DELAY DEPENDENT POWER OPTIMISATION OF COMBINATIONAL CIRCUITS USING AND-INVERTER GRAPHS

Rashmi Mehrotra*, Tom English*, Emanuel Popovici*†, Michel Schellekens†
*Dept of Microelectronic Engineering / †Centre for Efficiency-Oriented Languages
University College Cork
Cork, Ireland
r.mehrotra@student.ucc.ie, tom.english@ue.ucc.ie
e.popovici@ucc.ie, m.schellekens@cs.ucc.ie

ABSTRACT

Dynamic power dissipation due to switching activity has been one of the major concerns in power optimisation. By approximating the switching activity of circuit nodes as internal switching probabilities using *AND Inverter graphs* (AIGs), it is possible to estimate and optimise power dissipation. In our work, the internal switching probabilities are derived via probabilistic estimation method under a variable delay model. Local reordering delay dependent rules are applied on the AIG nodes for the minimisation of overall sum of switching probability. Optimisation techniques such as simulated annealing for conversions from higher switching probability network to lower switching probability network are used in this paper. Combinational circuits used in our work are up to 100k gates and they are implemented using ROM.

I. INTRODUCTION

The dynamic power dissipation, also called the switching power is given by

$$P_{dynamic} = \alpha.C_l.V_{dd}^2.f_{clk} \quad (1)$$

where α is the switching activity factor (also called transition probability), C_l is the overall capacitance to be charged and discharged in a reference clock cycle, V_{dd} is the supply voltage and f_{clk} is the clock frequency. In recent years, a variety of probabilistic methods have been developed to estimate and reduce the average switching activity of the combinational circuits [1], [2] and [3]. Similar probabilistic methods of switching activity estimation are used in our work but using *And Inverter Graphs* (AIGs) [4]. The AIGs represent the functionality of the circuit to be realised. We have used AIGs in our paper as they provide a much compact representation of combinational circuits than *Binary Decision Diagrams* (BDDs) [5] . Formal verification is easier as they can be easily mapped to a netlist. Delay parameter can be easily incorporated in AIGs unlike BDDs. This makes switching probability estimation under a variable delay model much easier and faster than using BDDs. The tool *switching probability estimator* (SPE) locates switching instants (possibility of glitches) for accurate estimation. Once the switching probability is known at each of the nodes, various delay dependent reordering rules are applied via tool RESWITCHD implemented in C as a sub-package in ABC [4]. A previous version of the tool with delay independent rules has been implemented in our previous work. The tool RESWITCHD minimises the switching probability at each node by trading off some delay. Also since switching power is a product of switching probability and capacitance, we have defined capacitive load at each node to be the number of fanouts at that node. The tool actually minimises the product of switching probability and capacitive load.

II. SWITCHING PROBABILITY ESTIMATION METHOD

Estimation of switching activity is a challenging process, input probability, nodes functionality, gate delays are some of the factors that affect switching activity. We will deal with all these factors gradually. In this paper we have assumed that the inertial gate delays (node delays) are given and the primary input signals are mutually independent. Let $t_{d,n}$ denote the delay at node n.

A. Gate Delay

There are various types of gate delay models, zero delay, unit delay and variable delay are a few among them. In this work, we construct a variable delay model assuming delays of the gate (AIG nodes) to be proportional to their fanout [3]. Assuming arrival time at the primary inputs to be zero, we can calculate the arrival time at each of the AIG node. Let $t_{arr,a}$

978-1-4244-6683-2/10/$26.00 ©2010 IEEE

represents the arrival time at node a. Consider node c to be an AND node with fanins a and b as shown in Fig. 1. So the arrival time at node c will be

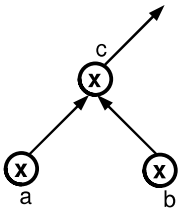

Fig. 1. AIG node.

$$t_{arr,c} = max(t_{arr,a}, t_{arr,b}) + t_{d,c} \quad (2)$$

The difference in arrival times of signals at a gate input, (the difference in the arrival times of the fanins at the AND node), leads to spurious transitions, also called as glitches. These spurious transitions play a major role in power dissipation. Based on the fanins arrival times and the delay of the node, time instants at which a possible signal transition can occur have been calculated and associated with the node. Let $t_{sw,a}$ denote a switching instant at node a. Since an AIG network can be easily mapped to a netlist, delay assignation, arrival time estimation and switching instant calculation is easier than BDDs.

B. Nodes Functionality

The time parameter plays a critical role in the power consumption estimation. Hence, the exact description of a logic circuit should include not only its logic behavior, but also the time dependency among the logic signals. Furthermore, since the glitch generation is strongly dependent on time, a modified Boolean function, which describes the logic and timing behavior of each signal, is needed. We call this *Real Delay Boolean Function* (RDBF). For more details refer to [1] and [3]. Such RDBF functions can also be built using BDDs, but for very large circuits it becomes time consuming and memory consuming. AIGs are comfortable with delay manipulation unlike BDDs. Consider the AIG of the logic function f = $x_1x_2x_3$ as shown in Fig. 2.

The logic behavior of the node f is described in time domain by the following RDBF:

$$f = x_1(t-5d)x_2(t-5d)x_3(t-3d) \quad (3)$$

The signal f may switch at two time instants then i.e. $t_{sw,f}$=3d and $t_{sw,f}$=5d. These two time instants are actually the switching instants. Set of switching instants of node f is denoted as $T_{sw,f}$.
Hence this method aims at a series of switching

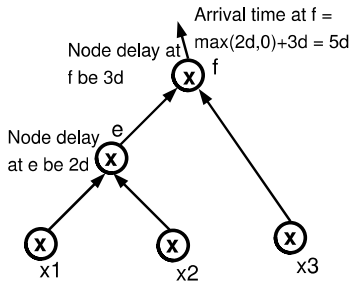

Fig. 2. AIG of logic function $x_1x_2x_3$.

activity calculations at certain switching time points. Let $P_{sw,t_{sw,x}}$(x) denote the switching probability at node x at the switching instant $t_{sw,x}$. The set of all nodes of the network is denoted as N. Hence in an AIG network, the total sum of switching probability at all the nodes of the network is represented as

$$SP(f) = \sum_{x \in N} \sum_{t_{sw,x} \in T_{sw,x}} P_{sw,t_{sw,x}}(x) \quad (4)$$

In an AIG network these switching instants can be easily calculated as we traverse from lower levels to higher levels. Once we associate the arrival times and node delays corresponding to each node of the network, the switching instants can be easily calculated. For the switching probability calculation, consider the same node c with fanins a and b. Based on the technique mentioned in [7], consider the value of c at two different observation times c_{t1} and c_{t1+dt}. Let t1 be one of the switching instants $\in T_{sw,c}$ and 'dt' be a small interval of time. If the value of c at those instants is not same, c is considered to switch. Hence the switching probability $P_{sw,t1}$(c) can be estimated as follows. Let the node delay at c be $t_{d,c}$. P_t(a) is the probability of 'a' being '1' at time 't'.

$$\begin{aligned} P_{sw,t1}(c) &= P((c_{t1}=0) \cap (c_{t1+dt}=1)) \\ &\quad + P((c_{t1}=1) \cap (c_{t1+dt}=0)) \end{aligned}$$

$$\begin{aligned} P_{sw,t1}(c) &= P_{t1-t_{d,c}}(a) P_{t1-t_{d,c}}(b). \\ &\quad [1 - (P_{t1+dt-t_{d,c}}(a) P_{t1+dt-t_{d,c}}(b))] \\ &\quad + P_{t1+dt-t_{d,c}}(a) P_{t1+dt-t_{d,c}}(b). \\ &\quad [1 - (P_{t1-t_{d,c}}(a) P_{t1-t_{d,c}}(b))] \end{aligned}$$

C. Input Probability

Design approaches, which take into account signal probability, show good results with respect to power optimisation. Most recently a lot of methods have been presented which primarily focus on the probabilistic behavior of the processed data, like bus encoding techniques [8]. This led researchers to

exploit redundancy in the data transmitted on the bus. According to the equations above, we need a probability profile or a waveform corresponding to each primary input, with respect to each bit of an n-bit bus used at every interval of time. The Real Delay Boolean Function of each node is written in term of primary inputs whose probability profiles are known and hence switching probability is predicted.

III. BACKGROUND BEHIND THE RESWITCHD TOOL

The RESWITCHD tool is based on four delay dependent reordering rules namely R1, R2, R3 and R4. The four reordering rules help us to reduce the overall switching probability keeping in mind the delay distribution. These rules are applied on the AIG network once the synthesis is done in ABC. The scripts used in ABC are namely 'resyn' and 'resyn2' which are most efficient synthesis scripts for reducing complexity. After this, the four rules are applied on each node of the AIG network. These four rules are actually based on the common rules of Boolean Algebra. The rules of associativity i.e. a ∪ (b ∪ c) = (a ∪ b) ∪ c and a ∩ (b ∩ c) = (a ∩ b) ∩ c are dealt in Rule R1. Rule R2 works on the same lines as Rule R1 but it also keeps a check on the redundant node creation. The rules of associativity with four variables i.e. (a ∪ b) ∪ (c ∪ d) = (a ∪ c) ∪ (b ∪ d) and (a ∩ b) ∩ (c ∩ d) = (a ∩ c) ∩ (b ∩ d) and the rule of commutativity i.e. a∩b = b∩a and a∪b = b∪a are dealt in Rule R3. The rule of distributivity i.e a ∩ (b ∪ c) = (a∩b) ∪ (a∩c) are dealt in Rule R4. The rest of the boolean algebra rules are already dealt in ABC synthesis scripts. In this way using the Boolean Algebra rules, we cover all possible local reordering rules on each node so as to alter the switching probability keeping the functionality of the circuit same. The rules R1, R2, R3 and R4 are briefly described below. Note that in all these reordering rules, we have considered the product of switching probability and number of fanouts as the parameter for the reordering to take place.

The rule R1 is explained in Fig. 3. Consider node d to be one of the fanins of node e with no other fanout than e and no compliment edge between node d and node e. At this stage if we swap node c with either of the nodes a or b, the functionality of node e will not change. Secondly since node d does not have any other fanouts, the swapping will not effect the overall AIG network. Also due to swapping, the arrival times at node d and node e will change and hence the switching instants will also change. After each swapping we need to update the arrival times

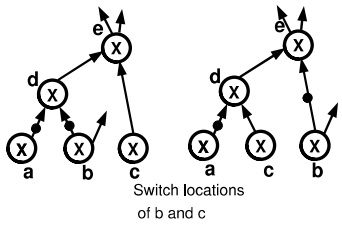

Fig. 3. Rule R1 on AIG

for the rest of the nodes. So due to this swapping, we have different fanins and different switching instants at node d and node e. The switching probability at node d and node e might decrease or increase. Consider t1 to be the switching instant at node d before swapping and t2 be the switching instant at node d after swapping. Consider node a and c to be grouped after swapping. Node delay at node d be $t_{d,d}$. The initial and final switching probability at node d will be as follows.

$$\begin{aligned} P_{sw,t1}(d) &= P_{t1-t_{d,d}}(a)P_{t1-t_{d,d}}(b). \\ &\quad [1-(P_{t1+dt-t_{d,d}}(a)P_{t1+dt-t_{d,d}}(b))] \\ &\quad +P_{t1+dt-t_{d,d}}(a)P_{t1+dt-t_{d,d}}(b). \\ &\quad [1-(P_{t1-t_{d,d}}(a)P_{t1-t_{d,d}}(b))] \end{aligned}$$

$$\begin{aligned} P_{sw,t2}(d) &= P_{t2-t_{d,d}}(a)P_{t2-t_{d,d}}(c). \\ &\quad [1-(P_{t2+dt-t_{d,d}}(a)P_{t2+dt-t_{d,d}}(c))] \\ &\quad +P_{t2+dt-t_{d,d}}(a)P_{t2+dt-t_{d,d}}(c). \\ &\quad [1-(P_{t2-t_{d,d}}(a)P_{t2-t_{d,d}}(c))] \end{aligned}$$

We have made cases within the tool to direct the swapping of a particular node only when the switching probability is getting reduced.

The rule R2 is explained in Fig. 4. In rule R2, consider the conditions required for rule R1 holds true and we are swapping b and c such that a and c are grouped. There can be a case when there already exists an AIG node with same fanins a and c and with the same complemented attributes as they have with nodes d and e respectively. Hence such kind of swapping will create and deduce redundant nodes with same fanins a and c. These nodes are removed from the AIG network leaving one in the graph. The rule R3 is explained in Fig. 5. Consider node g with fanins e and f and with no complimented attributed with either of them. Secondly e and f does not have any other fanouts except g. This case allows us to swap either of the nodes a and b with either of the nodes c and d. The functionality at node g will not change and hence will not affect the overall AIG network. This swapping will change the

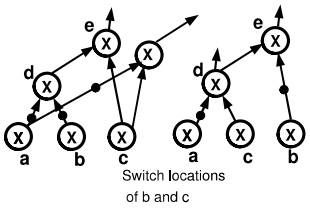

Fig. 4. Rule R2 on AIG

switching probability as well as the arrival times at node e, node f and node g. We need to compare the present and the final sum of switching probabilities at nodes e, f and g for the implementation of Rule R3. This swapping will give new option of nodes which may either decrease the switching probability or may create redundant nodes. In the Fig. 5, if we swap nodes b and c, b and d nodes are then grouped together. Hence rule R3 actually combines the options available by rules R1 and R2. The rule

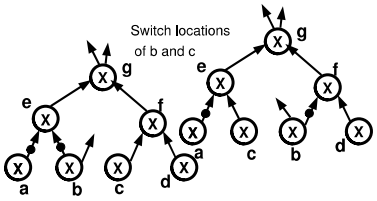

Fig. 5. Rule R3 on AIG.

R4 is explained in Fig. 6. Rule R4 is based on the rules of distributivity. Consider two consecutive nodes named d and e such that d is the fanin of e with no other fanouts except e and there is complemented edge between d and e. In this case, we can write e = (a'b')'c as (ca)+(cb). So we need to create one extra node with fanins c and b, while node d has fanins c and a. This is done to make an AIG according to the functionality (ca)+(cb). The new node shall have the same node delay as of node d. Rule R4 although increases the node count by one but sometimes reduces the switching probability of the network to a great extend.

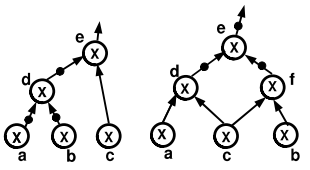

Fig. 6. Rule R4 on AIG.

IV. OPTIMISATION ALGORITHMS

The reordering rules namely R1, R2, R3 and R4 can be applied on each node of the AIG network depending upon the conditions mentioned above. To decide which rule will be highly advantageous for the timing driven power optimisation, we apply 'simulated annealing' (SMA) [6] as the optimisation algorithm in this work. In the SMA method, we intend to minimise the function SP(f) i.e total sum of the product of the switching probability and capacitive load at each node of the AIG network representing the functionality 'f' of the combinational circuit. At each step, the SMA considers some neighbour s' of the current state s, and probabilistically decides between moving the system to state s' or staying in state s. Here s and s' stands for initial and final AIG network respectively. This procedure is explained in Fig. 7. Instead of probabilities, we have

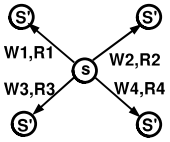

Fig. 7. Simulated annealing method.

defined weights namely W1,W2,W3 and W4 for each movement at each node. For the weight definitions, we define two new variables $\triangle$c and $\triangle$t. $\triangle$c is the difference in switching probability from initial network to final network on the application of a particular rule i.e. SP_i(f) - SP_f(f). $\triangle$t is the difference in the arrival times at the node fanout n in final network and initial network on the application of that particular rule i.e $t_{arr,n}$ - $t_{arr,n}$'. The weights are defined as follows:-
If $\triangle$c is greater than zero and $\triangle$t is less than zero,

$$W = e^{\triangle c*(-1*\triangle t)} \tag{5}$$

If $\triangle$c is greater than zero and $\triangle$t is greater than zero,

$$W = e^{\triangle c/\triangle t} \tag{6}$$

If $\triangle$c is greater than zero and $\triangle$t is equal to zero,

$$W = e^{\triangle c} \tag{7}$$

If $\triangle$c is less than zero

$$W = 0 \tag{8}$$

Lastly if the increase in percentage delay at a node due to a particular rule is greater than 10%, the value of W goes to zero. The weights W1, W2, W3 and W4 correspond to rule R1, R2, R3 and R4 individually. After the application of SMA at each node of the AIG, the initial and final networks are checked via Synopsys Formality for the formal verification.

V. EXPERIMENTAL RESULTS

We applied the tool RESWITCHD on a set of MCNC Benchmark circuits and large combinational ROM circuits. For the implementation of ROM combinational circuits, given a set of input variables and number of output lines, the designer needs only to specify a ROM program table that provides information for the required paths in the ROM. In our experiments, we made a matrix of size 16x2^{14} which is actually a look up table specifying paths in ROM circuit. Number of rows are 16 which are actually the number of output lines. Number of columns are 2^{14} which are actually the combinations of 14-bit binary digit corresponding to 14 input variables. The matrix is filled with random 4 digit hex data column wise. Depending upon the size of ROM required, we range the number of input variables from 1 to 14, and range the number of output lines from 1 to 16. The circuits have names specifying the number of input variables and output lines. Like ROM$_{o8,i6}$ specifies that there are 6 input variables and 8 output lines. Table I presents the percentage decrease in switching probability along with the node count and delay information. The average decrease in switching probability is 27.21% with 3.4% percent increase in delay and 7% increase in node count. NC stands for node count. SP stands for switching probability and max t_{arr} implies the maximum arrival time among all the nodes hence means the longest path in the network. After each reordering technique, the arrival time at each node changes due to change in the path or level. On each reordering we have to ensure that there is large decrease in switching power and a small increase in delay. However there can be a case when a particular reordering not only reduces switching power but also alter the longest path which is smaller than the earlier longest path. This will lead to over all decrease in switching power and delay. Also here node count is actually a measure of the complexity of the network. Lower number of node count will hence show that the network has been optimised structurally.

VI. POWER RESULTS

According to the Fig. 8, we performed power experiments on a set of some of the above circuits. In the figure, the tool AIG2Net is used for the manual mapping of the AIG network to a mapped netlist in Verilog using the TSMC 65GP CMOS standard cell library cells of ANDs, Inverters and buffers. A user defined SAIF is provided which actually specifies the switching probabilities at each node of the network. We used Synopsys Design Compiler for power estimation on the two netlist which are obtained with and without using the tool RESWITCHD.

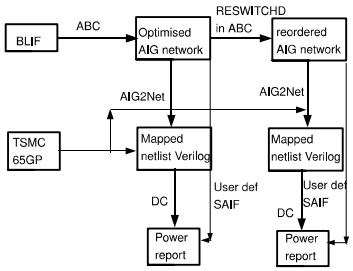

Fig. 8. Power estimation and optimisation.

Table II shows some experimental results. Average power decrease is 18.08% with slight increase in delay and area. The power reduction ranges from 13% to 36%. Here P_{ABC}, T_{ABC} and A_{ABC} are the power, timing and area results from the netlist obtained via ABC. P_{RW}, T_{RW} and A_{RW} are the power, timing and area results from the netlist obtained via RESWITCHD. The power optimisation results are improved further when we consider the product of switching probability and capacitive load as a parameter for reordering, not just switching probability.

VII. CONCLUSION

We presented an efficient switching probability estimation and optimisation tool using a specific delay model using AIG networks. We optimise the total sum of switching probability on an AIG which is our choice for representing the functionality of the combinational circuit. We also kept a check on the delay of the circuit. This meant that reordering a node was only allowed when the percentage increase in delay at that node was less than 10%. Applying the various reordering rules via RESWITCHD, we decreased the switching probability with only a slight increase in delay and node count. We applied the tool RESWITCHD on a set of MCNC Benchmark circuits and large combinational ROM circuits with node count from 200 to 100k. The average decrease in switching probability is 27.21% with a slight increase of 3.42% delay. The resulting circuit that is obtained by mapping the AIG after the application of RESWITCHD has shown in simulation reduced power dissipation characteristic with minimal overhead with respect to timing and area. The average power reduction is 18.08%. The networks before reordering and after reordering are verified via Synopsys Formality. In this way, the tool RESWITCHD allowed us to have a timing driven power optimisation on large combinational circuits.

TABLE I
NC, DELAY, SP ON THE APPLICATION OF RESWITCHD

Circuit	NC before	max t_{arr} before	SP before	NC inc	max t_{arr}inc	SP dec
i10	1878	118	91.31	3.1%	3.27%	18.58%
t481	957	55	38.7	4.5%	-1.8%	33.44%
dalu	1111	91	37.42	8.3%	5.2%	24.73%
vda	643	37	44.23	0%	5%	25.54%
pair	1292	65	97.8	5.7%	2.9%	25.48%
apex7	186	44	14.88	6%	4%	26%
frg2	730	38	94.14	6.6%	0%	41.32%
apex6	614	49	48.39	4.3%	-8.8%	12.5%
ROM$_{o10,i8}$	1217	50	147.4	9%	1.9%	26.23%
ROM$_{o11,i9}$	2331	54	250.27	9%	8.4%	26.18%
ROM$_{o15,i13}$	40071	86	2688.23	9%	8.5%	29.81%
ROM$_{o16,i14}$	79343	98	4330.27	9%	6.66%	29.83%
ROM$_{o10,i12}$	20801	79	780.65	9%	4.8%	26.75%
ROM$_{o11,i11}$	10514	73	780.65	9%	6.4%	27.24%
ROM$_{o12,i14}$	95059	95	4364.7	9%	2%	29.27%
ROM$_{o13,i13}$	49072	91	2553.8	9%	3.19%	30.48%
ROM$_{o14,i14}$	105978	100	4430.72	9%	6.54%	29.25%

TABLE II
COMPARISION OF THE PROPOSED TECHNIQUE WITH THE BEST ALGORITHM IN ABC

Circuit	Gates	P_{ABC}	T_{ABC}	A_{ABC}	P_{RW}	T_{RW}	A_{RW}	P dec	T inc	A inc
dalu	2000	281.69	25.48	2740572	235.42	24.59	2776628	19.65%	-3.61%	1.29%
vda	850	173.49	8.38	1360040	150.76	9.01	1427556	15.07%	6.88%	4.72%
x3	1200	341.57	12.40	1509817	300.75	12.46	1551429	13.85%	0.48%	2.68%
des	6250	1746.4	14.16	8916154	1525.3	14.59	9114893	14.50%	2.84%	2.18%
i8	1900	392.61	10.60	2632622	287.74	11.48	2644593	26.71%	6.66%	0.45%
i9	1025	145.29	11.49	1368924	92.66	11.78	1387254	36.22%	2.46%	1.32%
ROM$_{o10,i8}$	1910	679.85	11.15	2631797	597	11.45	2728439	13.87%	2.62%	3.54%
ROM$_{o9,i7}$	920	378.42	9.47	1249644	322.18	9.84	1298947	17.45%	3.7%	3.7%
ROM$_{o12,i10}$	8530	2644.4	16.72	11653079	2215.2	16.86	12079592	19.37%	0.8%	3.5%
ROM$_{o13,i11}$	17860	5380.7	20.02	26461521	4639.8	20.36	26786046	16%	1.66%	1.21%
ROM$_{o10,i10}$	9380	2870.5	16.74	13169811	2497.9	16.96	13586011	15%	1.29%	3%
ROM$_{o14,i12}$	36270	10373	21.61	53068261	9170.9	21.58	53674448	13.11%	0%	1.1%
ROM$_{o11,i11}$	18850	5338.6	19.61	27286107	4669.9	19.19	27926652	14.31%	-2.18%	2.29%
Average	10435	-	-	-	-	-	-	18.08%	1.81%	2.38%

REFERENCES

1. S. Theoharis, G. Theodoridis, D. Soudris, C. Goutis and A. Thanailakis. "A fast and accurate delay dependent method for switching estimation of large combinational circuits". Journal of Systems Architecture, vol. 48, pp. 113-124, 2002.
2. Jose Monteiro, Srinivas Devadas, Abhijit Ghosh, Kurt Keutzer and Jacob White. "Estimation of average switching activity in combinational logic circuits using symbolic simulation". IEEE transations on computer-aided design of integrated circuits and systems, vol. 16, no. 1, 1997.
3. Vivekanandan Srinivasan. "Real Delay Graphical Probabilistic Switching Model for VLSI Circuits". Thesis submitted to Department of Electrical Engineering, College of Engineering, University of South Florida .
4. A. Mishchenko and S. Chatterjee and R. Brayton. "Dag-aware aig rewriting a fresh look at combinational logic synthesis". Proceedings of the 43rd annual conference on Design automation, pp. 532-535, 2006.
5. R.Bryant. "Graph-based algorithms for boolean function manipulation". IEEE Transactions on Computers, vol. 35, no. 8, pp. 677-691, 1986.
6. Sabih H.Gerez "Simulated annealing". Algorithms for VLSI Design Automation, pp. 71-72.
7. M. T. P. Lindgren, M. Kerttu and R. Drechsler. "Low power optimisation technique for BDD mapped circuits". ASP-DAC, 2001.
8. Aghaghiri, Y.Fallah and F.Pedram "Irredundant Address Bus Encoding for Low Power". International Symposium on Low Power Electronics and Design, pp. 182-187, Huntington Beach, CA, August 2001.