PROPAGATION ALGORITHM OF BEHAVIOR PROBABILITY IN POWER ESTIMATION BASED ON MULTIPLE-VALUED LOGIC

Xunwei Wu
Institute of CAS, Ningbo University, Ningbo 315211, CHINA
Massoud Pedram
Dept. of E.E.-Systems, Univ. of Southern California, CA 90007, USA

Abstract — This paper analyses the propagation operations of signal's multiple-valued behavior while passing through the basic gates. Based on it the propagation algorithm of behavior probability is proposed. In comparison with the propagation algorithm of signal probability this algorithm have advantages that the calculation is direct one and the glitches resulted from race hazard can be taken into account, whereby the estimation accuracy of power dissipation can be improved.

I. INTRODUCTION

According to the development of CMOS VLSI technology, the continuing increase in chip density and operating frequency have made power consumption a major concern in VLSI design. For example, the PC chip from Motorola consumes 8.5W, the Pentium chip from Intel consumes 16W, and DEC's Alpha chip 21164 consumes 50W. The excessive power dissipation in integrated circuits not only discourages their use in a portable environment, but also causes overheating, which degrades performance and reduces chip life-time.[1] All of these factors drive designers to devote significant resources to reduce the circuit power dissipation. Indeed, the Semiconductor Industry Association has identified low-power design techniques as a critical technological need.[2] The design for low power cannot be achieved without accurate power prediction tools. Therefore, there is a critical need for a technique to estimate power dissipation during the design process to meet the power budget without having to go through a costly redesign effort.[3] From the power analysis of CMOS circuit, it is necessary to develop a comparatively precise and efficient power estimation algorithm.

The dominant power consumption of CMOS circuit is dynamic power dissipation. The energy required to charge or discharge the load capacitance of each device's output node i is $\frac{1}{2}C(i)\cdot V_{\mathrm{DD}}^2$.[4] If the number of total nodes in the circuit is M and the input sequence length is L, we can use the following formula to calculate the average power dissipation:

$$P=\sum_{\text{node}\,i}^{M}P_i=\sum_{\text{node}\,i}^{M}\{\frac{1}{T}\sum_{\text{cycle}\,j}^{L}\Delta x_i(j)[\frac{1}{2}c(i)\cdot V_{\mathrm{DD}}^2]\}, \tag{1}$$

where $T = L/ f_{\mathrm{CLK}}$ is the period of the sequence, $\Delta x_i(j)$ represents the switching activity of signal x_i at node i from cycle j to cycle j+1:

$$\Delta x_i(j)=\begin{cases}1 & \text{if } x_i(j)\neq x_i(j+1);\\ 0 & \text{if } x_i(j)=x_i(j+1).\end{cases}$$

From Eq.(1), we can obtain the corresponding flow chart as shown in Fig.1. When the input sequence is provided, we can calculate the switching activity of each node gradually, then the total power dissipation can be obtained in the end. However, this flow has the following two problems in application.

1) The large number of nodes *M* and the demand of the input sequence's length will lead to the huge calculation, which will go beyond the actual memory and exerting time of computer.
2) If the signal delay is taken into account, because of the race between input signals there may exist glitches at output, which will increase the transition number at the output node and introduce additional power dissipation. It has been observed that this additional power dissipation is typically 20% of the total power, but can be as high as 200% of the total power in some cases such as in a multiplier.[2] But the calculation of $\Delta x_i(j)$ in Eq.(1) will omit this power dissipation and lower the precision.

As to the first problem, the conventional solution is to exert the average step first in the flow, that is to average the *L* cycle's input signal in the beginning so as to decrease the calculation to 1/*L* of the original calculation. But we still do not have a simple and efficient method to solve the last problem.

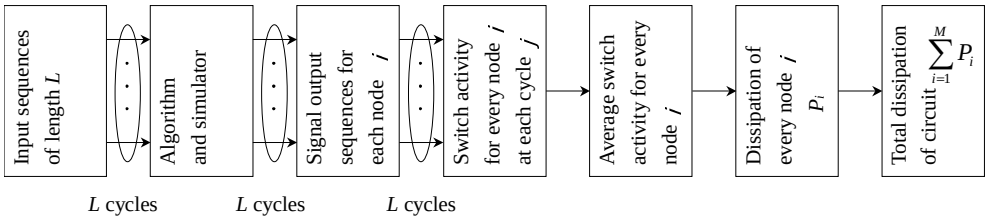

Figure 1 Basic algorithm flow for power estimation

This paper mainly presents a probability algorithm of power estimation based on multi-valued logic. It is because that the power consumption of CMOS circuit is related to the signal behavior, which is a multi-valued variable. In the next section, we introduce the concept of signal probability, further we describe the propagation algorithm and power estimation based on signal's probability by providing some examples. Section Ⅲ discusses the multi-valued description and the transition algorithm of signal's behavioral. Section Ⅳ develops a transition algorithm based on behavioral probability. Conclusion are given in Section Ⅴ.

II. PROPAGATION ALGORITHM OF SIGNAL PROBABILITY AND POWER ESTIMATION

In Eq.(1), $\frac{1}{T}\sum_{\text{cycle } j}^{L} \Delta x_i(j)$ represents the average of measuring result of $\Delta x_i(j)$ during the period of $T = L/f_{\text{CLK}}$. So a concept of average switching activity $\overline{\Delta x_i(j)}$ can be introduced, thus Eq.(1) changes to:

$$P = \sum_{\text{node } i}^{M} P_i = \sum_{\text{node } i}^{M} \{ f_{\text{CLK}} \overline{\Delta x_i(j)} [\frac{1}{2} c(i) \cdot V_{\text{DD}}^2] \}, \qquad (2)$$

Obviously, $\overline{\Delta x_i(j)}$ in Eq.(2) can be considered as the probability of $\Delta x_i(j)$=1 in a cycle. In fact, Ref.[5] had already introduced the signal probability. The signal's probability is defined as:

Definition In a long period, the ratio of the time of x = 1 to the total time is the signal probability of x = 1.

The propagation operations of the basic gates shown in Fig.2 have been introduced by Ref.[5]. For the inverter shown in Fig.2(a), since the probability of $\bar{x}=1$ and the probability of x = 0 are the same we have

$$P(f)=P(\bar{x})=1-P(x).$$

If we denote $1-P(x)$ as $\overline{P(x)}$, the propagation operation of the inverter is:

$$P(f)=P(\bar{x})=\overline{P(x)}. \tag{3}$$

For the AND gate shown in Fig.2(b), if x and y are independent with each other the probability of $x = y = 1$ should be the product of two signal probabilities:

$$P(g)=P(x)\cdot P(y). \tag{4}$$

As to the OR gate shown in Fig.2(c) we can transform it into the form of AND gate by using De Morgan Law, as shown in Fig.2(d). Thus, if x and y are independent with each other we have

$$\overline{p(h)}=\overline{P(x)}\cdot\overline{P(y)}. \tag{5}$$

Substituting $\overline{P(h)}=1-P(h)$ and so on we obtain

$$P(h)=P(x)+P(y)-P(x)\cdot P(y). \tag{6}$$

Since a circuit is composed of the above three basic gates, based on Eqs.(3)-(6) we can calculate the signal probabilities from input to output, node by node. If there exists re-convergence of signal in the circuit the revision for calculation is necessary. We will explain it by an example later.

(a) (b) (c) (d)

Figure 2. Basic gates

We should notice that $\overline{\Delta x_i(j)}$ in Eq.(2) is signal probability other than transition probability. The remained question is how to transform the signal probabilities at nodes into the corresponding transition probabilities. Now we discuss the relation between signal probabilities $P(x), P(\bar{x})$ and transition probability $P(\Delta x_i)$ in the successive cycles. Since the signals in the former and the latter cycles are thought of independent, the rising probability is $P(\bar{x})\cdot P(x)$, and the falling probability is $P(x)\cdot P(\bar{x})$. So they are equal, and we have

$$P(\Delta x) = 2P(x) \cdot P(\bar{x}) = 2P(x)[1 - P(x)]. \quad (7)$$

As a basic formula for transforming signal probability into transition probability the above equation has been widespread adopted for calculating the transition probability.[6-9]

Example 1. In the circuit shown in Fig.3 three inputs, *a*, *b*, *c*, are independent with each other. Their signal probabilities are $P(a) = 0.9$, $P(b) = P(c) = 0.5$, respectively. Other three nodes in the circuit are *x*, *y*, *f*. The probability for all nodes are required.

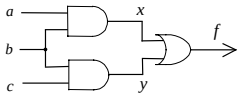

Figure 3. Circuit of Example 1

First we can obtain the signal probabilities for the nodes *x*, *y*, and *f*:

$$P(x) = P(a) \cdot P(b),$$

$$P(y) = P(b) \cdot P(c),$$

$$\overline{p(f)} = \overline{P(x)} \cdot \overline{P(y)}.$$

The last equation can be rewritten as

$$P(f) = P(x) + P(y) - P(x) \cdot P(y) = P(a) \cdot P(b) + P(b) \cdot P(c) - P(a) \cdot [P(b)]^2 \cdot P(c),$$

where the term with a higher power reflects the re-convergence of signal *b* at the OR gate. The term $[P(b)]^2$ represents the "probability of *b* = 1 during *b* = 1". It should be the pure probability of *b* = 1 rather than its square, therefore the above equation should be revised as

$$P(f) = P(x) + P(y) - P(x) \cdot P(y) = P(a) \cdot P(b) + P(b) \cdot P(c) - P(a) \cdot P(b) \cdot P(c).$$

Substituting $P(a) = 0.9$, $P(b) = P(c) = 0.5$ into the above equation we have $P(x) = 0.45$, $P(y) = 0.25$ and $P(f) = 0.475$. By using Eq.(7) the corresponding transition probabilities can be obtained from all of the calculated signal probabilities:

$P(\Delta a) = 0.18$, $P(\Delta b) = P(\Delta c) = 0.5$, $P(\Delta x) = 0.495$, $P(\Delta y) = 0.375$, $P(\Delta f) = 0.4988$.

If considering average switching activities for all nodes in the circuit we have

$$\sum E_{sw} = P(\Delta a) + 2P(\Delta b) + P(\Delta c) + P(\Delta x) + P(\Delta y) + P(\Delta f) = 3.05,$$

where the factor 2 is taken into account for the doubled load capacitance at the node *b*.

The above introduced propagation algorithm based on signal probability simplifies the flow chart in Fig.1 into the counterpart in Fig.4, where averaging is taken to the input sequence at the first so as to save the calculation. However, we should point out the following two problems with the use of Eq.(13):

1. Equation (7) shows a stationary relationship between the signal probability and the transition probability. It means that as long as the signal probability is given the corresponding transition probability can be derived. However, for a certain signal sequence the relationship given in Eq.(7) usually is not met accurately. Therefore, the calculation accuracy is weakened if we calculate the signal probability from a given sequence.[10]

2. Equation (7) is on the basis of signal probability, which is a quiescent description for the circuit. It cannot reflect the signal race hazard due to delay. However the possible glitches at various nodes will significantly increase transition number, which is neglected by Eq.(7). Equation (7) is accurate only for the "clean" input signal sequences.

According to the above points, we hope that we can find a direct algorithm for transition probability instead the above bypass way. At the same time we should take into account the glitches, i.e. extra transition behaviors due to signal delay.

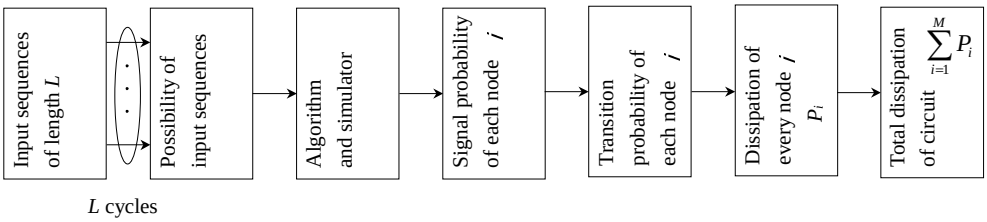

Figure 4 Algorithm flow for power estimation based on signal probability

III. TERNARY DESCRIPTION OF SIGNAL BEHAVIOR AND OPERATION RULES

Because the power dissipation of CMOS circuit is mainly decided by the signal transition at each node other than the signal itself, so we should introduce the signal's behavior to describe the working of the circuit. For any signal x in the circuit, we denote its logic values in successive cycles as $x(j)$ and $x(j+1)$ respectively. Correspondingly, four combinations can be used to express all behaviors of the signal, as shown in Table 1, where a special quaternary variable $\bar{x}$ denotes the signal behavior. Its four values are (0, α, β, 1), where α, β represent the two kinds of transition behavior and 0,1 represent the two kinds of holding behavior. (Note that while the latter have the same forms as signals 0 and 1, their meanings are different.)

Table 1 Quaternary representation for behaviors of a signal

$\bar{x}$	$x(j) \to x(j+1)$		Behavior
0	0	0	0-holding
α	0	1	α-transition
β	1	0	β-transition
1	1	1	1-holding

In fact, if we consider only power dissipation, we don't have to distinguish between an α–transition and a β–transition since the power dissipation for rising and falling transitions are the same.[4] Therefore we use $\frac{1}{2}$ to express both α and β, and represent the behavior $\bar{x}$ by using a ternary variable which takes three values (0, ½, 1), as shown in Table 2.

Table 2 Ternary representation for behavior of a signal

$\bar{x}$	$x(t) \to x'(t)$		Behavior
0	0	0	0-holding
$\frac{1}{2}$	$x(t) \neq x'(t)$		transition
1	1	1	1-holding

We consider the operation of three basic gates on signal behavior. If we don't take delay into account, the AND operation of α transition (0→1) and β transition (1→0) is 0-holding behavior, the OR operation is 1-holding behavior. However, if two transitions don't happen at the same time due to delay, then a glitch may be created. Since a glitch consists of two transitions, if we consider the probabilities for α being earlier or later than β to be equal, we could count the average number of transitions for $\alpha \cdot \beta$ and $\alpha + \beta$ as 1. Here we have taken the signal delay into account, three basic gates' operation on signal behavior are shown in Fig.5. Note that $\vec{x}$ =½ and $\vec{y}$ =½ in Fig.5 has considered the various transitions between *x* and *y*.

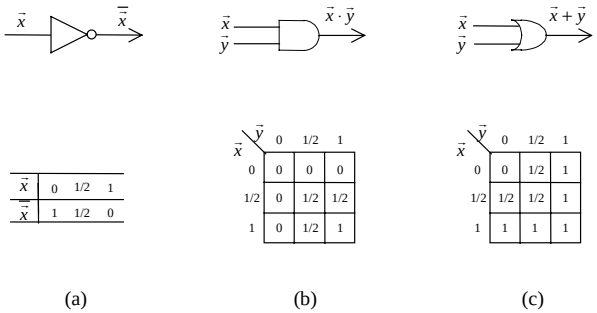

Figure 5. Operations of three basic gates for ternary behaviors

(a) $f = \bar{x}$, (b) $g = x \cdot y$, (c) $h = x + y$

We find that the relationship between inputs and outputs in Fig.5 happen to conform to the definitions of NOT, AND, and OR operations for ternary signals in ternary logic. Therefore, we can unite the representations of both signals and behaviors by a simple variable. That is, we no longer use an arrow to mark the behavior. Correspondingly, the operations on signal and behavior will have unified definitions in form as follows.

NOT operation

$$\bar{x} \overset{\Delta}{=} 1 - x, \tag{8}$$

AND operation

$$x \cdot y \overset{\Delta}{=} \min(x, y), \tag{9}$$

OR operation

$$x + y \overset{\Delta}{=} \max(x, y). \tag{10}$$

In the above equations, if *x*, *y* are binary signals, i.e. $x, y \in \{0, 1\}$, the operations are only for signals; If *x*, *y* are behaviors, i.e. $x, y \in \{0, ½, 1\}$, the operations are only for behaviors. Thus, for a logic circuit, the output signal of each gate in the circuit can be obtained by the traditional logic operation rules if the input signals are given. Meanwhile, if the input behaviors are given, the output behavior of each gate in the circuit can be derived by using the corresponding ternary operation rule.

In addition, we also can define the Literal operation as follows:

Literal operation

$$x^k = \begin{cases} 1 & \text{if} \quad x = k \\ 0 & \text{if} \quad x \neq k \end{cases} \tag{11}$$

where $x, k \in \{0, ½, 1\}$. Thus, the transition behavior of signal x_i at node i can be expressed as $x_i^{½}$. Obviously, $x_i^{½}$ is the Δx_i in Eq.(2).

In fact, if the input signal sequence is "clean", i.e. without any glitches, its corresponding input behavior sequence will be transformed easily. For example, for the following signal sequence:

x = 0 0 1 0 0 1 1 1 0 0 1 1 1 0 0 1 0 ⋯

we obtain the corresponding behavior sequence by averaging the adjacent two entries:

x = 0 ½ ½ 0 ½ 1 1 ½ 0 ½ 1 1 ½ 0 ½ ½ ⋯

Notice that in the above sequence symbols 0 and 1 represent the 0-holding and 1-holding behaviors rather than signal values, and symbol ½ represents the transition behavior.

As long as the input behaviors are given, it will be easy to derive the output behaviors for each output node from the circuit. The race hazards resulting from delay have been taken into account. As an example, all input behaviors of the circuit shown in Fig.4 are given and the signal values in the successive cycles have been given in the brackets. If we don't consider the possible glitches resulting from delay, the output behaviors for each gate in Fig.6 will be A:$(0 \to 1)$, B:$(1 \to 0)$, C:$(0 \to 0)$, D:$(1 \to 0)$, E:$(1 \to 1)$, F:$(1 \to 0)$, G:$(1 \to 0)$. The total number of transitions is 5. However, if delay is taken into account, the operation rules in Eqs.(9) and (10) can be used to get A = ½ , B = ½, C = 0, D = ½ , E = ½ , F = ½ , G = ½ . The consideration of glitches associated with operation $\alpha + \beta$ results in an additional transition: E = ½ .

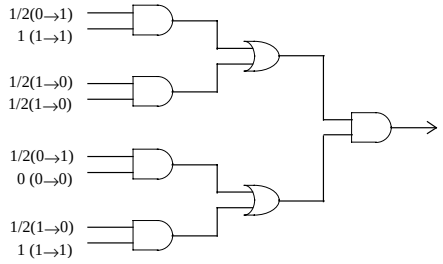

Figure 6. Transition calculation of logic circuit

IV. PROPAGATION ALGORITHM OF BEHAVIOR PROBABILITY AND POWER ESTIMATION

In section II, transforming the signal probabilities into transition probabilities is necessary to estimate power dissipation. The discussion in the last section also pointed out that the propagation algorithm of behavior can take into account the race-hazard due to signal delay so as to get an accurate result. All of those imply that in the flow chat of Fig.4 we can directly average the behavior of input signals and develop a propagation algorithm based on behavioral probabilities to estimate the power consumption, as shown in Fig.7.

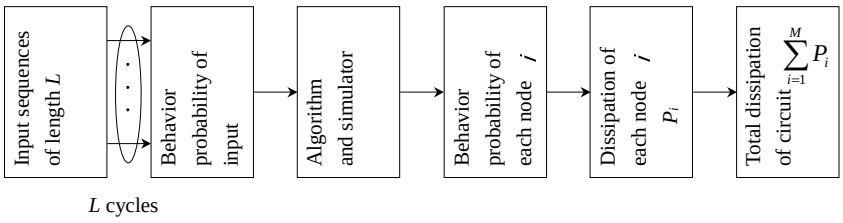

Figure 7 Algorithm flow for power estimation based on behavior probability

According to the above discussion the operations of three basic gates to the ternary behavior can be obtained while the signal delay is taken into account. Based on the operations of three basic gates shown in Fig.5, we can derive their corresponding propagation operations of the behavior probabilities. For the inverter shown in Fig.5(a) , we have

$$\begin{aligned} P(f^0) &= P(x^1), \\ P(f^1) &= P(x^0), \\ P(f^{1/2}) &= P(x^{1/2}). \end{aligned} \tag{12}$$

As to the output of the AND gate shown in Fig.5(b) we can obtain its 1-holding probability $P(g^1)$ and 0-holding probability $P(g^0)$:

$$P(g^1) = P(x^1) \cdot P(y^1), \tag{13}$$

$$P(g^0) = P(x^0) + P(y^0) - P(x^0) \cdot P(y^0).$$

The above equation can be expressed as

$$\overline{p(g^0)} = \overline{P(x^0)} \cdot \overline{P(y^0)}. \tag{14}$$

Note that the forms of Eqs.(13), (14) are similar to ones of Eqs.(4), (5) for signal probability. Besides, we can get the transition probability through the corresponding holding probabilities:

$$P(g^{1/2}) = 1 - P(g^0) - P(g^1). \tag{15}$$

With a similar discussion for the OR gate shown in Fig.5(c) we can derive the corresponding propagation relationships:

$$P(h^0) = P(x^0) \cdot P(y^0) \tag{16}$$

$$\overline{p(h^1)} = \overline{P(x^1)} \cdot \overline{P(y^1)} \tag{17}$$

$$P(h^{1/2}) = 1 - P(h^0) - P(h^1) \tag{18}$$

By using Eqs.(12)-(18), we can calculate all behavior probabilities for each node based on the circuit structure from the given input behavioral probabilities.

Example 2. We calculate the circuit in Fig.3 again by the propagation algorithm proposed above. First, we derive the holding probabilities for outputs of two AND gates:

$$\overline{p(x^0)} = \overline{P(a^0)} \cdot \overline{P(b^0)}, \qquad P(x^1) = P(a^1) \cdot P(b^1),$$

$$\overline{p(y^0)} = \overline{P(b^0)} \cdot \overline{P(c^0)}, \qquad P(y^1) = P(b^1) \cdot P(c^1).$$

As to the output of the OR gate in the circuit we can calculate its holding probabilities by Eqs.(16), (17) and revise its term with higher power due to re-convergence:

$$P(f^0) = P(b^0) + P(a^0)\cdot P(c^0) - P(a^0)\cdot P(b^0)\cdot P(c^0),$$

$$P(f^1) = P(a^1)\cdot P(b^1) + P(b^1)\cdot P(c^1) - P(a^1)\cdot P(b^1)\cdot P(c^1).$$

If we use the same signal probabilities of inputs in Example 1, i.e. $P(a) = 0.9$, $P(b) = P(c) = 0.5$. Similar to the discussion for deriving Eq.(7), the following holding probabilities can be derived from the corresponding signal probabilities:

$$P(x^0) = P(\bar{x})\cdot P(\bar{x}) = [P(\bar{x})]^2, \quad (19)$$

$$P(x^1) = P(x)\cdot P(x) = [P(x)]^2, \quad (20)$$

thus we can get the following holding probabilities of inputs:

$$P(a^0) = 0.01,\ P(a^1) = 0.81,\ P(b^0) = P(b^1) = 0.25,\ P(c^0) = P(c^1) = 0.25.$$

Substituting them into the above equations for *x*, *y* and *f*, we have $P(x^0) = 0.2575$, $P(x^1) = 0.2025$, $P(y^0) = 0.4375$, $P(y^1) = 0.0625$, $P(f^0) = 0.2519$, $P(f^1) = 0.2144$, respectively. Further we obtain all transition probabilities for each node in the circuit:

$$P(a^{1/2}) = 0.18,\ P(b^{1/2}) = P(c^{1/2}) = 0.5,\ P(x^{1/2}) = 0.54,\ P(y^{1/2}) = 0.5,\ P(f^{1/2}) = 0.5338.$$

Finally, the average switching activities for the circuit will be

$$\sum E_{SW} = P(a^{1/2}) + 2P(b^{1/2}) + P(c^{1/2}) + P(x^{1/2}) + P(y^{1/2}) = + P(f^{1/2}) = 3.2538.$$

By comparing the calculation with one in Example 1, we find that except for the input nodes all other three nodes (*x*, *y*, *f*) have higher transition probabilities since we have taken into account the possible glitches due to signal race. In contrast with the original signal propagation algorithm, we list both of them in Table 3, where $P(x=1)$ is $P(x)$ and $P(x=0)$ is $P(\bar{x})$. We can find that the propagation algorithm of behavior probability proposed in this section well correspond to the traditional propagation algorithm of signal probability.

Table 3 Signal probability algorithm vs. behavior probability algorithm

	Signal probability algorithm	**Behavior probability algorithm**
Normalization	$P(x=0)+P(x=1)=1$	$P(x^0) + P(x^{1/2}) + P(x^1) = 1$
Inverter $f = \bar{x}$	$P(f=0) = P(x=1)$ $P(f=1) = P(x=0)$	$P(f^0) = P(x^1)$ $P(f^1) = P(x^0)$ $P(f^{1/2}) = P(x^{1/2})$
AND gate $g = x\cdot y$	$P(g=1) = P(x=1)\cdot P(y=1)$ $\overline{p(g=0)} = \overline{P(x=0)}\cdot\overline{P(y=0)}$	$P(g^1) = P(x^1)\cdot P(y^1)$ $\overline{p(g^0)} = \overline{P(x^0)}\cdot\overline{P(y^0)}$
OR gate $h = x + y$	$P(h=0) = P(x=0)\cdot P(y=0)$ $\overline{p(h=1)} = \overline{P(x=1)}\cdot\overline{P(y=1)}$	$P(h^0) = P(x^0)\cdot P(y^0)$ $\overline{p(h^1)} = \overline{P(x^1)}\cdot\overline{P(y^1)}$
Transition probability	$P(x^{1/2}) = 2P(x=0)\cdot P(x=1)$	$P(g^{1/2}) = 1 - P(g^0) - P(g^1)$

V. CONCLUSIONS

A comparative precise and efficient power estimation technique is important for design of low power VLSI. However, the very large scale of the integrated circuits and the needed very long input sequence lead to the huge calculation, which will go beyond the actual memory and exerting time of computer. Thus, the probability algorithm is paid attention in power estimation since it can be released from the affection of long sequence. The previous researches focused on the algorithm based on signal probability. Although it avoids the huge calculation due to long sequence, it still suffers from the following two problems: (1) it is an algorithm based on signal values. However, the power dissipation depends on the transition behavior of signal rather than its value. Therefore the transforming procedure from the signal probability into the transition probability is necessary. (2) This transforming procedure from the signal probability into the transition probability does not take the glitches resulted from race hazard into account, thus the estimation accuracy is weakened.

Since the main power dissipation of CMOS circuits depends on signal's transition behavior we can discuss the issue directly from signal's behaviors. According to various signal values taken in the successive clock periods the signal behavior should be a quaternary variable: 0-holding, 1-holding, rising transition, and falling transition. If don't care the transition direction it can be simplified as a ternary one. Therefore, we believe that the method of multiple-valued logic may be useful to guide the related research about power analysis. In our previous work we have investigated the behavior propagation in CMOS circuits. [11] This paper focuses on the propagation algorithm of behavior probability. The presented research shows that power affection of glitches resulted from race-hazard has been taken into account in the power estimation based on propagation algorithm of behavior probability. Besides, the propagation algorithm of behavior probability well corresponds to the traditional propagation algorithm of signal probability. It offers a new probability algorithm for power estimation of VLSI. The further research in on the way.

Acknowledgement:

This work was supported in part by NNSF of China (No. 69773034) and NSF Grand of USA (No.53-4503-2694).

REFERENCES

[1] F. N. Najm, "A survey of power estimation techniques in VLSI circuits," *IEEE Transactions on VLSI Systems*, vol.2 , no.4, pp.446-455, 1995.

[2] Semiconductor Industry Association, Workshop Working Group Reports, (Irving, TX) 22-23,1992.

[3] M. Pedram, "Power minimization in IC Design: Principles and applications," *ACM Trans. on Design Automation of Electronic Systems*, vol.1, no. 1, pp.3-56, Jan. 1996.

[4] N. H. E. Weste and K. Eshraghian, Principles of CMOS VLSI Design: A Systems Perspectives, 2nd Edition, (Addison-Wesley Publishing Company, New York), 1993.

[5] K. P. Parker and E. I. McCluskey, "Probabilistic treatment of general combinational networks," *IEEE Transactions on Computers*, vol.24, no.6, pp.668-670, 1975.

[6] A. Ghost, et al., "Estimation of average switching activity in combinational and sequential circuits," in *Proc. 29th ACM/IEEE DAC*, pp.253-259, 1992.

[7] R. Marculescu, D. Marculescu, M. Pedram, "Switching activity analysis considering

spatiotempral correlations," in *Proc. IEEE ICCAD*, pp.294-299, 1994.

[8] T. L. Chou, K. Roy, S. Prasad, "Estimation circuit activity considering signal correlations and simultaneous switching," in *Proc. IEEE ICCAD*, pp.300-303, 1994.

[9] T. Uchino, et al., "Switching activity analysis using Boolean approximation method," in *Proc. IEEE ICCAD*, pp.20-25, 1995.

[10] M. Pedram, Q. Wu, and X. Wu, "A note of the relationship between signal probability and switching activity," in *Proc. of ASP-DAC*'97, Chiba, 117-120, Jan. 1997..

[11] X. Wu, B. Chen, M. Pedram, "Power Estimation in CMOS Circuits Based on Multiple Valued Logic," *Journal of MVL*, vol.4 , no.4, 1999.