Minimizing Leakage Power in Aging-Bounded High-level Synthesis with Design Time Multi-V_{th} Assignment

Yibo Chen, Yuan Xie
Pennsylvania State University
University Park, PA 16802
{yxc236, yuanxie}@cse.psu.edu

Yu Wang
Tsinghua University
Beijing, China 100084
yu-wang@tsinghua.edu.cn

Andres Takach
Mentor Graphics Corporation
Wilsonville, OR 97070
andres_takach@mentor.com

***Abstract*—Aging effects (such as Negative Bias Temperature Instability (NBTI)) can cause the temporal degradation of threshold voltage of transistors, and have become major reliability concerns for deep-submicron (DSM) designs. Meanwhile, leakage power dissipation becomes dominant in total power as technology scales. While multi-threshold voltage assignment has been shown as an effective way to reduce leakage, the NBTI-degradation rates vary with different initial threshold voltage assignment, and therefore motivates the co-optimizations of leakage reduction and NBTI mitigation. This paper minimizes leakage power during high-level synthesis of circuits with bounded delay degradation (thus guaranteed lifetime), using multi-V_{th} resource libraries. We first propose a fast evaluation approach for NBTI-induced degradation of architectural function units, and multi-V_{th} resource libraries are built with degradation characterized for each function unit. We then propose an aging-bounded high-level synthesis framework, within which the degraded delays are used to guide the synthesis, and leakage power is optimized through the proposed aging-aware resource rebinding algorithm. Experimental results show that, the proposed techniques can effectively reduce the leakage power with an extra 26% leakage reduction, compared to traditional aging-unaware multi-V_{th} assignment approach.**

I. Introduction

As technology scales, Negative Bias Temperature Instability (NBTI) has become a major reliability concern for circuit designers. NBTI manifests itself as an increase in the transistor threshold voltage, causing the logic gates to slow down, and the critical paths may no longer be able to meet the timing constraints. Circuit level simulations have shown that NBTI can result in a 10% circuit delay degradation after 10 years of service time [1], [2]. Meanwhile, the leakage power of circuits has an exponential dependence on threshold voltage [3]. During circuit operation time, NBTI-induced threshold voltage degradation may severely affect the leakage power. Therefore, ways to accurately analyze and reduce leakage power under the impact of NBTI needs to be explored.

Recently, various techniques have been proposed to mitigate the impact of NBTI, including gate sizing [4], synthesis [5], Input Vector Control (IVC) [6], and Internal Node Control (INC) [7]. Most of the techniques, however, focus at gate level or physical design level. As the number of transistors integrated in a single chip reaches billions, the pace of productivity gains has not kept up to address the increases in design complexity. Consequently, we have seen a recent trend of moving design abstraction to a higher level. **High-level synthesis (HLS)**, which is also known as behavioral synthesis, enables this shift by providing automation to generate optimized hardware from a high-level description of the functionality or algorithms to be implemented in hardware. During HLS, many circuit optimization techniques can be applied at the higher abstraction level (module level), such as Multiple Supply Voltage (multi-V_{dd}) [8], Multiple Threshold Voltage (multi-V_{th}) [9], [10], and Adaptive Body Biasing (ABB) [11]. HLS provides an optimization platform for tackling the NBTI degradation problem with reduced tuning complexity.

This work was supported in part by NSF CAREER 0643902, NSF 0916887, NSFC 60870001, and a grant from SRC.

A principal approach for NBTI mitigation is *guardbanding*, in which extra delay headroom is reserved at design time, allowing the circuit to be degraded to a bounded extent. In order to keep the degraded delay under the bound, circuits can be adaptively adjusted at run time, using Adaptive Supply Voltage (ASV) [12], [13] or Adaptive Body Biasing (ABB) [14], [15]. As run-time tuning usually incurs extra control overhead, this paper focus on design time optimization using multi-V_{th} assignment. Multi-V_{th} assignment has been shown as an effective way to reduce circuit leakage power [9], [10], [16]. However, prior research didn't take into account the temporal degradation of threshold voltage. In terms of NBTI mitigation, this paper is based on the fact that high-V_{th} circuits degrades slower than low-V_{th} circuits [2], [14], [15]. Therefore, the difference between the delay of a high-V_{th} circuit and that of its low-V_{th} equivalent, will become smaller and smaller as the degradation goes on. Given the same delay guardband, high-V_{th} and low-V_{th} circuits may reach the delay bound at about the same time (that means both circuits guarantee the same lifetime), while high-V_{th} circuits consume much less leakage power. Therefore, comparing with the NBTI-unaware multi-V_{th} techniques, more high-V_{th} circuits can be used in favor of leakage power saving. In this paper, the dependencies of leakage and degradation rate on initial V_{th} settings are explored at the behavioral synthesis level, yielding minimal leakage power under the given aging bound.

This paper starts from accurate evaluation of delay degradation and leakage power for architectural function units under different threshold voltages, using the long-term dynamic NBTI model considering the impact of input signal probabilities [1], [2]. After the multi-V_{th} library is characterized, a HLS framework with aging bounds is presented, within which an initial scheduling and resource binding is done according to the anticipated degraded delay of units at the attainable lifetime bound, using only low-V_{th} resource units. Static timing analysis is then performed on the scheduled and bound result, generating timing slack information based on the degraded delay. The "anticipated" slacks are used to guide a resource rebinding algorithm, which iteratively replaces the resource unit with their high-V_{th} equivalents where the delay difference can be fit into the slacks, to reduce leakage power without violating the timing constraints. The effectiveness of the proposed technique is demonstrated on a set of industrial HLS benchmarks, and the improvements are compared with the conventional aging-unaware multi-V_{th} implementations.

To the best of our knowledge, this is the first work to tackle the co-optimization problem that minimize leakage power and mitigate aging effect simultaneously at the behavioral synthesis level. The contributions of this paper can be summarized as follows:

1) A fast evaluation approach for NBTI-induced degradation of architectural function units is introduced, to build multi-V_{th} resource libraries with modeled NBTI-induced degradation;
2) A framework for high-level synthesis with aging bounds is established based on conventional HLS design flow;

978-1-4244-5767-0/10/$26.00 2010 IEEE

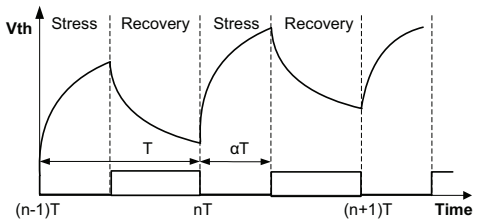

Fig. 1. Threshold voltage degradation during stress and recovery cycles.

3) A heuristic resource binding algorithm is proposed to minimize leakage power under given aging bounds, using multi-V_{th} resource libraries.

II. NBTI and Leakage Characterization

This section presents the NBTI model and the characterization flow to capture the degradation and leakage power of architectural resource units under different threshold voltages. A multi-V_{th} resource library with degraded delay and leakage information is built for the proposed aging-bounded high-level synthesis.

A. NBTI Modeling

NBTI can be described as the generation of interface charges at the Si/SiO_2 interface [17]. Depending on the bias condition of the 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 hydrogen species, correspondingly, threshold voltage (V_{th}) of the PMOS transistors increases. During the recovery phase ($V_{gs} = Vs$), the interface traps can be annealed by the hydrogen species and thus, V_{th} degradation (ΔV_{th}) is partially recovered. Dynamic NBTI model [2] captures the degradation when the PMOS transistor undergoes alternate stress and recovery periods, as shown in Fig. 1.

In order to predict the long term threshold voltage degradation (ΔV_{th}) due to NBTI, a compact model based on reaction-diffusion is proposed in [2], in which ΔV_{th} is modeled as a function of the cycle period T_{clk}, duty ratio α, and circuit running time t:

$$|\Delta V_{th,t}| = \left(\frac{\sqrt{K_v^2 \alpha T_{clk}}}{1-\beta_t^{1/2n}}\right)^{2n}, \beta_t = 1 - \frac{2\xi_1 t_e + \sqrt{\xi_2 C(1-\alpha T_{clk})}}{2t_{ox} + \sqrt{Ct}}$$

$$K_v = \left(\frac{qt_{ox}}{\epsilon_{ox}}\right)^3 K^2 C_{ox}(V_{gs} - V_{th})\sqrt{C}\exp\left(\frac{2E_{ox}}{E_o}\right) \quad (1)$$

All the model parameters, if not aforementioned, are constants from real measurement or data fitting. For the sake of brevity, their meanings and values are not listed here (they can be found in [2]). The model assumes a periodic rectangular waveform as the gate input signal, while in real circuits, the signal waveforms are usually random. In [1], it is analytically proven that, these random waveforms can be converted to equivalent periodic rectangular signals by ensuring that the signal probabilities of the random waveform and that of the deterministic periodic waveform are the same. Therefore, the above model is applicable to any input waveforms with the duty ratio α to be set as the signal probability[1]. The model also shows an exponential dependence between ΔV_{th} and initial V_{th}. For a single PMOS transistor at 45nm technology, the threshold voltage degradation at 10-year lifetime against input signal probabilities and initial threshold voltages is plotted in Fig. 2, which shows the maximum V_{th} degradation varies from $0.11V$ to $0.16V$ with

[1]Here signal probability (SP) is defined as the probability that the signal is at logic 0, since NBTI stress on PMOS devices is caused by logic 0 signals.

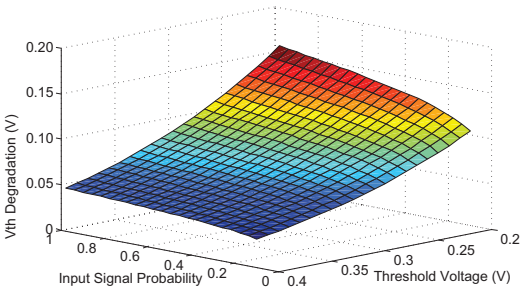

Fig. 2. Threshold voltage degradation against input signal probabilities and initial threshold voltages for a PMOS transistor, showing higher input signal probability and lower initial threshold voltage lead to larger degradation.

different input signal variabilities, and demonstrates the necessity of considering the impact of signal probabilities during NBTI modeling, especially for low-V_{th} circuits.

B. NBTI and Leakage Characterization for Multi-V_{th} Library Components

With transistor-level NBTI models, gate-level NBTI evaluation can be done by propagating gate input signal probabilities to internal transistors, computing the corresponding V_{th} degradation, and calibrating the gate delay by SPICE simulations with the updated V_{th} values [5]. However, for large-scale circuits such as architectural function units with thousands of gates, a fast and efficient evaluation flow utilizing existing analysis tools is needed.

Fig. 3 shows the NBTI and leakage characterization flow used to characterize architectural function units in this paper. The flow starts with the creation of NBTI-characterized technology libraries. Operation conditions, such as initial threshold voltage and anticipated circuit lifetime (e.g., 10 years), are set priorly. Gate-level NBTI models together with netlists of library cells are then fed to the library characterization tool *Liberty NCX* from Synopsys [18], generating standard cells with nominal delays to serve as a baseline, as well as degraded cells with delays based on the appropriate ΔV_{th} resulting from the cells' input probabilities. The names of the degraded cells are annotated with the corresponding input probabilities as suffixes. Leakage power of each cell is also characterized according to the degraded threshold voltage. All the characterized cells are then compiled into technology libraries for targeting and linking in subsequent analysis steps.

Following cell library creation, synthesis is performed in *Design Compiler* taking as input the Verilog/VHDL description of a function unit, using the standard cells with nominal delay to produce a cell netlist of the desired unit. The synthesized netlist is then fed to *Primetime PX* to propagate the primary input probabilities to the internal nodes of the netlist. As the signal probability of each internal node is reported, the cells taking that node as input are annotated with the signal probability value. In the case that a cell takes multiple inputs, the value corresponding to the worst-case NBTI-degradation is selected. With the annotated cell netlist, static timing and power analysis using *Primetime* is performed against the NBTI-characterized technology libraries, generating the NBTI-induced delay and leakage power values for the given function unit.

As implementing multiple threshold voltages in a single chip induces extra manufacturing cost, to reduce the tuning overhead, three voltage levels are used in this work, and the multi-V_{th} technique is applied at the granularity of function units. That means all the gates inside a function unit operate at the same threshold voltage, and threshold voltage only varies from function unit to function unit. Correspondingly, the components in the resource library are then

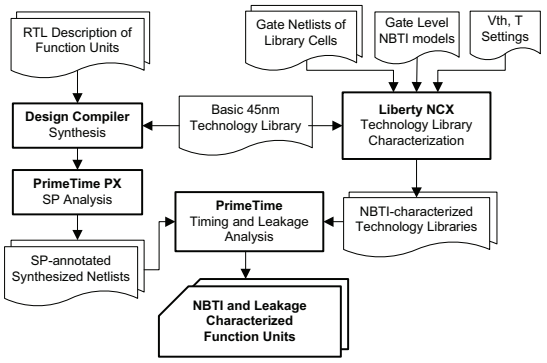

Fig. 3. NBTI and leakage power characterization flow for function units. Design compiler, Primetime and Literby NCX are commercial tools from Synopsys.

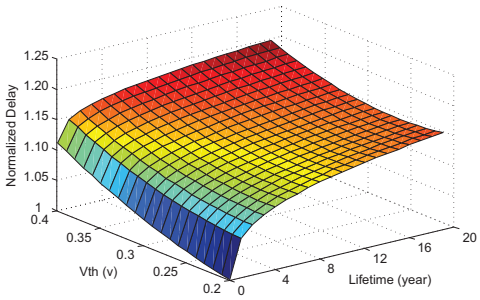

Fig. 4. NBTI-induced delay degradation of a 16-bit adder against different threshold voltages and circuit running time, showing that high-V_{th} adder has a larger initial delay but a lower degradation rate.

characterized under low-V_{th} (LVT), medium-V_{th} (MVT) and high-V_{th} (HVT) settings respectively, using the flow presented in previous subsection. For a given set of resource library components, the characterized results are compiled into a multi-V_{th} resource library, in which each unit has multi-V_{th} implementations with equivalent functionalities but different NBTI-induced delay and leakage power values. Note that a finer-granularity multi-V_{th} assignment for function units could be done at the gate level, which would increase the complexity of the characterization and design space exploration at high-level synthesis.

C. Motivation Example

Using the proposed characterization flow, the delay degradation of a 16-bit adder (as a representative function unit) is sampled and interpolated with different initial threshold voltages at different circuit running times, as shown in Fig. 4. The figure shows that, high-V_{th} circuits have a smaller degradation rate than low-V_{th} circuits.

Fig. 5 shows a motivation example for this paper. The circuit has two paths with various number of function units in each path. Given a performance (delay) requirement D_{req}, a high-level synthesis tool would try to assign high V_{th} to as many function units as possible, such that the leakage reduction is maximized while the delay requirement is still met. However, under the influence of NBTI, during the V_{th} assignment, one must consider the delay degradation as time goes by, and make sure that any path delay during the specific product life time $[0, T_{life}]$ is not larger than the performance requirement D_{req}.

Fig. 5(a) shows a simple *guardbanding* solution to take into account the delay degradation due to NBTI, with extra delay headroom is reserved at design time. One can simply tighten the performance constraint to include the aging effect. For example, by simply setting

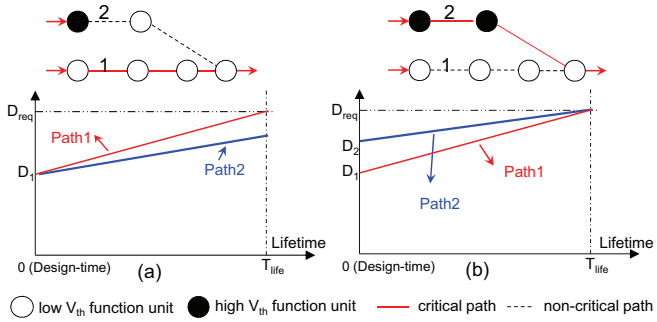

Fig. 5. The conceptual comparison of different optimization strategies: (a) Conventional dual V_{th} assignment with tighter timing constraint $D1$ at design-time; (b)NBTI-aware dual V_{th} assignment with timing constraint D_{req} at T_{life}. Due to NBTI, gates with higher V_{th} has slower degradation.

a new timing constraint $D1$ ($D1 = D_{req} - \Delta D$, in which ΔD is the maximum (worst-case) delay degradation) at *design time*, one can obtain a V_{th} assignment as shown in Fig. 5(a), with $D_{path_1} = D_{path_2} \leq D1$ at design time.

However, such a simple solution ignores the fact that *function units with lower V_{th} tend to age faster, while function units with higher V_{th} has a slower degradation* [2]. For example, in Fig. 5(a), path 2 has a function unit with high V_{th}, while all function units in path 1 are assigned low V_{th}. Consequently, path 2 has a slower aging rate. Being aware of such a difference, one may be more aggressive to assign more function units on path 2 with high V_{th} (Fig. 5(b)), even make it slower than path 1 at *design time* (Fig. 5(b), $D2 > D1$), as long as the path delay D_{path_1} and D_{path_2} at *the end of life time* T_{life} can still meet the timing constraint D_{req}. Such an approach can achieve extra leakage savings (Fig. 5(b) has one extra high-V_{th} function unit than Fig. 5(a)). Note that Fig. 5 is only a simple illustrative example, without considering resource sharing and pipelining, which makes the analysis of performance/power analysis more complicated.

Consequently, based on the fact that *function units with lower V_{th} tend to age faster, while function units with higher V_{th} has a slower degradation*, and the leakage and delay degradation characterization for resource library (in Section II-B), we propose a leakage optimization behavioral synthesis framework considering aging effect. In this framework, we use a new timing constraint (i.e.,D_{path_i} at the end of life time T_{life} can meet the delay requirement D_{req}), instead of a design-time timing constraint (i.e.,D_{path_i} at the design time can meet the delay requirement $D_{req} - \Delta D$), such that extra leakage savings can be achieved, by using more high-V_{th} function units.

III. Leakage Optimization in Aging-bounded HLS

In this section, we present the aging-bounded high-level synthesis framework, and then propose the resource rebinding algorithm for leakage power minimization under aging bounds.

A. Aging-bounded HLS

High-level synthesis (HLS) is the process of transforming a behavioral description into a RTL description. Operations such as additions and multiplications in the data flow graph (DFG) are scheduled into control steps. During the resource allocation and binding stages, operations are bound to corresponding function units in the resource library meeting resource type and latency requirements.

In a conventional HLS flow, given the clock cycle period D_{clk} (which is usually required by the design specification), the timing requirement can be represented as follows:

$$\forall i \in 1 \ldots n, \quad Slack_i \doteq D_{clk} - D_i \geq 0 \tag{2}$$

where n is the number of control steps, D_i and $Slack_i$ are the total delay (the maximum arrival time) and slack at control step i, respectively. The resource binding step binds operations in each control step to optimal function units from the resource library, ensuring that all control steps have non-negative slacks.

In the case of NBTI, the circuit is degraded and the delay D_i gradually increases as time goes on. Eventually the slacks are used up by the NBTI induced delay degradations, and the circuit fails due to a timing violation. A common way to prevent circuits from failing is *guardbanding*, which reserves extra timing headroom at design time by relaxing D_{clk} (thus lower frequency), allowing circuits to degrade to a certain extent. Generally, users will set a lower bound T on the circuit lifetime, expecting the circuit to work flawlessly until time T. How to set the optimal guardband on D_{clk} under the attainable lifetime constraint and perform the design accordingly, are the key problems to be solved in *Aging-bounded HLS*.

Aging-bounded HLS takes as input the lower bound of attainable circuit lifetime, computes the "anticipated" degraded delay of function units at the lifetime bound, and uses the degraded delay values to guide the resource selection and binding. In this case, given an attainable lifetime bound of 10 years, the timing requirement in Expression (2) will be changed to:

$$\forall i \in 1 \ldots n, \quad Slack_{i,10year} \doteq D_{clk} - D_{i,10year} \geq 0 \qquad (3)$$

where $D_{i,10year}$ and $Slack_{i,10year}$ are the total delay and slack at control step i, measured at the service time of 10 years considering degradation, respectively. Resource library characterized in Section II is used in aging-bound HLS, and resource unit are selected according to the new timing requirement. The circuit lifetime is then guaranteed to be more than 10 years by this requirement. As mentioned in Section II-C, an optimized resource selection between multi-V_{th} function units considering both delay degradation and leakage power, will lead to better design decisions.

B. Leakage Optimization in Aging-bounded HLS

With the NBTI-characterized multi-V_{th} resource library and the aging-bounded HLS framework, the leakage power optimization problem can be solved using traditional low-power resource binding algorithms such as integer linear programming [19] and maximum weight independent set [9]. However, the use of multiple threshold voltages enlarges the design space by times and increases the computational complexity. Consequently, this paper uses a greedy heuristic guided by search-and-replace, which has been shown to be practical and effective [10].

The basic flow for leakage optimization in this paper is shown in Fig. 6. The flow takes DFG descriptions of circuits as input, performance initial scheduling and resource binding within conventional HLS algorithms under the new lifetime bound, using only the basic (low-V_{th}) resource units from the resource library characterized in Section II. After that, the leakage power optimization is performed in steps listed as follows.

1) Slacks Analysis in HLS: Timing slacks of each control step at a given lifetime bound are defined in Expression (3). However, in most HLS tools, operation chaining is used which schedules multiple chained operations into one control step. In this case, each operation may has its own non-zero slack. We borrow the methodology of slack analysis at gate-level and apply it for HLS. The chained operations are classified to different levels according to their "logic" (actually architectural) depths. At each level, the maximum arrival time at the output nodes is calculated, and each operation's slack is calculated as the difference between its arrival time and the maximum arrival time at its level.

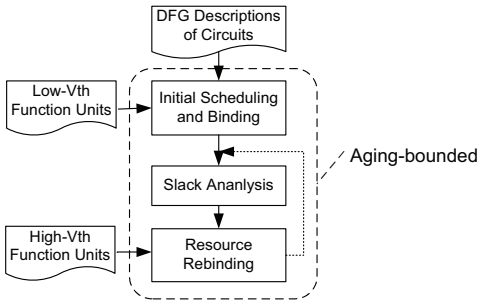

Fig. 6. The flow of leakage optimization in aging-bounded HLS.

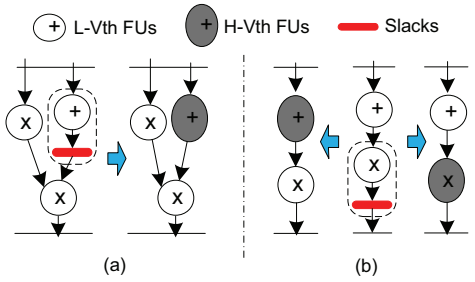

Fig. 7. Resource replacements used in the rebinding: (a) Replacing according to resource slacks; (b) Replacing according to control step slacks.

2) Resource Replacements Used in the Rebinding: Corresponding to the slack analysis, in order to fully explore the design space, two types of resource replacements are used in the search for resource binding:

- **Replacing according to resource slacks**, as shown in Fig. 7-(a). In this case, the slacks are dedicated to the target resources to be replaced. Therefore, the replacing is straightforward to find function-equivalent units while the delay difference can be fit into the slacks.
- **Replacing according to control step slacks**, as shown in Fig. 7-(b). In this case, the slacks can be shared between the chained operations. This complicates the problem. In order to find out the best combination of resource bindings, we assign the whole slack to each level of the chained operations in turn, converting the control step slacks to resource slacks in each level, and finding out the assignment that yields the best result. We omit the possibility that control step slack is distributed among different levels enabling simultaneous replacements in multiple levels, according to the observation that control step slacks are usually not significantly larger than the delay differences of resource units.

3) The Resource Rebinding Algorithm: According to the resource replacing strategies discussed above, a resource rebinding algorithm is proposed to find out all the low-V_{th} candidates, and to replace them with high-V_{th} equivalents for leakage power reduction. The outline of the algorithm is shown in Fig. 8, where a DFG is initially scheduled and bound to low-V_{th} (LVT) resource units, under a given lifetime bound (Lines 1-3). The algorithm then traversals all the control steps (Line 4). In each control step, chained operations are levelized and slack analysis is performed (Lines 5-6), following by the assignment of control step slack to each level of operations, and the subsequent updating of resource slacks (Line 9). Resource rebinding is done by replacing the low-V_{th} (LVT) units with the optimal equivalences found by searching the medium-V_{th} (MVT) and high-V_{th} (HVT) libraries, with the anticipated delay differences that can be fit into the slacks (Line 10, 16-21). The optimal assignment of control step

slack is then determined by comparing the leakage savings resulting from the corresponding resource replacement (Line 12-14).

```
NBTI-BINDING(DFG, T_lifetime)
    ▷ Initialization
 1  Multi-Vth Library Characterization with Lifetime Bound T_lifetime
 2  List Scheduling under Resource Usage Constraints
 3  Initial Binding to LVT resources
    ▷ Aging-aware resource binding
 4  for i ← 1 to NumCSteps
        do {
 5          Levelize Operations in CStepi
 6          Perform Slack Analysis
 7          for j ← 1 to NumLevels
                do {
 8                  Save Slacks(1..Levels, 1..Ops)
 9                  Slacks(j, −) ← Slacks(j, −) + CStepSlack(i)
10                  PSaving(j), Replaces(j) ← RESREPLACE(Slacks)
11                  Restore Slacks(1..Levels, 1..Ops)
                    }
12          Find j so that PSaving(j) = Max(Gain(1..NumLevels))
13          Apply Replaces(j) to DFG
14          TotalPSaving ← TotalPSaving + PSaving(j)
            }
15  Report TotalPSaving and Updated DFG

16  RESREPLACE(Slacks)
17  for all ops in Current CStep
        do {
18          Find Best Resource Candidates from MVT and HVT Libraries
            according to Slacks
19          Perform Resource Replacement for ops
20          Record Replaces and Compute PSaving
            }
21  Return Replaces and PSaving
```

Fig. 8. Outline of the aging-aware resource binding algorithm

As for the computational complexity, in the proposed algorithm, the levelization and slack analysis on each control step can be done by depth-first search which has the complexity of $O(|V|log(|V|))$, and the sizes of the graphs ($|V|$) are usually small in each control step of the DFG. For the resource replacement, according to the slack updating strategy, each operations can have at most two slack values. Assume that for each slack value the resource libraries are searched exhaustively, the maximum loop depth with respect to all operations is 2. Therefore, the overall run time for the proposed resource binding algorithm is $O(n^2)$.

IV. Experiments and Result Analysis

In this section, we present the experimental results of our leakage power optimization framework for aging-bounded high-level synthesis.

We first show the NBTI-induced delay degradation and leakage power characterization of function units. The work is based on a 45nm technology and NCSU FreePDK 45nm cell library [20] is used for the characterization. The threshold voltages are set as: *LVT* = $0.200V$, *MVT* = $0.315V$, *HVT* = $0.423V$, while the supply voltage is $V_{dd} = 1.1V$. Figs. 9 and 10 show the example of characterization results for a set of function units in our resource library, including two 16-bit adders (*bkung16* and *kogge16*), two 32-bit adders (*bkung32* and *kogge32*), two 8-bit×8-bit multipliers (*pmult8x8* and *booth9x9*). Note that the characterization of other components (such as multiplexer and registers) are not depicted here due to space limitation.

From Fig. 9 we can see that the differences between the initial delays of function units with different V_{th}s are significant, as the NBTI-induced degradation (shown by the error bars) goes on, the "anticipated" degraded delays are getting much closer. Meanwhile, Fig. 10 lists the leakage power of function units at the time of first use and the service time of 10 years, under different initial threshold voltages. Note that the y axis is logarithmically plotted, which shows

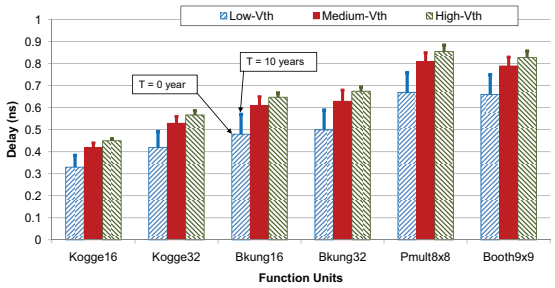

Fig. 9. NBTI-induced delay degradation of function units with different initial threshold voltages, at the circuit lifetime of 10 years. The pattern-filled bars show the original delays without degradation, and the error bars show the NBTI-induced degradations.

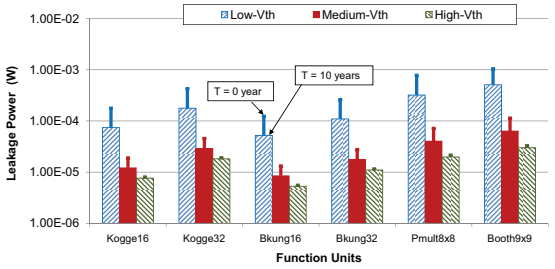

Fig. 10. Leakage power of function units with different initial threshold voltages. The y axis is logarithmically plotted. The pattern-filled bars show the leakages at the circuit lifetime of 10 years, and the error bars show the change of leakage power due to NBTI-induced V_{th} degradation.

the potentials of leakage power savings if high-V_{th} units are used instead of low-V_{th} units, and this motivates the resource rebinding work presented in this paper.

With the NBTI-aware multi-V_{th} resource library characterized, our proposed resource rebinding algorithm for leakage minimization is applied on a set of industrial HLS benchmarks. The profile of benchmarks, as well as the initial scheduling results, are listed in Table I, where the 2nd and 3rd columns show the number of nodes and edges in each benchmark, respectively. The 4th column shows the number of control steps resulted from the initial scheduling, and the last 2 columns show the number of resource instances used in each schedule.

The proposed resource binding algorithm is implemented in C++ and experiments are conducted on a Linux workstation with Intel Xeon 3.2GHz processor and 2GB RAM. All the experiments run in less than 10s of CPU time. All the leakage reduction values in the experimental results are computed against single-V_{th} implementations using only low-V_{th} units.

Fig. 11 shows the comparison of total leakage energy reduction with the traditional aging-unaware multi-V_{th} assignment. In the aging-unaware approach, multi-V_{th} assignment is performed according to the original (non-degraded) delays of function units, yielding an average leakage reduction of 14%, while in our proposed aging-bounded approach, the degraded delays at the lifetime bound of 10 years are used to guide the resource rebinding, and an average leakage reduction of 26% is achieved. The comparison shows that with the proposed aging-bounded approach, the leakage power consumption

TABLE I
BENCHMARK PROFILE AND INITIAL SCHEDULING RESULTS

Name	# nodes	# edges	# CCs	# adders	# multipliers
PR	44	132	12	4	2
WANG	52	132	14	4	2
MCM	96	250	18	7	3
HONDA	99	212	15	6	6
DIR	150	312	16	8	8
STEAM	222	470	19	11	10
CHEM	348	729	29	9	10

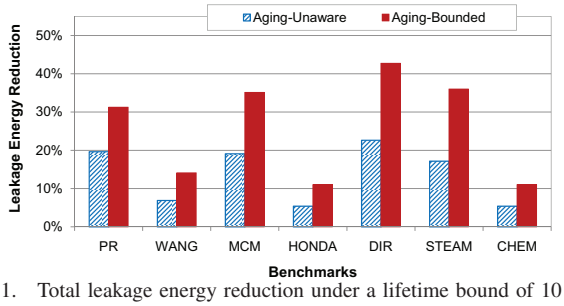

Fig. 11. Total leakage energy reduction under a lifetime bound of 10 years, compared with the traditional aging-unaware multi-V_{th} assignment.

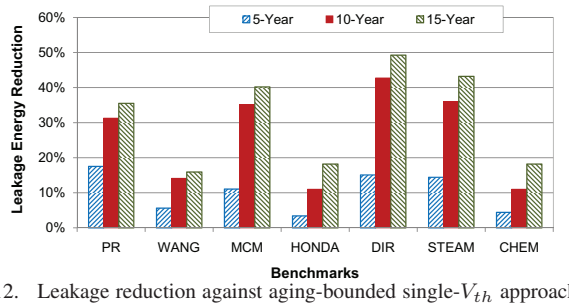

Fig. 12. Leakage reduction against aging-bounded single-V_{th} approach with different lifetime bounds.

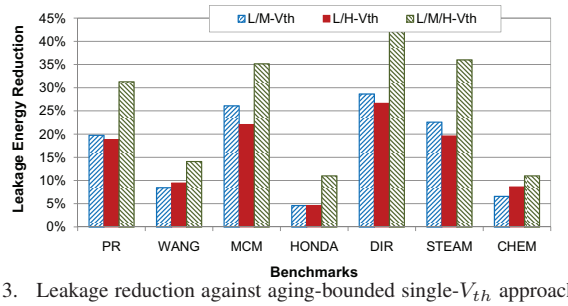

Fig. 13. Leakage reduction against aging-bounded single-V_{th} approach with different threshold voltage settings, under a lifetime bound of 10 years.

can be more effectively reduced without affecting the attainable circuit lifetime.

Fig. 12 explores the impact of different lifetime bounds at the effectiveness of the proposed leakage reduction technique, against aging-bounded single-V_{th} implementations. The average leakage energy reduction under lifetime constraints of 5, 10 and 15 years, are 11%, 26% and 32%, respectively. This suggests that higher lifetime bounds are more favorable for leakage energy reduction. The reason behind is that as the circuit running time gets longer, the delay difference between high-V_{th} and low-V_{th} units decreases, so more high-V_{th} units can be used in the design under higher lifetime bounds. However, higher lifetime bounds also require more guardbanding for the larger overall degradations, yielding lower clock frequencies. Therefore, the tradeoff between leakage power reduction and circuit performance need to be balanced.

Fig. 13 explores the impact of different settings of threshold voltage levels at the effectiveness of the proposed leakage reduction technique, against aging-bounded single-V_{th} implementations. In the comparison experiments, threshold voltage levels are reduced to 2, where only *Medium*-V_{th} units or *High*-V_{th} units can be used for the replacement. The attainable lifetime bound is set as 10 years. The average leakage energy reductions under three threshold voltage settings L-M-V_{th} (using LVT and MVT), L-H-V_{th} (using LVT and HVT), and multi-V_{th} (using all three levels) are 16%, 17% and 26%, respectively. This comparison is raised by the consideration of manufacturing overhead of the multi-V_{th} technology. As total leakage reduction is determined by both the number of units replaced and the leakage saving of each single replacement, if only two levels of threshold voltages are allowed, the results depend on which factor dominates. Fig. 13 shows that in some cases, L-M-V_{th} scheme beats L-H-V_{th} scheme because more LVT units can be replaced with MVT units, while in other cases L-H-V_{th} is more favorable, because the leakage saving brought by HVT units is more significant. This leads to the optimal second threshold voltage selection problem that is left to be explored in future work. Nevertheless, multi-V_{th} design using three levels of threshold voltages can exploit more benefits at the cost of extra manufacturing overhead.

V. Conclusion

This paper investigates the impact of different threshold voltages on the rates of circuit degradation due to NBTI, from the view of high-level synthesis. As the delay difference between low-V_{th} circuits and high-V_{th} equivalents diminishes with degradation, more high-V_{th} can be used in the design, bringing in great potentials for leakage power savings. The paper then proposes a framework to accurately evaluate the delay degradation as well as the leakage power for architectural units, to perform the synthesis under a new metric *Lifetime Bound*, and to optimize the leakage power consumption during the new synthesis process. Experimental results show that, compared to traditional aging-unaware multi-V_{th} assignment approach, the proposed techniques can more effectively reduce the leakage power under the given attainable circuit lifetime bound.

References

[1] S. V. Kumar, C. H. Kim, and S. S. Sapatnekar. An analytical model for negative bias temperature instability. In *ICCAD*, 2006.
[2] S. Bhardwaj *et al.* Predictive modeling of the NBTI effect for reliable design. In *CICC*, 2006.
[3] K. Cao *et al.* BSIM4 gate leakage model including source-drain partition. In *IEDM*, 2000.
[4] K. Kang, H. Kufluoglu, M. A. Alain, and K. Roy. Efficient transistor-level sizing technique under temporal performance degradation due to NBTI. In *ICCD*, 2007.
[5] S. V. Kumar, C. H. Kim, and S. S. Sapatnekar. NBTI-aware synthesis of digital circuits. In *DAC*, 2007.
[6] Y. Wang *et al.* Temperature-aware NBTI modeling and the impact of input vector control on performance degradation. In *DATE*, 2007.
[7] Y. Wang *et al.* Gate replacement techniques for simultaneous leakage and aging optimization. In *DATE*, 2009.
[8] W. Shiue and C. Chakrabarti. Low power scheduling with resources operating at multiple voltages. In *IEEE Transactions on Circuits and Systems*, 2000.
[9] X. Tang, H. Zhou, and P. Banerjee. Leakage power optimization with dual-Vth library in high-level synthesis. In *DAC*, 2005.
[10] K. Khouri and N. Jha. Leakage power analysis and reduction during behavioral synthesis. In *IEEE Transaction on VLSI*, 2002.
[11] F. Wang, X. Wu, and Y. Xie. Variability-driven module selection with joint design time optimization and post-silicon tuning. In *ASPDAC*, 2008.
[12] L. Zhang and R. P. Dick. Scheduled voltage scaling for increasing lifetime in the presence of NBTI. In *ASPDAC*, 2009.
[13] X. Chen *et al.* Variation-aware supply voltage assignment for minimizing circuit degradation and leakage. In *ISLPED*, 2009.
[14] S. Kumar, C. Kim, and S. Sapatnekar. Adaptive techniques for overcoming performance degradation due to aging in digital circuits. In *ASPDAC*, 2009.
[15] A. Tiwari and J. Torrellas. Facelift: Hiding and slowing down aging in multicores. In *MICRO*, 2008.
[16] V. Sundararajan and K. K. Parhi. Low power synthesis of dual threshold voltage CMOS VLSI circuits. In *ISLPED*, 1999.
[17] V. Huard, M. Denais, and C. Parthasarathy. NBTI degradation: From physical mechanisms to modeling. In *Microelectron. Reliab.*, 2006.
[18] Synopsys. Liberty NCX. http://www.synopsys.com/.
[19] W.-T. Shiue. High level synthesis for peak power minimization using ILP. In *ASAP*, 2000.
[20] NCSU. 45nm FreePDK. http://www.eda.ncsu.edu/wiki/FreePDK.