Re-synthesis for Delay Variation Tolerance

Shih-Chieh Chang
Department of CS
National Tsing Hua University
Hsinchu, Taiwan
scchang@cs.nthu.edu.tw

Cheng-Tao Hsieh
Department of CS
National Tsing Hua University
Hsinchu, Taiwan
jdshieh@nthucad.cs.nthu.edu.tw

Kai-Chiang Wu
Department of CS
National Tsing Hua University
Hsinchu, Taiwan
Alexe@nthucad.cs.nthu.edu.tw

ABSTRACT

Several factors such as process variation, noises, and delay defects can degrade the reliabilities of a circuit. Traditional methods add a pessimistic timing margin to resolve delay variation problems. In this paper, instead of sacrificing the performance, we propose a re-synthesis technique which adds redundant logics to protect the performance. Because nodes in the critical paths have zero slacks and are vulnerable to delay variation, we formulate the problem of tolerating delay variation to be the problem of increasing the slacks of nodes. Our re-synthesis technique can increase the slacks of all nodes or wires to be larger than a pre-determined value. Our experimental results show that additional area penalty is around 21% for 10% of delay variation tolerance.

Categories and Subject Descriptors

B.8.1 [**Performance and Reliability**]: Reliability, Testing, and Fault-Tolerance

General Terms

Design, Performance, Reliability

Keywords

Delay variation, tolerance

1. INTRODUCTION

Due to the design trend of shrinking device geometries, lower power voltages, and higher frequencies, circuit performance is increasingly sensitive to factors such as process variation, noises, and delay defects [1][2]. These factors can negatively affect the timing behavior of a circuit and therefore, can cause delay variation in a chip. To alleviate delay variation problems, designers often have to adopt the worst-case delay model or employ a timing margin to protect the performance from delay fluctuation. However, such conservatism is becoming unnecessary pessimism [4][6]. It is reported in [3] that a fabricated ASIC may run up to 40% faster than predicted by the standard (worst-case) timing analysis. On the other hand, even if we can afford the over-design penalty, determining an appropriate worst corner is difficult due to multiple sources of delay variation and their complex influence on circuit performance. Moreover, adding timing margin may not be possible for a timing critical design. In this paper, instead of sacrificing the performance, we propose a novel way to trade area for delay variation tolerance.

DAC 2004, June 7–11, 2004, San Diego, California, USA

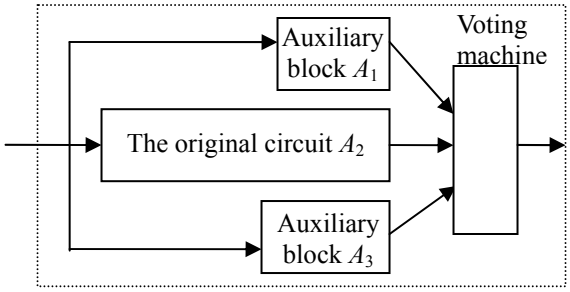

Figure 1: A delay tolerance structure.

In a circuit, some gates (wires) such as those along the critical paths are vulnerable to delay variation because any delay variation in those gates (wires) may adversely affect the whole circuit delay. The vulnerability can be best characterized by a gate's slack, the quantity that represents the affordable margin without violating the circuit's delay. The smaller the slack of a gate is, the more vulnerable the gate will be.

We say a circuit has d_t *delay tolerance* if the delay of each gate (or wire) can increase d_t without affecting the circuit's delay; in other words, the slack of each gate (or wire) is at least d_t. Given a delay tolerance value d_t and a circuit, our goal is to re-synthesize the circuit such that every gate (or wire) in the new circuit can tolerate at least delay variation d_t. In Figure 1, our technique builds a new structure consisting of a voting machine, the original circuit A_2, and two additional auxiliary blocks A_1 and A_3. Several properties of the new structure are briefly summarized as follows. First, the delays of auxiliary blocks, A_1 and A_3, are always smaller than or equal to the delay of the original circuit A_2. By introducing the area penalty of A_1 and A_3, the whole circuit can tolerate at least a

pre-defined tolerance d_t. Moreover, the pre-defined tolerance value determines the sizes of the auxiliary blocks that are generally much smaller than that of the original one.

When the delay tolerance is 10% (15%) of the original circuit's delay, our experimental results show that on the average, the new structure has 21% (41%) of area overhead. We also run another set of experiments by assuming the delay of each gate is given as a probability density function [5]. We estimate the statistical delay of a circuit by running 10,000 times of Monte-Carlo experiments. The results show that on the average, 68% of samples of a circuit can achieve some delay requirement. On the other hand, 87% of samples of the re-synthesized circuit can achieve the same delay requirement.

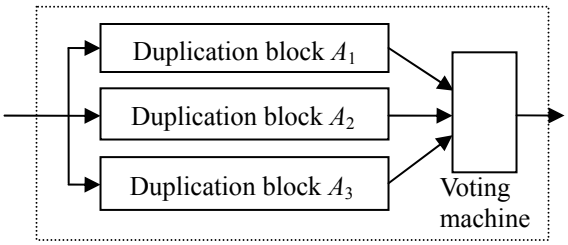

Figure 2: A TMR structure.

2. DELAY VARIATION TOLERANCE IN A TMR STRUCTURE

Many previous papers presented architectures for functional fault tolerance. However, to the best of our knowledge, there has been no research aiming for delay variation tolerance. Among the functional fault tolerance techniques, Triple Modular Redundancy (TMR) [7], illustrated in Figure 2, is a widely used scheme. In a TMR structure, a given circuit is replicated into three duplication blocks (A_1, A_2, A_3) whose outputs are connected to the inputs of a (majority) voting circuit. The use of the voting circuit allows a TMR to produce correct results as long as any two duplication blocks generate correct results. In a TMR, each wire or gate is *redundant* because removal of one wire or gate will not affect the circuit functionality.

Though a TMR structure is primarily used for functional tolerance, a TMR structure can also tolerate delay variation because each node in a TMR has an infinite slack. (We will explain the property of infinite slack later.) On the other hand, our objective of delay variation may cause 10% - 20% more delay than the original circuit's delay. Therefore, the infinite slack for each gate in a TMR is over-protective. Besides, a TMR requires three times the area of the original circuit, making the scheme impractical for our objective.

Again, our objective is to re-synthesize a circuit so that the slack of each node is at least d_t. Though a TMR structure is not practical for our objective, it does provide a good starting point for improving the slacks of nodes. To reduce the required area, we can remove redundant wires in a TMR structure. However, when a redundant wire is removed, some nodes' slacks are changed. The main idea of this paper is to remove redundant wires while keeping the slacks of all nodes to be equal to or greater than d_t. Let us discuss some important properties in a TMR structure.

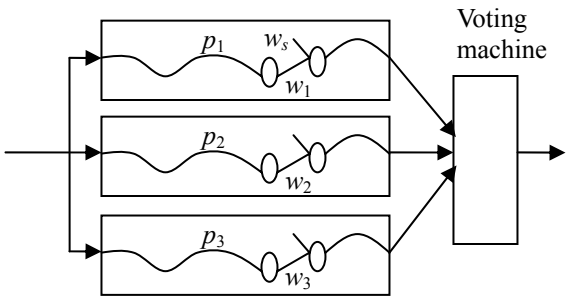

Figure 3: Isomorphic paths in a TMR.

In a TMR, each component such as a wire, a gate, and a path has three replications. We say three replications of a component are *isomorphic* components. For example in Figure 3, paths p_1, p_2, and p_3 are isomorphic paths and wires w_1, w_2, and w_3 are isomorphic wires. Consider a path p_1 and its two other isomorphic paths p_2 and p_3. Let the delay of a path p be $d(p)$, which is equal to the summation of all delay elements along the path. Ideally, all three isomorphic paths have the same delay, $d(p_1)=d(p_2)=d(p_3)$. Suppose due to delay variation, $d(p_1)$, $d(p_2)$, and $d(p_3)$ are different. Since a voting machine determines its output when two of its inputs have generated correct results, the final delay will be dominated by the second arriving signal. In other words, the final delay of a TMR for the same computations is not determined by the latest delay. The property of choosing the second arriving signal for a voting machine makes all three isomorphic paths not vulnerable individually to delay variation.

We define a path to be a ***strictly-false*** path if the path remains or becomes a false path for **any** increment on the path delay. In a TMR, if three isomorphic paths have the same delay $d(p_1)=d(p_2)=d(p_3)$, all three paths are strictly-false paths because of choosing the second arriving signal. In fact, all the paths in a TMR are all strictly-false paths. Moreover, if all paths passing a node are all strictly-false, the node has an infinite slack. The reasons are explained in the following.

First, we would like to clarify the concept of a node's slack. We define the slack of a node to be the largest affordable margin that can be added to the node's delay without increasing the whole circuit delay. The exact slack is difficult to compute. However, we

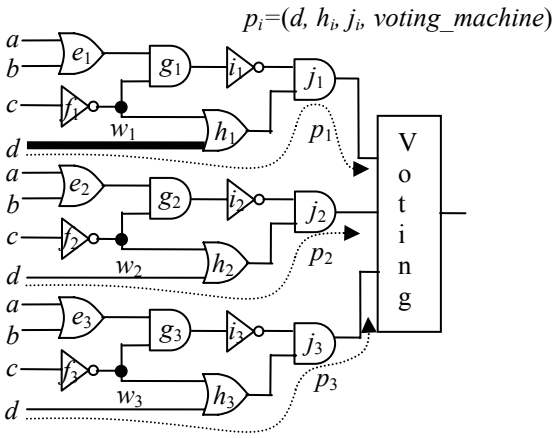

Figure 4: Removal of wire w_1 causes the slacks of some gates (wires) to change.

can quickly estimate the value using the formulas of (1) the required time minus the arrival time of a node or (2) the circuit delay minus the delay of the longest path passing the node. The above computation for a node's slack is conservative or an upper bound because strictly-false paths should not be considered. We can obtain a more accurate estimation by the circuit delay minus the delay of the longest non-strictly-false path passing the node. Since all paths are strictly-false in a TMR, the slacks of all nodes in a TMR are infinite.

3. REMOVING WIRES IN A TMR WHILE MAINTAINING d_t DELAY TOLERANCE

We now discuss the effect of wire removal on slack. Basically, removal of wires will cause some paths to change from strictly-false paths to true paths. Consider three isomorphic wires (w_1, w_2, w_3) in Figure 3, where wire w_i is along path p_i. In a TMR, three isomorphic wires are redundant individually but removing one may cause other two irredundant. Suppose wire w_1 is removed and hence path p_1 ceases to exist. Because only two paths p_2 and p_3 are isomorphic and produce correct results, the voting machine will choose the latest arriving result of p_2 and p_3, which implies paths p_2 and p_3 become true paths. Because paths p_2 and p_3 are no longer strictly-false paths, the slacks of nodes along p_2 and p_3 are no longer infinite.

For example, consider a TMR structure in Figure 4. Assume a gate and the voting machine have the delay of 1. Wires (w_1, w_2, w_3) and paths (p_1, p_2, p_3) are isomorphic. Suppose bold wire w_1 is removed (or replaced by a constant zero). After removal, let us compute the slack for node h_2. Since paths p_2 and p_3 become true paths, the longest true path passing node h_2 is p_2. Therefore, the slack of node h_2 is the circuit's delay minus $d(p_2)$=3. If the circuit's delay is 5, the slack of node h_2 is 2.

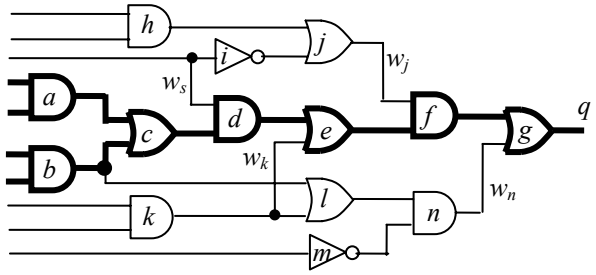

Figure 5: The original circuit.

In a TMR, removing a wire can alter some paths from strictly-false paths to become true paths and also alter some wire's infinite slacks to become finite. In the following, we show the necessary and sufficient conditions for wire removal while maintaining the slack of each node to be at least d_t. In the original circuit, we say a node is a *d_t-critical node* if the node's slack is less than d_t. A circuit region is called a *d_t-critical region*, which consists of only d_t-critical nodes and wires between d_t-critical nodes. For example consider the original circuit in Figure 5 where each gate's delay is 1. Suppose the delay tolerance value d_t is 2. Node g is a d_t-critical node because the slack of g is 0 less than d_t=2. In fact, nodes {a, b, c, d, e, f, g} (drawn by bold lines) are all d_t-critical nodes. Moreover, the d_t-critical region consists of those highlighted (bold) gates and wires in the figure.

Lemma 1: For d_t delay tolerance, all isomorphic paths in d_t-critical regions cannot be true paths.

Proof: Omitted.

Theorem 1: A wire w in d_t-critical regions cannot be removed to maintain d_t delay tolerance requirement.

Proof: Omitted.

The above theorem says that for d_t delay tolerance, wires in three isomorphic d_t-critical regions cannot be removed. Within a d_t-critical region, we say a node is a *d_t-dominator* if all paths from primary inputs to primary outputs must pass the node. Also, a node (wire) is a *side-input* to a d_t-dominator if the node (wire) is an immediate input to the d_t-dominator but does not belong to a node (wire) in the d_t-critical region. For example, consider the same example in Figure 5. There are four paths (from primary inputs to output q) in the d_t-critical region. Nodes {c, d, e, f, g} are d_t-dominators because all four paths pass those nodes. Moreover, node n (wire w_n) is a side-input to d_t-dominator g.

Theorem 2: A side-input wire w to a d_t-dominator can be removed (replaced by a non-controlling value) without violating the requirement of d_t delay tolerance.

Proof: Omitted.

Consider the same example in Figure 5. A TMR can be constructed by duplicating three copies of the original circuit.

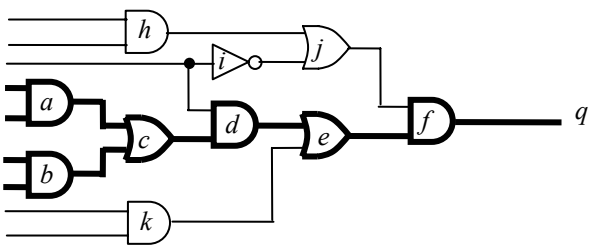

Figure 6: Removal of side-input w_n.

Since wire w_n is a side-input to a d_t-dominator, according to Theorem 2, we can remove wire w_n. After removal, the resulting circuit is shown in Figure 6. There can be many side-input wires to d_t-dominators. The following theorem will show that removal of several side-input wires still satisfies the d_t delay tolerance requirement.

Let wires (w_{11}, w_{12}, w_{13}) are three isomorphic wires and are side-input wires. In addition, wires (w_{21}, w_{22}, w_{23}) are isomorphic wires and are side-input wires.

Theorem 3: One wire among three isomorphic wires (w_{11}, w_{12}, w_{13}) and one wire among three isomorphic wires (w_{21}, w_{22}, w_{23}) can be removed simultaneously without violating the requirement of d_t delay tolerance.

Proof: Omitted.

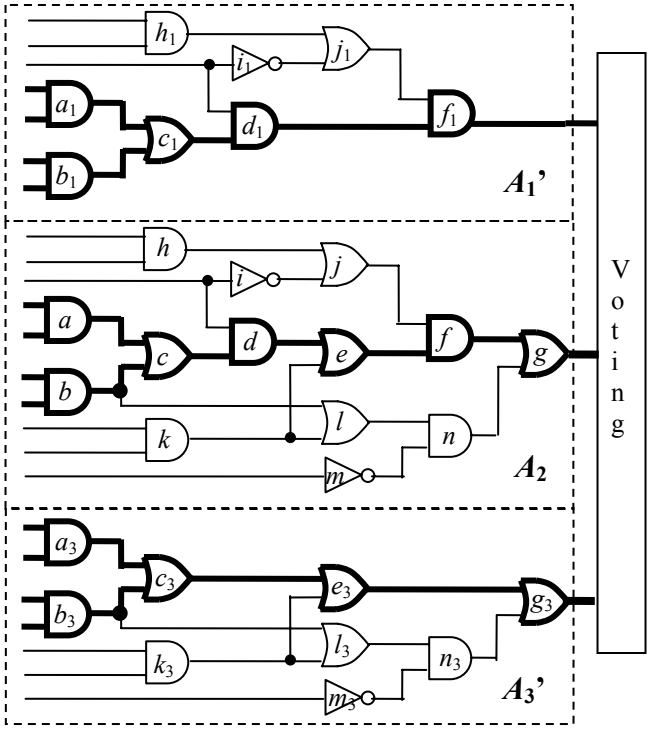

Figure 7: Removing side-input wires in the two duplication blocks.

According to the above theorem, one can remove several side-input wires at the same time. If there are k side-input wires to d_t-dominators, each of which has two isomorphic side-input wires, we can remove k side-input wires. Though there are many ways to remove k side-input wires, our algorithm evenly distributes the removal to two duplication blocks such as A_1 and A_3 in Figure 1. The reasons are as follows. Note that removing a wire will shorten the delay of a path. In addition, a voting machine chooses the second arriving input. To reduce overall circuit's delay, it is desirable to reduce two duplication blocks' delays. Again, in Figure 5 one can find the original circuit totally has four side-input wires $\{w_j, w_n, w_k, w_s\}$. We remove two isomorphic wires to $\{w_n, w_k\}$ in duplication block A_1 and two isomorphic wires to $\{w_j, w_s\}$ in A_3 and the resulting circuit is shown in Figure 7.

4. SIGNAL SHARING OUTSIDE THE d_t-CRITICAL REGIONS

Note that gates not in the d_t-critical regions have slacks at least d_t. For those gates not in the d_t-critical regions, we may further reduce the area by sharing (logically) equivalent signals which implement the same function. For example in Figure 7, the output function of node j_1 in A_1' and the output function of node j in A_2 have the same functionality. We can share the output function of node j and node j_1 in Figure 8. After sharing, the required time for node j does not change but the arrival time may increase due to the additional fanout from node j to node f_1. We need to re-compute the slack of node j. If the slack of node j is equal to or greater than d_t, we can perform the sharing; otherwise, the sharing is not allowed. Suppose all equivalent signals are allowed to share, the final circuit is shown in Figure 8. Block A_1'

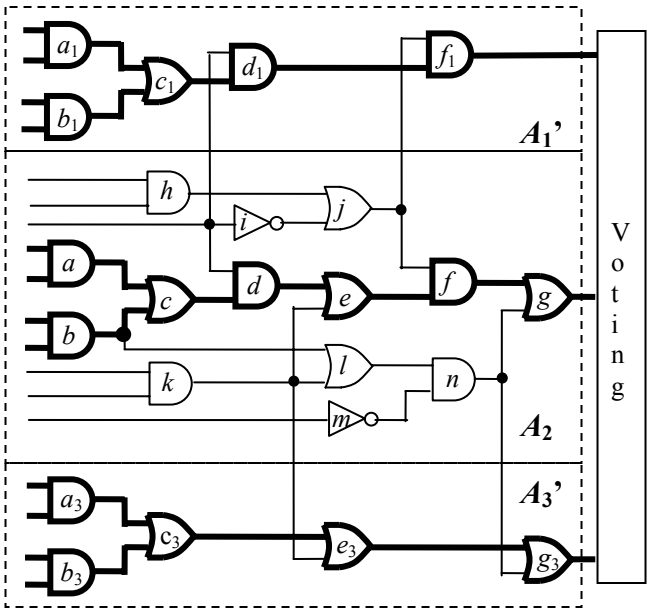

Figure 8: The three blocks in the resulting circuit.

Table 1: Comparison between the original circuit and the corresponding delay tolerance structure

Circuit	Original circuit			Delay tolerance structure (d_t = 10%)					Delay tolerance structure (d_t = 15%)					Statistical analysis	
	Area	Circuit delay	Avg. slack	Area	Area overhead (%)	Circuit delay	#nodes with infinite slack	Avg. slack	Area	Area overhead (%)	Circuit delay	#nodes with infinite slack	Avg. slack	Original circuit	Delay tolerance d_t=10%
Alu2	487664	18.71	2.12	531744	9.0	18.66	0	3.62	617584	26.6	19.06	10	3.80	6828	7083
Alu4	1254656	25.71	5.03	1440720	14.8	26.23	16	5.94	1796144	43.2	25.65	38	7.05	5001	8799
Apex6	1165568	11.66	3.55	1357200	16.4	11.29	6	4.43	1442112	23.7	11.47	40	4.38	4062	7929
Apex7	365632	11.41	3.72	411568	12.6	11.24	2	4.01	505296	38.2	11.49	12	3.63	8820	9924
B9	165184	6.54	1.81	202304	22.5	6.64	4	1.94	222256	34.6	6.46	5	2.30	6411	6930
Frg1	186528	11.81	2.26	221792	18.9	10.89	0	3.74	250560	34.3	11.55	4	3.29	8361	9081
Frg2	1425408	11.40	2.56	2065264	44.9	11.82	10	2.28	2269888	59.2	12.02	48	2.53	4179	8385
Pair	2469408	14.46	3.63	2894432	17.2	14.07	12	4.52	3050336	23.5	13.85	24	5.06	6192	8775
Rot	1031936	14.78	6.01	1203152	16.6	14.47	26	6.67	1286672	24.7	15.05	26	6.09	8775	9861
S344	273760	12.34	3.42	307168	12.2	11.86	0	4.13	378624	38.3	11.39	4	4.58	8247	8835
S349	268192	12.80	3.65	327120	22.0	12.27	0	4.33	403216	50.3	12.07	18	4.59	7812	8565
S526	306240	8.98	1.61	364704	19.1	8.83	4	2.06	521536	70.3	8.94	36	2.05	6759	8460
S641	261696	12.94	3.99	301600	15.2	12.40	0	5.31	307168	17.4	11.75	0	5.99	9078	9765
S713	261696	12.92	3.99	294640	12.6	12.38	0	5.36	308560	17.9	11.99	0	5.91	8319	8412
S1196	967904	13.28	2.23	1154432	19.3	13.47	12	2.68	1436544	48.4	14.23	36	2.08	7515	8346
S1238	1026368	14.11	2.11	1116384	8.8	13.93	12	3.10	1402208	36.6	14.69	12	2.52	5607	9714
S1488	900624	13.17	1.79	1253728	39.2	12.74	34	2.91	1530272	69.9	13.27	107	2.52	6933	9990
S1494	857472	12.69	1.77	1314048	53.2	12.78	62	2.08	1494544	74.3	12.86	202	2.23	3351	7842
Avg.					20.8					40.6				6792	8705

and A_3' in Figure 8 become the two auxiliary blocks in our delay tolerance structure in Figure 1.

Note that the delay of a re-synthesized circuit can be larger or smaller than that of the original one. The delay of a re-synthesized circuit may be larger because of the extra delay of a voting machine. On the other hand, because some critical paths may become false, the delay of a re-synthesized circuit may be smaller. For example in Figure 8, all the bold paths in A_2 become false so they should not be considered in the final timing.

5. EXPERIMENTAL RESULTS

We have implemented our algorithm and experimented on a large set of MCNC and ISCAS benchmark circuits. For each circuit, we first use script.delay to minimize the delay of the circuit. Then, we use the delay tolerance value of 10% and 15% of the original circuit's delay to re-synthesize a circuit to two delay tolerance circuits. The experimental results are demonstrated in Table 1. Note a circuit *C* and the two corresponding delay tolerance circuits may have different delays. To compare slacks fairly among three circuits, the required time for all three circuits is set to be the delay of the original circuit *C*.

In Table 1, column one gives the name of an original circuit. Column two shows the area, column three shows the delay, and column four shows the average slack of gates in the original circuit. Column five to nine report the experimental results when d_t is 10% of the original circuit's delay. Column five shows the area. Column six gives the area penalty in the re-synthesized circuit. Column seven presents the delay of the circuit. Column eight reports the number of gates with infinite slack. Column nine shows the average slack for those gates with finite slacks. Consider circuit Alu2 as an example. The circuit has the area of 487,664 and the delay of 18.71. The average slack is 2.12. Assume the delay tolerance is 1.87 (=0.1*18.71). The re-synthesized circuit has the area of 531,744 and the delay of 18.66. The new circuit has no gates with infinite slack. The average slack is 3.62 assuming the required time is 18.71. Column ten to fourteen show the experimental results when d_t is 15% of the original circuit's delay. Assume the delay tolerance is 2.81 (=0.15*18.71). The re-synthesized circuit has the area of 617,584 and the delay of 19.06. The new circuit has 10 gates with infinite slack. Again, we assume the required time to be 18.71 and the average slack is 3.80. On the average, we find that to have 10% of delay tolerance, the area overhead is about 21% while to have 15% of delay tolerance, the area overhead is about 41%. The delay of the re-synthesized circuit is about the same as that of the original circuit. In addition, for re-synthesis, all benchmark circuits can be finished within minutes of CPU time on Sun Blade 2000 workstation.

We have preformed another set of experiments assuming the delay of each gate is given as a probability density function similar to the way in [5]. We then run 10,000 times of the Monte-Carlo experiment. During the experiment, we can calculate the circuit's delay of each sample circuit and compare to a pre-defined delay requirement which is set to {1.1*the circuit's delay in column 3} for each benchmark circuit. We then count the number of samples whose calculated delays are less than the pre-defined delay

requirement. The results are shown in column fifteen in Table 1. When d_t is 10% of the original circuit delay, column sixteen shows the number of samples whose delays are less than the delay requirement for a re-synthesized circuit. Take circuit S1488 as an example. Among 10,000 samples, 6,933 samples have delay less than 14.49(=1.1*13.17) while after re-synthesis, 9,990 samples have delay less than 14.49. On the average 68% of samples of original circuits can achieve the delay requirement. On the other hand, 87% of samples of re-synthesized circuits can achieve the same delay requirement. The experimental results show that on the average of 19% more circuit samples can achieve some delay requirements after our re-synthesis for delay tolerance.

6. CONCLUSION

We have presented a framework to re-synthesize a given circuit for d_t delay tolerance. Our method adopts wire removal and signal sharing to reduce the area overhead in our delay tolerance structure. Our experimental results show that the area penalty is about 21% for 10% delay variation tolerance.

REFERENCES

[1] K. Baker, G. Gronthoud, M. Lousberg, I. Schanstra, and C. Hawkins, "Defect-based delay testing of resistive vias-contacts, a critical evaluation," *Proc. of IEEE International Test Conference*, pp. 467-476, Sep. 1999.

[2] M. A. Breuer, C. Gleason, and S. Gupta, "New validation and test problems for high performance deep sub-micron VLSI circuits," *Tutorial Notes, IEEE VLSI Test Symposium*, April 1997.

[3] D. G. Chinnery and K. Keutzer, "Closing the gap between ASIC and custom: an ASIC perspective," *Proc. of Design Automation Conf.*, pp. 637-642, June 5-9, 2000.

[4] Kurt Keutzer and Michael Orshansky, "From blind certainty to informed uncertainty," Proc. of the 8th ACM/IEEE Int. Workshop on Timing Issues in the Specification and Synthesis of Digital Systems, pp. 37-41, 2002.

[5] Jing-Jia Liou, Angela Krstic, Li-C. Wang, and Kwang-Ting Cheng, "False-path-aware statistical timing analysis and efficient path selection for delay testing and timing validation," *Proc. of Design Automation Conf.*, pp. 566-569, June 10-14, 2002.

[6] Enrico Malavasi, Stefano Zanella, Julian Uschersohn, Mike Misheloff, and Carlo Guardiani, "Impact analysis of process variability on digital circuits with performance limited yield," *IEEE Int. Workshop on Statistical Methodology*, pp. 60-63, June 2001.

[7] Von Neumann, J., "Probabilistic logics and the synthesis of reliable organisms from unreliable components," *Automata Studies, Ann. of Math. Studies*, no. 34, C. E. Shannon and J. McCarthy, Eds., Princeton University Press, pp. 43-98, 1956.