Body Bias Clustering for Low Test-Cost Post-Silicon Tuning

Shuta Kimura Masanori Hashimoto Takao Onoye
Department of Information Systems Engineering, Osaka University
hasimoto@ist.osaka-u.ac.jp

Abstract— Post-silicon tuning is attracting a lot of attention for coping with increasing process variation. However, its tuning cost via testing is still a crucial problem. In this paper, we propose tuning-friendly body bias clustering with multiple bias voltages. The proposed method provides a small set of compensation levels so that the speed and leakage current vary monotonically according to the level. Thanks to this monotonic leveling and limitation of the number of levels, the test-cost of post-silicon tuning is significantly reduced. During the body bias clustering, the proposed method explicitly estimates and minimizes the average leakage after the post-silicon tuning. Experimental results demonstrate that the proposed method reduces the average leakage by 25.3 to 51.9% compared to non clustering case. We reveal that two bias voltages are sufficient when only a small number of compensation levels are allowed for test-cost reduction. We also give an implication on how to synthesize a circuit to which post-silicon tuning will be applied.

I. Introduction

With the advance of semiconductor technology, chip manufacturing variability has become a crucial problem, and design for variability has been studied. However, robustness improvement only in design time inherently has its limit. To overcome the problem, post-silicon tuning is now drawing a lot of attention, and has been studied.

Body biasing and supply voltage scaling can make circuits faster, yet they increase leakage and dynamic power consumption. Traditionally, chip-level and block-level tuning have been studied [1][2][3] and adopted in some commercial chips. On the other hand, to improve timing while keeping power dissipation low, fine-grained tuning only for portions with timing violation is effective, since most of other portions in fabricated chips and blocks do not violate timing specification. However, fine-grained tuning involves implementation overhead, because of well separation, distribution of multiple body and supply voltages and level shifters, and all of them need extra area and/or power dissipation. The finest granularity is logic gate, but the overhead is impractically huge and is not totally acceptable.

Another problem of fine-grained tuning is the test cost required to obtain an optimal tuning result after fabrication. If there are M tuning variables and each variable can take N values, the number of combinations is N^M, and finding an optimal combination for every chip after fabrication becomes less practical, as M and N increases.

To minimize power dissipation after post-silicon tuning with acceptable implementation overhead, gate clustering has been proposed [4]. The aim of this approach is to attain a tuning quality that is close to gate-level tuning while reducing the implementation overhead and the tuning variable M to $M'(<< M)$. Although the number of combinations is reduced to $N^{M'}$, the test cost is still expensive for most products, which prevents post-silicon tuning.

To further reduce the test cost, our preliminary work [5] proposed to preset the testing order of combinations in design time. We limited $N = 2$, and determined an ordered cluster set that allows only $M+1$ combinations for test and guarantees monotonic decrease/increase in delay/power irrelevant to manufacturing variability. In terms of the reduction in test cost, this approach is promising. However, there are critical remaining issues that prevent a practical use. The first problem is that only random variability is considered in cluster generation in [5], whereas manufacturing variability in actual chips consists of several components, such as die-to-die, within-die spatial and random components. It is predicted that random variability due to RDF (random dopant fluctuation) and LER (line edge roughness) becomes significant according to the technology advance. However, the die-to-die variability is still a dominant factor that induces delay defects, and hence the clustering result ignoring die-to-die variability is not optimal. In addition, a method in [5] cannot handle three or more body bias voltages. Therefore, it is not validated that limiting the number of body voltages to two is appropriate. Given the number of body voltages, we have to choose a set of optimal body voltages from producible voltages. However, this selection is not possible with [5] as well. Furthermore, [5] focuses on subthreshold circuits, and the effectiveness to ordinary superthreshold circuits has not been clarified.

This paper presents a method to obtain a cluster assignment for test-friendly post-silicon tuning. The proposed method takes into account correlated (die-to-die and within-die) variability components in timing and leakage estimation in addition to random variability. To efficiently carry out the clustering, we derive two canonical forms having identical random variables; one expresses the circuit timing and the other represents the circuit leakage, and use them for conditional probability computation involved in estimation of post-tuning leakage current. We also extend the compensation method so that it can handle three or more bias voltages keeping delay/leakage monotonicity. Using the feature that multiple bias voltages are exploitable and a set of body voltages is selectable according to the given circuit and delay constraint, we experimentally reveal that two voltage levels are reasonably sufficient for the small number of compensation levels which is equal to or less than the number of clusters. We further discuss how to synthesize a circuit to which will be applied post-silicon tuning, and give a design implication.

978-1-4673-0772-7/12/$31.00 ©2012 IEEE

The rest of this paper is organized as follows. In Section II, we introduce the compensation approach proposed in [5], and present a novel approach for handling multiple body bias voltages. Section III gives the problem formulation of this work. Section IV explains how to derive canonical forms that express statistical models of delay and leakage. We explain the proposed optimization flow and each step in Section V. We show experimental results in Section VI, and conclude the discussion in Section VII.

II. Compensation approaches for low test-cost tuning

A. Basic approach for low test-cost tuning

Post-silicon tuning based on the conventional body bias clustering [4] suffers from its large test-cost for tuning. The tuning is carried out by sweeping body voltages for each cluster and searching for a combination that attains the minimum leakage current while satisfies given timing constraints. This exhaustive tuning involves large test-cost because there are a lot of combinations to be tested and it requires leakage current measurement. Let the number of available body bias voltages be N_{bias} and the number of clusters be $N_{cluster}$. There are $N_{bias}^{N_{cluster}}$ combinations, and finding an optimal combination for each fabricated chip becomes less practical, as N_{bias} and $N_{cluster}$ increase.

To reduce test-cost, we adopt a small test-cost approach proposed in [5]. Figure 1 illustrates an example of the tuning flow in the case that the number of clusters is three. Each cluster is pre-ordered in design time simultaneously with clustering itself, and the subscript i of the cluster name C_i denotes the order. We call this tuning approach "pre-ordered clustering". Here, [5] supposes that only two bias voltages V_{low} and V_{high}, are available. V_{high} makes the circuit faster than V_{low}, while it causes larger leakage.

First V_{low} is applied to all clusters (tuning level 0, hereafter described as level 0 in short). If the chip does not satisfy timing constraints at level 0, then cluster C_1 is sped up by applying V_{high} to C_1 (level 1). If the chip still does not satisfy timing constraints at level 1, V_{high} is given to C_2 in addition to C_1 (level 2). In the same way, the body voltage of the next cluster is changed to V_{high} in order until the circuit satisfies timing constraints. If the circuit does not satisfy timing even at the maximum level (level 3 in the example), the chip should be discarded. In this approach, the number of levels is $N_{cluster} + 1$.

An important property that makes the tuning with pre-ordered clustering realizable is that the circuit speed becomes faster and leakages become larger monotonically as the tuning level increments, and this property is necessarily satisfied for every chip, because changing body voltage from V_{low} to V_{high} always improves the speed and increases leakage. Thanks to the monotonic property, testing can be simplified because no leakage measurement is needed and testing can be finished once the chip satisfies timing constraints. This means the average number of levels to be tested is smaller than $N_{cluster} + 1$. For example in Fig. 1, the average number of test per chip is 7/3 and smaller than the number of available levels 4.

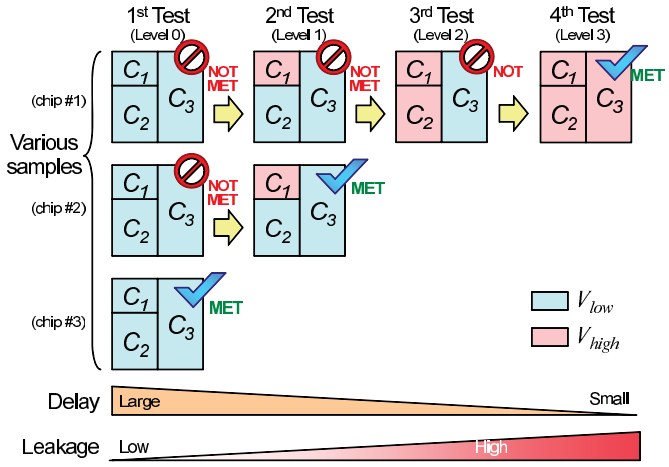

Fig. 1. Basic pre-ordered compensation [5].

B. Post-silicon tuning with more than two bias voltages

The tuning with pre-ordered clustering [5] assumes two bias voltages, and then this tuning approach is not applicable when three or more bias voltages are available. We here discuss how to extend the tuning with pre-ordered clustering while keeping the advantage of low test-cost.

In the extension, the monotonic property in speed and leakage with respect to the compensation level must be satisfied to remain the test cost low. We adopt the following two extensions of "voltage-first alteration" and "cluster-first alteration" as ones of those that satisfy the monotonic property.

The order of clusters is pre-determined in design time similarly to [5]. Figure 2 illustrates an example of voltage-first alteration. In this example, there are three clusters($C_1 - C_3$) and three bias voltages (V_{low}, V_{mid} and V_{high}). First we apply V_{low} to all clusters (level 0). Then, we alter bias voltage of C_1 to V_{mid} (level 1) and to V_{high} (level 2) in sequence. When the timing constraint is not satisfied even though the highest bias voltage is given to C_1, we next alter the bias voltage of C_2. We continue this alternation until the timing constraint is satisfied. With this extension, the number of levels becomes $(N_{cluster} \times (N_{bias} - 1) + 1)$, and in this example it is $3 \times (3 - 1) + 1 = 7$.

Figure 3 exemplifies cluster-first alteration. First we apply V_{low} to all clusters (level 0). Then, we apply V_{mid} to C_1 (level 1) and to C_2 (level 2) in sequence. When the timing constraint is not satisfied even though all clusters are applied the V_{mid}, we next alter the bias voltage of C_1. We continue this alteration until the timing constraints are satisfied. With this extension, the number of levels is the same as voltage-first alteration.

Even with the pre-ordered clustering, the number of compensation levels tends to be large as the number of bias voltages increases, which results in the increase in test-cost for tuning. To maintain the test-cost for tuning within the allowable range, we choose N_{level} compensation levels from $(N_{cluster} \times (N_{bias} - 1) + 1)$ levels. In the proposed method, we determine the order of clusters and choose the compensation level simultaneously during the body bias clustering. We call this clustering "extended pre-ordered clustering".

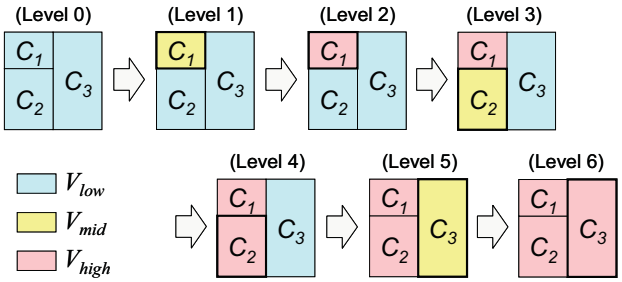

Fig. 2. Voltage-first alteration,

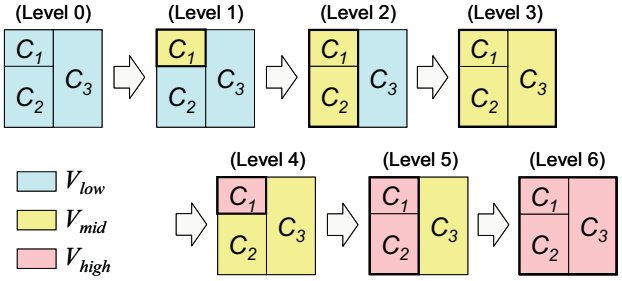

Fig. 3. Cluster-first alteration,

III. Problem Formulation

We formulate the problem to solve in this work. Given the number of compensation levels N_{level}, the number of clusters $N_{cluster}$, the target yield Y_{target}, the number of distributable body voltages N_{dist_bias}, the number of producible body voltages $N_{prod\_bias}(\geq N_{dist\_bias})$ and their voltage values, the proposed method provides a clustering result that minimizes the average leakage after post-silicon tuning with the definition of N_{level} compensation levels. In this problem, we simply divide the given chip layout into N_{island} islands spatially and cluster the islands for layout consideration, where N_{island} is assumed to be given (Fig. 4). A solution is that each island belongs to one and only one of the clusters, and the number of islands included in each cluster is not zero. In addition, a body voltage from the given body voltages is assigned to each cluster for each compensation level such that the circuit delay decreases and leakage increases monotonically as the compensation level increments in every chip. The total number of body voltage used for defining N_{level} compensation levels is N_{dist_bias}.

The objective function of clustering $Cost$ is expressed as the product of average leakage after post-silicon tuning $P_{leakage}$ and a penalty function $penalty$.

$$Cost = P_{leakage} \cdot penalty, \tag{1}$$

where:

$$P_{leakage} = \sum_{i=1}^{N_{level}} \left\{ \int_{cond.} L^{(i)}(\mathbf{X}) \cdot pdf(\mathbf{X}) d\mathbf{X} \right\}, \tag{2}$$

$$cond. = \begin{cases} d^{(i)}(\mathbf{X}) \leq d_{const} < d^{(i-1)}(\mathbf{X}) & (i > 0) \\ d^{(i)}(\mathbf{X}) \leq d_{const} & (i = 0), \end{cases} \tag{3}$$

where $\mathbf{X}$ is a vector of random variables, which corresponds to, for example, channel length, threshold voltage and so on. $L^{(i)}(\mathbf{X})$ and $d^{(i)}(\mathbf{X})$ are leakage current and delay at compensation level i, which means the cluster order in post-silicon tuning is considered in $P_{leakage}$ computation. $pdf(\mathbf{X})$ is the probability density function of $\mathbf{X}$. d_{const} is a given timing constraint that must be satisfied after post-silicon tuning. An important point here is that the integration of $L^{(i)}(\mathbf{X}) \cdot pdf(\mathbf{X})$ must be carried out in the subspace of $\mathbf{X}$ satisfying $d^{(i)}(\mathbf{X}) \leq d_{const} < d^{(i-1)}(\mathbf{X})$, where this subspace corresponds to the post-silicon tuning as explained in Fig. 1. $penalty$ is a penalty function introduced to satisfy the yield constraint Y_{target}, and it is a function of Y_{target} and $Y_{cluster}$. Without this constraint, the timing yield would be sacrificed for leakage reduction.

$$Y_{cluster} = \mathrm{Prob}[d^{(N_{level})}(\mathbf{X}) < d_{const})], \tag{4}$$

where
$\mathrm{Prob}[condition]$ represents the probability that $condition$ is satisfied. There are many function candidates for $penalty$, and we chose an exponential function that $penalty$ increases exponentially as $Y_{cluster}$ decreases when $Y_{cluster} - Y_{target}$ is smaller than or close to zero and $penalty = 1$ in the other range.

An efficient computation of Eq. (1) will be explained in Sections IV and V.A. In [5], only random variability is assumed and then the integration in the subspace is approximated as the integration in the entire space. However, in actual chips, die-to-die variability has a strong impact on delay defects, and such an approximation degrades the quality of the clustering result.

IV. Statistical Modeling of Delay and Leakage Current

Under process variation, delays, arrival times and leakage currents should be expressed statistically. This section discusses the statistical computation of delay and leakage current for computing the objective function of Eq. (2).

A. Delay computation

To efficiently express and compute delays and arrival times, a canonical form that expresses statistical delay and arrival time is widely used [6].

$$d = d_0 + \sum_{p=1}^{N_{var}} c_p X_p + c_{rnd} X_{rnd}, \tag{5}$$

where $X_p \sim N(0,1)$ represents inter-die and intra-die spatially-correlated variations, $X_{rnd} \sim N(0,1)$ represents random variation. c_p and c_{rnd} are delay coefficients which are determined by the sensitivity of delay to magnitude of variation. N_{var} is the number of variational parameters and d_0 is the average of d. Note that by applying PCA (principal component analysis) beforehand to the correlated random variables of inter-die variation and intra-die spatially-correlated variations, $X_p(p = 1, \cdots, N_{var})$ and X_{rnd} become all independent and have no correlation to each other [6].

To calculate circuit delay, sum and max operations are needed. The sum of two canonical forms can be expressed as a canonical form. Although the max operation is non-linear, [6] presents a practical computation to approximately express the max of two canonical forms as a canonical form. With these

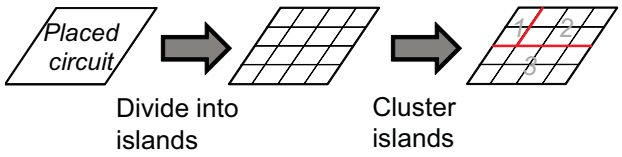

Fig. 4. Island generation and clustering (N_{island}=16, $N_{cluster}$=3).

sum and max operations, circuit delay d_{cir} can be expressed with the canonical form of Eq. (6).

$$d_{cir} = d_{cir,0} + \sum_{p=1}^{N_{var}} c_{cir,p} X_p + c_{cir,rnd} X_{rnd}. \quad (6)$$

B. Leakage computation

We next explain the leakage computation. In order to obtain the entire circuit leakage L_{cir}, we must compute

$$L_{cir} = \sum_{j=1}^{N_{cell}} L_j, \quad (7)$$

where N_{cell} is the number of cells. The leakage follows not a normal distribution but a lognormal distribution, and hence the summation procedure used in delay are not applicable to leakage computation. We therefore use a lognormal summation method proposed in [7] to obtain the canonical form of leakage.

First, we express leakage as follows.

$$L = \exp\left(k_0 + \sum_{p=1}^{N_{var}} k_p X_p + k_{rnd} X_{rnd}\right), \quad (8)$$

where $X_p \sim N(0,1)$ is a random variable that corresponds to a variation source, and it is identical to that in Eq. (5). $X_{rnd} \sim N(0,1)$ represents random variation. k_p and k_{rnd} are coefficients.

Then, mean and variance of leakage can be expressed as follows.

$$E(L) = \exp\left(k_0 + \frac{1}{2}\sum_{p=1}^{Nvar} k_p^2\right), \quad (9)$$

$$\begin{aligned}\mathrm{Var}(L) &= E(L^2) - \{E(L)\}^2 \\ &= \exp\left(2k_0 + 2\sum_{p=1}^{Nvar} k_p^2\right) - \exp\left(2k_0 + \sum_{p=1}^{Nvar} k_p^2\right).\end{aligned} \quad (10)$$

Here, covariance between each variable and leakage can be calculated by

$$E(L \cdot e_{X_i}) = \exp\left(k_0 + \frac{1}{2}\left(\sum_{p=1,p\neq i}^{N_{var}} k_p^2 + (k_i+1)^2\right)\right). \quad (11)$$

Covariance between leakages can be written as

$$\begin{aligned}E(L_1 \cdot L_2) &= \exp\left((k_{0,1}+k_{0,2}) + \frac{1}{2}\sum_{p=1}^{N_{var}}(k_{p,1}+k_{p,2})^2\right. \\ &\quad \left. + \frac{1}{2}k_{rnd,1}^2 + \frac{1}{2}k_{rnd,2}^2\right). \end{aligned} \quad (12)$$

Here, we can express summation of two leakages as follows.

$$\begin{aligned}L_{sum} &= L_1 + L_2 \\ &= \exp\left(k_{0,sum} + \sum_{p=1}^{N_{var}} k_{p,sum} X_p + k_{rnd,sum} X'_{rnd}\right),\end{aligned} \quad (13)$$

where $k_{0,sum}$, $k_{p,sum}$ and $k_{rnd,sum}$ are coefficients of the sum of leakage, which can be computed by formulas presented in [7].

Using the sum operation described above, the entire circuit leakage can be expressed with the canonical form of Eq. (14).

$$L_{cir} = k_{cir,0} + \sum_{p=1}^{N_{var}} k_{cir,p} X_p + k_{cir,rnd} X_{rnd}. \quad (14)$$

Now, we obtained the canonical form of the entire circuit leakage whose variables X_p are identical to the canonical form of the circuit delay. This means that the correlation between circuit delay and leakage are embedded in the coefficients of c_{cir} and k_{cir}. Exploiting this embedded correlation information, we compute Eq. (2), which will be explained in the next section.

V. Design-Time Optimization Procedure

This section presents the proposed procedure for body bias clustering. We first explain an efficient computation of the objective function in Section V.A. We next describe the optimization flow using simulated annealing in Section V.B, and explain the generation of neighboring solution in Section V.C.

A. Computation of objective function

Given the canonical forms of delay and leakage of Eqs. (6) and (14), we now compute Eq. (2). The difficulty of computing Eq. (2) is the integration of $L^{(i)}(\mathbf{X}) \cdot pdf(\mathbf{X})$ in the subspace of $\mathbf{X}$ satisfying $d^{(i)}(\mathbf{X}) \leq d_{const} < d^{(i-1)}$. We in this paper adopt Monte Carlo approach to carry out the integration among existing numerical computation methods, since the accuracy and computation time are easily tradable.

The computation process is as follows.

Step 1: Prepare the canonical forms of circuit delay and leakage in Eqs. (6) and (14) for each compensation level i.

Step 2: Generate $N_{var}+1$ random numbers for X_p and X_{rnd}, where X_p and X_{rnd} are independent normal random variables. This step can be regarded as the fabrication of a chip.

Step 3: Calculate circuit delay $d^{(i)}$ using Eq.(6) and find the level i such that the maximum delay constraint is satisfied at level i and is not at level $(i-1)$.

$$d^{(i)} \leq d_{const} < d^{(i-1)}. \quad (15)$$

This step can be interpreted as the post-silicon tuning in Fig. 1 for the chip fabricated in Step 2. From the viewpoint of mathematical computation, this inequality corresponds to the subspace for the integration in Eq. (1).

Step 4: Calculate leakage current using Eq. (14) at the level of i found in Step 3, and accumulate it as the cost value.

Step 5: Go back to Step 2 and repeat Steps 2-4 N_{MC} times. Here, N_{MC} is the number of required trials. If the number of trials reach N_{MC}, the accumulated cost value is divided by N_{MC}, and then we obtain the average leakage current after post-silicon tuning expressed in Eq. (1).

It should be noted that the proposed computation carries out Monte Carlo simulation on the canonical forms and does not require iterative timing analysis on the timing graph, which means Monte Carlo simulation is not as expensive as we thought.

B. Optimization by simulated annealing

The cluster generation problem here we are solving is a combinatorial optimization problem. As one of the optimization algorithms, we adopt simulated annealing method in this work.

The initial solution is generated randomly. Inside the optimization loops in simulated annealing, we generate a neighboring solution from the current solution. The generation of neighboring solution is explained in Section V.C. We then derive the canonical expressions of circuit delay and leakage based on the generated neighboring solution, and calculate the objective function of Eq. (1), $Cost^{new}$. If the cost of the neighboring solution $Cost^{new}$ is less than that of the current solution $Cost^{cur}$, we replace the current solution with the neighboring solution. Moreover, if $Cost^{new}$ is less than the cost of the best solution $Cost^{best}$, we record the neighboring solution as the best solution. On the other hand, in the case of $Cost^{new} > Cost^{cur}$, we also update current solution based on the probability $\exp(-\Delta/T)$, where $\Delta = Cost^{new} - Cost^{cur}$ and T is the temperature parameter.

Following the manner of simulated annealing, we gradually decrease T in the optimization loops. When T is high, the accurate computation of the objective function is not necessary since $exp(-\Delta/T)$ is mostly dependent on T, whereas the higher accuracy is required when T is low. This tendency can be satisfied by changing N_{MC} depending on T, which helps to reduce the optimization run time. If an exiting condition defined by, for example, execution time, the minimum temperature and/or the number of loops is satisfied, the optimization finishes.

C. Generation of neighboring solution

In the optimization problem defined in Section III, we have to determine a composition of $N_{cluster}$ clusters, select N_{level} levels from $(N_{cluster} \times (N_{bias} - 1) + 1)$ levels, and choose N_{dist_bias} voltages from N_{prod_bias} voltages. This means that we need to explore a solution space consisting of three subspaces. We thus have to generate a neighboring solution in simulated annealing so as to efficiently explore the subspaces.

When generating a neighboring solution, we first pick up one subspace randomly according to the probabilities proportional to the size of each subspace, and slightly modify the subsolution in the selected subspace while preserving the neighborhood. The subspace selection probabilities for island clustering, level selection and bias selection are $Pr_{cluster}$, Pr_{level} and Pr_{bias}, respectively, and they satisfy $Pr_{cluster} + Pr_{level} + Pr_{bias} = 1$. We estimate the size of each subspace using the number of possible combinations. The number of combinations for clusters is obtained using the recurrence relation below.

$$C_{cluster}(N_{cluster}) = N_{cluster}^{N_{island}} - \sum_{k=1}^{N_{cluster}-1} \binom{N_{cluster}}{k \cdot C_{cluster}(k)}. \quad (16)$$

The numbers of combinations for levels and bias are computed by

$$C_{level} = \binom{N_{possible\_level}}{N_{level}}, \quad (17)$$

$$C_{bias} = \binom{N_{prod\_bias}}{N_{dist\_bias}}, \quad (18)$$

where $N_{possible\_level}$ is $(N_{cluster} \times (N_{bias} - 1) + 1)$. Using $C_{cluster}$, C_{level} and C_{bias}, $Pr_{cluster}$, Pr_{level} and Pr_{bias} are expressed as

$$Pr_{cluster} = C_{cluster}/(C_{cluster} + C_{level} + C_{bias}), \quad (19)$$

$$Pr_{level} = C_{level}/(C_{cluster} + C_{level} + C_{bias}), \quad (20)$$

$$Pr_{bias} = C_{bias}/(C_{cluster} + C_{level} + C_{bias}). \quad (21)$$

VI. Experimental results

This section experimentally evaluates the proposed clustering method. The circuits used for experiments, which are listed in Table I, were synthesized by a commercial logic synthesizer and placed by a commercial placer with an industrial 65nm library. c1908 is a circuit in ISCAS85 benchmark set, mult16, mult32, and mult64 are 16-bit, 32-bit and 64-bit multipliers, and dsp_alu is an ALU in a DSP for mobile phone. We characterized the nominal delay and leakage, and the sensitivities of delay and leakage to variabilities with HSPICE. The proposed method was implemented in C++ language and was run on a 3.2 GHz Opteron processor. $\sigma_{global} = 25$ [mV] and $\sigma_{random} = 15$ [mV] are assumed in experiments. All circuits are spatially divided to 4×4, thus $N_{island} = 16$. The target yield Y_{target} is 98%. We provided 7(=N_{prod_bias}) voltages for body voltages; RBB(Reversed Body Bias) 300mV, RBB200mV, RBB100mV, ZBB (Zero Body Bias), FBB(Forward Body Bias) 100mV, FBB200mV and FBB300mV. RBBs make circuits slower and reduce leakage current, and FBBs make circuits faster and increase leakage current. In the case that three or more bias voltages are available, better result in VFA and CFA was selected.

A. Performance improvement with proposed clustering

We first validate the effectiveness of circuit performance improvement thanks to the proposed clustering. We applied the proposed clustering to mult64 by changing the delay constraint d_{const}. Here, mult64 was synthesized with the achievable minimum delay constraint, and the number of bias voltages N_{bias} was set to two. Figure 5 shows delays and leakages of mult64. Curves labeled with "w/ clustering" and "w/o clustering" represent compensation results with the proposed clustering ($N_{cluster}$=4, N_{level}=5) and no clustering (equivalent to

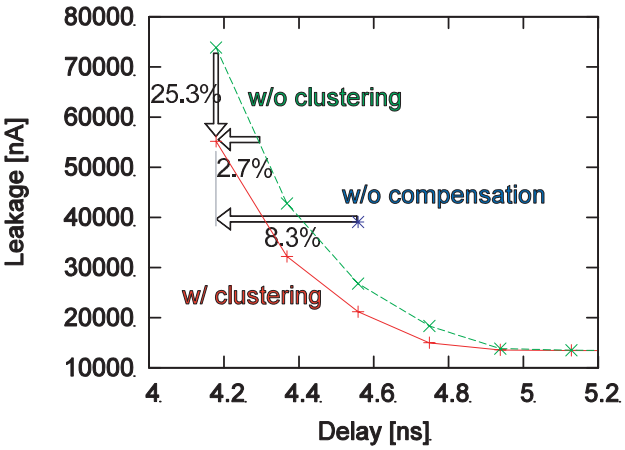

Fig. 5. Performance improvement with proposed clustering (mult64).

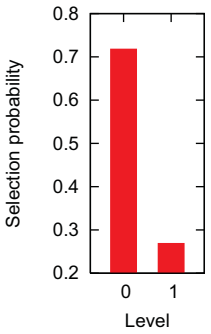

Fig. 6. Distribution of level selection probability w/o clustering.

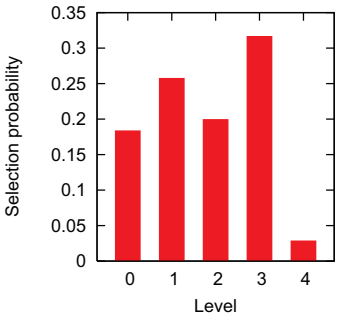

Fig. 7. Distribution of level selection probability w/ clustering.

TABLE I
PERFORMANCE IMPROVEMENT WITH PROPOSED CLUSTERING.

Circuit	#cells	Delay reduction w/ same leakage	Leakage reduction w/ same delay
c1908	919	3.7%	28.8%
mult16	4,145	3.7%	35.3%
mult32	15,291	3.4%	32.4%
dsp_alu	17,302	6.3%	51.9%
mult64	73,473	2.7%	25.3%

$N_{cluster}$=1), and a dot with "w/o compensation" corresponds to a result without compensation. With FBB 300mV, the circuit delay can be reduced by 8.3% in this configuration.

With performance compensation with the proposed clustering, the leakage current is reduced by 25.3% attaining the same circuit delay (4.17ns) compared to that without clustering. Looking at the same leakage, the circuit delay is decreased by 2.7%. Table I summarizes the delay and leakage reduction in various circuits. We can obtain 25.3 to 51.9% leakage reduction and 2.7 to 6.3% delay reduction.

Table II lists the selected bias voltages in the case that 4.17ns delay constraint is given to mult64. Thanks to clustering with $N_{cluster}$=4, ZBB can be used for non timing-critical gates, which contributes to leakage reduction. Figures 6 and 7 show the probability distribution of each level being selected. When clustering is applied, levels 0 to 4 are selected almost uniformly, whereas level 0 is mostly selected without clustering.

B. Importance of correlation between delay and leakage

The proposed method considers the correlation between delay and leakage, which originates from die-to-die and within-die spatial variations, by expressing delay and leakage with the same random variables, though [5] ignored the correlation. We here evaluate how much leakage reduction can be obtained by the explicit consideration of the correlation in the optimization.

Table III lists the leakage currents after post-silicon tuning

TABLE II
SELECTED BODY VOLTAGES (MULT64, 4.17NS CONSTRAINT).

$N_{cluster}$	N_{level}	Selected bias voltages
1 (w/o clustering)	2	FBB 100mV, FBB 200mV
4 (w/ clustering)	4	ZBB, FBB 200mV

TABLE III
LEAKAGE CURRENTS OF CIRCUITS OPTIMIZED W/ AND W/O CORRELATION CONSIDERATION.

Circuit	Leakage current [μA]		Leakage reduction [%]
	w/o correlation consideration	w/ correlation consideration	
c1908	0.376	0.369	1.9
mult16	1.41	1.38	2.2
mult32	5.43	4.84	10.9
dsp_alu	5.29	5.24	1.0
mult64	27.0	25.1	6.8

when the circuit was optimized with and without considering the correlation. The number of clusters $N_{cluster}$ is 4 and the number of bias voltages N_{bias} is 2 in this experiment. With the correlation consideration in the design-time optimization reduces the leakage current up to 10.9%.

C. Number of bias voltages

We next show how the number of bias voltages affects leakage current. We changed the number of levels N_{level} from 3 to 9 and evaluated the leakage in three cases of N_{dist_bias}=2, 3 and 4. Figure 8 shows results of mult16 with a tight timing constraint. In the figure, x-axis means the number of levels used in compensation N_{level}, and y-axis represents leakage current after compensation. In the case of N_{dist_bias}=2, the maximum number of levels is 5, because $N_{cluster} = 4$ in this experiment, and hence no points are plotted in the range of $N_{level} > 5$.

We can see that as N_{level} increases, the leakage becomes smaller for all three cases. However, larger N_{level} needs more test cost after fabrication. Please recall that even for chips without post-silicon tuning, testing is expensive. We then focus on small N_{level} which requires small test cost. When $N_{level} \leq N_{cluster} + 1$, there are small difference of effectiveness in leakage reduction between there cases. This result indicates that two bias voltages are sufficient for leakage reduction in case of $N_{level} \leq N_{cluster} + 1$ and large resource for body voltage distribution is not necessary.

We also carried out the similar experiment with a looser timing constraint, and the same conclusion was obtained. Besides, it was observed that the superiority of cluster-first alternation and voltage-first alternation was dependent on the tightness of the timing constraint. With a looser timing constraint, voltage-first alternation is better because it assign a high body voltage

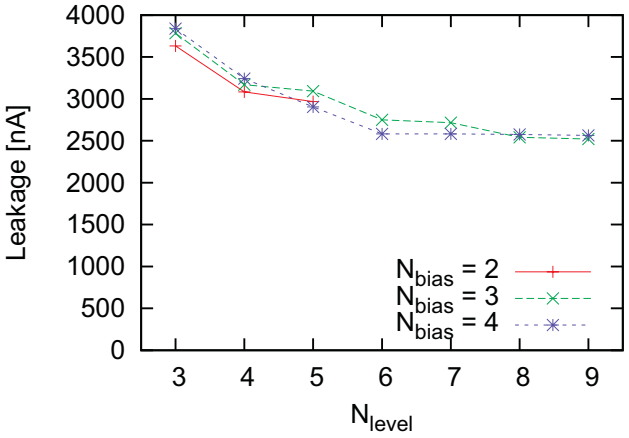

Fig. 8. N_{levels} versus leakage (mult16).

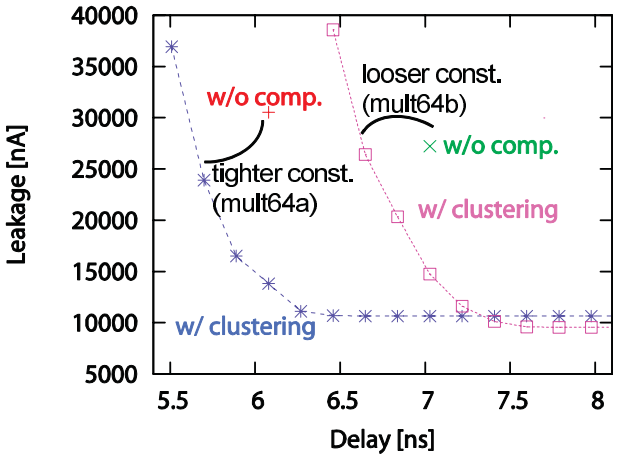

Fig. 9. Dependency of leakage on delay constraints given to logic synthesis and body bias clustering (mult64).

to only a few timing-critical clusters. On the other hand, given a tight timing constraint, cluster-first alternation was superior because most of clusters are timing critical and low bias voltage should be eliminated at lower levels. Note that in both the alternations, although a few levels could be less useful, these levels are often eliminated in the selection process of N_{level} levels.

D. Implication for circuit synthesis

We finally discuss how to synthesize a circuit to which the post-silicon tuning will be applied. Depending on the timing constraint given to logic synthesis, the obtainable leakage after post-silicon tuning is different, and we have to give an appropriate constraint to logic synthesis.

Figure 9 shows delays and leakages of two mult64 implementations. One was synthesized with a delay constraint that is 120% of the tightest delay (mult64a), and another was synthesized with 150% constraint (mult64b). The number of body voltages is two. We can see that the leakage starts to increase when the delay constraint becomes smaller than the delay of no compensation case. For example, let us suppose that the delay constraint given to clustering is 7.0ns. In this case, mult64a attains smaller leakage current, because the leakage of mult64b increased at this point. Therefore, as long as the given delay constraint can be satisfied even without post-silicon tuning, the post-silicon tuning should not be used for delay reduction and should be used for leakage current reduction. In this case, a small number of clusters are enough. On the other hand, when the delay constraint cannot be satisfied without post-silicon tuning, the body bias clustering is effective for performance improvement as shown in Section VI.A.

An example of CPU time on an Intel 3.0GHz processor is roughly 1400 seconds in this experiment for multi64b including 71k cells. We observed that CPU time is proportional to circuit scale, which is mostly determined by the complexity of SSTA. Monte Carlo simulation on the derived canonical forms was not significant in the CPU time.

VII. Conclusion

We proposed a body bias clustering method that enables low test cost tuning. The proposed method explicitly computes the average leakage current after post-silicon tuning taking into account the correlation between delay and leakage. We experimentally demonstrated that the proposed clustering method reduced the achievable minimum delay by 2.7 to 6.3% as well as leakage current by 25.3 to 51.9%. Experiments also indicate that two bias voltages are sufficient for low test-cost post-silicon tuning. Except a case that post-silicon tuning is used for reducing the achievable circuit delay, body bias clustering should be used for leakage reduction, and a circuit must be synthesized with an appropriate delay constraint for exploiting this implication.

Acknowledgement

This work is supported by NEDO.

References

[1] M. Takahashi *et al.*, "A 60-mW MPEG4 video codec using clustered voltage scaling with variable supply-voltage scheme," *IEEE JSSC*, vol. 33, no. 11, pp. 1772 –1780, 1998.

[2] J. Tschanz *et al.*, "Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage," *IEEE JSSC*, vol. 37, no. 11, pp. 1396 – 1402, 2002.

[3] Y. Nakamura *et al.*, "1/5 power reduction by global optimization based on fine-grained body biasing," in *Proc. CICC*, pp. 547 – 550, 2008.

[4] S. Kulkarni *et al.*, "Design-time optimization of post-silicon tuned circuits using adaptive body bias," *IEEE Trans. CAD*, vol. 27, no. 3, pp. 481 –494, 2008.

[5] K. Hamamoto, M. Hashimoto and T. Onoye, "Tuning-friendly body bias clustering for compensating random variability in subthreshold circuits," in *Proc. ISLPED*, pp. 51–56, 2009.

[6] H. Chang *et al.*, "Statistical timing analysis under spatial correlations," *IEEE Trans. CAD*, vol. 24, no. 9, pp. 1467 – 1482, 2005.

[7] A. Srivastava *et al.*, "A novel approach to perform gate-level yield analysis and optimization considering correlated variations in power and performance," *IEEE Trans. CAD*, vol. 27, no. 2, pp. 272 –285, 2008.