논문 01-26-10A-14 한국통신학회논문지 '01-10 Vol.26 No.10A

게이트 단계에서의 소모전력 예측

정회원 황 인 기*, 조 준 동**

Gate-Level Power Estimation

In-Ki Hwang*, Jun-Dong Cho** *Regular Members*

요 약

최근의 전자업계의 동향을 살펴보면, 휴대 가능한 제품의 요구가 증대되고, 고 집적화 됨에 따라 제품의 크기와 동작속도 뿐만 아니라, 소모하는 전력의 양이 큰 문제로 대두되었다. 더욱이 휴대 장비에서는 전지의 양이 제한되어 있기 때문에, 소모 전력을 줄이는 것은 중요한 문제이다. 휴대 장비가 아니라고 해도, 높은 전력소모를 보이는 제품은 안정된 동작을 위해 값비싼 냉각장치 등을 필요로 한다. 이와 같이 전력소모를 줄이거나 예측할 수 있는 CAD tool에 대한 개발이 시급한 상황이다. 이제까지의 업계의 경향은 물리적 단계의 소모전력을 분석하는 tool의 개발 쪽에 한정되어 있었다. 하지만 이러한 하위 단계에서의 tool은 제품 생산 직전의 단계에서 이루어짐으로, 제품이 원하는 규격에 맞지 않을 경우, 재생산의 비용과 시간의 손실이 크다. 따라서 보다 상위 단계에서의 소모전력 예측 tool의 필요가 증가하고 있다.

본 논문에서는 이러한 기대에 발맞춰 gate 단계에서 소모전력을 예측할 수 있는 알고리즘을 제안하였다. 제안한 알고리즘은 입력 신호와의 의존성을 줄이기 위해 확률을 이용한 방법을 기초로 하였으며, 알고리즘의 정확성을 입증하기 위해 시스템을 설계, HSPICE를 이용한 시뮬레이션 결과와 비교하였다. 본 논문에서 제한한 알고리즘을 이용하여, 널리 알려진 시스템(ISCAS'85, ISCAS'89)의 소모전력을 예측한 결과, 시뮬레이션을 통해 얻은 결과와 비교해 봤을 때, 10% 이내의 오차 한도를 가진 것으로 분석되었다.

ABSTRACT

In recent, the power consumption was becoming the major problem in portable system because limited battery life. Not only portable devices but also high powered system need expensive cooling system and packaging. It's the reason why power estimation tool is so demanded.

In this paper, we proposed power estimation algorithm in gate-level. We analyze several system using our algorithm and compare this with result achieved using HSPICE. The error bound is under 10 %.

Ⅰ. 서 론

In the early of this decade, it becomes clear that power consumption was becoming a major problem. The demand for portable electric devices, very large integrated system and higher operation frequency has led to an increase emphasis on power consumption. Due to limited battery life, high power consumption is a major problem in the design of portable or mobile electronics. Even in line-powered equipment, such high power levels require expensive packages and heat-sinks. Thus, there is a need for CAD tools to help with the power management problem. While tools have long existed for analyzing power consumption at the lower levels of abstraction only recently have efforts been directly towards developing a high-

* 한국전자통신연구원 멀티미디어 통신팀 (ikhwang74@etri.re.kr)
** 성균관대학원 설계자동화 연구실(jdcho@skku.ac.kr)
논문번호 : 010158-0627, 접수일자 : 2001년 6월 27일

level power estimation capability.

In order to avoid costly redesign steps, power estimation tools are required that can assess the power dissipation early in the design process, before the final circuit-level design has been specified. This allows the designer to explore design trade-offs at a higher level of abstraction than was previously possible, reducing design time and cost[1][2].

The dominant source of power consumption in digital CMOS circuits is due to the charging and discharging of the node capacitances. The term dynamic power consumption refers to the sum of the short-circuit and capacitive power dissipations.

In this paper, we propose new algorithm for high-level power estimation. It estimates charging/ discharging ratio, the base of dynamic power. We verified this algorithm by applying to some system. First, we analyze the system with algorithm, then simulate the system with specific inputs until the variance of result is adequately small. Second, we compare analysed result and simulated data. We got simulation data from HSPICE simulation. There's under 10% error bound and some difference according to the types of architecture. In Chap.2, We enumerate previous work and describe proposed algorithm in Chap. 3. Chap4. shows the experimental result and Chap.5 is conclusion

Ⅱ. Previous Work

There are two major estimation technique. One is simulation based estimation and the other is non-simulation based estimation.

Simulation based techniques measure average power for specific inputs using circuit simulator like SPICE[5]. These techniques can test specific input such as hazard and spatial and temporal correlation and can handling various device model. It's the main advantage of these techniques. But, these techniques are so input dependent that it is hard to generate compact input set to test all node and condition. Also, it needs large memory space and long execution time. IRSIM[7], switch-level simulator, is the kind of this technique[8].

McPOWER[9] and MED[10] proposed another simulation based approach. The issues are how to select the input patterns to be applied in the simulators and how to decide when the measures power has converged close enough to the true average power. Normally, the inputs are randomly generated and statistical mean estimation techniques are used to decide when to stop, essentially a Monte Carlo method.

Non-simulation based techniques estimate power using probability, entropy, or macro modeling, etc. These approach proposed to solve pattern dependence problem and simplify delay model. Thus their accuracy is limited by the quality of delay models and the input specification. There are many proposed algorithm in non-simulation based approach like BDD(Binary Decision Diagram) [3][12], CREST[11], Correlation Coefficient[13], DENSIM[4], and Entropy based approach[14]. etc.

Ⅲ. Proposed High-level Power Estimation

In this chapter, We propose the method to estimate power with Boolean function. We assumed that there are no delay. In any gate, the input transition led the transition of output. The basic issue of my propose is that transition at input signal and some condition of output make transition of output.

We applied this algorithm to some basic system for verifying the accuracy of algorithm.

As the result, We have got under 10% error bound.

1. Power Estimation

The key issue to estimate power dissipation of any system is estimate its capacitance and transition density. In general, capacitance means area of the system and it is very hard to estimate why there are lots of way to achieve any given system. For the purpose of the system, speed, area, and power, etc., we can achieve system lots of way with same function. So what we called power estimation is bounded only estimate

transition density in many cases.

When the transition is occurred? Its the starting point of this algorithm. I thought that transition at input side and some condition of output led transition at output side. As an example, think about NAND gate. The Boolean function of NAND is $f=\overline{A \cdot B}$ and the transition conditions at input side are transition A only, transition B only, and transition both A and B. It is not true that every transition at input side led transition at output side. In the case of transition A only, input pair translate from (0,0) to (1,0) and from (1,0) to (0,0) are the condition for transition at output side. In the case of transition only B and both A and B at input side, we can get the condition of transition in same way as it is shown at Table. 1.

We can formula this as follow

$$\begin{aligned} P(tran\_out) = & P(tran\_only\ A) * P(tran\ condition) \\ & + P(tran\_only\ B) * P(tran\ condition) \\ & + P(tran\_both\ A,B) * P(tran\ condition) \end{aligned} \quad (1)$$

where P(x) means the probability that event x is occurred. The transition condition is the shadowed pair in Table. 1.

Table1. Transition condition for NAND gate

Input	tran only A	tran only B	tran both A and B
00	10	01	
01		00	10
10	00	11	
11			

We specified the input sequence as probability and transition density 0.5. If we apply this specified input signal to NAND gate, then based on (1)

$$\begin{aligned} P(tran\_out) = & 0.5*0.5*(0.5^2+0.5^2) \\ & +0.5*0.5*(0.5^2+0.5^2) \\ & +0.5*0.5*(0.5^2+0.5^2) \\ & = 0.375 \end{aligned} \quad (2)$$

then, we simulate NAND gate with HSPICE to obtain its experimental transition density. We simulate until the variance of input probability and output transition density is under 1%. As shown in Fig. 1.

We simulate about 350 times and the value of transition density is 0.375, thus it is very accurate value with no error compared with analyzed value.

It is sure that the result of NAND gate can be applied to AND, because AND gate is consisted with NAND and Inverter gate, and output transition density of Inverter gate is same as transition density of input. So, the difference between AND and NAND gate is only probability. If we assume that the probability of AND gate is x then the probability of NAND gate is (1-x).

In Fig. 1, 2, X-axis represents the run time and Y-axis represents density.

We test another basic gate such as NOR and OR gate. Like AND and NAND gate, there's no difference except probability. The analyzed transition density of NOR gate is 0.375 and the simulation result is Fig. 2.

We simulate NOR gate about 350 times, and the transition density obtained is about 0.385. The result has about 2.6% error compared with analyzed result 0.375.

In such a way, We extend the algorithm. For large system, we analyze primary gate then apply these data to the rest system.

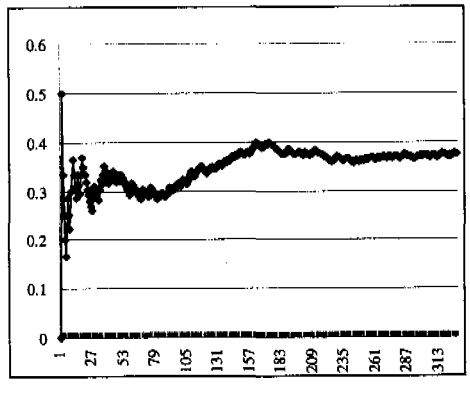


Fig. 1 Transition density for NAND gate

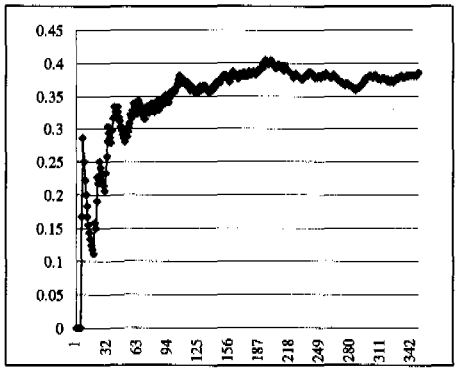


Fig. 2 Transition density for NOR gate

There are lots of way to construct same Boolean function for the purpose of system. As an example, think about Boolean function $f=A\cdot B\cdot C$, we can construct this function two ways, like $f=(A\cdot B)\cdot C$ and $f=A\cdot B\cdot C$. In first case, we can analyze the algorithm easily but we must do two times. In second case, we can analyze the algorithm just one time but it is hard because of long formulae. We must consider each of transition case like transition A only, B only, C only, both A and B, and transition A, B, and C in same time, etc. So, we consider each seven cases. It is shown in Table. 2.

Table2. Transition condirion for 3-input AND gate

Input	only A	only B	only C	both A,B	both A,C	both B,C	A,B,C
000	100	010	001	110	101	011	
001	101	011	000		100	010	110
010	110	000	011	100		001	101
011		001	010	101	110	000	100
100	000	110	101	010	001		011
101	001		100	011	000	110	010
110	010	100		000	011	101	001
111							

In case $f=(A\cdot B)\cdot C$, the analyzed transition density is 0.266, and the other case, the analyzed transition density is 0.219 as shown in Fig. 3.

We simulate about 700 times, and the simulation result for $f=A\cdot B\cdot C$ is about 0.252, so the error is 5.5% and 13% in each case. Thus, for the Boolean function $f=A\cdot B\cdot C$, the case of $f=(A\cdot B)\cdot C$ is more accurate than $f=A\cdot B\cdot C$.

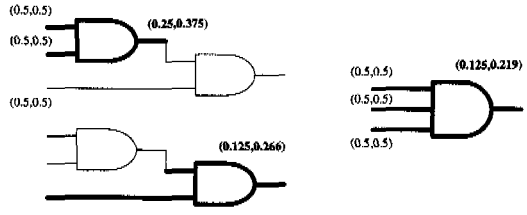


Fig. 3 3-input AND gate architecture

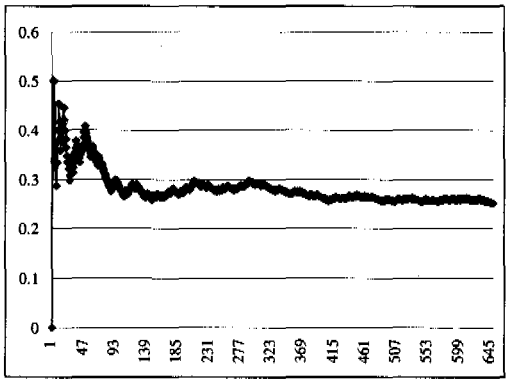


Fig. 4 Transition density for 3-input AND gate

Ⅳ. Experimental Result

As a first step toward a high-level power estimation capability, If the function represented $y=f(x_1, x_2, \ldots, x_n)$ then, we normalize (1) like (3).

$$P(tran\_out)=\sum_{i=1}^{n}\{P(tran\_x_i)\prod_{k\neq i}P(tran\_x_k)\}\{P(\prod y_{0\to 1})+P(\prod y_{1\to 0})\} \quad (3)$$

We assumed zero-delay model and specified input set, 0.5 probability and 0.5 transition density. We simulated some basic circuit, ISCAS'85, and ISCAS'89 circuits until the transition density was settled. The simulation time was from 800 to 1600 times.

1. Half Adder

The half adder consisted with two inputs and two outputs. The function of each output is $carry=a_0\cdot b_0$, and $sum=a_0\oplus b_0$. The following is the architecture of half adder(Fig. 5)

The analyzed transition density of output carry is 0.375 as it was shown before chapter, and

output sum is 0.439. The simulation result was shown in Fig. 6, 7.

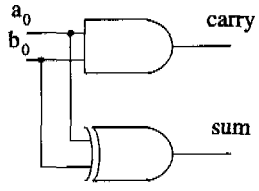


Fig. 5 Architecture for half adder

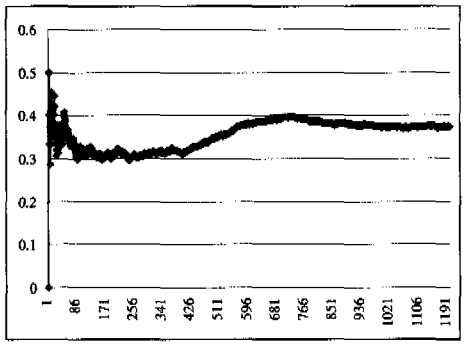


Fig. 6 Transition density for output carry

So the error of each output is 0% and 3.5%. In case of output sum, we analyzed two ways. First, we construct $sum = a_0 \cdot \overline{b_0} + \overline{a_0} \cdot b_0$, then the analyzed result was 0.439, so the error was 3.5%. Second, we construct $sum = a_0 \oplus b_0$, then the analyzed result was 0.5, so the error was 9%. Like three input AND gate, there was some difference according to architecture types.

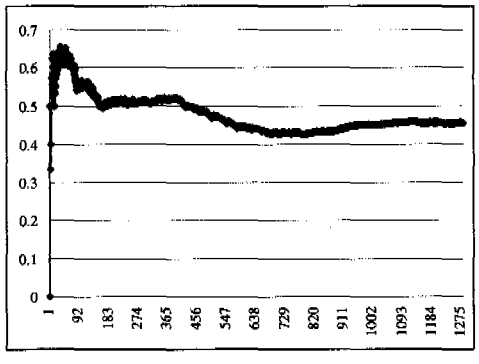


Fig. 7 Transition density for output *sum*

2. Full Adder

The full adder consisted with three inputs and two outputs. The function of each output is $carry = a_0 \cdot b_0 + b_0 \cdot c_0 + c_0 \cdot a_0$, and $sum = a_0 \oplus b_0 \oplus c_0$. The following is the architecture of full adder(Fig. 8)

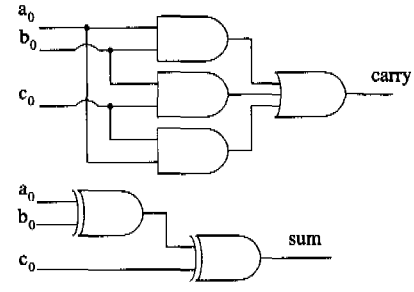


Fig. 8 Architecture for full adder

The analyzed transition density of output carry is 0.439 as it was shown before chapter, and output sum is 0.5. The simulation result was shown in Fig. 9, 10.

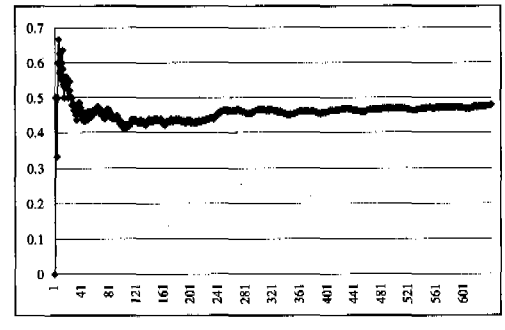


Fig. 9 Transition density for output carry

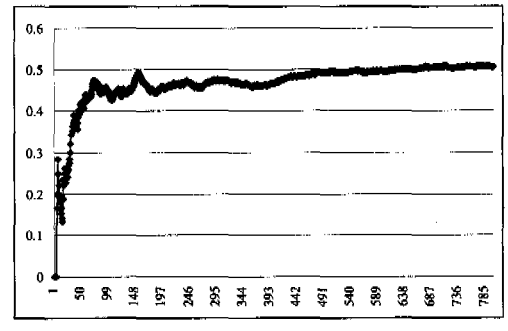


Fig. 10 Transition density for output sum

The simulation result of carry is 0.48, then the error is 8.5% compared with analyzed result. The simulation result of sum is 0.505, then the error is 0.1%.

3. Bypass Adder

We constructed 4-bit bypass adder consisted with 4 full adder blocks and carry selector block. So, it has total 33 gates. It was shown in Fig. 11.

We represent output carry and sum1, why output sum has same function. The function sum was same with represented in chapter full adder, and the function of carry was

$f = carry_0 \cdot cont.sig. + carry_4 \cdot \overline{cont.sig.}$, where

$cont.sig. = (a_0 \oplus b_0)\cdot(a_1 \oplus b_1)\cdot(a_2 \oplus b_2)\cdot(a_3 \oplus b_3)$The simulation result was shown in Fig. 12, 13.

The analyzed result for carry was 0.439 and sum1 was 0.5, so the error for each output was 10%, 0.6%.

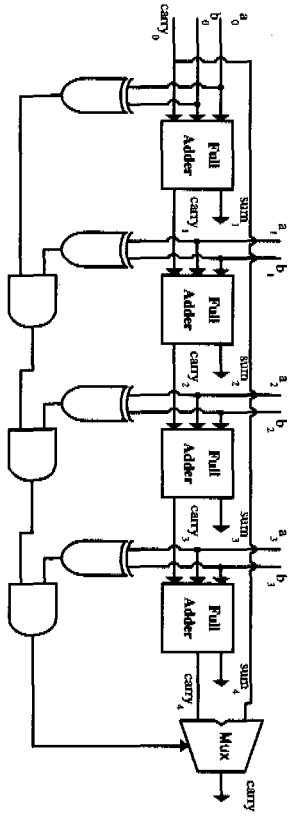


Fig. 11 Architecture for 4-bit bypass adder

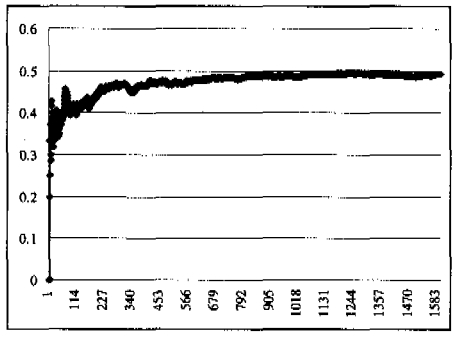


Fig. 12 Transition density for output *carry*

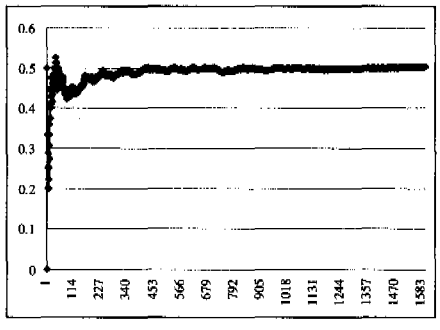


Fig. 13 Transition density for output *sum1*

4. Carry Look Ahead Adder(74182)

We constructed CLA, and it has 19 gates. We represented output cx, cy and p. The function for each output was like follow.

$$c_x = \overline{G_0} + P_0 c_n$$
$$c_y = G_1 + (G_0 + c_n P_0) P_1$$
$$\overline{p} = \overline{(P_3 P_2 P_1 P_0)} \quad (4)$$

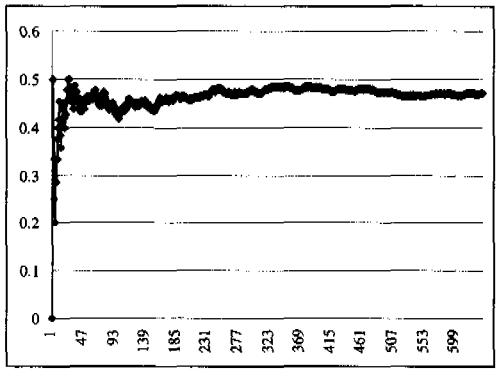


Fig. 14 Transition density for output *cx*

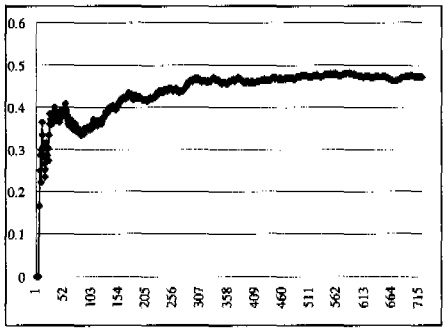


Fig. 15 Transition density for output *cy*

The analyzed result were 0.44, 0.42, and 0.16 in order and the simulated result were 0.47, 0.471 and 0.172. So the error were 6.4%, 10.8%, and 6.9%.

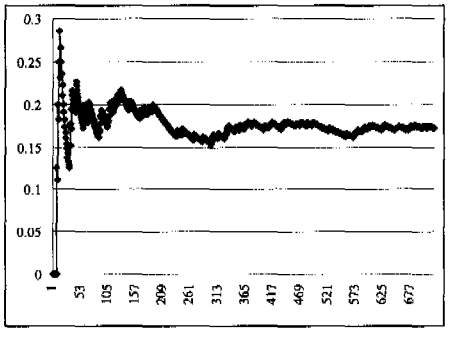


Fig. 16 Transition density for output p

5. Fast Adder (74283)

We constructed fast adder, and it has 36 gates. We represented output c0, c1, c2 and s.

The analyzed result were 0.472, 0.466, 0.468, and 0.5 in order and the simulated result were 0.44, 0.503, 0.444, and 0.459. So the error were 6.8%, 7.4%, 5.1% and 8.2%.

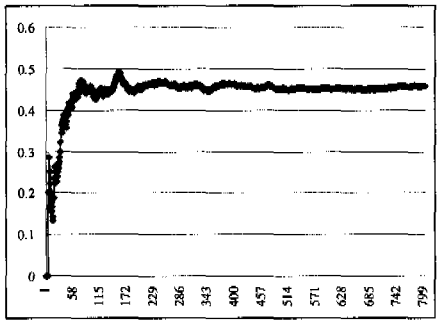


Fig. 17 Transition density for output *s*

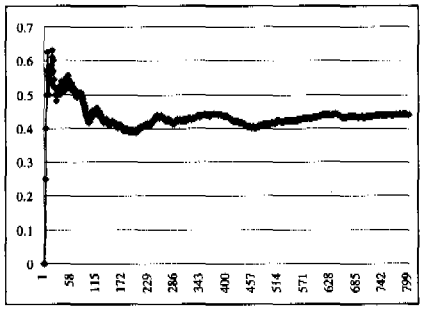


Fig. 18 Transition density for output *c0*

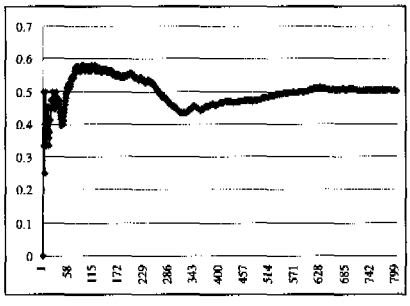


Fig. 19 Transition density for output *c1*

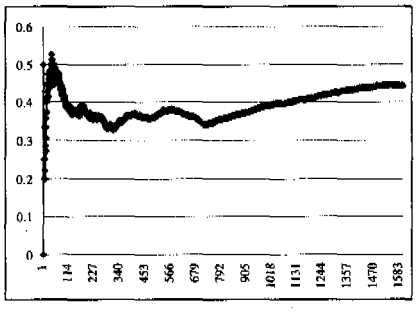


Fig. 20 Transition density for output *c2*

6. 4-bit Magnitude Comparator (74L85)

We constructed 4-bit magnitude comparator, and

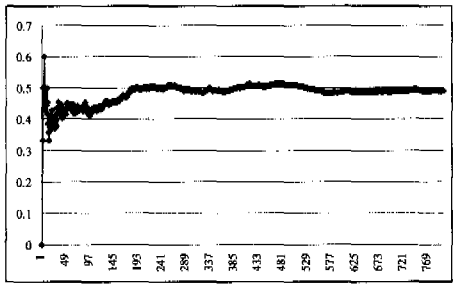


Fig. 21 Transition density for output *agb*

it has 21 gates. We represented output aeb, agb, and bga.

The analyzed result were 0.223, 0.494, and 0.494 in order and the simulated result were 0.24, 0.49, and 0.478. So the error were 7.1%, 0.8%, and 3.2%.

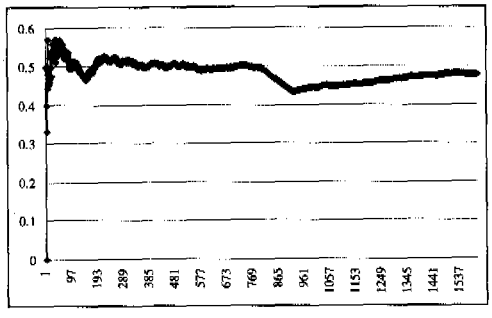


Fig. 22 Transition density for output *bga*

V. Conclusion

In this paper, we proposed an algorithm for high-level power estimation. First, we established algorithm based on the concept of probability and transition density. We thought that transition was occurred according to the transition at input side. But, it is not true that transition at input side always make transition at output side. So, we thought that there's some condition for transition. The answer was condition of input pair. We insert this issue to the formulae.

$$P(tran\_out)=\sum_{i=1}^{n}\{P(tran\_x_i)\prod_{k\neq i}P(tran\_x_k)\}\{P(\prod y_{0\to 1})+P(\prod y_{1\to 0})\}$$

After, establishing algorithm, we tested started from basic circuit like NAND, NOR gates, to complex circuit like 74L85, 74283(ISCAS), etc. We do simulate until the variance of value of transition density was settled.

Together with simulating, we analyze the circuits with our algorithm. The result was shown at table 3.

As result, we could get under 10% error bound. But there's some problem that there are difference according to the type of architecture. It is important why we construct any system matching to the purpose of system, such as power, area, speed. Another problem is that it need much time for large circuit why according to the algorithm, we do analyze gate to gate from primary inputs to outputs.

Table 3. The result of proposed algorithm

system	output	Analyzed value	Simulated value	error
half adder	carry	0.375	0.375	0.00%
	sum	0.439	0.455	3.50%
full adder	carry	0.439	0.48	8.50%
	sum	0.5	0.505	0.10%
bypass adder	carry	0.439	0.493	10.00%
	sum1	0.5	0.503	0.60%
74182	Cn+x	0.44	0.47	4.40%
	Cn+y	0.42	0.471	10.80%
	P	0.16	0.172	6.90%
74283	C0	0.472	0.44	6.80%
	C1	0.466	0.503	7.40%
	C2	0.468	0.444	5.10%
	S	0.5	0.459	8.20%
74L85	aeb	0.223	0.24	7.10%
	agb	0.494	0.49	0.80%
	bga	0.494	0.478	3.20%

Bibliography

[1] Farid N. Najm, "A Survey of Power Estimation Techniques in VLSI Circuits", *IEEE TVLSI*, 1994.

[2] W. Nabel and J. Mermet, *Low Power Design in Deep Submicron Electronics*, ASI, pp. 135-174, 1996.

[3] M. A. Cirit, "Estimating dynamic power consumption of CMOS circuits", *IEEE Int. Conf. Computer Aided Design*, pp. 534-537, Nov. 9-12, 1987.

[4] F. Najm, "Transition density, a stochastic measure of activity in digital circuits", *28th ACM/IEEE Design Automation Conference*, Jun. 17-21, pp. 644-649, 1991.

[5] T. Quarles, "The SPICE3 implementation guide," *Tech. Rep. M 89-44*, Electronics Research Laboratory, Univ. of California, Berkeley, Apr. 1989.

[6] C. X. Huang, B. Zhang, A. C. Deng, and B. Swirski, "The design and implementation of PowerMill," *Proc. IEEE int. Symp. on Low Power Design*, pp. 105-110, Apr. 1995.

[7] A. Salz and M. A. Horowitz, "IRSIM: An

incremental MOS switch-level simulator," *Proc. 26th IEEE/ACM Design Automation Conference*, pp. 173-178, Jun. 1989.

[8] K. K. Parhi and T. Nishitani, *Digital Signal Processing for Multimedia Systems*, Marcel Dekker, pp. 741-769, 1999.

[9] R. Burch, F. Najm, P. Yang and T. Trick, "McPOWER: A Monte Carlo approach to power estimation," *IEEE/ACM Int. Conf. Computer-Aided Design*, pp. 90-97, Nov. 8-12, 1992.

[10] M. Xakellis and , F. Najm, "Statistical estimation of the switching activity in digital circuits," *31st ACM/IEEE Design automation Conf.*, pp. 728-733, 1994.

[11] F. Najm, R. Burch, P. Yang, and I. Hajj, "CREST-A current estimator for CMOS circuit," *IEEE Int. Conf. Computer-Aided Design*, Nov. 7-10, pp. 204-207, 1988.

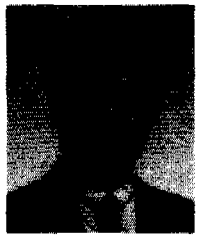
[12] A. Ghosh, S. Devadas, K. Keutzer, and J. White, "Estimation of average switching activity in combinational and sequential circuits," *29th ACM/IEEE Design Automation Conf.*, Jun 8-12, pp. 253-259, 1992.

[13] C. Y. Tsui, M. Pedram, and A. M. Despain, "Efficient estimation of dynamic power consumption under a real delay model," *IEEE Int. Conf. Computer-Aided Design*, Nov. 7-11, pp. 224-228, 1993.

[14] M. Nemani and F. Najm, "Towards a high-level power estimation capability," *IEEE Trans. Computer-Aided Design*, Vol. 15, No. 6, pp. 588-598, Jun. 1996.

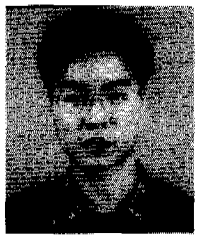
[15] S. Gupta and F. Najm, "Power macromodeling for high level power estimation," *Proc. 34th Design Automation Conf.*, pp. 365-370, Jun. 1997.

조 준 동(Jun-Dong Cho) 정회원



1980년 2월 : 성균관대학교 전자공학과 졸업
1983년 6월~1987년 8월 : 삼성전자 CAD 팀 근무 (연구원, 팀장)
1989년 9월 : Polytechnic Univ. Brooklyn, NY 전산학과 졸업
1993년 7월 : Northwestern Univ. 전산학과 졸업
1995년 3월~현재 : 성균관대학교 전기전자컴퓨터공학부(부교수)
IEEE Senior Member
<주관심 분야> 디지털통신, 무선통신, 이동통신, 저전력 설계 기술, VLSI CAD

황 인 기(In-Ki Hwang) 준회원



1999년 2월 : 성균관대학교 전자공학과 졸업
2001년 2월 : 성균관대학원 전기전자컴퓨터공학과 졸업
2001년 2월~현재 : 한국전자통신연구원 근무 (연구원)
<주관심 분야> 이동통신, 저전력 설계 기술