)$ are the supports of y_i and y_j with respect to y, respectively. The support of node y_i with respect to node y is a subset of the MSTII that are

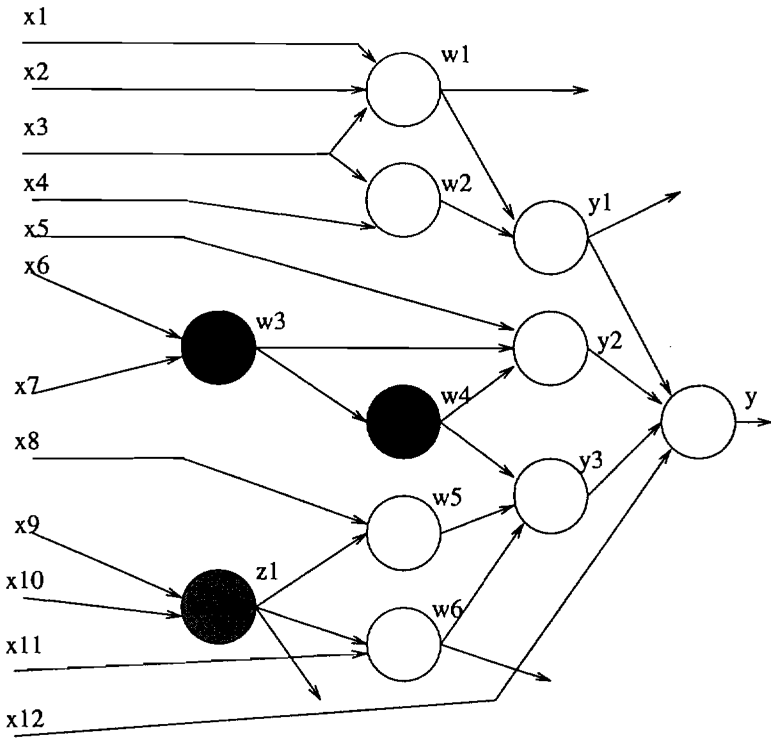

Figure 2: The Minimum Set of Topologically Independent

y_i's ancestors (MSTII$\cap$ { ancestors of y; }). The cardinality (number of elements) of $S(y_i)$ is denoted as $|\Lambda(y_i)|$. The fanin that is not in the MSTII and has a minimum cost is chosen. Then we assume this fanin to be independent of any other nodes in the circuit. The error caused by this assumption is minimized since we have chosen the fanin which has least correlation to the other fanins. After partitioning, some dependencies in the circuit have been eliminated. Consequentley, we must recompute the *MSTII(y)* including the node with the minimum cost. If the cardinality of the new *MSTII(y)* is still greater than the preset threshold value, we repeat the same scheme to partition the circuit. Otherwise, we compute the probability and activity of all the nodes involved. After y is computed, all the nodes which are both descendants of the *MSTII* and ancestors of y are calculated.

For example, in Figure 2 the cardinality of the *MSTII(y)* is 7. If we set the threshold number of inputs to be 5, some of the fanins need to be partitioned. The supports and costs of y_1, y_2, y_3, and x_{12} are ({ y_1 }, 0), ({ x_5, w_3 }, 1), ({ w_3, x_8, z_1, x_{11} } , 1), and ({ x_12 }, 0), respectively. Since y_1, x_{12} belongs to the MSTII, we can only chose from y_2 and y_3. If we ranclomly choose y_2, the new *MSTII* is { y_1, y_2, y_3, x_{12} }. This is because y_2 has been assumed to be independent after partitioning and we can not find any primary reconvergent node of y. Therefore, all fanins of y become independent after the second level partitioning.

3.5 Algorithm to Compute Activity Sensitivity

The algorithm to compute activity sensitivity for each node of a circuit is given below.

```
Compute Activity and Activity Sensitivity ∂a_j/∂a_i
Input : Circuit, signal probability and activity of each input
Output : Signal probability, activity, and activity sensitivity for each node of the circuit
For each primary output or internal node that is not in the fanin of y {
      If (y has not been calculated (not marked yet) ) {
         Calculate(y);
      }
}
```

```
Calculate(y) {
      Do
         Find out all the reconvergent nodes with respect to node y;
         Determine MSTII(y) and compute the support and cost of each fanin of y;
         Select the minimum-cost fanin (∉ MSTII(y)) of y to be partitioned.
      }
      While (the cardinality of MSTII(y) is greater than the limit)
      For each y_i ∈ the MSTII(y) {
         If (y_i is not yet calculated) {
            Calculate(y_i);
         }
      }
      Compute probability in terms of the MSTII(y) for each node in the MCR(y);
      Compute activity and activity sensitivity at each node in the MCR(y);
      Mark all nodes that have been calculated;
}
```

MCR(y) is *Minimum* Cone Region of node y and is defined as the set of such nodes that they are descendants of some node in *MSTII(y)* and ancestors of node y.

After the activity sensitivity $\partial a_j/\partial a_i$ of each primary output and internal node has been computed, we can use equation 7 to compute the power sensitivity to activity of each primary input.

4 Monte-Carlo Based Method to Compute Power Sensitivity

4.1 Monte-Carlo Method to Estimate Activity

The basic idea of Monte Carlo method for estimating node activity is to simulate a circuit with random patterns assigned to primary inputs. Stopping criterion is used to determine when node activities have converged[8, 9].

We can use random number generators to generate primary input patterns conforming to the given probabilities and activities. During a given period of duration T_D (clock cycles), we count the number of transitions at each node n_1 and call the value n_1/T_D a random sample. T_D denotes the sample length. The process is repeated K times in order to obtain K independent samples, $a_j = n_j/T_D, \quad j = 1 \cdots K$. A different seed is used each time for the random number generator. The sample mean is defined as a $= (\sum_{j=1}^{n} a_j)/K$. For large K, a will approach the expected value of $\bar{a}$ ($\lim_{T_D \to \infty} n_{T_D}/T_D$), denoted as a since the signal at each node is mean-ergodic. n_{T_D} is the number of transitions in the time interval $(\frac{-T_D}{2}, \frac{T_D}{2}]$. Similarly, for large K the sample standard deviation s will approach the true standard deviation a. Furthermore, according to the Central Limit Theorem [13] $\bar{a}$ is a random variable with mean a and has a distribution .approaching the normal distribution if K is large (typically ≥ 30). Likewise a $\approx s/\sqrt{K}$. It has been shown in [8, 9] that for $(1 - $ a) x 100% confidence the following inequality holds:

$$\frac{|a - \bar{a}|}{\bar{a}} \leq \frac{z_{\alpha/2}s}{\bar{a}\sqrt{K}}, \tag{21}$$

where $z_{\alpha/2}$ is a specific value such that the area under the standard normal distribution from $z_{\alpha/2}$ to ∞ is $\alpha/2$. Therefore, if

$$K \geq (\frac{z_{\alpha/2}s}{\bar{a}\epsilon'})^2, \tag{22}$$

we have $\frac{|a - \bar{a}|}{\bar{a}} \leq \frac{z_{\alpha/2}s}{\bar{a}\sqrt{K}} \leq \epsilon'$, and hence $\frac{(a - \bar{a}|}{a} \leq \frac{\epsilon'}{1 - \epsilon'} =$ E.

Equation 22 is the stopping criterion for $(1 -$ a) x 100% confidence and ϵ is an upper bound on the relative error. If any node in the circuit has a very low activity ($\bar{a} \ll 1$) the number of samples required by equation 22 can be very large. This results in slow convergence. However, since these low-activity nodes contribute little to power dissipation, a modified stopping criterion is proposed in [9]. One can specify a particular threshold value a;,, below which the activities of nodes are less important. Hence, one does not have to wait for those nodes to converge to a value within a certain percentage of error. Furthermore, if

$$K \geq (\frac{z_{\alpha/2}s}{a_{min}\epsilon'})^2, \tag{23}$$

we have $\frac{|a - \bar{a}|}{\bar{a}} \leq \frac{z_{\alpha/2}s}{\bar{a}\sqrt{K}} \leq \frac{a_{min}\epsilon'}{a}$, and hence $|a - \bar{a}| \leq a_{min}\epsilon'$.

Therefore, equation 23 becomes the stopping criterion (with $\bar{a} <$ a;,,) for $(1 - \alpha) \times 100\%$ confidence. $a_{min}\epsilon'$ is an absolute error bound (not a percentage error bound).

4.2 Monte-Carlo Based Method to Compute Power Sensitivity

The procedure to compute power sensitivity to primary input activity by using Monte Carlo Method is outlined in Figure 3. First we use Monte-Carlo method outlined in the previous section to get

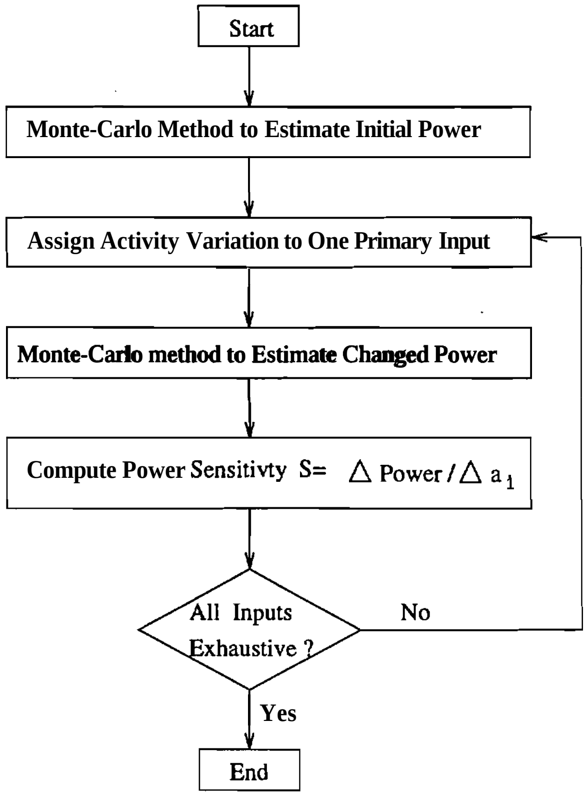

Figure 3: Monte-Carlo method to compute power sensitivity

the initial power. Then we repeat the same procedure n times, where n is the number of primary inputs. Each time when we repeat the procedure, we assign activity variation to only one primary input. Using expression $\Delta Power_{avg}/\Delta a_i$ we can get the power sensitivity to the activity of that primary input. Similarly, we can compute power sensitivity to primary input probability provided that we replace activity with probability in the above procedure.

5 Implementation and Experimental Results

The symbolic method and Monte-Carlo based method to compute power sensitivity to primary input specification have been implemented in C under the Berkeley SIS environment. In this section we will investigate the effect that the primary input specification uncertainty has on the power dissipation.

When we compute power sensitivity, the probability and activity of each primary input is assumed to be 0.5 and 0.2, respectively. Some results using Monte-Carlo method are shown in Figures 4, 6, 8, 10 where the y-axes represent power sensitivity to primary input activity and x-axes represent primary inputs. The results for the corresponding circuits using symbolic method are

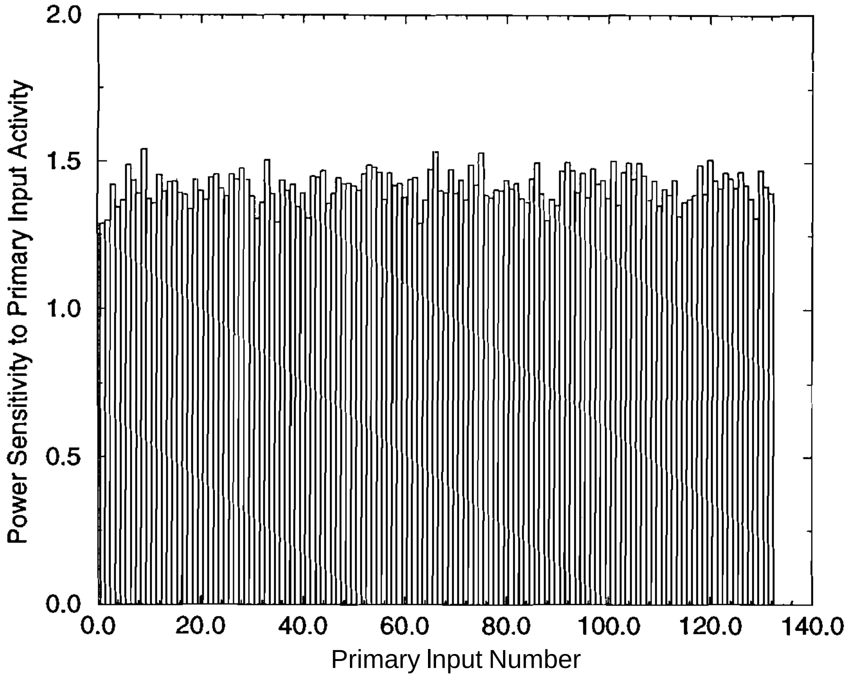

Figure 4: ζ_{a_i} for circuit i3 by Monte-Carlo based method

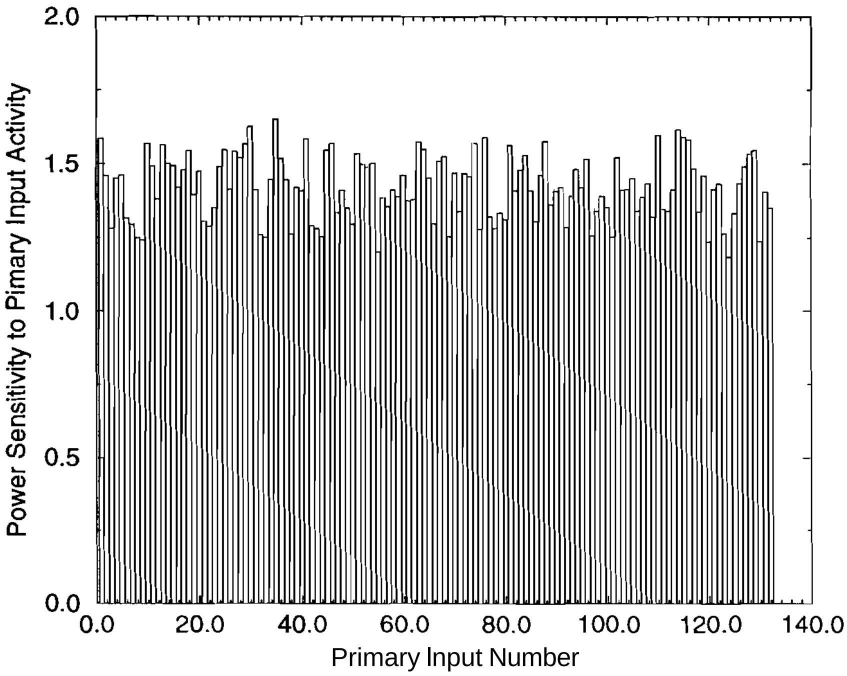

Figure 5: ζ_{a_i} for circuit i3 by symbolic based method

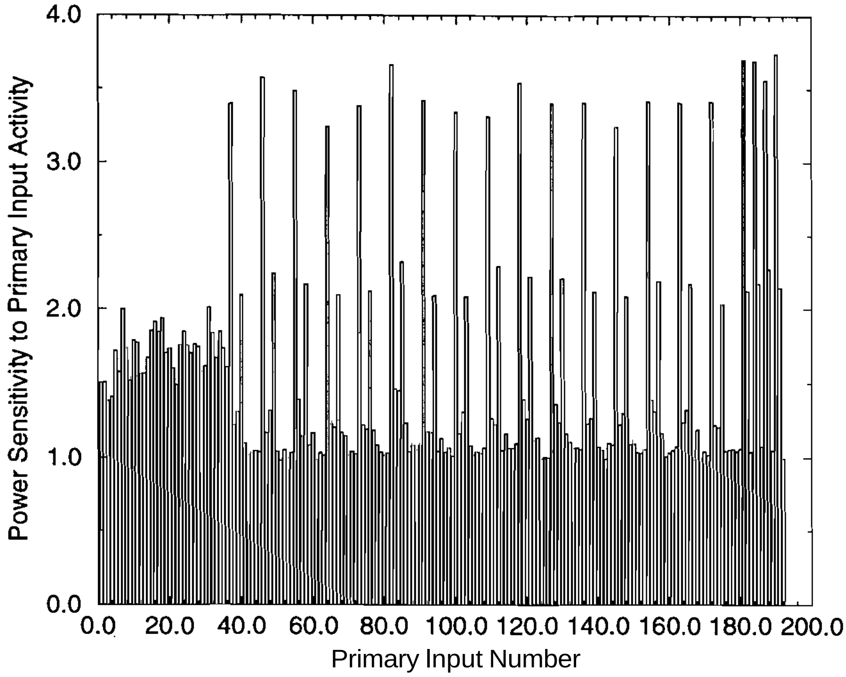

Figure 6: ζ_{a_i} for circuit i4 by Monte-Carlo based method

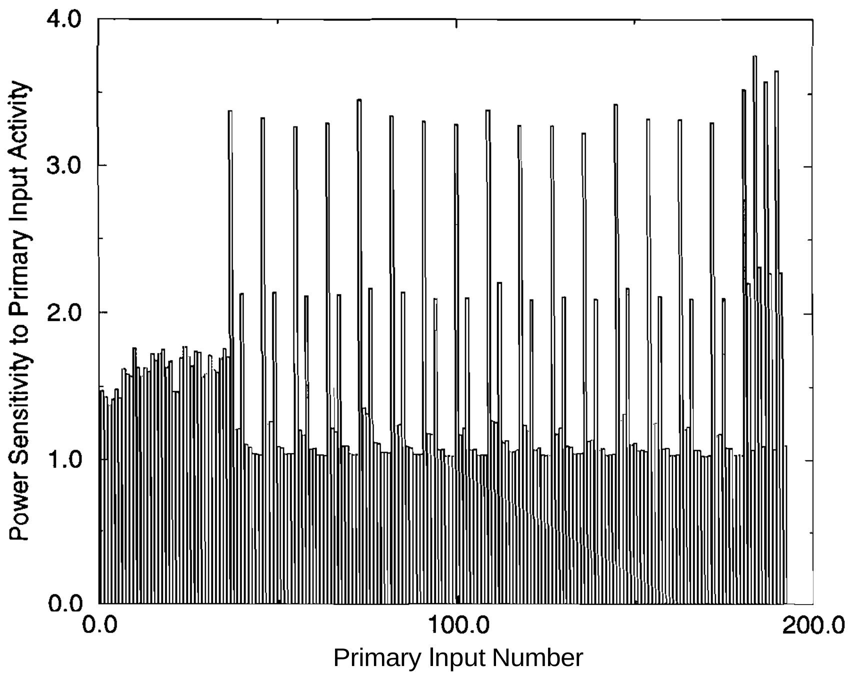

Figure 7: ζ_{a_i} for circuit i4 by symbolic based method

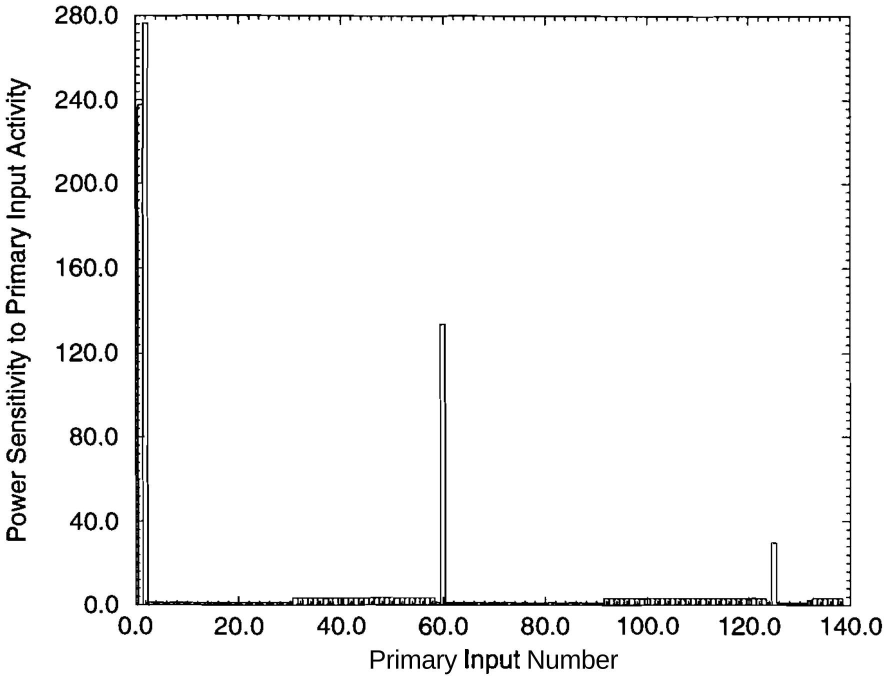

Figure 8: ζ_{a_i} for circuit i6 by Monte-Carlo based method

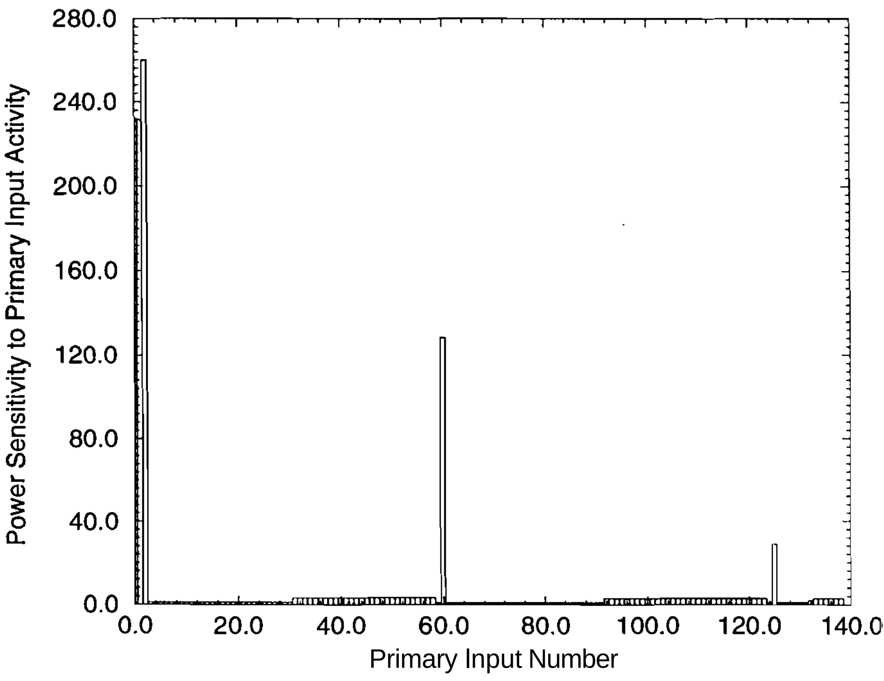

Figure 9: ζ_{a_i} for circuit i6 by symbolic based method

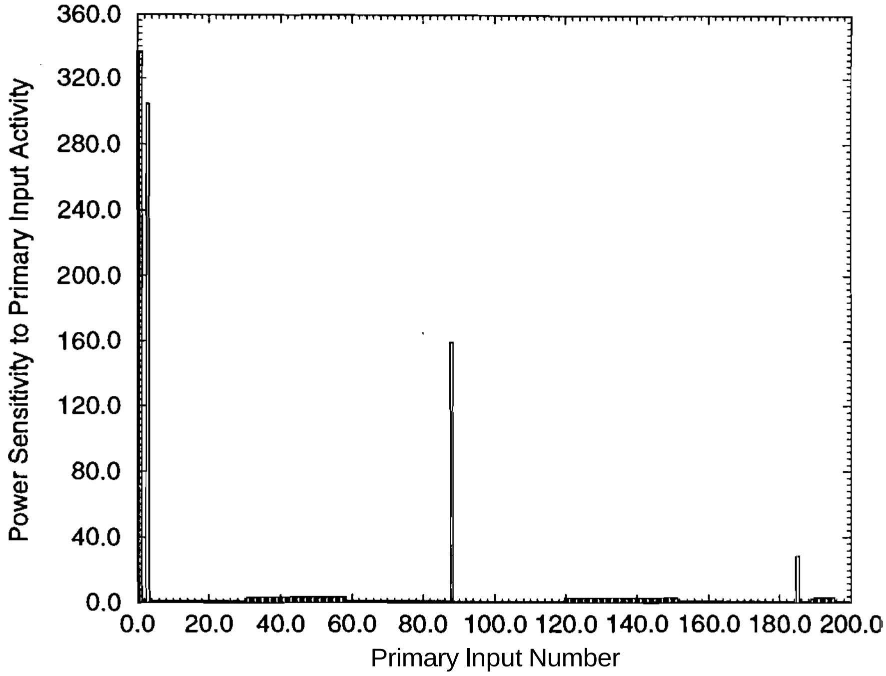

Figure 10: ζ_{a_i} for circuit i7 by Monte-Carlo based method

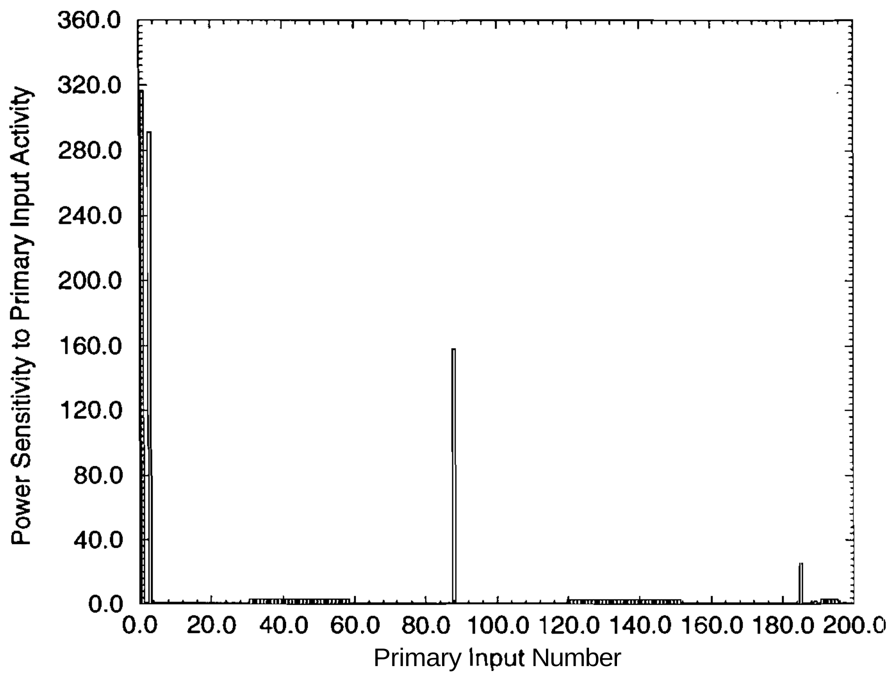

Figure 11: ζ_{a_i} for circuit i7 by symbolic based method

shown in Figures 5, 7, 9, 11. Results show that probabilistic and statistical methods of sensitivity estimation back each other fairly well. Results also show that for some circuits power dissipation is much more sensitive to the activity of some primary inputs than other inputs. A small variation of the activity of highly sensitive primary inputs will result in a dramatic change of the average power. Consider circuit i6. The power sensitivity to activity of the lst, 2nd, 60th, and 124th primary input is 237, 276, 133, and 30, respectively. The power sensitivity to activity of each of other primary inputs is less than 4. If the activity of the 1st and 2nd primary input have a variation of 0.1, the power dissipstion may change 30%. Therefore, for power conscious designs, either the sensitive inputs should be accurately specified For some circuits, the difference among power sensitivities to different primary inputs is small. For example, for circuit i3, the value of power sensitivity to activity of each primary input is between 1.3 and 1.6. This means that the activity variation of different primary inputs may have almost the same effect on average power dissipation.

To compare the results obtained by the symbolic method with those obtained by the Monte-Carlo based method, we compute percent error using the expression $\sum_{i \in allinput} \frac{|\zeta_{a_i}(Symb) - \zeta_{a_i}(MC)|}{\zeta_{a_i}(MC)}$ $\zeta_{a_i}(MC)$ is the power sensitivity to the activity of the i^{th} primary input by using Monte-Carlo method. $\zeta_{a_i}(Symb)$ is the power sensitivity to the activity of the i^{th} primary input by using symbolic method. The comparison is shown in the following table. The CPU time is also shown for a SPARC 5 workstation.

Since Monte-Carlo method to estimate power sensitivity repeats the estimation procedure $n+1$ times (n is the number of primary inputs), execution time may be unacceptably long for large n. Consider circuit C7552. It takes 4093 seconds of CPU time to finish one general Monte-Carlo power estimation procedure. The circuit has 207 primary inputs. It would take approxirnately 8.5 x 10^5 seconds (9.8 days) of CPU time. Circuits with prohibitively long Monte-Carlo execution time are identified by dashes in the "MC" and "percent error" columns of table 1. The symbolic method takes much less CPU time. For small circuits, the symbolic and Monte-Carlo are very close. For large circuits, circuit partitioning can introduce some error.

Note that in this paper we have only presented results on power sensitivity to primary input activities. The same method can be applied and similar results obtained on power sensitivity to primary input probabilities.

Table 1: Results of power sensitivity for some benchmarks

Circuit Chosen	PI's Number	Level Number	Gate Number	CPU Time (s)		percent error
				Symbolic	MC	
i1	25	5	33	0.16	519	5.6
i3	132	2	70	0.34	7896	0.4
i4	192	4	94	5.39	16960	2.5
i5	133	6	199	1.76	23326	0.8
i6	138	3	344	4.16	38877	4.4
i7	199	3	406	7.04	78514	4.8
i8	133	8	1183	134.3	119088	11.1
i9	88	7	353	209.0	33581	5.9
i10	257	54	2497	423	290744	15.4
C432	36	17	160	2.82	8085	12.8
C499	41	11	202	14.3	11615	0.1
C880	60	23	357	143.3	5675	1.6
C1355	41	23	514	14.2	15791	9.1
C1908	33	40	880	50.9	19007	11.4
C2670	233	32	1161	34.9	-	-
C3540	50	47	1667	114.8	48719	13.0
C5315	178	49	2290	55.6	-	-
C6288	32	124	2416	6127	46219	14.6
C7552	207	43	3466	112.4	-	-

6 Conclusions

In this paper we present a new concept: *power sensitivity to primary input specification.* Considering spatio-temporal correlations among signals, we propose a symbolic method to compute *power sensitivity to primay input specification.* Since the size of the activity equation may exceed available mernory resources, we partition the circuit in a way that makes the problem tractable without introducing an unacceptable amount of error. The symbolic method to compute power sensitivity has been implemented under the Berkeley SIS environment. For small circuits, the results show that the symbolic method produces very accurate results in a short CPU time. For large circuits., circuit partitioning may introduce an error on the order of 5-15% error compared to Monte-Carlo results. Our results on ISCAS and MCNC benchmark circuits indicate that for some circuits the power dissipation is much more sensitive to activity variation of some primary inputs than others. A small activity variation of those sensitive primary inputs will have a dramatic effect on power dissipation. Therefore, for power conscious designs, the sensitive inputs should be accurately specified.

References

[1] A.P.Chandrakasan, S.Sheng, and R.W.Brodersen, "Low-Power CMOS Digital Design," Journal of Solid-State Circuits, Vol.27, No.4, pp.473-483, April,1992.

[2] F.N.Najm, "Transition Density, A Stochastic Measure of Activity in Digital Circuits" 28th ACM *IEEE* Design Automation Conf., pp.644-649, 1991.

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

[4] A. Ghosl, S. Devadas, K. Keutzer, and J. White, "Estimation of Average Switching Activity in Combinational and Sequential Circuits," ACM IEEE Design Automation Conf., pp.253-259, 1992.

[5] S.Devadas, K.Keutzer and J.White, "Estimation of Power Dissipation in CMOS Combinational Circuits'", IEEE Custom Integrated Circuits Conf., pp. 19.7.1-19.7.6, 1990.

[6] J. Monteiro, S. Devadas, and B. Lin, "A Methodology for Efficient Estimation of Switching Activity in Sequential Logic Circuits," *ACM/IEEE* Design Automation Conf., 1994, pp. 12-17.

[7] C.-Y. Tsui, M. Pedram, and A. M. Despain, "Exact and Approximate Methods for Calculating Signal and Transition Probabilities in FSMs," *ACM/IEEE* Design Automation Conf., 1994, pp. 18-23.

[8] R.Burch, F.Najm, P.Yang and T.Trick, "A Monte Carlo Approach for Power Estimation", IEEE Trans. VLSI Systems, Vol.1, No. 1, pp.63-71, March, 1993.

[9] M. G. Xakellis and F. N. Najm, "Statistical Estimation of the Switching Activity in Digital Circuits," *ACM/IEEE* Design Automation Conf. , 1994, pp. 728-733.

[10] F.N.Najm, "A Survey of Power Estimation Techniques in VLSI Circuits," IEEE Trans. on VLSI *Systems*, pp. 446-455, December, 1994.

[11] K.P.Parker and E.J.McCluskey, "Probabilistic Treatment of General Combinatorial Networks," IEEE Trans. on Computers, Vol.C-24, pp. 668-670, June, 1975.

[12] K. Roy and S. Prasad, "Circuit Activity Based Logic Synthesis for Low Power. Reliable Operations," IEEE Trans. on VLSI Systems, Dec. 1993, pp. 503-513.

[13] A. Papoulis, Probability, "Random Variables, and Stochastic Processes", 3rd Edition, New York: McGraw-Hill, 1991.