M-IVC: Using Multiple Input Vectors to Minimize Aging-induced Delay

Song Jin[1,2], Yinhe Han[1], Lei Zhang[1], Huawei Li[1,*], Xiaowei Li[1], Guihai Yan[1,2]
[1]Key Laboratory of Computer System and Architecture
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, P.R. China
[2]Graduate University of Chinese Academy of Sciences, Beijing, P.R. China
{jinsong, yinhes, zlei, lihuawei, lxw, yan_guihai }@ict.ac.cn

***Abstract*—Negative bias temperature instability (NBTI) has been a significant reliability concern in current digital circuit design due to its effect of increasing the path delay with time and in turn degrading the circuit performance. NBTI degradation has strong dependence on input pattern and duty cycles. Based on this observation, we propose to apply multiple input vectors to the combination circuit in a non-uniform way during standby mode. Multiple input vectors can enhance the capability to control the circuit nodes, achieve smaller duty cycles to reduce the stress time of gates and thus mitigate static NBTI. A constrained multi-object optimization model is formalized to find the optimal combination of duty cycles for timing-critical paths, which in turn minimizes the increase of path delay. An ATPG-like procedure is then presented to generate the corresponding input vectors. Experimental results demonstrate that the delay increase of timing-critical paths can be mitigated significantly under long time NBTI effect (10-year) by only applying a small number of vectors.**

Ⅰ. INTRODUCTION

With aggressive scaling of the feature size, transistor aging presents a serious reliability concern for current and future integrated circuits design. Negative bias temperature instability (NBTI) is recognized as a major aging mechanism due to its effect of elevating the threshold voltage of the gate and in turn increasing the path delay, which decreases the maximum frequency a circuit can deliver gradually with time[1, 2, 3, 6, 7, 8]. In the worst case, path delay can be increased by up to 20% under long time (e.g., 10 years) NBTI effect [2].

NBTI effect occurs when a negative bias voltage (Vgs=-Vdd) is applied on a PMOS transistor (stress scenario). Its degradation is strongly dependent on the input pattern and duty cycle (i.e., the percentage of time the gate is applied with negative voltage) [2, 5, 7]. Based on the long term prediction model of NBTI [3,11], the delay increase of a gate due to NBTI effect can be expressed as:

$$\Delta T_{p(i)} = c_i \cdot \alpha_i^n \cdot t^n \qquad (1)$$

where $\Delta T_{p(i)}$ is the delay increase between input node i to the output of a gate, c_i is constant parameter related to node i. t is the operational time of the circuit. n is 0.16, which represents the diffusion species are H_2. The parameter α_i is the duty cycle of the input node i. In the static stress scenario, the gate has 1-α due to the persistent 0-signal on its input, which increases the threshold voltage exponentially with time. However, due to the special self-healing property of NBTI [3], switching between 0-signal and 1-signal in dynamic stress scenario leads to α less than 1, which reduces the stress time of the gate and in turn results in much less threshold voltage increase. Wang et al demonstrated that various input patterns and duty cycles can result in 3X~5X difference on the degradation rate [2].

Low power designs generally employ sleep or clock gating techniques to reduce dynamic power. Clocks are gated in the standby mode for idle function units, thus prevent unnecessary switching activities on the gate input. However, persistent 0-signals on gate input during this standby mode will exacerbate static NBTI stress and degrade performance of the circuit significantly.

In this paper, M-IVC, a kind of multiple input vectors control method is proposed. It applies multiple input vectors in a non-uniform way in standby mode to minimize static NBTI degradation on timing-critical paths. Multiple input vectors can enhance the capability to control the circuit nodes since different vectors may cover different portions of nodes. More importantly, by applying multiple input vectors in a non-uniform way (depicted in Sec. Ⅳ), smaller duty cycles can be obtained, which reduce the stress time of the gates. A multi-object optimization model is formalized to find the optimal combination of duty cycles for the critical gates on timing-critical paths. An ATPG-like process is then presented to generate the corresponding input vectors.

Technologies to resist NBTI degradation were proposed in recent years, such as the NBTI-aware sizing algorithm to set the feature size of transistor [6], the technology mapping method considering NBTI effect [9] and the reliability monitor design for NBTI [10]. They alleviated NBTI degradation by introducing extra area or power consumption overhead. In [5], the author used input vector pair to resist NBTI degradation on the combinational circuit. The input vector pair was generated randomly in a very coarse-grained manner (32-bit word wide). This could hardly achieve the optimal combination of duty cycles to reduce the stress time of the gates effectively.

Different with these methods, M-IVC applies multiple input vectors in a non-uniform way in standby mode to achieve the optimal combination of duty cycles for timing-critical paths, which reduces the increase of path delay significantly. For a smaller number of input vectors, the

*To whom correspondence should be addressed.

interval between applying time of two vectors can be large (e.g., 100000 clock periods), thus the average dynamic power consumption resulted from applying these input vectors are negligible.

The rest of the paper is organized as follows. Section Ⅱ depicts the process of obtaining timing-critical path set. The multi-object optimization model is set up in Section Ⅲ. Section Ⅳ presents an ATPG-like process to generate vectors and discusses the related implementation issues. Experimental results are given in Section Ⅴ and finally, we conclude in Section Ⅵ.

Ⅱ. TIMING-CRITICAL PATHS SELECTION

Generally, designer always reserves 10% delay of the longest path as timing margin to tolerant the effects induced by process variation or some aging mechanisms. However, Wang et al pointed out that the path delay can be increased up to 20% under long time (10 years) NBTI in worst case scenario [2]. Therefore in this paper, for a path i, if 20% increase of the path delay in worst NBTI scenario violates the prescribed timing margin (e.g., 10%), it's considered as a timing-critical path (TP). Of course, arbitrary timing margin can be set to tolerant the NBTI degradation, and this will not affect the validity of our method.

The original timing information for all of the paths in the circuit without considering NBTI effect can be obtained through static timing analysis (STA). $D_{p(i)}$ is used to denoted the delay of the path i, $D_{\max}$ is used to denoted the delay of the longest path. Thus the path delay of the timing-critical path i which is selected by STA should satisfy the formula (2):

$$D_{p(i)} \times (1+20\%) \geq D_{\max} \times (1+10\%) \qquad (2)$$

Then, a set of TPs are selected using STA based on formula (2).

Above method to select TPs is over conservative, since the delay increase of a path is simply recognized as the sum of delay increases of all the gates on the path. Next, TPs are pruned further by considering different influences on the increase of path delay induced by rising and falling transition separately.

1. Pruning TPs based on the rising or falling transition

NBTI elevates the threshold voltage of PMOS while does not affect NMOS. So it only affects falling transition on the CMOS gate input and slows down rising transition on the output accordingly. Therefore, for a gate on the TP which has rising transition on its input, its delay increase induced by NBTI effect should not be counted as a part of the delay increase of the TP. For example, as shown in Figure 1, path A and B both consist of 3 inverters. It's assumed that all of the inverters are suffered NBTI degradation, resulting in the increases of their threshold voltages. When a falling transition is applied on the primary input (PI) of the path A, the rising transitions on the outputs of gate 1 and 3 are delayed, while the falling transition on the gate2 output does not be affected due to the gate2 input has rising transition. Therefore, the delay increase of the path A is actually the sum of delay increases of gate1 and gate3. Similarly, for the rising transition on PI of path B, the increase of path delay is only equal to that of gate2.

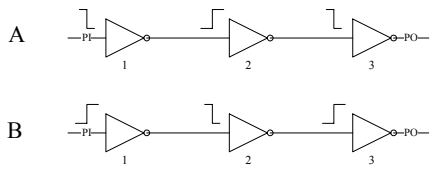

Figure 1. Inverter with rising and falling transition

Based on the above analysis, it can be seen that the path has different delay increase corresponding to the rising or falling transition on its PI. Therefore, TPs, which are obtained from STA previously, are pruned by recalculating their delay increase considering rising and falling transition separately. A path will be weeded out from the set of TPs if its delay increase is not satisfying formula (2) under both of rising and falling scenario. The pruning process is depicted as below.

For a constant operational time t (e.g., 10 years) and the diffusion specie H_2 (i.e., $n = 0.16$), the delay increase between input node i to the output of a gate is obtained from HSPICE simulation assuming different α (from 0 to 1, 0.1 increment). The simulation results are then fitted with formula (1) to obtain c_i for all of primitive gates (INV, NAND, NOR). Then, corresponding to the rising and falling transition on the PI, the delay increase of a TP is calculated separately by using formula (1), assuming worst case NBTI scenario (i.e. $\alpha = 1$) for 10 years. During the calculation, only the delay increase of the gate which having falling transition on its input is counted as a part of delay increase of the TP. The pruning results are listed in table 1. It can be seen that the number of TPs is reduced significantly.

2. Critical Gates Identification

After obtained the pruned set of TPs, the critical gates (CGs) are needed to be identified since the duty cycles of the CGs are the decision variables in the optimization model. A gate on the TP is considered as CG if its delay increase is counted as a part of delay increase of the TP during the pruning process. Therefore, if a TP is timing critical in both rising and falling transition scenario, all the gates on the path are marked as critical gates. However, if a TP is timing critical only in rising or falling transition scenario separately. The gates, which having falling transition on their inputs, are marked as critical gates.

TABLE 1. PRUNING RESULTS ON TPS

Circuits	Num(CS)	Num(PR)	R%
c880	80	18	77.5
c1908	693	27	96.1
c3540	1621	273	83.1
c5315	1244	598	51.9
s298	3	2	33.3
s820	2	2	0
s1196	25	8	68
s1238	15	8	46.6
s9234	2280	322	85.8

Num(CS): number of TPs using conservative selection; Num(PR): number of TPs after pruning; R%: the percentage of reduction.

III. THE MULTI-OBJECT OPTIMIZATION MODEL

In this section, a constrained multi-object optimization model is formalized, which can aid us to find an optimal combination of duty cycles for all of the TPs.

1. Formalizing the Optimization Object

For suffering NBTI degradation for a period of time t, the propagation delay between input node i and the output of a gate can be expressed as below based on formula (1),

$$T_{g(i)} = c_{0(i)} + c_i \cdot \alpha_i^{\,n} \cdot t^n$$

where $c_{0(i)}$ refers to the intrinsic delay between input node i and output of the gate, it's obtained from HSPICE simulation corresponding to the specific technology node. For a period of operational time, t is constant, so $c_i \cdot t^n$ can be replaced with a new constant φ_i and the final propagation delay of the gate can be rewrite as formula (3),

$$T_{g(i)} = c_{0(i)} + \varphi_i \cdot \alpha_i^n \qquad (3)$$

Some notations are defined as follows. I_k refers to the inputs set of gate k, L_i refers to the critical gates set of path i. Therefore, for a timing-critical path i having l critical gates, the final path delay under NBTI for a period of time t is formulated as formula (4).

$$D_{p(i)} = \phi_i + \sum_{j=1}^{l} \varphi_{j(k)} \cdot \alpha_{j(k)}^n \qquad (4)$$

where $k \in I_k$, $j \in L_i$. $\phi_i = \sum_{j=1}^{l} c_{0\,j(k)}$, refers to the sum of intrinsic delays of all the critical gates on path i. For the diffusion species H_2, $n = 0.16$. the duty cycle $\alpha_{j(k)} \in [0,1]$. For m timing-critical paths in the circuit, $1 \le i \le m$.

For a circuit having m timing-critical paths, the optimization object is minimizing the maximum delay increase of the longest path resulted from NBTI degradation. It's formalized as a multi-object optimization problem:

$$\min \max(D_{p(1)}, D_{p(2)} \ldots D_{p(m)})$$

The optimization process is needed to satisfy two constraints, which are formulated as inequalities constraints and are depicted as follows.

2. Constraint-1: No Timing Violation Problem

For all of the timing-critical paths, the optimization should ensure that the delay increase of the paths will never violate the timing margin. We formulate this constraint as formula (6),

$$D_{p(i)} - D_{\max} \times \gamma \le 0 \qquad (6)$$

where $D_{\max}$ refers to the delay of longest path, γ refers to the timing margin. For m timing-critical paths in the circuit, $1 \le i \le m$.

3. Constraint-2: Logic Conflictions among Critical Gates

Ideally, the timing-critical paths have no delay increase under NBTI degradation if all of the inputs of critical gates can be set 1-signal in standby mode, that is, all of the inputs of critical gates have 0- α. However, due to the logic conflictions among the critical gates in different timing-critical paths, it could hardly achieve this purpose. This also means that the logic conflictions among critical gates restrict the value of αs of these gates. Therefore, the logic conflictions are formulated as constraints in the optimization model based on the implication sets of critical gates and the logic function of the gate. The implication sets are used to restrict the integer value of α (i.e., 0 or 1) while the logic function of the gate is used to restrict the real value of α (i.e., $0 < \alpha < 1$).

Logic implications [12, 13, 14] can capture the effect of assigning logic value to a net (input or output of a gate) on other net values. Based on the implication sets, we can learn whether some inputs of critical gates can be set to 1 simultaneously or not. This also means whether αs of these gate inputs can be 0 simultaneously. A simple example is used to illustrated how to determine the integer value of αs for a 2-input NAND gate with inputs a, b and output c based on its implication sets. For a 2-input NAND gate, implication sets can be obtained easily from its truth table: imp(a/0)={c/1}, imp(b/0)={c/1} and imp(c/0)={a/1,b/1}. The possible integer values of α_a, α_b and α_c then can be formulated as formula (7).

$$\begin{aligned} &1 \le \alpha_a + \alpha_b + \alpha_c \le 2 \\ &\alpha_a + \alpha_c \le 1 \\ &\alpha_b + \alpha_c \le 1 \end{aligned} \qquad (7)$$

Formulas corresponding to other kinds of primitive gates (INV, NOR) can be derived easily in a similar way.

Formula (7) can only be used to determine the integer values of α_a, α_b and α_c. However, the situation is more complicated when it's considering the real values

of α_a , α_b and α_c . Note that duty cycle denotes *the percentage of time* the gate input has 0-signal. Thus, corresponding to the α_a and α_b , the real value of α_c is not a unique value while is in a range. As shown in figure 2, it's

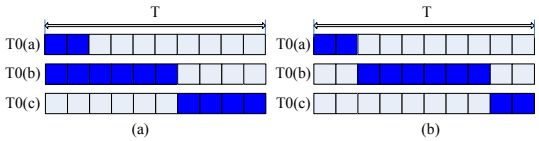

Figure 2. 0-signal time of c.

assuming that total operational time is T , $\alpha_a = 0.2$ and $\alpha_b = 0.6$. The 0-signal time of a and b are denoted by $T0_{(a)}$ (i.e., $0.2T$) and $T0_{(b)}$ (i.e., $0.6T$) separately. Considering figure 2(a), it shows the scenario that the 0-signal time of a and b are overlapped. It's can be seen obviously that the 0-signal time of c is determined by the maximum of $T0_{(a)}$ and $T0_{(b)}$. That is $T0_{(c)} = 1 - \max(T0_{(a)}, T0_{(b)})$. While figure 2(b) shows the scenario that the 0-signal time of a and b is non-overlapped. It's easy to denote $T0_{(c)}$ as $T0_{(c)} = 1 - (T0_{(a)} + T0_{(b)})$. Combining the above two scenarios, the possible real values of α_a , α_b and α_c can be formulated as formula (8),

$$1 - (\alpha_a + \alpha_b) \le \alpha_c \qquad (8-1)$$

$$\alpha_c \le 1 - \max(\alpha_a, \alpha_b) \qquad (8-2)$$

Formula (8-2) can also be replaced with (8-3) and (8-4).

$$\frac{\alpha_a + \alpha_b}{2} + \frac{\alpha_a - \alpha_b}{2} + \alpha_c \le 1 \qquad (8-3)$$

$$\frac{\alpha_a + \alpha_b}{2} + \frac{\alpha_b - \alpha_a}{2} + \alpha_c \le 1 \qquad (8-4)$$

It's easy to validate that for any real values of α_a and α_b , formula (8) can ensure to obtain the correct value of α_c .

Formula (7) and (8) can be extended to include more gates to reflect the logic conflictions among critical gates.

Under Constraint-1 and Constraint-2, the optimization model is solved to find an optimal combination of αs for timing-critical paths. In the next section, an ATPG-like process is presented to generate corresponding vectors based on the combination of αs .

Ⅳ. INPUT VECTORS GENERATION

1. ATPG-like Process to Generate Input Vectors

Generating a set of input vectors which can achieve the optimal combination of αs is an ATPG-like process because it only need to set the desirable values to the inputs of critical gates and doesn't need to propagate any fault effect to the primary output (PO).

The inputs of critical gates are classified based on their αs which are obtained from solving the optimization model. The ATPG-like process firstly generates a primitive input vector, which sets 1 on the inputs of all critical gates having 0-value α , on the contrary, the inputs having 1-value α will be ignored. This primitive vector has many x bits since only a portion of inputs of the critical gates are set deterministic value. Then, the x bits are filled to generate other vectors based on different values of αs .

The ATPG-like process performing on C17 circuit is illustrated in Figure 3. There are two TPs: N3-N11-N16-N22, N6-N11-N16-N23 and four critical gates: G2, G3, G5 and G6. It needs two input vectors to achieve the combination of αs .The square brackets denote [the duty cycle α | the input signal set by the first vector| the input signal set by the second vector]. N3, N6, N11 and N16 are the inputs of critical gates. Their αs are listed on figure 3. αs of the other input which are not on the timing-critical paths are denoted by x (i.e., don't care value). Besides, the input signal will also be denoted by x if it has don't care value under these two vectors.

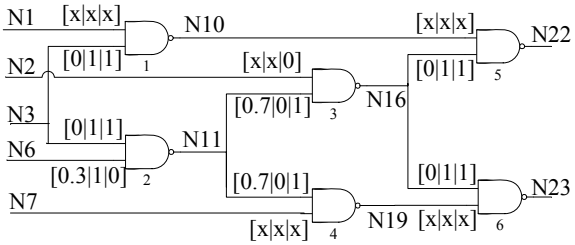

Figure 3. ATPG-like process on C17.

As shown in figure 3, N3 and N16 have 0-value α , so a primitive vector (xx10x) is generated to set 1-signal to N3 and N16. Then, corresponding x bits in the primitive vector are filled to set 1-signal to N6 and N11 separately. Filling process obtains two new vectors {x110x, x010x}. The αs listed on the figure denote that during a period of time t in standby mode, N3 and N16 should have 1-signal persistently while N6 and N11 should have 1-signal for $0.7t$ and $0.3t$ separately. This optimal combination of αs can be achieved by applying the first vector for $0.7t$ and the second vector for $0.3t$ successively.

2. Hardware Implementation Issues

Nowadays, power management units generate sleep signal to indicate that the functional circuit is stepping into standby mode. This sleep signal can also be used to indicate the start of applying input vectors. The applying time of each vector is the multiple of base applying time (e.g., 100000 clock periods) corresponding to its' duty cycle.

The framework of hardware implementation is illustrated in Figure 4. Input vector, combining with several extra bits which indicate the applying time of the vector, form a new storage word and is stored into a small on-chip ROM. After receiving sleep signal, control logic fetches a storage word from the ROM, decodes it, and then, put the vector into the input data latch of functional circuit. At the same time, a counter in the control unit starts to count by sampling clock signal. When the applying time is reached, control logic fetches next storage word, resets the counter and makes it restart to count.

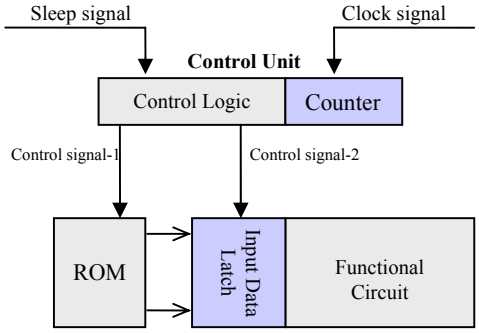

Figure 4. Hardware Implementation.

Ⅴ. EXPERIMENT SETUP AND RESULTS

Experiment is performed on ISCAS benchmark circuit. We modify net list of the circuit, replace the gates having more than 4-input with the combination of 2-4input gates. The intrinsic propagation delay c_0 and coefficient c_i in formula (3) for all kinds of gates is fitted by HSPICE simulation. All HSPICE simulation is performed under PTM [15] 65nm technology. PrimeTime, a static timing analysis tool of SYNOPSYS, combining with a C-language back-end processing procedure, is used to generate the timing-critical path set. The increase of path delay is calculated considering rising and falling transition separately. Gates on the TPs, whose delay increases are counted as a part of delay increase of the TPs during the pruning process, are marked as critical gates. The statistical information of the timing-critical paths and the critical gates is illustrated in table 2. Interestingly, number of timing-critical paths in the sequential circuit is generally small compared with that of the combinational circuit in ISCAS benchmark. The reason is that there are many short paths between two flip-flops. However, since all the flip-flops are in the same clock domain, the clock period (i.e., timing margin) is determined by the delay of the longest path between two flip-flops. So lots of short path is weeded out during the process of generating the timing-critical path set.

Total number of gates and the number of critical gates in the circuit are showed in the third and fourth column in Table 2. The fifth column shows the ratio of them. It's obviously that only a few portions of paths and gates in the circuit are needed to be concerned to resist NBTI degradation.

TABLE 2. STATISTIC OF TIMING-CRITICAL PATHS AND GATES

circuit	TPs	Gates	CGs	G%
c880	18	383	34	8.9
c1908	27	972	60	6.2
c3540	273	1706	126	7.4
c5315	598	2352	118	5.0
s298	2	120	6	5
s820	2	290	7	2.4
s1196	8	530	45	8.5
s1238	8	509	35	6.9
s9234	322	5598	212	3.8

TPs: Timing critical paths; Gates: number of gates in the circuit; CGs: number of the critical gates; G%: the radio of CGs to Gates

Multi-object optimization model is formulated based on the timing-critical paths and the critical gates. The Constraint-1 is set up given timing margin. A C-program are performed to find the implication sets of the critical gates and then, the Constraint-2 is set up based on the implication sets and the logic function of the gates.

A MATLAB script is written to implement a modified sequential quadratic programming (SQP) [16] algorithm, which solves the optimization model to find the optimal solution, starting at an initial value.

After obtained the optimal combination of duty cycles, input vectors are generated using TetraMax, an ATPG tool of SYNOPSYS. Different from ATPG in general post-manufacture test, it doesn't need to concern the propagation of fault effect from faulty site to the PO and just need to set the desirable values on the inputs of critical gates. We modify net list of the circuits, designate all the inputs of critical gates as POs and set all the inputs of critical gates as faulty sites with sa0 faults. During the ATPG process, the *-merge* option in *SET ATPG* command is set to *high* to generate minimum number of input vectors.

Table 3 illustrates the mitigating effect of M-IVC on the delay increase of longest path resulted from 10 years NBTI degradation, which is compared to the case that applying multiple vectors without considering the optimal combination of duty cycles. There are some undetectable faults reported from the ATPG-like process, which denote these inputs of critical gates can not be set to 1. The αs of these critical gates are set to 1 during the process of solving the optimization model.

Column 2 in the table 3 lists the original delay of the path. After implemented M-IVC, the increase delay of the path is mitigated significantly. The final delay of the path resulted from M-IVC implementation is listed in column 3. The ratio of increase is listed in column 4. It can be seen that M-IVC can prevent the emergence of timing violation problem for all the circuits, even under 10 years NBTI degradation. Compared with worst case NBTI degradation (20% increase in 10-year), the maximum delay increase of the longest path is only 6% (c5315). Since the critical gates in

TABLE 3. EXPERIMENTAL RESULTS ON ISCAS BENCHMARK CIRCUIT

circuit	M-IVC (proposed)				M-IVC(proportional)		
	$D_{ori(ps)}$	$D_{final\ (ps)}$	D%	N_{IV}	$D_{ori(ps)}$	$D_{final(ps)}$	D%
c880	1651	1683	2	3	1616	1758	8.8
c1908	2201	2236	1.6	3	2216	2370	7
c3540	3391	3588	5.8	6	3407	3794	11.3
c5315	2962	3142	6	5	2962	3271	10.4
s298	630	630	0	1	630	630	0
s820	812	812	0	1	812	812	0
s1196	2133	2186	2.6	5	2035	2247	10.4
s1238	1730	1740	0.6	7	1730	1961	13.3
s9234	3148	3202	5.7	19	3156	3480	10.3

Dori:orginal path delay;Dfinal:final path delay under M-IVC;D%:ratio of Dfinal and Dori;NIV:number of vectors generated by ATPG-like process

s298 and s820 have no logic conflictions, only one input vector is needed to set all of the inputs of critical gates as 1, which in turn eliminates NBTI degradation completely. Column 5 lists the number of input vectors which obtained from the ATPG-like process. It shows that in general, only a small number of vectors are needed to achieve the optimal combination of duty cycles.

Applying multiple vectors without considering the optimal combination of duty cycles is performed on the same circuit.

All of the inputs of critical gates are set to sa0 faults and are recognized as PO. ATPG-like process generates the vectors for all of the detectable faults. Each vector can set 1-signal on a portion of the inputs of critical gates. Then, the applying time of each vector are decided by the percentage of critical gates the vector covers. We called this kind of applying method as proportional M-IVC. Column 6-8 in table 3 illustrate experimental results using the proportional M-IVC. It can be seen that without considering optimal combination of αs for the critical gates, the delay increase of longest path resulted from NBTI effect is much larger than the one under M-IVC. The longest paths in five circuits have delay increase more than 10%, which results in timing violation problem given 10% timing margin.

Ⅵ. CONCLUSIONS

Some gates in the circuit will suffer static NBTI stress during the long period of time in standby mode. The resulting degradation has strong dependence on the input patterns and duty cycle. Applying multiple vectors in a non-uniform way can achieve the optimal combination of duty cycles for the critical gates, which reduces the stress time of the critical gates, in turn result in significant reduction of NBTI degradation on the timing-critical paths. A multi-object optimization model is formulated to find the optimal combination of duty cycles and an ATPG-like process is presented to generate the corresponding vectors. Experimental results showed that significant reduction of NBTI degradation is achieved by using the proposed M-IVC method.

ACKNOWLEDGEMENT

The work was supported in part by National Natural Science Foundation of China (NSFC) under grant No.(60633060, 60806014, 60831160526, 60606008, 60776031), in part by National Basic Research Program of China (973) under grant No. 2005CB321604, 2005CB321605, and in part by Hi-Tech Research and Development Program of China (863) under grant No.(2007AA01Z109, 2007AA01Z113, 2009AA01Z126, 2007AA01Z476).

REFERENCE

[1] N. Kimizuka. et al. The impact of bias temperature instability for direct-tunneling ultra-thin gate oxide on mosfet scaling. Symposium on VLSI technology., pp 73–74, 1999.

[2] W. Wang, S. Yang, S. Bhardwaj, R. Vattikonda, S. Vrudhula, F. Liu, and Y. Cao. The impact of nbti on the performance of combinational and sequential circuits. Design Automation Conference, pp 364–369, Jun. 2007.

[3] S. Bhardwaj, W. Wang, R. Vattikonda, Y. Cao, and S. Vrudhula. Predictive modeling of the nbti effect for reliable design. IEEE Custom Integrated Circuits Conference, pp 189--192, Sep. 2006.

[4] R. Vattikonda, W. Wang, and Y. Cao. Modeling and minimization of pmos nbti effect for robust nanometer design. Design Automation Conference, pp 1047–1052, Jul. 2006.

[5] Jaume Abella , Xavier Vera , Antonio Gonzalez, Penelope: The NBTI-Aware Processor, Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture, pp 85-96, December,2007.

[6] B. C. Paul, K. Kang, H. Kufluoglu, M. A. Alam, and K. Roy. Temporal performance degradation under NBTI: Estimation and design for improved reliability of nanoscale circuits. ACM/IEEE Design, Automation, and Test Europe, pp 780–785, 2006.

[7] Yu Wang. et al. Temperature-aware NBTI modeling and the impact of input vector control on performance degradation. ACM/IEEE Design, Automation, and Test Europe, pp 546-551, 2007.

[8] S. V. Kumar, C. H. Kim, and S. S. Sapatnekar. An analytical model for negative bias temperature instability. International Conference on Comuter-Aided Design, pp 493–496, 2006.

[9] Sanjay V. Kumar. et al. NBTI-aware synthesis of digital circuits. Design Automation Conference , pp 370-375, 2007.

[10] Zhenyu (Jerry) Qi, Mircea R. Stan. NBTI resilient circuits using adaptive body biasing. GLSVLSI, pp 285-290, 2008.

[11] Wenping Wang, Zile Wei, Shengqi Yang, Yu Cao. An Efficient Method to Identify Critical Gates under Circuit Aging. International Conference on Comuter-Aided Design, pp 735–740, 2007.

[12] Michael H.Schulz. et al. SOCRATES: a highly efficient automatic test pattern generation system. IEEE Transactions on Computer-Aided Design. Vol. 7, No. 1, 1988.

[13] M.H. Schulz and E. Auth, "Improved Deterministic Test Pattern Generation with Applications to Redundancy Identification," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 8, No. 7, pp. 811-816, July 1989.

[14] Kunz, W. Pradhan, D.K. Accelerated dynamic learning for test pattern generation incombinational circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pp 684-694, 1993.

[15] W. Zhao and Y. Cao. New generation of predictive technology model for sub-45nm early design explorations. Available at http://www.eas.asu.edu/_ptm. TED, 53(11):2816–2823, Nov. 2006.

[16] Brayton,R.K. et al. A new algorithm for statistical circuit design based on quasi-Newton methods and function splitting. IEEE Transactions on Circuit and Systems, Vol. CAS-26, pp 784-794, Sep, 1979.