

Leakage Power and Circuit Aging Cooptimization by Gate Replacement Techniques

Yu Wang, *Member, IEEE*, Xiaoming Chen, Wenping Wang, *Member, IEEE*, Yu Cao, *Senior Member, IEEE*, Yuan Xie, and Huazhong Yang, *Senior Member, IEEE*

Abstract—As technology scales, the aging effect caused by negative bias temperature instability (NBTI) has become a major reliability concern. In the mean time, reducing leakage power remains to be one of the key design goals. Because both NBTI-induced circuit degradation and standby leakage power have a strong dependency on the input vectors, input vector control (IVC) technique could be adopted to reduce the leakage power and mitigate NBTI-induced degradation. The IVC technique, however, is ineffective for larger circuits. Consequently, in this paper, we propose two gate replacement algorithms [*direct gate replacement (DGR) algorithm and divide and conquer-based gate replacement (DCBGR) algorithm*], together with optimal input vector selection, to simultaneously reduce the leakage power and mitigate NBTI-induced degradation. Our experimental results on 23 benchmark circuits reveal the following. 1) Both DGR and DCBGR algorithms outperform pure IVC technique by 15%–30% with 5% delay relaxation for three different design goals: leakage power reduction only, NBTI mitigation only, and leakage/NBTI cooptimization. 2) The DCBGR algorithm leads to better optimization results and save on average more than 10× runtime compared to the DGR algorithm. 3) The area overhead for leakage reduction is much more than that for NBTI mitigation.

Index Terms—Gate replacement, internal node control (INC), leakage power, negative bias temperature instability (NBTI).

I. INTRODUCTION

AS TECHNOLOGY scales, negative bias temperature instability (NBTI) is emerging as one of the major reliability degradation mechanisms [1]. NBTI occurs when pMOS transistors are negatively biased (i.e., $V_{gs} = -V_{dd}$) at elevated temperature, causing a shift in the threshold voltages. Over a long period of time, such V_{th} shifts can potentially cause a significant increase in the delay of pMOS devices [2], and result

Manuscript received June 22, 2009; revised September 28, 2009 and November 12, 2009. This work was supported by National Key Technological Program of China under Contract 2008ZX01035-001, by the National Natural Science Foundation of China under Contract 60870001 and Contract 90707002, by the National 863 Project of China under Contract 2009AA01Z130, by Tsinghua National Laboratory for Information Science and Technology Cross-Discipline Foundation. The work of Y. Cao was supported in part by GSRC/SRC. The work of Y. Xie was supported in part by grants from NSF 0643902, 0702617, and an SRC Grant.

Y. Wang, X. Chen, and H. Yang are with the Department of Electrical Engineering, Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: yu-wang@mail.tsinghua.edu.cn).

W. Wang and Y. Cao are with the Department of Electrical Engineering, Arizona State University, Tempe, AZ 85287-5706 USA.

Y. Xie is with the Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA 16802 USA.

Digital Object Identifier 10.1109/TVLSI.2009.2037637

in about 10%–20% degradation in the circuit speed, potentially leading to a functional failure [3]. Consequently, it is important to model, analyze, and mitigate the impact of the NBTI effect on the circuit performance.

Early research on NBTI mainly focused on the analysis of the threshold voltage degradation and the impact on the drive current of semiconductor devices [4]. Later, many researchers studied the NBTI modeling and mitigation techniques on various design abstraction levels. Analytical compact models [5]–[9] that evaluate NBTI effects using power-law timing degradation were proposed to help designers estimate the NBTI-induced performance degradation. Based on these transistor level compact models, circuit level NBTI degradation analysis models were proposed [10]–[12], and static timing analysis (STA) techniques considering NBTI degradation were proposed [13], [14]. Based on these NBTI-aware circuit performance degradation models and STA techniques, researchers have investigated design techniques that can mitigate the NBTI effects. These techniques can be classified into two types: 1) *NBTI compensation techniques*, which include gate/transistor sizing [11], [15], V_{dd}/V_{th} dynamic adjustment [16], [17], guard-banding [18], etc., and 2) *NBTI mitigation techniques*, which include lower supply voltage/temperature/signal probability [13], bit-flipping technique for SRAM [19], NBTI-aware synthesis [20], input vector control (IVC) [14], [21], [22], etc.

These previous works, however, estimated the NBTI-induced lifetime degradation *with the assumption that the circuits operate all the time*. Practically, not every application requires the underlying hardware to operate at the highest performance level all the time. Modules in which the computation is burst are often idle. There are periods during which the pMOS transistors are under static stress condition. Many pMOS transistors affected by NBTI can be found in both combinational and storage blocks when the gate inputs are set to be “0” during the standby time, leading to a larger degradation. Consequently, it is important to accurately estimate the NBTI-induced degradation at the standby time, in order to safely guard-band the circuit performance, and to find design techniques to mitigate such degradation.

IVC is a well-studied technique for leakage power reduction [23]–[31] at circuit standby time. Since NBTI also depends on the input patterns, IVC can be used to mitigate the NBTI effect during the standby mode. Fig. 1 shows the relation between the leakage power and the circuit delay degradation caused by NBTI, under different input vectors. One can observe that, given the required constraints for both leakage and delay degradation (the region in Fig. 1), a set of input vectors can be preselected

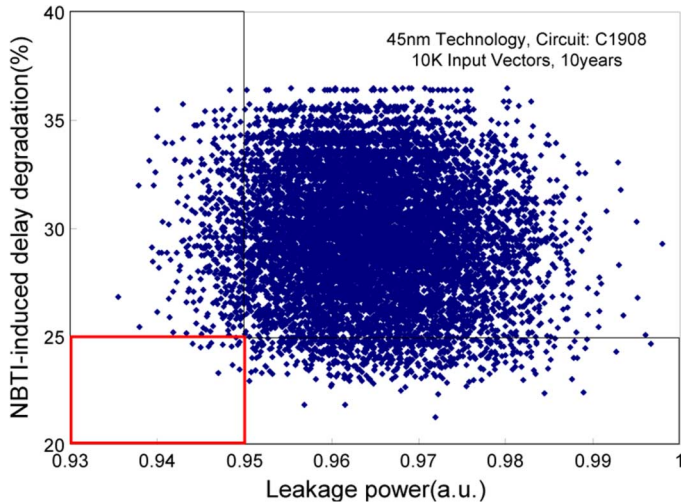


Fig. 1. Leakage power versus delay degradation for different input vectors.

and applied to the entire circuit at the standby mode, such that the total leakage power and the delay degradation are minimized simultaneously. In this example, less than 1% of the sampled input patterns provide the minimum of both circuit delay degradation and the leakage power.

Our previous work [14] proposed a method to select the best input vectors from the minimum leakage vector set. However, the best input vectors for minimizing the leakage power may not be the best ones to minimize NBTI-induced degradation. Furthermore, we did not consider the difference of NBTI effects between active and standby time, the results claimed only 3% saving of the circuit degradation at 90-nm technology node. Our recent research showed that IVC technique, which leads to around 30% circuit performance saving for 90/65/45-nm technology nodes, is proved to be effective for mitigating NBTI-induced degradation during the standby time [21]. Jaume *et al.* [22] used different input vectors to change the zero-probability of internal pMOS transistors, so that the degradation was evenly distributed. The effect of this technique on an adder was evaluated, but detailed research for generic random logic is needed.

Although pure IVC techniques have been evaluated for mitigating the NBTI effect, they are not sufficiently effective when the circuits become larger. How to efficiently find the optimal results for both leakage power and NBTI-induced circuit degradation remains a problem. Some researchers have proposed **internal node control (INC)** [32]–[37] techniques to reduce leakage power, such as gate replacement [32]–[34], control point insertion [36], [37], and sleep transistor insertion [35], in which control point insertion and sleep transistor insertion will change the gate structure. Because the NBTI effect strongly depends on the circuit internal logic values, INC can also be used to mitigate NBTI-induced circuit degradation. In this paper, we prefer one of the INC techniques: gate replacement technique. To the best of our knowledge, this is the first work to achieve NBTI-induced degradation mitigation and leakage power reduction simultaneously by using INC technique.

Bild *et al.* [38] and we [39] independently and simultaneously published similar INC techniques at the same conference. The

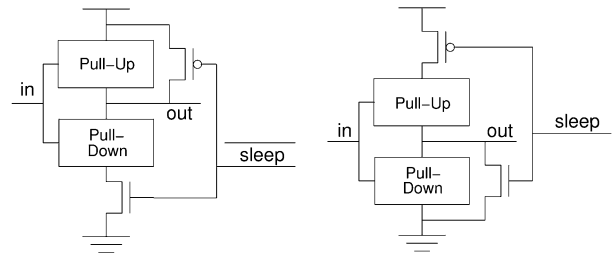


Fig. 2. INC: Control point insertion.

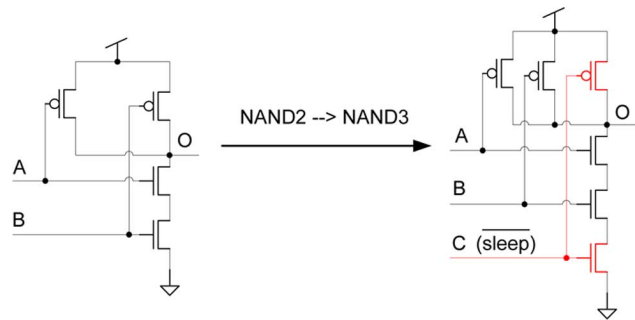


Fig. 3. How to replace a NAND2 gate by a NAND3 gate.

INC method proposed in [38] can be called “control point insertion” technique, which forces the gate output to a specific value when circuit is standby (see Fig. 2), to mitigate NBTI-induced degradation. In fact, their technique is similar to gate replacement proposed by us. For example, a NAND2 gate can be replaced by a NAND3 gate, by adding a pMOS transistor in parallel with the pull-up network and adding an nMOS transistor in series with the pull-down network (see Fig. 3), this operation is the same as the left figure in Fig. 2. The replacement of NOR gate is the same as the right figure in Fig. 2. The advantage of “control point insertion” is that any gate output can be forced either high or low, but with gate replacement method, only certain output value can be forced. However, gate replacement does not destroy gate structure, the standard cell library can be directly used; On the other hand, with “control point insertion”, it may introduce extra stacking effect [20], [38] (for example, we modify a NAND2 gate by the method of the right figure in Fig. 2 will introduce extra stacking effect).

In this paper, based on the previous gate replacement algorithm for leakage reduction [33], [34], we propose two fast gate replacement algorithms which simultaneously reduce the leakage power and mitigate NBTI-induced circuit degradation. The contributions of this paper can be summarized in the following aspects.

- 1) For the first time, the gate replacement technique is used for NBTI mitigation. Based on the basic gate replacement technique, we first propose a *direct gate replacement (DGR)* algorithm, and then propose a *divide and conquer based gate replacement (DCBGR)* algorithm, to improve the NBTI mitigation/leakage reduction achievement. The complexity of DGR algorithm is $O(n^2)$ in the worst case and $O(n)$ on average; while the complexity of DCBGR is $O(n)$. Consequently, our algorithms, especially DCBGR will serve well when circuit scale becomes larger.

- 2) The experimental results show that both DGR and DCBGR algorithms outperform pure IVC by 15%–30% with 5% delay relaxation at time 0 for three different design goals: for leakage power reduction only, for NBTI mitigation only, and leakage/NBTI cooptimization.
- 3) Our experimental results show that: for larger circuits, IVC technique is less effective, while INC through gate replacement technique is more effective for leakage reduction.
- 4) We also investigate four key issues in gate replacement algorithms. ① The NBTI-leakage sequence in cooptimization should be correct; ② To get an optimal solution, the delay relaxation at time 0 should be well chosen; ③ The tradeoff between leakage and NBTI should be well balanced; ④ The impact of RAS (ratio of active and standby time) on delay improvement is large, but on leakage improvement is very small.
- 5) Although the gate replacement technique is compatible with standard cell design flow, the area penalty remains a problem. The DCBGR algorithm results for leakage reduction only/NBTI mitigation only show that the area penalty for leakage reduction is on average 11.08%, while that for NBTI mitigation is on average 4.18%. The area penalty for leakage reduction is much larger than that for NBTI mitigation.

The rest of this paper is organized as follows. Section II presents the NBTI/leakage model and the cosimulation flow. Section III presents the basic gate replacement technique for both leakage reduction and NBTI mitigation. Section IV proposes our fast algorithms to simultaneously reduce leakage power and mitigate NBTI effect. Section V evaluates our algorithms with ISCAS85 benchmarks and some typical arithmetic logic unit (ALU) circuits. Section VI concludes this paper.

II. PRELIMINARIES

This section presents the NBTI modeling and the leakage modeling, as well as the NBTI/leakage cosimulation framework used in our research.

A. Degradation Model Under NBTI Effect

Depending on the bias condition of pMOS transistor, NBTI has two phases: stress phase and recovery phase. In the stress phase ($V_{gs} = 0$), the holes in the channel weaken the Si-H bonds, which results in the generation of the positive interface charges and the hydrogen species. Correspondingly, threshold voltage (V_{th}) of the pMOS transistor increases. During the recovery phase ($V_g = V_{DD}$), the interface traps can be annealed by the hydrogen species and thus, V_{th} degradation (ΔV_{th}) is partially recovered. If a pMOS device is always under stress condition, it is referred as *static* NBTI. Otherwise, both stress and recovery exist during active circuit operation, it is described as *dynamic* NBTI.

Based on the reaction-diffusion (R-D) mechanism, Alam *et al.* proposed analytical models for NBTI-induced degradation in the first stress and recovery periods [5], [6]. The transistor threshold voltage increase caused by positive interface charges generation was analyzed in [10], [11]. An analytical model for multicycle dynamic NBTI model was proposed in [12], where a recursion formula was used to evaluate the NBTI effect. [7], [9]

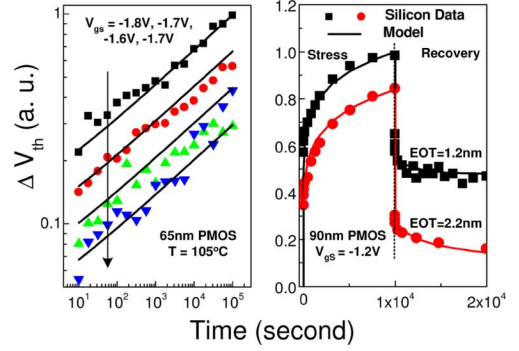


Fig. 4. Threshold voltage degradation model verification for both static and dynamic NBTI [9].

proposed a predictive NBTI model, which described the impact of various process and design parameters on the NBTI effect.

Furthermore, the real time NBTI model was developed in [9], [40]. A long-term prediction model is derived for both static and dynamic NBTI in [9]

$$\begin{aligned} \text{Static: } \Delta V_{th} &= A((1 + \delta)t_{ox} + \sqrt{C(t - t_0)})^{2n} \\ \text{Dynamic: } \Delta V_{th} &= \left(\sqrt{K_v^2 \cdot T_{clk} \cdot \alpha} / (1 - \beta_t^{1/2n}) \right)^{2n} \quad (1) \end{aligned}$$

where A , K_v and β_t are functions of voltage, temperature, stress time and signal probability.

The proposed model is verified by 65- and 90-nm silicon data, as shown in Fig. 4. From the right figure, we know that for dynamic NBTI, there is a sudden change at the beginning of the recovery phase, which has a significant impact on the estimation of the NBTI degradation. This sudden drop can be explained by the fast diffusion in the gate dielectric or trapping/detrapping. Using static NBTI model, which ignores the recovery phase, to predict V_{th} degradation for a gate operating under dynamic condition will lead to a dramatic overestimation in the V_{th} degradation. Therefore, the exact amount of degradation relies on the period of time in which the circuit stays in stress or recovery.

Fig. 5 shows ΔV_{th} prediction by the proposed model. Note that there is a big difference between the static and dynamic NBTI, even the signal probability in the dynamic model is close to 1, which means the pMOS transistors are under stress mode during most of the time, the degradation will be still less than the static case. This happens because the recovery is a highly non-linear function of time, even small periods of recovery results in significant reduction in the threshold voltage. Latest silicon measurements [41], [42] have shown this behavior of NBTI, and our model provides accurate prediction for both static and dynamic NBTI. Therefore, the simple static analysis may cause an extremely pessimistic estimation of the NBTI-induced degradation and consequently, resulting in over-margining in the design stage. On the contrary, only dynamic NBTI model for the total lifetime without considering the static NBTI effect during the standby time may lead to an underestimation of NBTI-induced performance degradation. **In this paper, we use dynamic NBTI model in the active time and static NBTI model in the standby time.** Note that our work in this paper does not constrained by using the R-D NBTI model, and other NBTI models can also be used. Actually, the physical explanation of NBTI

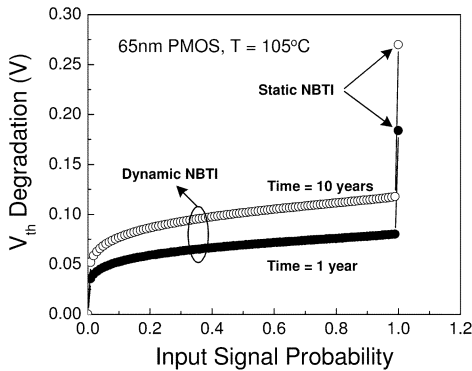


Fig. 5. Static and dynamic NBTI degradation for different input signal probabilities.

is still under investigation by the research community. For example, in addition to R-D theory, trapping/detrapping mechanism is also used to explain the NBTI effect [43]–[46]. The proposed simulation flow and gate replacement techniques in this work is compatible with the models derived by other theories, such as trapping/detrapping mechanism.

Recently, Zheng *et al.* [47] has analyzed the impact of the active-standby sequence and the initial condition on NBTI-induced degradation ratio. The results indicate that NBTI under low switching activity is relative insensitive to the initial condition and the active-standby sequence (the error ratio is less than 5%–10%). Consequently, in this paper, we only consider the RAS (ratio of active and standby time) but not the detailed active-standby sequences.

The delay difference due to ΔV_{th} is given by [11], [14]

$$\Delta d(v) = \frac{\alpha \Delta V_{th}}{V_{gs} - V_{th}} \times d(v) \quad (2)$$

where $d(v)$ is the original delay of gate v which can be extracted from the commercial STA tools. There could be several ΔV_{th} of different pMOS's in one gate. In such cases, we just select the largest one to calculate the gate delay degradation, which is the worst case delay degradation for the gate.

B. Leakage Power Modeling

A leakage lookup table is created by simulating all the cell types in the standard cell library under all possible input patterns. Thus, the leakage power P_{leak} can be expressed as

$$P_{leak}(v) = V_{dd} \times \sum_{input} I_l(v, input) \times Prob(v, input) \quad (3)$$

where $I_l(v, input)$ and $Prob(v, input)$ are the leakage current (including subthreshold and gate leakage current) and the probability of gate v under input pattern $input$. Along the circuit lifetime, the circuit leakage power will be smaller due to the NBTI-induced V_{th} shifts. We take the leakage power at the starting time of the circuit, which is of the maximum value, as the design objective to be optimized.

C. Different Input Vector Dependency of NBTI and Leakage

Both the NBTI and leakage mechanisms have dependencies on the technology and design parameters related to the gate

drive. The NBTI effect on a pMOS transistor depends on V_{gs} and the stress time (duty cycle), which are both related to the input state of a gate. Consequently, all 1's will be the best input pattern with the smallest NBTI-induced degradation for all gate types. However, leakage power varies among different input vectors. Our previous work [21] has simulated all the cells in a commercial 65-nm library, and find out that the best case input patterns to reduce leakage power for NAND/AND/INV gates are all 0's at the inputs, while for NOR/OR/BUF gates are all 1's. Therefore, although both the NBTI and leakage depend on the input patterns, we can see the discrepancy: for NAND/AND/INV gates, the input pattern for minimum leakage will lead to worst NBTI-induced delay degradation; on the contrary, for NOR/OR/BUF gates, the input pattern for minimum leakage will lead to best case NBTI-induced delay degradation. Therefore, we have to get a thorough control of internal node states through the INC techniques, so that the internal node states can be carefully chosen to meet both leakage power and lifetime requirements.

D. NBTI/Leakage Cosimulation Flow

Fig. 6 shows our NBTI/leakage cosimulation flow. For a given circuit, a commercial static timing analysis tool is firstly used to generate the potential critical paths (PCPs). When the circuit is in the active mode, statistical information for input signal probability (SP) is used to generate the internal node SP. When the circuit is in the standby mode, a logic simulator is used to generate the voltage level of each internal node. The active time internal node SP and the standby time internal node states are used to estimate the NBTI-induced V_{th} degradation through transistor level NBTI model. The leakage power is estimated based on the input vector aware leakage lookup tables. Based on the V_{th} degradation estimation and the original timing libraries, a fast path-based NBTI-aware timing analysis is performed. We modify the input vector generation module to implement our gate replacement algorithms.

Note that the proposed flow is not constrained by how the NBTI effect is modeled: one can use NBTI modeling based on R-D theory (as we do in this work) or use NBTI modeling based on trapping/detrapping mechanism [43]–[46].

III. GATE REPLACEMENT (GR) TECHNIQUE

The gate replacement technique is to replace a gate $G(\vec{x})$ by another corresponding library gate $G(\vec{x}, sleep)$ [32], [34], where \vec{x} is the input vector of gate G , $sleep$ is the sleep signal of the circuit, such that it follows:

- 1) $G(\vec{x}, 0) = G(\vec{x})$, when the circuit is active ($sleep = 0$);
- 2) $G(\vec{x}, 1)$ has smaller leakage power or can serve as an internal node control point to mitigate NBTI effect when the circuit is standby ($sleep = 1$).

1) *Gate Replacement for NBTI*: The NBTI effect on a pMOS transistor depends on the stress condition: V_{gs} and stress time, which are both related to the input state of a gate. Consequently, all 1's will be the best input pattern with the smallest NBTI-induced degradation for all gate types. Fig. 7 is an example that shows how to mitigate NBTI-induced degradation by gate replacement. In Fig. 7(1), the NAND2 gate G_2 's delay will be larger if G_1 's output is "0" at the circuit standby time. Through the gate

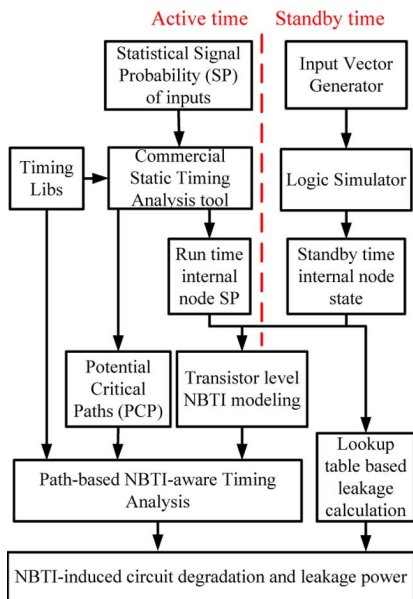


Fig. 6. NBTI and leakage cosimulation flow.

replacement technique, we replace G_1 by an NAND3 gate so that its output is changed to be “1” when the circuit is standby, hence the NBTI effect on G_2 is mitigated. However, if the gate G_1 is not an NAND gate, for example, an AND2 gate (Fig. 7(2)), and its input logics are not all 1’s during the standby time, then the replacement of G_1 is ineffective (G_1 ’s output will not change to be “1” unless all its inputs are 1’s). In this case, we will consider all the fan-in gates of G_1 , and try to make all these fan-in gates’ output be 1’s; then the output logic of G_1 will change to be “1”. Note that the replaced fan-in gates of G_1 may not be in the critical paths at all. Consider an NOR gate in Fig. 7(3), its output will not change to be “1” unless all its inputs are 0’s. So we consider all the fan-in gates of G_1 , and try to make all these fan-in gates’ output be 0’s. This case is similar to Fig. 7(2). In this work, after replacing one gate G , we will calculate the delay of critical path in which G locates, if the replacement of G does have negative effect to the path delay (for example, Fig. 7(3) may produce this problem), then the replacement of G is canceled. The replacement of other gate types are not illustrated in detail, and they are similar to Fig. 7.

Gate replacement technique is adopted to change the behavior of transistors/gates. More exactly, the effect of INC can be summarized “when the circuit is standby, it breaks stress state to complete-non-stress state of one gate/transistor (bring the input “0” to “1”)", so INC can drastically (but not partly) solve NBTI of one gate/transistor during the standby mode.

2) *Gate Replacement for Leakage* [33], [34]: We also call a gate at its worst leakage state (WLS) [34] when its input vector leads to the largest leakage power. Fig. 8 shows how to replace an NAND2 gate to reduce its leakage power. The NAND2 gate is at its WLS with leakage power 775.94 pW, when its input is “11”. We replace it by an NAND3 gate, of which the leakage power is 88.496 pW during the standby time. Then we can save up to 88.60% of the leakage power. (the result is based on the 45-nm library [48]).

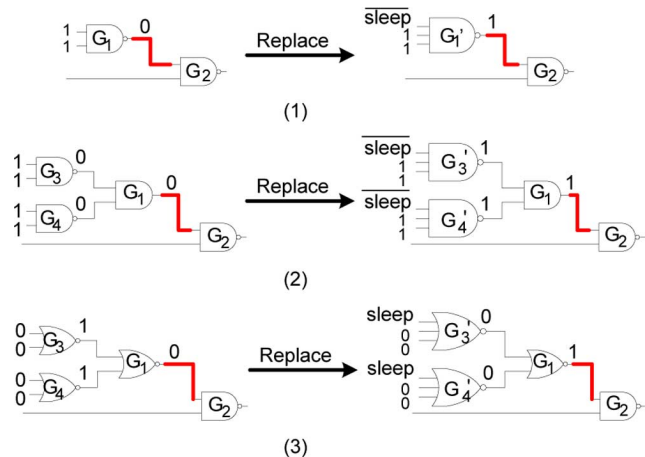


Fig. 7. Gate replacement examples for NBTI mitigation.

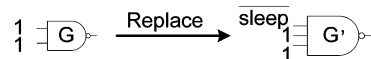


Fig. 8. Gate replacement example for leakage reduction.

3) *Overhead Analysis of Gate Replacement*: From the design view, the gate replacement technique does not require gate structure modification and is compatible with the current standard cell based design flow. This technique can also be combined with optimal input vectors at the primary inputs to get the optimal results for leakage reduction and NBTI mitigation.

The gate replacement technique introduces delay, power, and area overhead. Such overhead, however, can be controlled by adding delay and area constraints in the optimization algorithms, or by performing transistor resizing [34]. In this paper, the delay constraint is set to be less than 5% of the original delay increase at time 0 after gate replacement. From our experimental results, although the delay requirement at time 0 is relaxed, we will get a better circuit delay during most of the circuit operation time. For power overhead, the dynamic power overhead is trivial [34], because the sleep signal remains constant at both active and standby mode.

The sleep signal generation and distribution is a general problem for all the circuits which have two working states: active and standby. Since our experiments are performed in logic level, we do not have the place and route information to get the power and area overhead of sleep signal. Such issues should be carefully handled at the physical design stage, together with other standby time leakage reduction techniques.

Gate replacement and other INC techniques are effective for gates in standby mode. When the circuit is active, they do not change any gate outputs and the circuit maintains the original function. Consequently, INC techniques provide no benefit in active time. In other words, the benefits of INC techniques are independent of the input signal probability, but depend on the length of the standby time.

IV. GATE REPLACEMENT ALGORITHMS

In this section, we propose our two fast gate replacement algorithms: DGR algorithm and DCBGR algorithm.

Direct Gate Replacement (DGR) algorithm

Input: $\{G_1, G_2, \dots, G_N\}$: all the circuit gates in topological order; *SLEEP*: The circuit sleep signal; The circuit input vector \vec{x} .
Output: a circuit 1) of the same functionality when *SLEEP*=0 and 2) with less leakage power and NBTI-induced degradation when *SLEEP*=1.

```

1. for i=1 to N do
2.   if  $G_i$  is in critical paths
3.     search  $G_i$ 's fan-in critical gate  $G_c$ ;
4.     if output( $G_c$ )=0
5.       temporarily replace  $G_c$ ;
6.       if output( $G_c$ )=1
7.         mark the replacement of  $G_c$ ;
8.       else
9.         search all the fan-in gates of  $G_c$ ;
10.        according to the type of  $G_c$ , try to replacement all the fan-in gates of  $G_c$ , to make output( $G_c$ )=1;
11.        if output( $G_c$ )=1
12.          mark the replacement of all the fan-in gates of  $G_c$ ;
13. end for
14. for i=1 to N do
15.   if  $G_i$  is at WLS and not visited
16.     if  $G_i$  isn't critical or is critical but its output will not change after replacement
17.       include  $G_i$  in  $S$ ;
18.       while there is new addition to  $S$  do
19.         for each newly gate  $G$  in  $S$  do
20.           temporarily replace  $G$ ;
21.           if output( $G_c$ ) is changed
22.             include all  $G$ 's un-visited fan-out gates in  $S$ , which is not critical or is critical but its output will not change after replacement;
23.         end for
24.       end while
25.       calculate the circuit leakage power and circuit delay;
26.       if there is leakage reduction and the delay satisfies the requirement
27.         mark all the gates in  $S$  as visited;
28.         mark all the replacement in  $S$ ;
29.       else mark  $G_i$  as visited only;
30.       empty  $S$ ;
31.       else mark  $G_i$  as visited only;
32. end for

```

Fig. 9. Pseudo code for DGR algorithm.

A. DGR Algorithm

Similar to the previous gate replacement algorithm [34], there are also two key steps for the DGR: 1) get the optimal input vector for circuits and 2) gate replacement based on the optimal input vector. We follow the two steps and amend the previous algorithm to further consider NBTI induced circuit degradation together with leakage power.

1) *Get the Optimal Input Vector*: An optimal input vector is chosen from 10 K random input vector search. Since we are considering the NBTI effect and leakage power simultaneously, the object function is as follows:

$$F(D_{\text{circuit}}, P_{\text{leakage}}) = A \times D_{\text{circuit}} + B \times P_{\text{leakage}} \quad (4)$$

where D_{circuit} is the circuit delay after 10 years; P_{leakage} is the circuit leakage power at time 0, which is the maximum leakage during the circuit lifetime. A and B are two weight constants for circuit designers to balance the leakage power requirement and circuit lifetime requirement. The best leakage and circuit delay results of random search are used as our reference for evaluating the results of the gate replacement algorithms.

2) *Direct Gate Replacement Based on the Optimal Input Vector*: In the DGR algorithm, we first arrange all the gates in the circuit into a topological order. The topological order guarantees that when we evaluate a gate, all its predecessor gates have been already considered. Then all the gates are evaluated one by one according to this order. The detailed algorithm is shown in Fig. 9. First, all the critical paths and the gates near the critical paths (the path whose delay is more than 90% of the maximum path delay is called “near-critical path”). In the below, we will call both critical and near-critical gates as “critical gates”) are investigated to mitigate the NBTI

effect, and then we evaluate other gates in the circuits to further reduce the leakage power.

NBTI mitigation in and near the critical paths (line 1–13): In the NBTI mitigation part, when we consider a gate G_i in the topological order, the G_i 's critical fan-in gate G_c in the critical path is first selected. To mitigate the effect of NBTI in the critical path, the output value of G_c should be “1”. If the output of G_c is “0” and there is a corresponding library gate G'_c that can replace G_c , then we temporarily replace G_c by G'_c (line 5). After this replacement, if the output logic is changed to be “1”, then we mark the replacement of G_c . Else the output is not changed to be “1”, then we will find all the fan-in gates of gate G_c , and try to replace them according to G_c 's type, to make the output logic of G_c be “1” (line 9–12).

Leakage power reduction (line 14–32): After the gate replacement for NBTI mitigation, all the gates are visited by a topological order again. This part is modified from [34]. We skip the gates that: 1) are not at WLS or 2) are in critical paths and their outputs will be changed after replacement or 3) have already been visited, until we find a new gate G_i at its WLS (line 15–17). Then we temporarily replace G_i and keep a set S that includes all the unvisited gates affected by the replacement of G_i . All the newly added gates in S are temporarily replaced (line 18–24). The total leakage change caused by the replacement and the circuit delay at time 0 are calculated (line 25). If there is leakage reduction and the circuit delay satisfies the requirement (5% delay relaxation at time 0 caused by gate replacement is allowed), all the gates in the set S are marked as replaced and visited (line 26–28). Otherwise we only mark G_i as visited (line 29). This algorithm will not finish until all the gates have been visited.

Complexity analysis: In the DGR algorithm, each gate is evaluated by the topological order, the loop is linear to N , the

total gates number in the circuit. When we consider a gate G for NBTI mitigation or leakage reduction, all G 's fan-in gates or fan-out gates may be investigated. In the leakage reduction part [34], a set S is used to include gate G and all its fan-out gates in most cases. In the worst case, S may include all the gates in the circuit. Consequently, the complexity of this algorithm is $O(N^2)$ in the worst case and $O(KN)$ on average, where K is the maximum fan-out/fan-in number of the gates in the circuit.

B. DCBGR Algorithm

Although DGR algorithm described in the previous subsection can achieve better results compared with the results of pure IVC technique, the complexity is $O(n^2)$ which is not scalable when the circuit size becomes larger. The optimal input vector generation phase is also time-consuming. On the other hand, since the DGR algorithm is performed based on an initial input vector, and the NBTI mitigation part and leakage reduction part are performed step-by-step, so the final optimization results may still have a gap with the optimal ones.

We further propose a DCBGR algorithm based on the improved gate replacement algorithm in [33]: 1) the circuit is first divided into several tree circuits; 2) the dynamic programming algorithm is performed on each tree circuit to achieve better results faster for both NBTI and leakage; 3) we assign new values of dangling nodes in the whole circuit, and continue to perform the algorithm until it converges.

1) *Divide the Circuit Into Tree Circuits*: In the beginning, we divide the circuit into tree circuits by deleting some connections between gates until every gate fans out to at most one gate. For example, if a gate G has fan-outs to k gates G_1, \dots, G_k ($k > 1$), we keep only one connection G_i and delete the other $k - 1$ connections. We keep the connection that has the longest path from G_i to the outputs of the circuit. After deleting the connections, there are many dangling inputs. In this algorithm, all the dangling inputs always equal to the output of their fan-in gates before deleting the connections.

2) *Input Vector Selection and Gate Replacement for Each Tree Circuit*: Based on the previous tree-based gate replacement algorithm [33], we develop an improved dynamic algorithm to generate an optimal input vector together with gate replacement for tree circuits. The detailed algorithm is shown in Fig. 10, and the variables used in Fig. 10 are shown in Table I.

For each tree circuit, all the gates in the tree circuit are visited by the topological order. If the gate is a primary input signal, then all its property values are set to be initial values (line 2–4). When we are visiting an internal gate G_i , all its valid input vector combinations will be evaluated (line 5). For each valid input combination, we calculate the corresponding total leakage power of all the fan-in subtrees of G_i (line 7), as well as the maximum arrival time of G_i considering all its fan-in subtrees (line 8). Using the object function (4), the minimum object function value and the corresponding optimal subtree leakage power and subtree delay rooted at G_i are calculated considering all the valid input combinations of G_i ; meanwhile, the corresponding optimal input vector of the tree circuit can be also calculated (line 9–13). Then gate replacement is considered. If G_i is a critical gate, it is considered to mitigate NBTI effect (line 14–27), else it is considered to reduce leakage power (line 28–33). If it

Divide and Conquer based Gate Replacement algorithm (DCBGR) for a tree circuit
Input: $\{G_1, G_2, \dots, G_N\}$: all the gates in topological order of a tree circuit.
Output: V_{opt} : the optimal input vector of the tree circuit; $\{Rep_1, Rep_2, \dots, Rep_N\}$: Replace G_i when $Rep(i, z)=1$ and G_i 's output is z .

```

1. for  $i=1$  to  $N$  do
2.   if  $G_i$  is an input signal
3.      $LK(i, z)=0$ ;  $DL(i, z)=0$ ;  $V(i, z)=z$ ;
4.     continue;
5.   for each valid input vector  $\vec{x}$  of  $G_i$  do
6.      $z=Output(i, \vec{x})$ ;  $LK = \sum_{j=1}^{N(i)} LK(i_j, \vec{x}_j)$ ;  $DL = \max_{j=1}^{N(i)} DL(i_j, \vec{x}_j)$ ;
7.     if  $F(DL + D(i, \vec{x}), LK + L(i, \vec{x})) < F_{object}(i, z)$ 
8.        $F_{object}(i, z) = F(DL + D(i, \vec{x}), LK + L(i, \vec{x}))$ ;
9.        $LK(i, z) = L(i, \vec{x}) + LK$ ;  $DL(i, z) = D(i, \vec{x}) + DL$ ;
10.       $V(i, z) = \bigcup_{j=1}^{N(i)} V(i_j, \vec{x}_j)$ ;
11.      search  $G_i$ 's fan-in critical gate  $G_c$ ;
12.      if  $G_i$  is in critical path and output( $G_c$ )=0
13.        temporarily replace  $G_c$ ;
14.        if output( $G_c$ )=1
15.           $Rep(c, 1)=true$ ;  $z=Output_R(i, \vec{x})$ ;
16.          re-calculate  $F_{object}(i, z)$ ,  $LK(i, z)$ ,  $DL(i, z)$ ,  $V(i, z)$ ;
17.        else
18.          search all the fan-in gates  $G_{(k)}$  of  $G_c$ ;
19.          according to the type of  $G_c$ , temporarily try to
20.          replacement all  $G_{(k)}$ , to make output( $G_c$ )=1;
21.          if output( $G_c$ )=1
22.             $Rep(\{k\}, output\{G_{(k)}\})=true$ ;  $z=Output_R(i, \vec{x})$ ;
23.            re-calculate  $F_{object}(i, z)$ ,  $LK(i, z)$ ,  $DL(i, z)$ ,  $V(i, z)$ ;
24.          if  $G_i$  is not in critical paths
25.            temporarily replace  $G_i$ ;
26.            if  $L_R(i, x) \times C_{rep} < L(i, x)$  and  $L_R(i, x) + \sum_{j=1}^{N(i)} LK(i_j, \vec{x}_j) < LK(i, z)$ 
27.               $z=Output_R(i, \vec{x})$ ;  $Rep(i, z)=true$ ;
28.              re-calculate  $F_{object}(i, z)$ ,  $LK(i, z)$ ,  $DL(i, z)$ ,  $V(i, z)$ ;
29.            end for
30.          end for
31.           $V_{opt} = F_{object}(N, 0) > F_{object}(N, 1) ? V(N, 1) : V(N, 0)$ ;
32.          calculate  $\{Rep\}$  in reverse topological order;

```

Fig. 10. Pseudo code for DCBGR algorithm for a tree circuit.

TABLE I
VARIABLES USED IN FIG. 10

| Variable | Definition |
|----------------------|---|
| i_j | the j^{th} input of G_i |
| $N(i)$ | the input number of G_i |
| $LK(i, z)$ | the optimal total leakage power of the subtree rooted at G_i when its output is z |
| $DL(i, z)$ | the optimal delay of the subtree rooted at G_i , when its output is z |
| $F_{object}(i, z)$ | the minimum object function value of the subtree rooted at G_i when its output is z |
| $V(i, z)$ | the tree circuit input vector producing $F_{object}(i, z)$ |
| \vec{x} | the input vector of a gate |
| \vec{x}_j | the j^{th} bit of \vec{x} |
| $L(i, \vec{x})$ | the leakage power of G_i under input vector \vec{x} |
| $L_R(i, \vec{x})$ | the leakage power of replaced G_i under input vector \vec{x} |
| $D(i, \vec{x})$ | the gate delay after 10 years of G_i under input vector \vec{x} |
| $Output(i, \vec{x})$ | the output of G_i with its input vector \vec{x} |
| $Rep(i, z)$ | indicate whether to replace G_i when its output is z |

is critical, the gate replacement for NBTI mitigation part (line 14–27) is almost the same as the NBTI mitigation part in DGR algorithm (line 1–13 in Fig. 9); else it will be temporarily replaced to reduce leakage power (line 28–33), this part is as same as [33], and C_{rep} is a constant to control the area and delay increase penalty caused by gate replacement. If there is leakage reduction by replacing G_i , then we mark the replacement $Rep(i, z) = true$. After gate replacement, all the property values of replaced gates and all the fan-out gates of replaced gates must be recalculated (line 20, 27, 33).

After all the gates in the tree circuit are evaluated, the optimal input vector of the tree circuit is judged by the object function

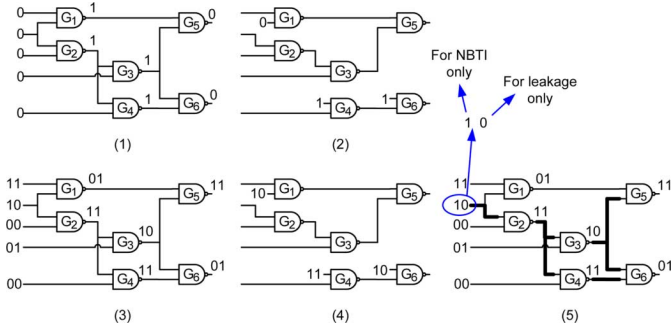


Fig. 11. Example of DCBGR algorithm for NBTI and leakage mitigation.

value of the last gate in the tree circuit (line 36), which is also the primary output gate of the tree circuit. At last we calculate Rep of each gate in a reverse topological order, which indicates whether each gate will be replaced or not.

3) *Assign New Dangling Values and Perform the Algorithm Until It Converges:* When we get the input vector of each tree circuit, the input vector of the whole circuit is also obtained. A logic simulator is used to calculate the logic value of each gate. Then the dangling inputs are assigned with new values again. If there are any dangling inputs that have been changed, the algorithm will be repeated from the most anterior gate in the topological order with new dangling input values until the algorithm generates the same input vector, then the algorithm converges. Otherwise if an input vector that has been appeared previously is generated, which means the algorithm repeats and will not converge to one input vector; we will stop the search, and choose the best input vector in one iteration cycle.

Complexity analysis: In the DCBGR algorithm, each gate is also evaluated by the topological order. For a critical gate, its critical fan-in gate or all the fan-in gates of its critical fan-in gate need to be considered. For a noncritical gate, only the gate itself needs to be considered. So the complexity of the DCBGR algorithm is $O(Kn)$ in the worst case, which is smaller than that of the DGR algorithm, where K is the maximum fan-in number of the gates in the circuit.

C. C17 Circuit as an Example of DCBGR Algorithm

Fig. 11 illustrates an example of the DCBGR algorithm for circuit C17 (at 45-nm technology node). In the beginning, we divide the circuit into tree circuits by deleting the connections in Fig. 11(1). G_1 , G_4 , and G_6 have dangling inputs. If the initial input vector of the circuit is set to all 0's then the value of these dangling inputs are "011", we set "011" to these dangling inputs as their initial values in Fig. 11(2).

Then we run the algorithm for two different object functions: leakage power reduction only and NBTI mitigation only. The critical paths are marked thick in Fig. 11(5). If NBTI mitigation is considered, the internal node values along these paths should be "1" as more as possible. Then the dynamic algorithm will generate the optimal input vectors for different object functions for the first time [see Fig. 11(3)]. With the new input vector, we calculate all the logic values in the circuit and assign new dangling inputs in Fig. 11(4). With the new dangling inputs, the algorithm is repeated again until the algorithm converges to

TABLE II
DCBGR RESULTS FOR C17 CIRCUIT(AT 45-nm TECHNOLOGY NODE,
RAS = 1 : 9, SP = 0.5, $T_{\text{active}} = T_{\text{standby}} = 378$ K, $T_{\text{life}} = 10$ YEAR)

| Object function | D_O (ns) | $D_{nbt\text{ti}}$ (ns) | Degradation | LK (nW) |
|-----------------|------------|-------------------------|-------------|-----------|
| Leakage only | 0.2418 | 0.2784 | 15.14% | 1.6738 |
| NBTI only | 0.2418 | 0.2448 | 1.24% | 2.3447 |

the same optimal input vectors for different object functions as shown in Fig. 11(5).

The best leakage input for NAND2 is "10", however the best NBTI input for NAND2 is "11". Hence, the optimal input vector for leakage is "10010" while the optimal input vector for NBTI is "11000". In the NBTI mitigation example, all the internal nodes on the critical paths are set to be "1" when the circuit is standby. Note that in this example, there is no gate replaced, because we set the 5% constraint of circuit delay increase at time 0, if any gate in C17 is replaced, then the circuit delay will exceed this constraint.

The detailed results are listed in Table II. D_O is the original delay at time 0. $D_{nbt\text{ti}}$ is the circuit delay after 10 years. LK is the leakage power at time 0. The optimal result for NBTI can save the circuit degradation from 15.14% to 1.24%, since the NBTI effect are eliminated when the circuit is standby. The optimal result for leakage power shows 28.6% reduction compared with the result for NBTI mitigation only.

V. IMPLEMENTATION AND SIMULATION RESULTS

A. Implementation

We implement our NBTI/Leakage cosimulation flow and the gate replacement algorithms by C++. A commercial static timing analysis tool, PrimeTime from Synopsys, is used to perform the timing analysis and generate the timing report, as well as the internal node signal probabilities. Benchmark circuits are synthesized using an open cell library (45 nm) [48] that are based on the PTM 45-nm transistor model [49]. Some key technology parameters are: $V_{\text{dd}} = 0.8$ V; $|V_{\text{th}}| = 0.18$ V for both nMOS and pMOS transistors; $T_{\text{ox}} = 1.1$ nm. ISCAS85 benchmark and some arithmetic component circuits are used to evaluate our algorithms. The active time temperature T_{active} and standby time temperature T_{standby} are both set to be 378 K corresponding to the worst-case NBTI-induced degradation and leakage power. Ratio of active and standby time (RAS) is set to be 1:9. We set input probabilities of all the input nodes to 0.5 for simplicity. The circuit lifetime is set to be 10 years.

B. Experimental Results

1) *Potential Analysis:* We define the *theoretic upper bound* for NBTI-induced circuit delay degradation D_{UB} is the maximum delay degradation when all the internal nodes are all 0's during circuit standby time; while the *theoretic lower bound* for NBTI-induced circuit delay degradation D_{LB} is the minimum delay degradation when all the internal nodes are all 1's during the circuit standby time. Of course, in a realistic design, there exists no such input vector that makes the internal nodes all 1's or all 0's, so D_{UB} and D_{LB} only define the maximum possible upper and lower bounds of NBTI-induced degradation, and

TABLE III
THEORETIC BOUNDS, POTENTIAL ANALYSIS AND THE RESULTS OF INPUT VECTOR RANDOM SEARCH. (RAS = 1 : 9, SP = 0.5, $T_{\text{active}} = T_{\text{standby}} = 378 \text{ K}$, $T_{\text{life}} = 10 \text{ YEAR}$)

| Benchmark Circuits | Gate# | D_O (ns) | D_{UB} (%) | D_{LB} (%) | Potential (%) | D_W (ns) | D_B (ns) | LK_W (μW) | LK_B (μW) |
|--------------------|-------|------------|--------------|--------------|---------------|------------|------------|--------------------------|--------------------------|
| array4x4 | 149 | 11.133 | 49.16 | 4.54 | 90.77 | 16.284 | 12.609 | 0.1003 | 0.0828 |
| pmult4x4 | 206 | 12.203 | 49.42 | 4.30 | 91.29 | 16.784 | 14.148 | 0.1215 | 0.1063 |
| bkung16 | 225 | 8.175 | 49.17 | 4.50 | 90.84 | 12.194 | 8.366 | 0.1313 | 0.1167 |
| log16 | 256 | 5.023 | 48.10 | 6.22 | 87.06 | 7.439 | 6.881 | 0.1581 | 0.1350 |
| kogge16 | 294 | 5.713 | 49.55 | 4.89 | 90.12 | 8.499 | 6.971 | 0.1777 | 0.1553 |
| c432 | 296 | 16.929 | 49.81 | 3.39 | 93.20 | 23.942 | 19.689 | 0.1661 | 0.1350 |
| c499 | 428 | 9.78 | 49.28 | 3.22 | 93.47 | 14.129 | 12.339 | 0.2313 | 0.1958 |
| bkung32 | 462 | 9.91 | 49.32 | 4.34 | 91.20 | 14.797 | 11.089 | 0.2633 | 0.2418 |
| c880 | 544 | 13.605 | 46.90 | 5.49 | 88.30 | 18.750 | 16.005 | 0.2982 | 0.2616 |
| log32 | 640 | 7.01 | 47.54 | 6.14 | 87.08 | 10.343 | 9.632 | 0.3947 | 0.3402 |
| kogge32 | 678 | 6.594 | 49.04 | 4.73 | 90.35 | 9.822 | 8.353 | 0.4022 | 0.3644 |
| array8x8 | 681 | 27.968 | 49.62 | 4.06 | 91.82 | 40.881 | 33.216 | 0.4571 | 0.3818 |
| pmult8x8 | 826 | 20.011 | 49.57 | 4.16 | 91.61 | 27.500 | 22.165 | 0.5247 | 0.4439 |
| c1355 | 942 | 11.868 | 49.24 | 3.65 | 92.59 | 16.046 | 14.997 | 0.4774 | 0.4631 |
| c1908 | 967 | 12.059 | 49.49 | 3.85 | 92.22 | 16.457 | 14.608 | 0.4651 | 0.4336 |
| c2670 | 1172 | 19.839 | 48.85 | 3.94 | 91.94 | 27.788 | 23.717 | 0.6422 | 0.5914 |
| booth9x9 | 1447 | 16.19 | 49.70 | 3.93 | 92.10 | 23.054 | 20.313 | 0.8060 | 0.7497 |
| log64 | 1536 | 9.95 | 47.60 | 6.08 | 87.23 | 14.686 | 13.941 | 0.9466 | 0.8213 |
| c3540 | 1612 | 21.515 | 48.90 | 3.54 | 92.75 | 30.351 | 26.397 | 0.8771 | 0.8277 |
| c5315 | 2358 | 17.938 | 49.56 | 3.76 | 92.42 | 24.923 | 20.763 | 1.3171 | 1.2421 |
| pmult16x16 | 3284 | 35.837 | 49.71 | 4.03 | 91.90 | 48.350 | 40.090 | 2.0109 | 1.8140 |
| c7552 | 3926 | 16.259 | 46.88 | 6.67 | 85.78 | 21.867 | 19.517 | 2.0203 | 1.9279 |
| c6288 | 6656 | 43.736 | 49.67 | 3.64 | 92.67 | 54.659 | 52.269 | 2.8935 | 2.7742 |
| Average | | | 48.96 | 4.48 | 90.81 | 41.70% | 20.81% | | 10.68% |
| Gate# > 800 | | | 49.02 | 4.30 | 91.2 | 37.60% | 21.20% | | 7.56% |

show the maximum potential of standby time NBTI mitigation techniques.

Table III (column 4–6) shows the theoretic bounds and potential analysis of 23 benchmark circuits, where D_O (column 3) is the circuit delay at time 0. From this table, for each circuit, the upper bound is around about 49% and the lower bound is around about 4%, and the difference between D_{UB} and $D_{LB}((D_{UB} - D_{LB})/D_{UB})$, which means the maximum potential of standby time NBTI mitigation techniques, is around about 90%. We also compare the results between the average results and the results of larger circuits (Gate# > 800). For potential analysis of standby time NBTI mitigation techniques, the average results and larger circuits results are almost the same.

We also calculate the theoretic NBTI improvement ratio through gate replacement technique. Assume that there are N gates on a path, the intrinsic delay of each gate is d on average; after T_{life} , the delay increase caused by NBTI of each gate is $p\%$ on average. So without any NBTI mitigation techniques, the NBTI-induced degradation of this path after T_{life} is also $p\%$. We further assume that the gate delay will increase $q\%$ when it is replaced, the maximum number of gates that can be replaced on the path is m , the delay constraint at time 0 is $r\%$. To ensure the delay constraint, which means $m \times d \times q\% < N \times d \times r\%$, m must be limited by $[(rN)/(q)]$, where $[\cdot]$ is the rounding function. After replacing m gates, the path delay degradation after T_{life} is $((N - m)p + mq)(N\%$ (this equation ignores the active NBTI, because it's much lower than the standby NBTI). The NBTI improvement through gate replacement technique is about:

$$\left(1 - \frac{(N - m)p + mq}{Np}\right) \times 100\% = \frac{m}{N} \frac{p - q}{p}. \quad (5)$$

This equation indicates that the approximative average NBTI improvement ratio through gate replacement technique is the product of (m/N) (the percentage of gates that can be replaced on the path) and $(p - q)/(p)$ (the average improvement of replacing one gate). Since $m = [(rN)/(q)]$, when N is sufficiently large $(m/N)(p - q)/(p) \approx r((1/q) - (1/p))$.

In this paper, $r\% = 5\%$; the delay increase of one gate caused by gate replacement is about $q\% = 5\% \sim 25\%$ for different cell types, the average is about 10%–15%, the NBTI-induced delay degradation of one gate is about $p = 30\% \sim 50\%$, so the improvement ratio of gate replacement technique is approximately 25%–50%. Since different circuits contain different cell types and have different structures, the improvement of each circuit may drastically vary. Furthermore, in most of our benchmark circuits, there are much more AND/NAND gates than OR/NOR gates, and the delay increase caused by replacement of AND/NAND (less than 10%) is much smaller than that of OR/NOR (15%–25%). In addition, from Fig. 7 we can see that some non-critical gates may be replaced to mitigate NBTI (their delay relaxation will not contribute to the $r\%$ circuit delay relaxation), so the optimal results may be larger than 25%–50%.

2) *Input Vector Random Search*: Table III (column 7–10) also shows the results of 10 K input vector random search for all the 23 benchmark circuits. D_W and D_B are the worst case and best case NBTI-induced delay after 10 years generated by IVC. LK_W and LK_B are the worst case and best case leakage power at time 0. The difference of NBTI induced delay degradation is on average 20.89% of the original delay; meanwhile the best leakage power can save on average 10.68% compared with the worst case leakage power. The results of larger circuits show that the IVC is less effective for larger circuits.

3) *DGR Algorithm*: Table IV shows the DGR optimization results for leakage reduction only and NBTI mitigation only. D_{imp} is the delay improvement after 10 years. LK_{imp} is the leakage improvement at time 0. a_{inc} is the circuit area increase caused by gate replacement; Rep is the gate replacement ratio; D_{inc} is the delay increase at time 0 caused by gate replacement. The leakage and NBTI improvements are compared with the best results of Random Search in Table III (LK_B and D_B), respectively, because the NBTI mitigation only algorithm and leakage reduction only algorithm are based on the minimum delay vector (MDV) and minimum leakage vector (MLV), respectively.

Our DGR algorithm can outperform the pure IVC about 14.03% and 29.53% for leakage reduction only and NBTI mitigation only respectively. For larger circuits, we have slightly more leakage saving, because there are more noncritical gates in larger circuits and they may be replaced to reduce leakage power. However, for larger circuits, the NBTI improvement is smaller than the average value. Because in larger circuits, there are more critical paths or near-critical paths. If we want to mitigate NBTI-induced circuit delay degradation, all the critical paths and the near-critical paths need to be considered to mitigate NBTI-induced degradation, and it is more difficult than smaller circuits, which have fewer critical paths and near-critical paths.

We also evaluate the runtime, area penalty (a_{inc}), replacement ratio (Rep) and delay penalty at time 0 (D_{inc}). The runtime

TABLE IV

RESULTS OF DGR ALGORITHM. (RAS = 1 : 9, SP = 0.5, $T_{\text{active}} = T_{\text{standby}} = 378 \text{ K}$, $T_{\text{life}} = 10 \text{ YEAR}$) (FOR LEAKAGE/NBTI ONLY ALGORITHM, $LK_{\text{imp}} = (1 - (LK_{\text{rep}})/(LK_B)) * 100\%$, $D_{\text{imp}} = (1 - (D_{\text{life,rep}} - D_B)/(D_{\text{life}} - D_B)) * 100\%$; FOR COOPTIMIZATION, $LK_{\text{imp}} = (1 - (LK_{\text{rep}})/(LK_S)) * 100\%$, $D_{\text{imp}} = (1 - (D_{\text{life,rep}} - D_S)/(D_{\text{life}} - D_S)) * 100\%$. WHERE LK_B AND D_B ARE THE BSET CASE LEAKAGE AND CIRCUIT DELAY OF IVC RESPECTIVELY FROM TABLE III, LK_S AND D_S ARE THE COOPTIMIZATION RESULTS OF IVC, LK_{rep} IS THE LEAKAGE POWER AFTER GATE REPLACEMENT, D_{life} IS THE CIRCUIT DELAY AT T_{life} WITHOUT GATE REPLACEMENT, $D_{\text{life,rep}}$ IS THE CIRCUIT DELAY AT T_{life} AFTER GATE REPLACEMENT)

| Benchmark Circuits | For leakage reduction only | | | | | For NBTI mitigation only | | | | | Co-optimization | | | | | | | | |
|---------------------|----------------------------|-------------|----------------------|-----------|----------------------|--------------------------|-------------|----------------------|-----------|----------------------|--------------------------|--------------|-----------------------|----------------------|----------------------|-----------|-------------|----------------------|--|
| | LK_{imp} (%) | Runtime (s) | a_{inc} (%) | Rep (%) | D_{inc} (%) | D_{imp} (%) | Runtime (s) | a_{inc} (%) | Rep (%) | D_{inc} (%) | LK_S (μW) | D_S (ns) | LK_{imp} (%) | D_{imp} (%) | a_{inc} (%) | Rep (%) | Runtime (s) | D_{inc} (%) | |
| array4x4 | 13.49 | 3.2 | 11.31 | 41.61 | 4.67 | 22.96 | 3.2 | 0.55 | 2.01 | 0.18 | 0.086 | 14.270 | 10.71 | 19.11 | 9.49 | 34.9 | 3.2 | 1.15 | |
| pmult4x4 | 1.66 | 5.1 | 0.70 | 2.43 | 0.00 | 2.20 | 5.1 | 1.96 | 6.80 | 1.23 | 0.106 | 14.572 | 10.31 | 17.91 | 6.73 | 23.3 | 5.1 | 1.57 | |
| bkung16 | 9.88 | 5.6 | 10.45 | 35.11 | 4.15 | 9.79 | 5.6 | 0.52 | 1.78 | 0.00 | 0.127 | 8.646 | 17.35 | 63.47 | 9.03 | 31.11 | 5.7 | 0.00 | |
| log16 | 8.62 | 6.5 | 6.25 | 21.88 | 3.05 | 1.77 | 6.5 | 0.33 | 1.17 | 3.05 | 0.136 | 6.881 | 7.17 | 1.77 | 3.35 | 11.72 | 6.5 | 3.05 | |
| kogge16 | 6.94 | 8.5 | 5.80 | 20.75 | 3.03 | 33.51 | 8.5 | 1.24 | 4.42 | 1.07 | 0.166 | 7.132 | 9.47 | 29.22 | 8.28 | 29.59 | 8.6 | 3.05 | |
| c432 | 13.14 | 8.5 | 11.95 | 38.18 | 1.21 | 42.48 | 8.4 | 2.33 | 7.43 | 1.45 | 0.140 | 19.802 | 8.25 | 45.27 | 6.45 | 20.61 | 8.5 | 1.56 | |
| c499 | 16.52 | 15.7 | 17.09 | 56.07 | 4.90 | 14.45 | 15.8 | 1.55 | 5.14 | 1.52 | 0.199 | 12.828 | 11.01 | 25.79 | 7.67 | 25.47 | 15.7 | 2.84 | |
| bkung32 | 9.10 | 18.0 | 7.65 | 26.41 | 1.76 | 83.66 | 17.8 | 1.82 | 6.28 | 0.00 | 0.255 | 11.114 | 16.00 | 59.73 | 9.66 | 33.33 | 18.1 | 0.12 | |
| c880 | 26.56 | 23.9 | 22.57 | 71.69 | 4.57 | 22.56 | 24.4 | 5.90 | 18.75 | 0.97 | 0.265 | 16.265 | 24.44 | 18.65 | 17.94 | 56.99 | 24.1 | 2.03 | |
| log32 | 13.68 | 29.5 | 10.63 | 37.19 | 4.63 | 53.51 | 29.4 | 0.94 | 3.28 | 2.19 | 0.341 | 9.632 | 8.96 | 9.74 | 4.2 | 14.69 | 29.7 | 2.19 | |
| kogge32 | 5.64 | 36.5 | 5.08 | 18.44 | 4.84 | 68.48 | 36.0 | 1.02 | 3.69 | 1.94 | 0.378 | 8.575 | 6.36 | 35.82 | 9.39 | 34.07 | 37.0 | 2.62 | |
| array8x8 | 10.75 | 36.5 | 11.92 | 42.44 | 5.00 | 29.20 | 36.4 | 1.12 | 4.11 | 0.84 | 0.396 | 36.470 | 11.21 | 2.98 | 6.64 | 24.38 | 37.4 | 1.24 | |
| pmult8x8 | 1.57 | 53.1 | 3.57 | 10.17 | 4.84 | 28.50 | 53.9 | 0.75 | 2.66 | 0.42 | 0.460 | 22.474 | 13.33 | 16.9 | 7.64 | 27.24 | 54.0 | 1.12 | |
| c1355 | 26.85 | 65.4 | 23.26 | 70.91 | 4.90 | 0.09 | 66.7 | 0.18 | 0.53 | 0.00 | 0.469 | 15.062 | 12.55 | 0.7 | 8.84 | 27.18 | 67.5 | 2.22 | |
| c1908 | 29.85 | 68.8 | 23.30 | 69.18 | 4.83 | 33.81 | 71.6 | 3.52 | 10.44 | 2.73 | 0.439 | 14.917 | 23.94 | 10.9 | 18.91 | 56.15 | 70.0 | 4.97 | |
| c2670 | 21.21 | 120.0 | 19.60 | 63.74 | 4.64 | 39.91 | 120.4 | 5.36 | 17.83 | 1.33 | 0.621 | 23.717 | 20.81 | 36.4 | 16.47 | 54.78 | 118.7 | 2.44 | |
| booth9x9 | 5.80 | 171.9 | 9.87 | 28.75 | 4.92 | 49.43 | 165.4 | 3.98 | 13.34 | 3.34 | 0.767 | 20.405 | 16.19 | 34.34 | 12.76 | 42.71 | 171.2 | 2.37 | |
| log64 | 15.99 | 184.4 | 12.54 | 43.88 | 4.85 | 54.47 | 184.6 | 0.52 | 1.82 | 1.54 | 0.822 | 13.941 | 8.69 | 24.84 | 3.79 | 13.28 | 184.2 | 1.54 | |
| c3540 | 21.39 | 207.7 | 21.87 | 74.07 | 4.86 | 51.70 | 215.1 | 7.09 | 24.01 | 3.28 | 0.837 | 26.397 | 19.21 | 22.43 | 18.02 | 61.04 | 215.6 | 2.41 | |
| c5315 | 18.26 | 570.0 | 16.54 | 56.66 | 4.96 | 15.83 | 600.6 | 6.82 | 23.75 | 4.91 | 1.288 | 20.838 | 17.29 | 0 | 15.72 | 54.75 | 571.3 | 4.99 | |
| pmult16x16 | 1.28 | 1214.9 | 2.56 | 7.89 | 4.61 | 6.66 | 1294.8 | 1.57 | 5.69 | 0.82 | 1.928 | 40.090 | 12.84 | 1.54 | 6.72 | 24.3 | 1283.4 | 1.13 | |
| c7552 | 24.45 | 1999.8 | 23.74 | 73.92 | 4.70 | 13.35 | 2327.4 | 9.11 | 29.14 | 4.76 | 1.956 | 19.559 | 25.30 | 5.98 | 20.69 | 66.17 | 2111.2 | 4.95 | |
| c6288 | 20.08 | 7131.3 | 15.80 | 42.92 | 4.96 | 0.88 | 8472.8 | 9.62 | 26.13 | 4.95 | 2.792 | 52.622 | 28.62 | 0.23 | 26.05 | 70.76 | 7287.1 | 5.00 | |
| Average | 14.03 | 521.0 | 12.78 | 41.49 | 4.13 | 29.53 | 598.7 | 2.95 | 9.57 | 1.85 | 2.83% | 2.01% | 14.78 | 21.03 | 11.06 | 36.47 | 536.3 | 2.37 | |
| Gate#>800 | 16.98 | 1071.6 | 15.70 | 49.28 | 4.82 | 26.78 | 1233.9 | 4.41 | 14.12 | 2.55 | 2.42% | 0.52% | 18.07 | 14.02 | 14.15 | 45.31 | 1103.1 | 3.01 | |

grows fast when the circuit becomes larger. For C6288, we may need about two hours. For leakage reduction only, the area increase and replacement ratio is much more than that for NBTI mitigation only, because for NBTI, only critical or near critical gates need to be considered, but for leakage reduction, all the rest gates need to be considered. The delay penalty at time 0 is less than 5% in our algorithm.

Table IV also shows the DGR optimization results for simultaneous leakage reduction and NBTI mitigation. Since DGR algorithm is based on the optimal input vector, the delay and leakage improvements are compared with the cooptimization results of Random Search: LK_S and D_S . LK_S and D_S are generated using a weighted object function ((4)) where leakage power and NBTI mitigation are treated with equivalent importance. Although IVC technique can simultaneously mitigate NBTI and reduce leakage, our DGR performs better: 14.78% more leakage saving and 21.03% more delay compensation. The results also shows that the leakage reduction results for larger circuits are better than the average results, NBTI mitigation results are worse than the average; while larger circuits will introduce larger area penalty since more gates are replaced.

The impact of NBTI-leakage sequence on the results: In the DGR algorithm, we firstly mitigate NBTI degradation in and near critical paths, and then reduce leakage power. We also analyze the case that the order of the two operations is exchanged, which means we firstly reduce leakage power, and then mitigate NBTI degradation. We get on average 8.04% leakage reduction and 14.14% delay compensation by the order-exchanged DGR algorithm compared with IVC. However, Fig. 12 illus-

trates the difference between the original DGR and the order-exchanged DGR algorithms. It shows that on average the results of order-exchanged DGR algorithm are worse than that of the original DGR algorithm. The reason is explained as follows. In the original DGR algorithm, NBTI mitigation is first implemented, after this operation, all the critical gates' logic values are fixed, and their values are beneficial to mitigate NBTI-induced degradation; then in the leakage reduction operation, it will not change these values and ensure the critical path delay. On the contrary, if the leakage reduction is firstly implemented, the critical logic values may be changed, so we may have more serious NBTI-induced degradation.

The impact of delay relaxation at time 0 on the results: Gate replacement will increase the circuit delay at time 0. In our simulation, less than 5% of circuit delay increase at time 0 is allowed. Here we analyze the impact of delay relaxation at time 0 on NBTI and leakage improvement. Fig. 13 shows the delay and leakage improvement under different delay relaxation at time 0 for C5315. The figure shows that the more delay relaxation is, the more delay and leakage improvement will be, since more replaced gate will lead to less leakage and smaller NBTI degradation. However, more delay relaxation at time 0 means the circuit delay is larger at the earlier time. Fig. 14 shows the NBTI-induced delay after gate replacement under different delay relaxation at time 0. It indicates that the circuit delay at time 0 has large impact on the circuit performance for about 10^7 s (120 days). To get a tradeoff between the circuit delay at time 0 and after 10 years, we prefer the 5% delay relaxation at time 0 in our simulation. The x -axis of Fig. 14 is log-

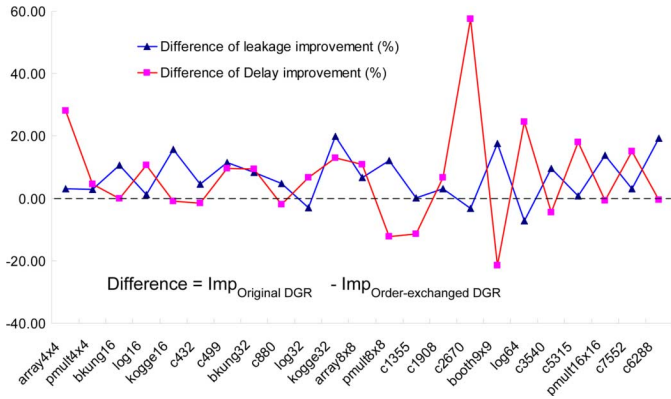


Fig. 12. Difference between the original and the **order-exchanged** DGR algorithms. For most cases the original DGR algorithm is better than the order-exchanged DGR algorithm. (RAS = 1 : 9, SP = 0.5, $T_{active} = T_{standby} = 378$ K, $T_{life} = 10$ year).

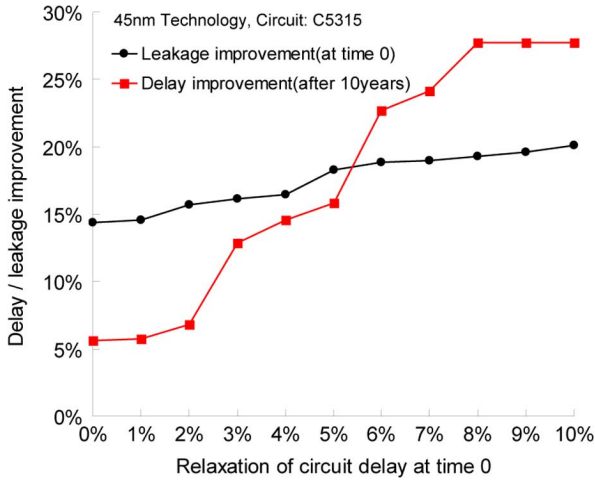


Fig. 13. Leakage and NBTI improvement under different delay relaxation at time 0. (RAS = 1 : 9, SP = 0.5, $T_{active} = T_{standby} = 378$ K, $T_{life} = 10$ year).

arithmetic, so we can see that we have a short time (about 120 days) working in the increased delay state and have a long time (10 year—120 day ≈ 9.6 year) working in the decreased delay state. Our technique keeps the circuit work in the decreased delay state during almost of the circuit life time.

The impact of the lifetime/leakage requirements on the results: Fig. 15 shows the DGR results for C880 under different weighted object function [different weight ratio A/B in the object function (4)]. The figure shows that the value of weight ratio A/B has large impact on the DGR algorithm results. If A/B is small, which means the leakage has larger weight than the delay in the object function, then we get more leakage reduction but more NBTI-induced delay degradation, vice versa. So if the NBTI and the leakage are considered simultaneously, the tradeoff between the leakage power requirement and the circuit lifetime requirement should be well balanced by designers.

4) **DCBGR Algorithm:** Table V shows the optimization results of DCBGR algorithm for leakage reduction only and NBTI mitigation only. The results are compared with the best results of random search in Table III (LK_B and D_B), respectively. The DCBGR results are better than those of DGR algorithm, while

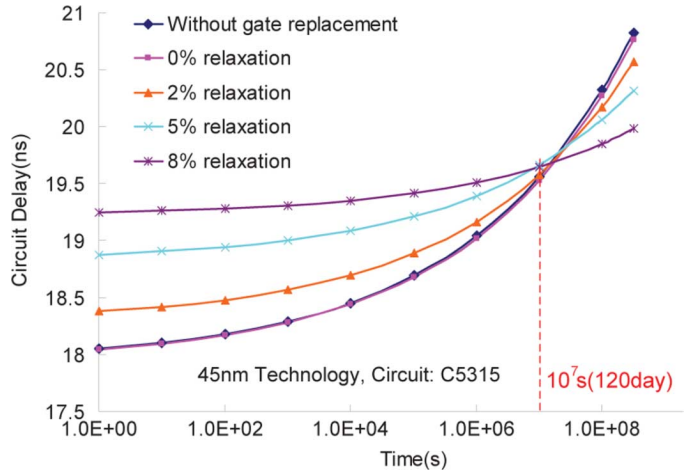


Fig. 14. NBTI-induced delay after gate replacement under different delay relaxation at time 0. (RAS = 1 : 9, SP = 0.5, $T_{active} = T_{standby} = 378$ K, $T_{life} = 10$ year).

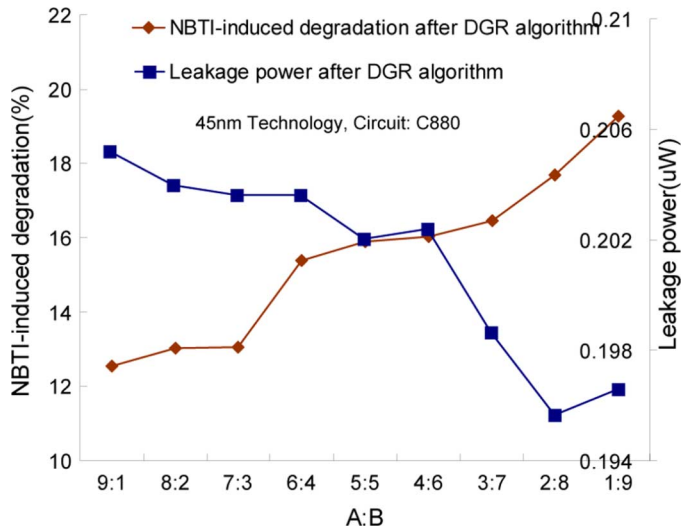


Fig. 15. Leakage and NBTI-induced degradation after DGR algorithm under different weighted object function. (RAS = 1 : 9, SP = 0.5, $T_{active} = T_{standby} = 378$ K, $T_{life} = 10$ year).

the DCBGR algorithm can save on average more than $10\times$ runtime compared with the previous DGR algorithm. For leakage reduction only, DCBGR can achieve on average 23.47% leakage power saving while DGR result is 14.03%. For NBTI mitigation only, DCBGR can compensate on average 51.75% NBTI-induced circuit degradation, while DGR result is 29.53%.

Table V also shows the optimization results of DCBGR algorithm for leakage reduction and NBTI mitigation simultaneously. The delay and leakage improvements are compared with the cooptimization results of Random Search: LK_S and D_S (shown in Table IV). Our DCBGR algorithm can reduce 15.44% more leakage power and mitigate 22.57% more NBTI-induced degradation than pure IVC technique. For larger circuits, leakage reduction ratio is more than the average and NBTI mitigation ratio is less than the average, which is consistent with our previous findings. From the DGR and DCBGR results, the area overhead for leakage reduction is larger than that for NBTI mitigation, since algorithm for leakage will consider all

TABLE V

RESULTS OF DCBGR ALGORITHM. (RAS = 1 : 9, SP = 0.5, $T_{\text{active}} = T_{\text{standby}} = 378$ K, $T_{\text{life}} = 10$ YEAR) (FOR LEAKAGE/NBTI ONLY ALGORITHM, $LK_{\text{imp}} = (1 - (LK_{\text{rep}})/(LK_B)) * 100\%$, $D_{\text{imp}} = (1 - (D_{\text{life,rep}} - D_B)/(D_{\text{life}} - D_B)) * 100\%$; FOR COOPTIMIZATION, $LK_{\text{imp}} = (1 - (LK_{\text{rep}})/(LK_S)) * 100\%$, $D_{\text{imp}} = (1 - (D_{\text{life,rep}} - D_S)/(D_{\text{life}} - D_S)) * 100\%$, WHERE LK_B AND D_B ARE THE BSET CASE LEAKAGE AND CIRCUIT DELAY OF IVC RESPECTIVELY FROM TABLE III, LK_S AND D_S ARE THE COOPTIMIZATION RESULTS OF IVC FROM TABLE IV, LK_{rep} IS THE LEAKAGE POWER AFTER GATE REPLACEMENT, D_{life} IS THE CIRCUIT DELAY AT T_{life} WITHOUT GATE REPLACEMENT, $D_{\text{life,rep}}$ IS THE CIRCUIT DELAY AT T_{life} AFTER GATE REPLACEMENT)

| Benchmark Circuits | For leakage reduction only | | | | | For NBTI mitigation only | | | | | Co-optimization | | | | | |
|---------------------|----------------------------|-------------|----------------------|--------------|----------------------|--------------------------|--------------|----------------------|--------------|----------------------|-----------------------|----------------------|----------------------|--------------|--------------|----------------------|
| | LK_{imp} (%) | Runtime (s) | a_{inc} (%) | Rep (%) | D_{inc} (%) | D_{imp} (%) | Runtime (s) | a_{inc} (%) | Rep (%) | D_{inc} (%) | LK_{imp} (%) | D_{imp} (%) | a_{inc} (%) | Rep (%) | Runtime (s) | D_{inc} (%) |
| array4x4 | 19.16 | 0.02 | 10.58 | 29.53 | 1.65 | 22.96 | 0.02 | 0.55 | 2.01 | 0.18 | 8.93 | 27.94 | 20.99 | 72.48 | 0.04 | 4.64 |
| pmult4x4 | 23.85 | 0.02 | 8.70 | 30.10 | 3.86 | 2.54 | 0.05 | 3.65 | 12.62 | 1.73 | 6.63 | 14.21 | 3.51 | 12.14 | 0.16 | 2.25 |
| bkung16 | 25.69 | 0.06 | 13.03 | 44.89 | 1.11 | 9.79 | 0.08 | 0.90 | 3.11 | 0.00 | 8.66 | 63.47 | 2.06 | 7.11 | 0.26 | 0.00 |
| log16 | 22.17 | 0.03 | 7.14 | 25.00 | 0.00 | 63.58 | 0.09 | 0.78 | 2.73 | 0.00 | 24.81 | 1.77 | 5.80 | 20.31 | 0.23 | 3.05 |
| kogge16 | 20.35 | 0.08 | 12.56 | 44.90 | 1.35 | 54.76 | 0.14 | 1.33 | 4.76 | 2.68 | 0.14 | 6.26 | 2.28 | 8.16 | 0.42 | 0.37 |
| c432 | 19.39 | 0.22 | 9.09 | 29.05 | 3.11 | 67.27 | 0.20 | 2.11 | 6.76 | 1.30 | 17.15 | 47.50 | 17.97 | 43.58 | 0.35 | 0.41 |
| c499 | 32.34 | 0.16 | 14.77 | 49.07 | 2.03 | 86.61 | 0.49 | 9.63 | 32.01 | 1.98 | 26.24 | 33.52 | 24.26 | 74.77 | 1.52 | 4.93 |
| bkung32 | 26.21 | 0.31 | 13.17 | 45.45 | 1.15 | 83.66 | 0.47 | 0.69 | 2.38 | 0.00 | 12.77 | 71.11 | 23.57 | 70.56 | 1.15 | 1.55 |
| c880 | 26.68 | 0.44 | 15.91 | 50.55 | 3.77 | 51.30 | 1.20 | 4.46 | 14.15 | 2.59 | 24.25 | 19.02 | 21.41 | 57.72 | 4.33 | 2.48 |
| log32 | 22.81 | 0.19 | 7.14 | 25.00 | 0.00 | 71.50 | 0.75 | 0.89 | 3.13 | 0.00 | 7.46 | 12.37 | 13.30 | 45.63 | 6.07 | 4.70 |
| kogge32 | 18.36 | 0.61 | 12.16 | 44.10 | 1.02 | 67.28 | 1.59 | 3.70 | 13.42 | 2.26 | 3.40 | 49.75 | 12.48 | 38.05 | 2.91 | 3.25 |
| array8x8 | 18.94 | 0.17 | 7.56 | 27.75 | 1.04 | 42.54 | 0.69 | 3.84 | 14.10 | 0.65 | 15.53 | 3.02 | 22.48 | 77.83 | 1.59 | 4.98 |
| pmult8x8 | 23.61 | 0.25 | 7.44 | 26.51 | 3.57 | 36.46 | 1.64 | 3.46 | 12.35 | 2.96 | 13.82 | 17.28 | 5.16 | 15.86 | 7.26 | 2.74 |
| c1355 | 24.95 | 0.81 | 13.54 | 41.30 | 3.34 | 53.39 | 4.27 | 4.81 | 14.65 | 3.88 | 15.06 | 5.11 | 5.54 | 16.88 | 11.63 | 0.59 |
| c1908 | 23.95 | 0.86 | 13.44 | 39.92 | 4.65 | 48.40 | 3.73 | 2.68 | 7.96 | 2.19 | 18.87 | 20.86 | 4.84 | 14.37 | 14.89 | 0.17 |
| c2670 | 18.53 | 5.83 | 13.83 | 45.99 | 2.13 | 71.02 | 7.73 | 4.08 | 13.57 | 0.89 | 26.80 | 39.07 | 31.58 | 81.66 | 26.34 | 2.53 |
| booth9x9 | 22.30 | 0.72 | 15.71 | 52.59 | 4.25 | 49.43 | 12.42 | 4.05 | 13.55 | 2.58 | 15.01 | 34.47 | 2.68 | 8.71 | 35.26 | 0.47 |
| log64 | 23.28 | 1.56 | 7.14 | 25.00 | 0.00 | 72.45 | 7.39 | 0.37 | 1.30 | 0.00 | 22.98 | 14.14 | 7.66 | 26.63 | 33.22 | 4.97 |
| c3540 | 34.31 | 7.09 | 11.89 | 40.01 | 4.09 | 57.33 | 12.72 | 5.70 | 19.23 | 3.77 | 22.22 | 22.54 | 29.89 | 82.88 | 67.73 | 4.45 |
| c5315 | 17.64 | 25.44 | 9.48 | 33.04 | 4.81 | 66.66 | 94.86 | 5.84 | 22.65 | 3.68 | 0.78 | 1.59 | 7.50 | 26.12 | 114.16 | 4.89 |
| pmult16x16 | 24.53 | 5.88 | 6.64 | 24.00 | 3.88 | 22.72 | 97.59 | 3.66 | 13.25 | 3.41 | 24.67 | 3.75 | 21.89 | 76.80 | 173.53 | 3.84 |
| c7552 | 21.21 | 26.16 | 12.05 | 38.54 | 4.82 | 61.80 | 302.13 | 6.82 | 21.83 | 3.25 | 18.15 | 6.79 | 4.09 | 13.07 | 155.36 | 0.00 |
| c6288 | 29.63 | 34.33 | 11.86 | 32.21 | 3.41 | 26.87 | 109.47 | 2.19 | 5.95 | 3.02 | 20.73 | 3.60 | 12.14 | 32.98 | 141.94 | 4.77 |
| Average | 23.47 | 4.84 | 11.08 | 36.72 | 2.57 | 51.75 | 28.68 | 3.31 | 11.19 | 1.87 | 15.44 | 22.57 | 13.18 | 40.18 | 34.79 | 2.70 |
| Gate#>800 | 24.00 | 9.91 | 11.18 | 36.28 | 3.54 | 51.50 | 59.45 | 3.97 | 13.30 | 2.69 | 18.10 | 15.38 | 12.09 | 36.00 | 71.03 | 2.68 |

the gates in the circuit; while that for NBTI only considers the critical and near-critical paths.

The impact of RAS on the results: We analyze the impact of RAS on leakage and delay improvement by DCBGR algorithm for cooptimization as shown in Fig. 16. Because the NBTI-induced degradation ratio in active time is much higher than that in standby time, while RAS becomes larger (i.e., the length of standby time is shorter), the delay improvement will become lower. For c880 circuit, the delay improvement varies from 19% to 3.3% when RAS varies from 1:9 to 9:1. On the other hand, the leakage improvement only increases a little (from 24.2% to 28.8%), because in this paper, we only consider the maximum leakage, i.e., the leakage power at time 0. When the delay improvement becomes lower, we have more energy and slack to reduce leakage power, so the leakage improvement increases when RAS increases. This phenomenon from another point of view indicates that the gate replacement technique is a standby technique, it provides no more benefit when the circuit is active.

VI. CONCLUSION

Power and reliability have become two key design goals with technology scaling down. In this paper, we have proposed two gate replacement algorithms, for leakage power reduction and NBTI-induced aging effect mitigation, based on our NBTI/leakage cosimulation platform. Both DGR algorithm and DCBGR algorithm are capable to achieve better results

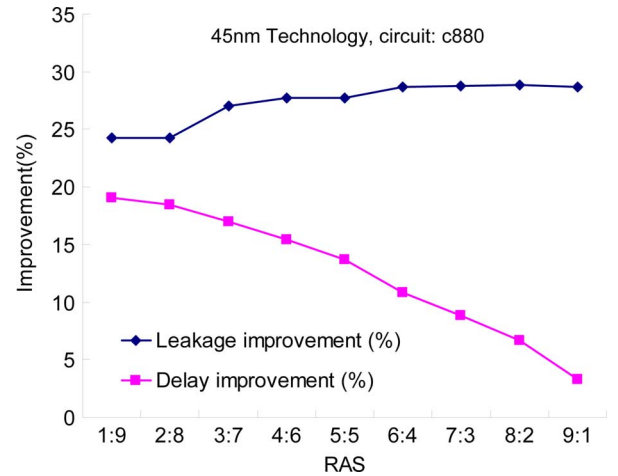


Fig. 16. Leakage and delay improvement under different RAS by DCBGR algorithm for cooptimization. (SP = 0.5, $T_{\text{active}} = T_{\text{standby}} = 378$ K, $T_{\text{life}} = 10$ year).

than pure IVC technique. The DCBGR algorithm with a complexity of $O(n)$ is much faster than the DGR algorithm. We also analyze the overhead of gate replacement technique. The area overhead for leakage power reduction is much larger than that of NBTI mitigation. Less than 5% of circuit delay at time 0 caused by gate replacement techniques will lead to about 15%–30% delay degradation saving compared with the pure IVC technique.

This work provides a generic methodology to optimize the circuit under NBTI and leakage concerns. In our current technology (65 and 45 nm), PBTI is still much lower than NBTI and thus, we ignore that in performing the experiments. In the future, if PBTI becomes comparable to NBTI, it could cancel some benefit from the proposed technique and shifts the optimal point.

If more gates in the circuit critical paths can achieve their best leakage power with all 1's as input, the circuit leakage power will be further reduced during the NBTI optimization phase. Hence, for future work, constrained logic synthesis combined with the gate replacement technique may lead to better cooptimization results.

REFERENCES

- [1] V. Huard, M. Denais, and C. Parthasarathy, "NBTI degradation: From physical mechanisms to modelling," *Microelectron. Reliab.*, vol. 46, no. 1, pp. 1–23, 2006.
- [2] D. K. Schroder and J. A. Babcock, "Negative bias temperature instability: Road to cross in deep submicron silicon semiconductor manufacturing," *J. Appl. Phys.*, vol. 94, no. 1, pp. 1–18, 2003.
- [3] S. Borkar, "Electronics beyond nano-scale cmos," in *Proc. DAC*, 2006, pp. 807–808.
- [4] S. Mahapatra and M. Alam, "A predictive reliability model for PMOS bias temperature degradation," in *IEDM Tech. Dig.*, 2002, pp. 505–508.
- [5] M. Alam, "A critical examination of the mechanics of dynamic NBTI for PMOSFETs," in *IEDM Tech. Dig.*, 2003, pp. 14.4.1–14.4.4.
- [6] S. Mahapatra, P. Kumar, and M. Alam, "Investigation and modeling of interface and bulk trap generation during negative bias temperature instability of p-MOSFETs," *IEEE Trans. Electron. Devices*, vol. 51, no. 9, pp. 1371–1379, Sep. 2004.
- [7] R. Vattikonda, W. Wang, and Y. Cao, "Modeling and minimization of pMOS NBTI effect for robust nanometer design," in *Proc. DAC*, Jul. 2006, pp. 1047–1052.
- [8] W. Wang, V. Reddy, A. T. Krishnan, R. Vattikonda, S. Krishnan, and Y. Cao, "An integrated modeling paradigm of circuit reliability for 65 nm CMOS technology," in *Proc. CICC*, 2007, pp. 511–514.
- [9] S. Bhardwaj, W. Wenping, R. Vattikonda, A. Y. C. Yu Cao, and S. A. V. S. Vrudhula, "Predictive modeling of the nbtI effect for reliable design," in *Proc. IEEE Custom Integr. Circuits Conf.*, 2006, pp. 189–192.
- [10] B. Paul, K. Kang, H. Kufluoglu, M. Alam, and K. Roy, "Impact of NBTI on the temporal performance degradation of digital circuits," *IEEE Electron Device Lett.*, vol. 26, no. 8, pp. 560–562, Sep. 2005.
- [11] B. C. Paul, K. Kunhyuk, H. Kufluoglu, M. A. A. M. A. Alam, and K. A. R. K. Roy, "Temporal performance degradation under nbtI: Estimation and design for improved reliability of nanoscale circuits," in *Proc. Des., Autom. Test Eur. (DATE)*, 2006, vol. 1, pp. 1–6.
- [12] S. V. Kumar, C. H. Kim, and S. S. Sapatnekar, "An analytical model for negative bias temperature instability," in *Proc. IEEE/ACM ICCAD*, 2006, pp. 493–496.
- [13] 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," in *Proc. DAC*, 2007, pp. 364–369.
- [14] Y. Wang, H. Luo, K. He, R. Luo, H. Yang, and Y. Xie, "Temperature-aware NBTI modeling and the impact of input vector control on performance degradation," in *Proc. DATE*, 2007, pp. 546–551.
- [15] K. Kunhyuk, H. Kufluoglu, M. A. Alain, and K. A. R. K. Roy, "Efficient transistor-level sizing technique under temporal performance degradation due to nbtI," in *Proc. Int. Conf. Comput. Des. (ICCD)*, 2006, pp. 216–221.
- [16] L. Zhang and R. Dick, "Scheduled voltage scaling for increasing lifetime in the presence of NBTI," in *Proc. ASP-DAC*, Jan. 2009, pp. 492–497.
- [17] S. Kumar, C. Kim, and S. Sapatnekar, "Adaptive techniques for overcoming performance degradation due to aging in digital circuits," in *Proc. ASP-DAC*, Jan. 2009, pp. 284–289.
- [18] K. Kang, S. Gangwal, S. Park, and K. Roy, "NBTI induced performance degradation in logic and memory circuits: How effectively can we approach a reliability solution?," in *Proc. Asia South Pac. Des. Autom. Conf. (ASPDAC)*, Mar. 2008, pp. 726–731.
- [19] S. Kumar, C. Kim, and S. Sapatnekar, "Impact of NBTI on SRAM read stability and design for reliability," in *Proc. ISQED*, 2006, pp. 210–218.
- [20] S. V. Kumar, C. H. Kim, and S. S. Sapatnekar, "NBTI-aware synthesis of digital circuits," in *Proc. DAC*, 2007, pp. 370–375.
- [21] Y. Wang, X. Chen, W. Wang, V. Balakrishnan, Y. Cao, Y. Xie, and H. Yang, "On the efficacy of input vector control to mitigate nbtI effects and leakage power," in *Proc. ISQED*, Mar. 2009, pp. 19–26.
- [22] J. Abella, X. Vera, and A. Gonzalez, "Penelope: The NBTI-aware processor," in *Proc. MICRO*, 2007, pp. 85–96.
- [23] A. Abdollahi, F. Fallah, and M. Pedram, "Leakage current reduction in CMOS VLSI circuits by input vector control," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. 2, pp. 140–154, Feb. 2004.
- [24] A. Ferre and J. Figueras, "Leakage power bounds in CMOS digital technologies," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 21, no. 6, pp. 731–738, Jun. 2002.
- [25] M. Johnson, D. Somasekhar, and K. Roy, "Models and algorithms for bounds on leakage in CMOS circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 18, no. 6, pp. 714–725, Jun. 1999.
- [26] L. Yuan and G. Qu, "Simultaneous input vector selection and dual threshold voltage assignment for static leakage minimization," in *ICCAD*, Nov. 2007, pp. 548–551.
- [27] F. Gao and J. Hayes, "Exact and heuristic approaches to input vector control for leakage power reduction," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 25, no. 11, pp. 2564–2571, Nov. 2006.
- [28] X. Chang, D. Fan, Y. Han, and Z. Zhang, "Soc leakage power reduction algorithm by input vector control," in *Proc. Int. Symp. Syst.-on-Chip*, Nov. 2005, pp. 86–89.
- [29] X. Chang, D. Fan, Y. Han, Z. Zhang, and X. Li, "Fast algorithm for leakage power reduction by input vector control," in *Proc. 6th Int. Conf. ASIC (ASICON)*, Oct. 2005, vol. 1, pp. 14–18.
- [30] F. Gao and J. Hayes, "Exact and heuristic approaches to input vector control for leakage power reduction," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des. (ICCAD)*, Nov. 2004, pp. 527–532.
- [31] A. Abdollahi, F. Fallah, and M. Pedram, "Leakage current reduction in cmos vlsi circuits by input vector control," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. 2, pp. 140–154, Feb. 2004.
- [32] L. Yuan and G. Qu, "Enhanced leakage reduction technique by gate replacement," in *Proc. 42nd Des. Autom. Conf.*, Jun. 2005, pp. 47–50.
- [33] L. Cheng, L. Deng, D. Chen, and M. Wong, "A fast simultaneous input vector generation and gate replacement algorithm for leakage power reduction," in *Proc. 43rd ACM/IEEE Des. Autom. Conf.*, Jul. 2006, pp. 117–120.
- [34] L. Yuan and G. Qu, "A combined gate replacement and input vector control approach for leakage current reduction," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 2, pp. 173–182, Feb. 2006.
- [35] N. Jayakumar and S. Khatri, "An algorithm to minimize leakage through simultaneous input vector control and circuit modification," in *Proc. DATE*, Apr. 2007, pp. 1–6.
- [36] H. Rahman and C. Chakrabarti, "An efficient control point insertion technique for leakage reduction of scaled CMOS circuits," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 52, no. 8, pp. 496–500, Aug. 2005.
- [37] A. Abdollahi, F. Fallah, and M. Pedram, "Leakage current reduction in CMOS VLSI circuits by input vector control," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. 2, pp. 140–154, Feb. 2004.
- [38] D. Bild, G. Bok, and R. Dick, "Minimization of NBTI performance degradation using internal node control," in *Proc. DATE*, 2009, pp. 148–153.
- [39] Y. Wang, X. Chen, W. Wang, Y. Cao, Y. Xie, and H. Yang, "Gate replacement techniques for simultaneous leakage and aging optimization," in *Proc. DATE*, Apr. 2009, pp. 328–333.
- [40] W. Wang, V. Reddy, A. Krishnan, R. Vattikonda, S. Krishnan, and Y. Cao, "Compact modeling and simulation of circuit reliability for 65 nm CMOS technology," *IEEE Trans. Device Mater. Reliab.*, vol. 7, no. 4, pp. 509–517, Aug. 2007.
- [41] V. Huard, C. Parthasarathy, N. Rallet, C. Guerin, M. Mammase, D. Barge, and C. Ouvrard, "New characterization and modeling approach for nbtI degradation from transistor to product level," in *Proc. IEDM*, Dec. 10–12, 2007, pp. 797–800.
- [42] T. Grasser, B. Kaczer, P. Hehenberger, W. Gos, R. O'Connor, H. Reisinger, W. Gustin, and C. Schlunder, "Simultaneous extraction of recoverable and permanent components contributing to bias-temperature instability," in *Proc. IEDM*, Dec. 10–12, 2007, pp. 801–804.
- [43] L. Zhang, R. Wang, J. Zhuge, R. Huang, D.-W. Kim, D. Park, and Y. Wang, "Impacts of non-negligible electron trapping/detrapping on the NBTI characteristics in silicon nanowire transistors with tin metal gates," in *Proc. IEEE Int. Electron Devices Meet. (IEDM)*, Dec. 2008, pp. 1–4.

- [44] G. Du, D. Ang, Y. Hu, S. Wang, and C. Ng, "Physical framework for NBTI: Insight from ultra-fast switching measurement of NBTI recovery," in *Proc. IEEE Int. Reliab. Phys. Symp. (IRPS)*, May 2008, pp. 735–736.
- [45] W.-L. Lin, J.-C. Lou, Y.-J. Lee, and T.-S. Chao, "Trapping/detrapping characteristics of electrons and holes under dynamic NBTI stress on HFO₂ and HFSION gate dielectrics," in *Proc. 16th IEEE Int. Symp. Phys. Failure Anal. Integr. Circuits (IPFA)*, Jul. 2009, pp. 122–125.
- [46] C. Parthasarathy, M. Denais, V. Huard, G. Ribes, E. Vincent, and A. Bravaix, "Characterization and modeling nbtI for design-in reliability," in *Proc. IEEE Int. Integr. Reliab. Workshop Final Report*, Oct. 2005, p. 5.
- [47] R. Zheng, J. Velamala, V. Reddy, V. Balakrishnan, E. Mintarno, S. Mitra, S. Krishnan, and Y. Cao, "Circuit aging prediction for low-power operation," presented at the CICC, San Jose, CA, 2009.
- [48] Nangate Inc., Sunnyvale, CA, "Nangate Open Cell Library," 2008. [Online]. Available: <http://www.nangate.com/>
- [49] Nanoscale Integration and Modeling (NIMO) Group, ASU, Tempe, AZ, "Predictive Technology Model (PTM)," 2009. [Online]. Available: <http://www.eas.asu.edu/~ptm/>



Yu Wang (S'05–M'07) received the B.S. and Ph.D. (with honor) degrees from the NICS Group, Electronics Engineering Department, Tsinghua University, Beijing, China, in 2002 and 2007, respectively.

Currently, he is an Assistant Professor with the Electrical Engineering Department, Tsinghua University. His research mainly focuses on fast circuit analysis, low power circuit design methodology, reliability-aware circuit design methodology, application specific FPGA design, and on-chip

communication strategies for MPSoC. He has authored and coauthored over 40 papers in refereed journals and conferences.

Dr. Wang is also TPC member for several conferences, such as ICCAD, ISQED, ISVLSI, and etc.



Xiaoming Chen received the B.S. degree from the Electronics Engineering Department, Tsinghua University, Beijing, China, in 2009, where he is currently pursuing the M.S. degree from the Electronics Engineering Department.

His research interests include CAD, low power design, reliability-aware design, and parallel circuit simulation.



Wenping Wang (S'06–M'08) received the B.S. and M.S. degrees from Changchun University of Science and Technology, Changchun, China, in 2000 and 2003, respectively, and the Ph.D. degree in electrical engineering from Arizona State University, Tempe, in 2008.

She did her master research work at Peking University, Beijing, China, from 2001 to 2003. Currently, she is a Reliability Engineer with Vitesse Semiconductor, Austin, TX. She has five patents and has published over 30 articles. Her research interests include

aging testing, modeling of transistor and circuit reliability degradation, particularly negative bias temperature instability (NBTI) and hot carrier injection (HCI)

effects, developing simulation method for both analog and digital design flow, and exploring design solutions for circuit aging.



Yu Cao (S'99–M'02–SM'09) received the B.S. degree in physics from Peking University, Beijing, China, in 1996, the M.A. degree in biophysics, and the Ph.D. degree in electrical engineering from University of California, Berkeley, in 1999 and 2002, respectively.

He worked as a summer intern with Hewlett-Packard Laboratories, Palo Alto, CA, in 2000, and at IBM Microelectronics Division, East Fishkill, NY, in 2001. After working as a Postdoctoral Researcher with the Berkeley Wireless Research Center (BWRC), he is now an Associate Professor of Electrical Engineering with Arizona State University, Tempe, Arizona. He has published numerous articles and coauthored one book on nano-CMOS physical and circuit design. His research interests include physical modeling of nanoscale technologies, design solutions for variability and reliability, and reliable integration of post-silicon technologies.

Dr. Cao was a recipient of the 2009 ACM SIGDA Outstanding New Faculty Award, 2009 Promotion and Tenure Faculty Exemplar, Arizona State University, 2009 Distinguished Lectured of IEEE Circuits and Systems Society, 2008 Chunhui Award for Outstanding Oversea Chinese Scholars, the 2007 Best Paper Award at International Symposium on Low Power Electronics and Design, the 2006 NSF CAREER Award, the 2006 and 2007 IBM Faculty Award, the 2004 Best Paper Award at International Symposium on Quality Electronic Design, and the 2000 Beatrice Winner Award at International Solid-State Circuits Conference. He has served on the technical program committee of many conferences and is a member of the IEEE EDS Compact Modeling Technical Committee.



Yuan Xie received the B.S. degree in electronic engineering from Tsinghua University, Beijing, China, and the M.S. and Ph.D. degrees in electrical engineering from Princeton University, Princeton, NJ.

He is an Associate Professor with the Department of Computer Science and Engineering, Pennsylvania State University, University Park. He has published more than 100 technical papers in the areas of computer architecture, design automation, VLSI design, and embedded system.

Prof. Xie was a recipient of the SRC Inventor Recognition Award in 2002, NSF CAREER Award in 2006, and IBM Faculty Award in 2009. He also received the Best Paper Award in ASP-DAC 2008.



Huazhong Yang (M'97–SM'00) was born in Ziyang, Sichuan Province, China, on Aug. 18, 1967. He received the B.S. degree in microelectronics, and the M.S. and Ph.D. degrees in electronic engineering from Tsinghua University, Beijing, China, in 1989, 1993, and 1998, respectively.

In 1993, he joined the Department of Electronic Engineering, Tsinghua University, Beijing, where he has been a Full Professor since 1998. His research interests include chip design for communication and multimedia applications, synthesis of analog integrated circuits (IC), power estimation and synthesis of digital ICs, noise and delay estimation of deep submicron ICs, yield enhancement, optimization and modeling. He has been in charge of several projects, including projects sponsored by the 863 program, NSFC, the 9th five-year national program and several international cooperation projects. He has authored and coauthored over 100 technical papers and 6 books.

Dr. Yang was recognized as '2000 National Palmary Young Researcher by NSFC.