Improving the Efficiency of Monte Carlo Power Estimation*

Chih-Shun Ding
Conexant Semiconductor Systems
Newport Beach, CA 92660

Cheng-Ta Hsieh Massoud Pedram
Department of Electrical Engineering - Systems
University of Southern California
Los Angeles, CA 90089

Abstract

In this paper, we propose two efficient statistical sampling techniques for estimating the total power consumption of large hierarchical circuits. We first show that, due to the characteristic of sampling efficiency in Monte Carlo simulation, granularity of samples is an important issue in achieving high overall efficiency. The proposed techniques perform sampling both temporally (across different clock cycles) and spatially (across different modules) so that smaller sample granularity can be achieved while maintaining the normality of samples. The first proposed technique, which is referred to as the module-based approach, samples each module independently when forming a power sample. The second technique, which is referred to as the cluster-based approach, lumps the modules of a system into a number of clusters on which sampling is then performed. Both techniques adapt stratification to further improve the efficiency. Experimental results show that these techniques provide a reduction of 2-3x in simulation run time compared to existing Monte-Carlo simulation techniques.

1 Introduction

Power dissipation has become one of the important design metrics in VLSI designs, along with area and speed. As a result, the issues of estimating the power consumption of a circuit in various design phases have been examined by researchers.

In early design phases of a circuit, the objective of power estimation [1, 2, 3] is to provide a relative comparison metric so that designers can efficiently explore various design alternatives.

*This research was supported in part by DARPA under contract no. F336125-95-C1627, SRC under contract no. 98-DJ-606 and a grant from Conexant Semiconductor Systems.

Absolute accuracies are compromised during these phases as information such as routing capacitances and gate/transistor level implementations is not finalized yet. As the design progresses, more design decisions are made and information required for highly accurate power estimation becomes available. Although low level optimization such as gate/transistor sizing can still be done in final phases of design, the objective of power estimation shifts more toward power verification, rather than design exploration. Examples of this type of techniques are [4, 5, 6, 7, 8, 9, 10, 11, 12, 13]

The purpose of this paper is to propose highly accurate yet efficient sampling techniques used in Monte Carlo simulation loops [7] in the final phases of the design process. We assume that a hierarchical netlist of the circuit with each module described at the register-transfer (RT)-level or below is available. For each module, a power simulation technique at the same abstraction level as the netlist is adopted. Surveys of existing techniques at RT-level and below are given in [14, 1, 15].

Our estimation target is the average total power consumption of a hierarchical circuit under a user-specified application input stream. In addition, the estimation itself is performed in a single Monte-Carlo simulation so that the power estimate at the top level circuit satisfies the user-specified confidence and error levels. This is different from the strategy which estimates each module in a separate Monte-Carlo simulation and sums up the estimates to produce the total average power. The drawback of the latter strategy is that it is difficult to determine *a prior* what confidence and error levels should be assigned to each module so that the sum of individual power estimates satisfies the confidence and error levels specified at the top circuit level.

Existing statistical sampling techniques are classified as either *parametric* [7, 9, 8, 11, 12] in which a basic simulation unit, called a (power) sample is assumed to follow near-normality, or *non-parametric* [10] in which no assumption about the distribution of samples is made. In the former techniques, the mean power consumption of a predetermined yet fixed number of randomly selected clock cycles constitutes a sample in order to ensure near-normality of samples. In non-parametric techniques, the power dissipation values of those units in a sample are ordered, resulting in order statistics of the sample. Different convergence (or stopping) criteria are applied in these techniques.

A sampling technique is differentiated from one another based on how the samples are selected and subsequently, how many samples are required for convergence. Therefore, the term "efficiency" should be referred to as the product of simulation effort spent on each sample (or sample granularity) times the number of required samples, instead of strictly the number of

required samples. After investigating the efficiency and oversampling characteristics of Monte Carlo simulation, we conclude that Monte-Carlo simulation strategy favors smaller sample granularity. In another words, if sample granularity is a choice, smaller sample granularity achieves higher overall efficiency. Consequently, our objectives in design sampling strategies are 1) reducing the variance of samples and 2) using smaller sample granularity while maintaining the near-normality of samples. We achieve the first objective by using stratification and second objective by sampling in both temporal (time) and spatial (module) domains, contrasted with existing sampling techniques which sample on the temporal domain only. In another words, the latter techniques, which are also referred to as clock-based techniques in this paper, treat the entire circuit as a single entity during sampling, and no circuit partitioning information is utilized.

The first proposed technique uses a module-based sampling strategy in which each module is sampled independently. By doing so, the normality of sample is significantly improved so that it allows us to use smaller granularity in a sample. When the number of modules in a circuit is very high, this technique still samples each module for at least one clock cycle. That is, there is a limitation in these techniques on how small the granularity of samples can be. Note that this is an issue only when the number of modules in a circuit is very high. In this case, we propose a second technique that will allow us to use an even smaller granularity in a sample. It uses a cluster-based sampling strategy in which the modules are grouped into a small number of clusters on which stratified random sampling techniques are performed. Experimental results show that these techniques reduce the simulation time by a factor of 4x.

We do not consider non-parametric techniques in this paper as the results reported in [10] indicates that oversampling problem is even worse than parametric techniques.

The rest of the paper is organized as follows. In Section 2, we introduce the notations. In Section 3, we consider issues associated with power estimation on large hierarchical circuits. A module-based and a cluster-based technique are proposed in Section 4 and 5, respectively. Practical issues are examined in Section 6. Experimental results are presented in Section 7 followed by concluding remarks in Section 8.

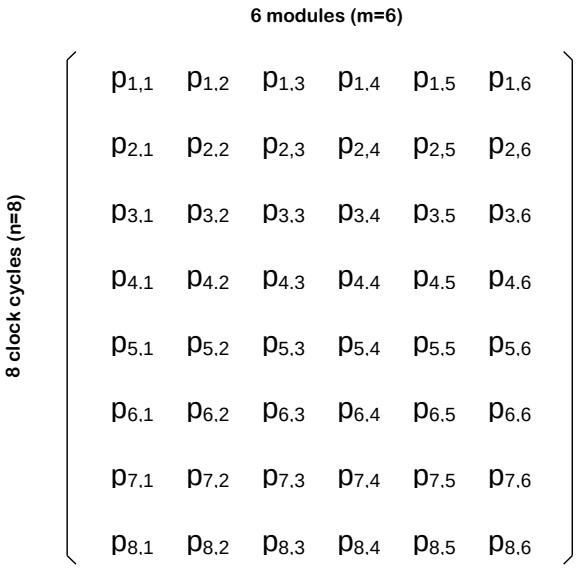

Figure 1: An example of power log matrix.

2 Background

2.1 Power Dissipation as a Random Variable

A *circuit* is defined as a collection of *modules* in which all inputs are either from circuit primary inputs or from the outputs of flip-flops in other modules. This partitioning scheme facilitates us to clearly define the signal arrival times at each module input so that we can accurately simulate each module independently for power. If the partitioning scheme used by the designer is different from the one proposed here, then for each module, the combinational cones between the module inputs and the outputs of flip-flops in other modules are duplicated and included in the module to aide simulation and their contributions to power consumption are ignored.

The netlist for each module is given at RT-level or lower and they can be at different levels. For each module, a power simulation technique at its own abstraction level can be adopted as long as it provides cycle-by-cycle power values. Next, we need to define the population used for sampling.

Depending on how the population is defined during sampling, sampling techniques can be either 1) survey sampling based [9, 13] or 2) signal statistic based [7, 8, 10, 11, 12]. In the former techniques, the actual population is obtained from an application vector stream at

the circuit inputs combined with the traversal of internal circuit states under the applied stream. The estimation target is the power consumption under the applied vector stream. The vector traces on sequential elements in the circuit need to be known in advance. While this is a computational overhead, it can be significantly reduced using functional or cycle-based simulations [13]. On the other hand, in the latter techniques, the population is described using signal statistics, such as signal probabilities and correlations. During sampling, the input vector stream is generated on the fly. While this class of techniques have been demonstrated on both combinational and sequential circuits, one of their shortcomings is that the complex correlations in a realistic input vector stream cannot be effectively recreated during the vector generation process. In particular, in realistic circuits, it requires specific vector sequences for the circuit to enter a particular operation mode and afterwards the signal statistics may change. In addition, it is not clear in these techniques on how the input signal statistics are obtained in the first place.

We adapt the survey sampling approach. We further assume that the vector traces of all sequential elements are given as a result of functional simulation on the circuit.

Let m and n denote the total number of modules in the circuit and clock cycles in the input stream, respectively. Power dissipation of the jth module, M_j, at the ith clock cycle is denoted as $p_{i,j}$ and represented by the $< i, j >$ entry of a *power log matrix*, as shown in Figure 1. Note that this matrix is just for conceptual convenience in defining the population; the actual power values $p_{i,j}$ are not known before estimation. Only the input vectors that will be used to simulate each entry are known from the vector traces of sequential elements.

The random variable representing the power consumption in each clock cycle of module M_j is denoted as P_j for $j = 1, \ldots, m$. $E[X]$ and $V[X]$ denote the mean and variance of random variable X, respectively. The random variable representing the circuit power in each clock cycle is denoted as P_s and can be written as

$$P_s = \sum_{j=1}^{m} P_j \tag{1}$$

One of the objectives in our techniques is to reduce sample granularity. To facilitate efficiency comparison against other sampling techniques, we define the notion of (*simulation*) *workload*. The workload for a sample, is calculated as the product of the number of transistors being simulated times the number of simulated cycles. Therefore, if a circuit consists of modules A and B (each having 10k transistors) is simulated for 30 clock cycles and the average power over the 30 cycles is used to produce one sample value, then the workload for the sample is

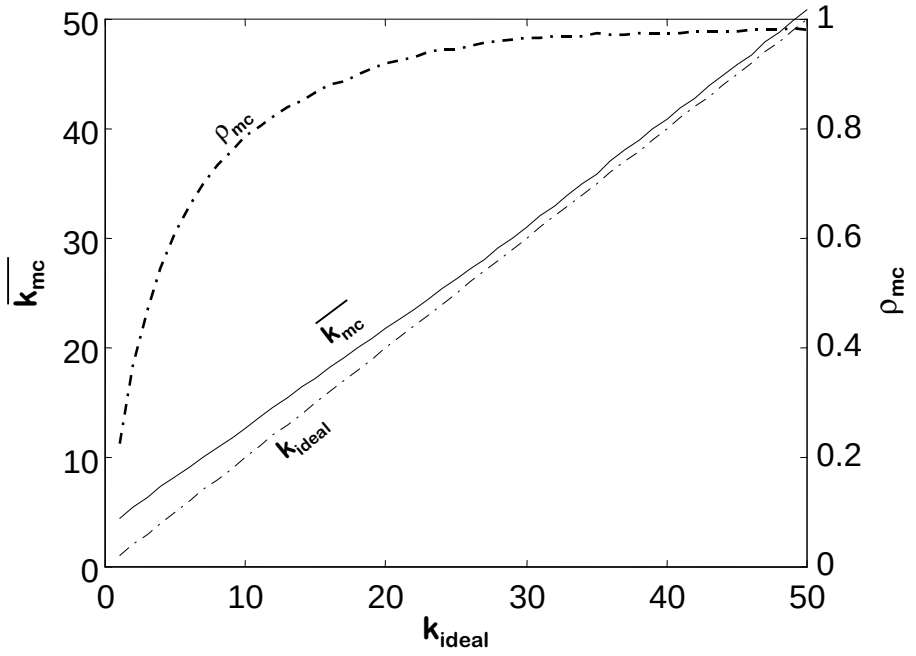

Figure 2: Oversampling in Monte Carlo simulation.

measured as 20k×30=600k transistor-cycles. On the other hand, if module A is simulated for 15 cycles and module B is simulated for 15 cycles and the 30 simulation results are averaged and multiplied by a factor of 2 to produce the sample value, then the workload for the sample is measured as 10k×15+10k×15=300k transistor-cycles.

3 Issues of Monte Carlo simulation on Large Circuits

3.1 Oversampling in Monte Carlo Simulation

Monte Carlo simulation has its own preferred region of operation in terms of *sampling efficiency*. Here, the sampling efficiency is strictly in terms of the number of samples required by Monte Carlo simulations, compared to the number obtained from analytical analysis. To investigate the sampling efficiency of MC simulation, we run a series of MC simulations with the population generated from a random number generator with normal distribution, a user-specified variance, and mean value of 1. Let X be the random variable representing the

random draw from this population. Furthermore we denoted this normal distribution as $N(E(X), V(X))$.

Before performing the MC simulation, since the population mean and variance are known in advance, we can compute the ideal number of required samples, k_{ideal}, to achieve an error level ϵ and a confidence level $(1-\alpha)$, as follows [16]:

$$\begin{aligned} k_{ideal} &= (\frac{z_{\alpha/2}}{\epsilon})^2 (V(X)/E^2(X)) \\ &= (\frac{z_{\alpha/2}}{\epsilon})^2 V_{rel}(X) \end{aligned} \tag{2}$$

where $z_{\alpha/2}$ is defined such that the area to its right under a standard normal distribution is equal to $\alpha/2$ and $V_{rel}(X)$ denotes the relative variance of X.

Next, on the same population (i.e. with the same population variance), we perform a Monte Carlo simulation for the same values of ϵ and $(1-\alpha)$. The simulation terminates when the number of samples, k, satisfies the following convergence criterion:

$$k \geq (\frac{t_{\alpha/2}}{\epsilon})^2 (\frac{S_k^2}{\mu_k^2}) \tag{3}$$

where where $t_{\alpha/2}$ is defined such that the area to its right under the t-distribution with degree $(k-1)$ is equal to $\alpha/2$; μ_k and S_k^2 are computed as follows (The values of these k samples are denoted as $x_1, \ldots, x_k$).

$$\mu_k = \frac{1}{k} \sum_{i=1}^{k} x_i \tag{4}$$

$$S_k^2 = \frac{1}{k-1} \sum_{i=1}^{k} (x_i - \mu_k)^2 \tag{5}$$

Since the value of k changes from one Monte Carlo simulation to next, a fair comparison to the analytical result is to compare the average value of k over a large number of Monte Carlo runs against k_{ideal}. We denote the average number of k of 100,000 Monte Carlo runs as $\overline{k_{mc}}$. Each variance setting gives us a pair of k_{ideal} and $\overline{k_{mc}}$. By varying the variance, we can get a plot of $\overline{k_{mc}}$ vs. k_{ideal}.

In this experiment there is no need to adjust the mean value as the random variable X with normal distribution $N(E(X), V(X))$ can be linearly transformed into another variable Y with normal distribution $N(1, \frac{V(X)}{E^2(X)})$ using $Y = \frac{X}{E(X)}$. The multiplicative factor does not change the results in Equation (2). Similar argument applies to the computation of Equation (3),

and hence k value. By varying the variance of our random number generator from 0 to ∞, we have virtually covered all normal distributions with positive mean values.

The relation between $\overline{k_{mc}}$ and k_{ideal} depends the confidence level. While we cannot prove the relation is independent of the error level, we have found that the plots of 5% and 1% error levels are almost identical. In Figure 2, we plot (with solid line) $\overline{k_{mc}}$ vs. k_{ideal} with k_{ideal} ranging from 1 to 50 under the parameters of 0.99 confidence level and 5% error level . In this figure, we also plot the line $\overline{k_{mc}} = k_{ideal}$ and label it as k_{ideal} for comparison purpose.

In Figure 2, we can use k_{ideal} as a base to define the efficiency coefficient as follows.

Definition 3.1 *The* **efficiency coefficient** ρ_{mc} *for Monte Carlo simulation is defined as*

$$\rho_{mc} = \frac{k_{ideal}}{\overline{k_{mc}}} \times 100\%$$

Definition 3.2 *Give an efficiency coefficient threshold* η*, the Monte Carlo simulation efficient region, called* **mc-efficient** *region, is defined as the range of* k_{ideal} *values where* $\rho_{mc} \geq \eta$*. The compliment set of the* mc-efficient *region is defined as the* **mc-inefficient** *region.*

Mc-efficient region depends on the user-specified efficiency coefficient threshold. In Figure 2, we plot ρ_{mc} vs. k_{ideal} as the heavy dashed line. The smaller the k_{ideal}, the lower ρ_{mc}. For instance, when $k_{ideal} = 2$, $\rho_{mc} = 37\%$. When $k_{ideal} = 1$, $\rho_{mc} = 22\%$, which translates into oversampling by a factor of 3.4. On the other hand, when $k_{ideal} = 10$, $\rho_{mc} = 80\%$, a mere 20% loss in efficiency. In a very large circuit (e.g. a circuit with 100K gates) the power simulation for a single clock cycle needs significant computation time. Therefore, although the k_{ideal} is very small in the mc-inefficient region, the impact of inefficient estimation technique is still enormous.

While we have demonstrated that Monte Carlo simulation could oversample, the question remains: "Will we run into mc-inefficient region when using clock-based Monte Carlo technique for power estimation on a large circuit?" In another words, what k_{ideal} value do we expect to encounter during this type of power estimation? It is difficult to give a quantitative answer. Our approach is to derive the relation between k_{ideal} at the module level and that at the system level. Based on results reported at the module level [7, 9], we can then infer the possible range of k_{ideal} value at top circuit level.

3.2 Relative Variance Analysis

We first look at the case where there is no correlation between P_i and P_j where $i \neq j$. The following theorem gives the bound on the relative variance of P_s. $MAX_{r.v.}$ denotes the maximum relative variance of a module in the circuit.

Theorem 3.1 *Consider a circuit with m modules $M_1, M_2, ..M_m$. Let the random variable representing the clock-by-clock power of module M_i be P_i and its mean and variance be $E[P_i]$ and $V(P_i)$, respectively. In addition, assume that P_i's are uncorrelated. Let the random variable representing the clock-by-clock power of the entire circuit be P_s, then*

$$V_{rel}(P_s) \leq \frac{\sum_{i=1}^{m} E^2[P_i]}{(\sum_{i=1}^{m} E[P_i])^2} MAX_{r.v.} \tag{6}$$

Proof

$$\begin{aligned} V(P_s) &= \sum_{i=1}^{m} V(P_i) \\ &= \sum_{i=1}^{m} V_{rel}(P_i) \cdot E^2[P_i] \\ &\leq MAX_{r.v.} \cdot \sum_{i=1}^{m} E^2[P_i] \\ V_{rel}(P_s) &= \frac{V(P_s)}{(\sum_{i=1}^{m} E[P_i])^2} \\ &\leq \frac{\sum_{i=1}^{m} E^2[P_i]}{(\sum_{i=1}^{m} E[P_i])^2} \cdot MAX_{r.v.} \end{aligned}$$

■

Let $k_{ideal}^{circuit}$ denote the k_{ideal} value for circuit level power estimation, whereas $MAX(k_{ideal}^{module})$ denote the maximum k_{ideal} for the modules in the same circuit. The following corollary gives the relationship between them.

Corollary 3.2 *Let the circuit be the same as the one defined in Theorem 3.1. Given that the same sample size, error level, and confidence level are used at both circuit level and module level, then*

$$k_{ideal}^{circuit} \leq \frac{\sum_{i=1}^{m} E^2[P_i]}{(\sum_{i=1}^{m} E[P_i])^2} MAX(k_{ideal}^{module}) \tag{7}$$

From Corollary 3.2, if there is no single module dominating the circuit power, then $k_{ideal}^{circuit}$ is much smaller than $MAX(k_{ideal}^{module})$. The best case is when $E[P_i]$ are the same for all modules. In this case, $k_{ideal}^{circuit} = \frac{1}{m} MAX(k_{ideal}^{module})$.

The following theorem gives the upper bound on the relative variance of circuit power for the case where modules are correlated.

Theorem 3.3 *Let the circuit be the same as the one defined in Theorem 3.1, except that the modules are correlated.* $V_{rel}(P_s)$ *is bounded from above by:*

$$V_{rel}(P_s) \leq MAX_{r.v.} \tag{8}$$

Proof

$$V(P_s) = \sum_{i=1}^{m} V(P_i) + 2 \sum_{i=1}^{m} \sum_{j>i}^{m} Cov(P_i, P_j)$$

where $Cov(P_i, P_j)$ denotes the covariance between two random variable P_i, P_j. Depending one the sign of the last term, the variance of circuit power may increase or decrease. It is shown in [16] that

$$Cov(P_i, P_j) \leq \sqrt{V(P_i)} \cdot \sqrt{V(P_j)}$$

Therefore,

$$\begin{aligned} V(P_s) &\leq (\sum_{i=1}^{m} \sqrt{V(P_i)})^2 \\ &\leq (\sum_{i=1}^{m} \sqrt{MAX_{r.v.}} \cdot E[P_i])^2 \\ &\leq (\sum_{i=1}^{m} E[P_i])^2 \cdot MAX_{r.v.} \\ V_{rel}(P_s) &= \frac{V(P_s)}{(\sum_{i=1}^{m} E[P_i])^2} \\ &\leq MAX_{r.v.} \end{aligned}$$

■

Corollary 3.4 *Let the circuit be the same as the one defined Corollary 3.2 except that modules are correlated, then*

$$k_{ideal}^{circuit} \leq MAX(k_{ideal}^{module}) \tag{9}$$

From the above analysis, we observe that:

1. When the covariances are positive, the $k_{ideal}^{circuit}$ can be comparable to $MAX(k_{ideal}^{module})$.
2. When the covariances are negative, the $k_{ideal}^{circuit}$ is even smaller than $MAX(k_{ideal}^{module})$.
3. For some positive and some negative covariances between different modules, one would expect that $k_{ideal}^{circuit}$ will still be much smaller than $MAX(k_{ideal}^{module})$.

The results reported from [7, 9] have indicated that about 10 samples are required to achieve 0.99 confidence level and 5% error level for ISCAS85 benchmarks using a sample size of 30 clock cycles. If the circuit of estimation target consists of 10 modules from those benchmarks and there are no correlations between two modules, the number of required samples can be as low as 1 in theory with a sample size of 30 clock cycles! Therefore, Monte Carlo simulation is very likely to oversample if we adopt the clock-cased approach at the top circuit level.

3.3 How to Reduce oversampling

The disadvantage of adopting clock-based approaches at the circuit level is that there is no good method to ensure normality of samples other than taking the average power of n_s randomly selected clock-cycles as a sample. Empirically, the smallest n_s value to satisfy near-normality is 30 [16]. Since the variance of clock-by-clock circuit power can be very low already, this method of ensuring near-normality of samples will force Monte Carlo simulation to utilize the mc-inefficient region.

In the next two sections, we will propose techniques to address this problem. To give an intuitive rationale for our approach, we need to revisit (2).

From (2), the total number of required simulated clock cycles is given by the number of required samples times the number of clock cycles simulated in a sample:

$$\begin{aligned} k_{ideal}^{circuit} \cdot n_s &= (\frac{z_{\alpha/2}}{\epsilon})^2 (V(P_s)/(n_s \cdot E^2(P_s))) \cdot n_s \\ &= (\frac{z_{\alpha/2}}{\epsilon})^2 (V(P_s)/E^2(P_s)) \end{aligned}$$

which is independent of n_s if 1) P_s itself already satisfies normality, and 2) we allow for non-integer $k_{ideal}^{circuit}$ value. With the above result and the efficiency characteristic of Monte Carlo simulation, we should use small workload per sample (equivalently, n_s) whenever possible. We achieve this goal by sampling both on temporal and spatial domains in the circuit.

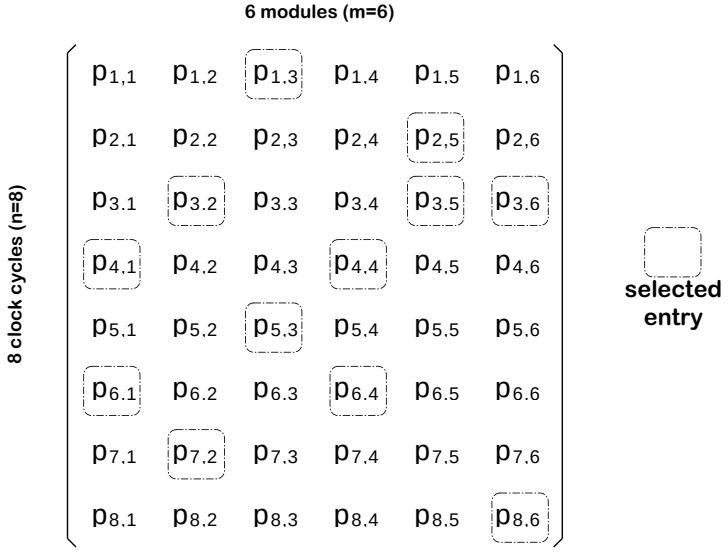

Figure 3: Module-based statistical sampling.

4 Module-Based Power Estimation

The module-based power simulation can be explained in terms of power log matrix as follows. Instead of randomly selecting a rowsum as a unit to be included in a sample, an entry is randomly selected from each column, independent of the selection made in the other columns, as shown in Figure 3. Let P'_j denote a random variable representing the power dissipation value of module M_j in the module-based approach. The new random variable representing the system power estimated in this approach is denoted by P'_s. We can write:

$$P'_s = \sum_{j=1}^{m} P'_j. \tag{10}$$

We use P'_s and P'_j's to accentuate the fact that now P'_j's are independent of each other whereas P_j's in (1) may be correlated.

Theorem 4.1 *Let P_s and P'_s be defined as in (1) and (10), respectively, then $E[P_s]$=$E[P'_s]$.*

Proof For P_s, each row is a unit in the population. All units in the population is equally likely to be selected. Therefore $E(P_j) = \frac{1}{n}\sum_{i=1}^{n} p_{i,j}$. On the other hand, for P'_j, each entry $p_{i,j}$ for the same j value and $1 \leq i \leq n$ is equally likely to be selected. Therefore $E(P'_j) = \frac{1}{n}\sum_{i=1}^{n} p_{i,j}$. $E[P_j]$=$E[P'_j]$. Consequently, $E[P_s]$=$E[P'_s]$. ∎

Since P_j''s are now independent, the module-based technique will have a better normality when compared with clock-based techniques using the same workload. This is because the empirical requirement for applying the central limit theorem to ensure near-normality of samples is that a minimum number of independent random variables must be included in a sample and that some 'general condition' is met. The general condition is informally stated as follows: "no single random variable makes a large contribution to the sum" [16]. One should not interpret this statement as about the mean values of those random variables. Instead, it should be interpreted as a statement about the 'fluctuation' of each random variable around its mean (i.e., its variance).

For clock-based techniques such as [7], each randomly selected clock cycles is identically distributed. The number independent random variables included in a sample is n_s. In contrast, in the module-based approach, the power dissipation of each module in a clock cycle counts as a random variable. And these random variables may have very different variances. For now, assume that each module has similar variance. If we have m modules, we only need to simulate each module for $\lceil \frac{n_s}{m} \rceil$ clock cycles to satisfy the requirement of n_s random variables. That means, the simulation load can be reduced by a factor of m. However this is the ideal situation. A difficult case is when one module dominate the sample variance. In this case, no reduction of workload is achieved. To alleviate this problem, we adapt stratified random sampling as outlined next. The goal is to allocate more randomly drawn units to the module with higher variance.

In stratified sampling, the population is divided into a set of disjoint groups (i.e. strata) such that units in the same stratum have similar power values. Next a predetermined and fixed number of units are randomly selected and simulated from each stratum and collected to form a sample. In our approach, the same number of units are drawn from each stratum. We use a predictor function to stratify the population. There are several proposed predictor functions [9, 17], ranging from the zero delay power estimate (the most computationally expensive) to the average switching activities on the module inputs, outputs, and internal sequential elements, weighted by estimated capacitances (or transistor counts) in each module. Since the predictor function is not directly used in computing the power consumption, reasonable deviation from absolute accuracy is acceptable. One should note that a more computational expensive power simulator can afford to use a more elaborated predictor function.

When allocating the number of strata to each module, since we cannot predict in advance how effective stratification will be, we completely ignore the future impact of stratification. We adapt the minimum variance allocation scheme proposed by [18]. This scheme is originally proposed as a solution to sample size allocation problem by allocating the number, n_l, of

randomly drawn units for l-th stratum as proportional to $(W_l S_l / \sum_l W_l S_l)$, where W_l and S_l^2 are the stratum weight (stratum size divided by population size) and variance of l-th stratum. In our application at this stage, each module is treated as a stratum and therefore their weights are the same. The module variances are estimated using predictor values. In the case where a single module contributes significantly higher variance, our scheme ensures more strata, and hence more drawn units, will be allocated to that module. After the number of strata is determined for each module, the units in each module are first sorted based on the predictor values and put in equal size bins with each bin representing a stratum.

To summarize, the module-based technique is more efficient than clock-based techniques as:

1. When power values of modules are strongly positively correlated, $k_{ideal}^{circuit}$ will be high and there may not be significant oversampling in clock-based techniques. By sampling each module independently, the correlations between modules are completely eliminated in module-based techniques, and consequently, smaller sample variances are achieved.

2. When power values of modules are strongly negatively correlated or weakly correlated, the required number of samples in clock-based techniques become so low that oversampling becomes an issue. By using smaller sample granularities in the module based technique, it reduces the amount of oversampling.

The main shortcoming of the stratified sampling is that the objective of reducing the workload conflicts with that of stratification. Using higher number of strata gives better efficiency improvement. However, at least one clock cycle needs to be simulated for each stratum. Therefore the minimum workload per sample will be raised. This is fine for mid-size systems. For large-scale systems, we could still enter the mc-inefficient region.

Our solution to this problem is to "lump" all modules of similar types into a super module and then perform stratification on each super module as described in the next section.

5 Cluster-based Power Estimation

In this section, we propose a technique that significantly reduces the workload of a power sample while maintaining the normality.

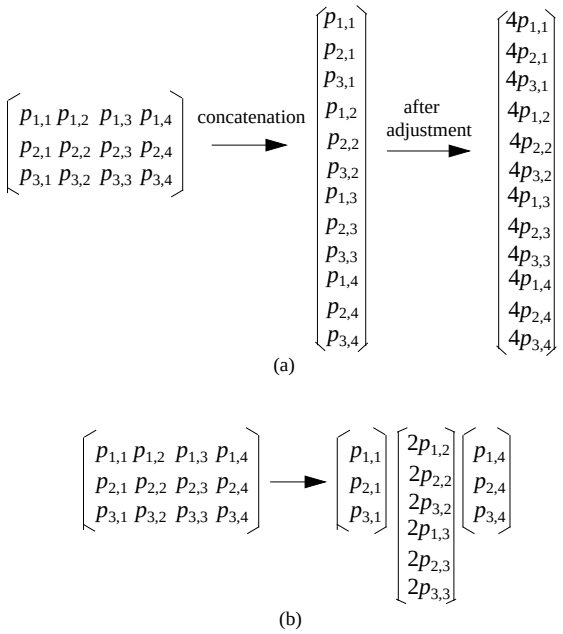

Figure 4: The transformation of two circuits for sampling.

5.1 Homogeneous Circuits

To give a rationale for this approach, consider a fairly homogeneous circuit for now (for instance, a circuit consisting of only adders). Almost all practical circuits are heterogeneous and they will be examined later in this section. There are n clock cycles and m modules in the circuit. The average power of this circuit is calculated as:

$$E(P_A) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij} \tag{11}$$

Next, consider circuit B which consists of a single adder but there are $n \cdot m$ clock cycles. Furthermore, the power log matrix of circuit B is obtained by concatenating all columns of the power log matrix of circuit A into a single column as shown in Figure 4(a). The average power of this circuit is:

$$E(P_B) = \frac{1}{mn} \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij} \tag{12}$$

Obviously, (12) is simply scaled down from (11) by m. Therefore all the entries in the single

column matrix should be multiplied by m if we want to estimate the average circuit power of circuit A by sampling on circuit B.

The overall strategy can be called a *mix-and-stratify* strategy as explained next. Consider a homogeneous circuit as circuit A above. First the predictor function are computed for all clock cycles and all modules and the units of all modules in the circuit are mixed together. The mixture is stratified into m' equal size strata. One unit is randomly sampled from each stratum. We can determine to which the unit originally belonged to and can perform gate-/transistor-level simulation on that module under the corresponding vector pair. The observed power from the simulation is multiplied by m and treated as a unit drawn in the standard stratified sampling. When calculating the sample value, we can exactly follow the steps in the standard stratified sampling. This approach can be easily extended to non-equal size stratum and/or non-equal sample sizes.

The advantage of this approach is that the minimum workload in a sample to maintain normality of samples can be less than that of simulating the entire circuit for one clock cycle. To give an example, consider a circuit with 60 adder modules. These 60 modules are lumped into a cluster (super module). Let the number of strata be 30. One unit is randomly drawn from each stratum and collectively becomes a sample. The workload of this sample is same as simulating only one half of the circuit for one clock cycle.

Next, we will address the issues of heterogeneous circuits.

5.2 Heterogeneous Circuits

By heterogeneous circuits, we mean circuits which contain more than one type of module. The classification scheme is based on the glitch activities in each module. For instance, multipliers should not be classified as the same type as that of adders, as the former has excessive glitch activities. On the other hand, random logic can be classified as the same type as that of adders. All modules of the same type are put into one cluster. As for power log matrix, all the columns corresponding to modules in the same cluster are concatenated into a single column matrix. Let the number of clusters be c and the number of modules in each cluster be m_i, where $i = 1, \ldots, c$. After the column concatenation, all the entries in the ith single column matrix are multiplied by m_i, as shown in the previous section. Then the estimation can be reduced to that of estimating the power on a new circuit consisting of c modules with each module representing a single columns matrix as shown in Figure 4(b). The simulation strategy is similar to the one described in the Section 5.1.

5.3 Stratified Sampling

In module-based techniques, stratified sampling is for improving efficiency. However, in cluster-based sampling, stratified sampling is mandatory. This is because when modules are merged into a cluster, it can act as a reverse process of stratification if each original module is already very homogeneous and yet at the same time, these modules have very different mean values. In another words, the clustering process alone can produce samples with multi-modal distributions. However, this issue can be easily prevented by stratification after clustering.

6 Practical Considerations

One may be concerned about the high overhead associated with storing the vector traces of sequential elements at each clock cycle. This overhead can be significantly reduced by using a two-stage sampling technique [9]. In the first stage, simple random sampling is performed during functional simulation to select a subpopulation of size b. The value of b is determined before functional simulation. When the functional simulation starts, b randomly selected clocks are marked. As the functional simulator reaches each marked cycle, the vector pairs of all sequential elements at that cycle are sampled and stored. The collection of all sampled vector pairs becomes the population for the second stage sampling where the techniques proposed in this paper are applied.

Higher subpopulation size increases stratification overhead whereas lower subpopulation size increases the inaccuracies of estimates. [9] gives detailed analysis on this issue and concluded that once subpopulation size reaches a certain value, the improvement on accuracy is negligible on further increase in subpopulation size. Our rule of thumb is that the subpopulation size should be at least twenty five times the largest average required number of simulated clock cycles used in the second-stage sampling techniques. The rationale behind the rule is that the error of a two-stage sampling technique is the root-mean-square of the errors from both stages. Our rule ensures that the error from the first stage is at least five times smaller than that from the second stage.

7 Experimental Results

We compare the sampling efficiency of four techniques : clock-based simple random sampling (SRS), clock-based stratified random sampling(STS), module-based stratified sampling(MODU),

Table 1: The number of modules in each circuit type

ckt	adder	subtracter	multiplier
CHEB	8	0	5
DS	1	3	7
IIR	6	2	5
DCT	24	24	10

Table 2: Results of 100,000 Monte Carlo simulation runs on music input streams

Scheme	CHEB							DS						
	s.s.	k_{ideal}	k_{avg}	avg W_{rel}	v.r. (%)	r.t. (s.)	o.h. (s.)	s.s.	k_{ideal}	k_{avg}	avg W_{rel}	v.r. (%)	r.t. (s.)	o.h. (s.)
SRS	30	2.61	6.0	180	0.2	7.5		30	8.97	11.6	348	0.4	18.9	
STS	30	1.17	4.6	137	0.1	6.6	0.7	30	2.04	5.5	163	0.0	8.4	1.0
MODU	6	5.17	8.3	50	0.8	2.4	0.8	6	6.18	9.7	55	0.8	3.5	1.1
CLUS	30	5.73	8.7	52	0.6	2.4	0.8	30	7.80	10.7	46	0.9	3.1	1.1
Scheme	IIR							DCT						
	s.s.	k_{ideal}	k_{avg}	avg W_{rel}	v.r. (%)	r.t. (s.)	o.h. (s.)	s.s.	k_{ideal}	k_{avg}	avg W_{rel}	v.r. (%)	r.t. (s.)	o.h. (s.)
SRS	30	3.79	7.1	211	0.7	9.0		30	2.20	5.2	156	0.2	16.2	
STS	30	1.58	5.0	149	0.0	7.2	0.8	30	0.59	3.9	117	0.0	13.2	2.0
MODU	6	5.76	8.3	51	0.8	2.4	0.8	3	4.83	8.4	25	0.4	3.0	2.1
CLUS	30	5.64	8.7	53	0.6	2.5	0.8	30	4.30	7.5	17	0.4	2.1	2.3

and cluster-based stratified sampling(CLUS).

The circuits are selected from popular high level synthesis benchmarks: a Chebyshev filter (CHEB), a differential solver (DS), an IIR filter (IIR), and a discrete cosine transformation circuit (DCT). The numbers of modules in each circuit type are summarized in Table 1. All modules are of 16-bit type. The first input sequence is obtained from a music CD and it will be referred to as the music stream. This stream contains strong low frequency components (due to the drums in the music) and hence is correlated. The second input stream is generated randomly. Both streams are 100,000 clock cycles in length. For stratification (STS, MODU, CLUS), we use a bit-parallel algorithm to compute the zero-delay power estimate as the predictor. The target simulator is a general-delay logic simulator. The power value of each

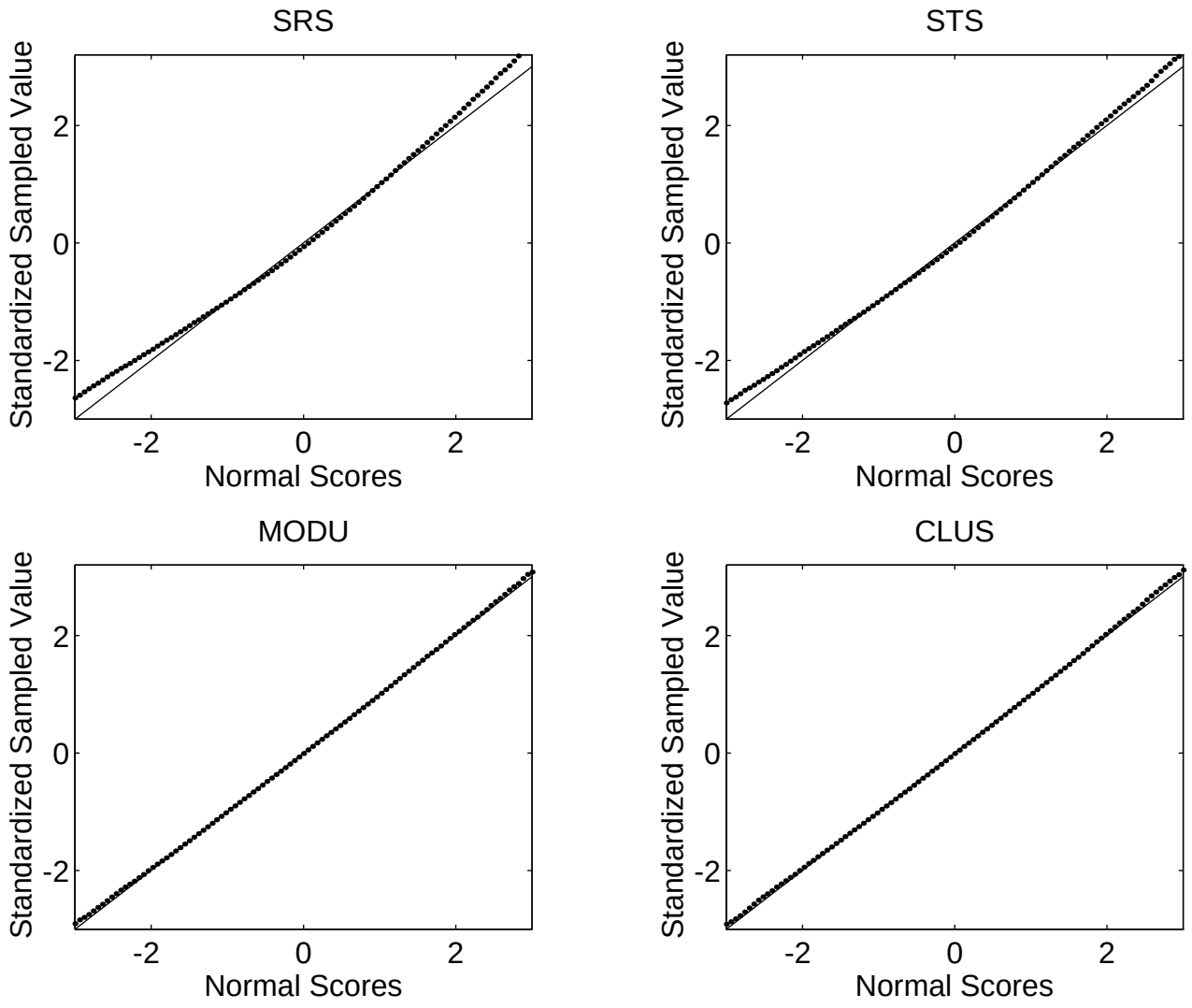

Figure 5: Normality score plots.

Table 3: Results of 100,000 Monte Carlo simulation runs on random input streams

Scheme	CHEB							DS						
	s.s.	k_{ideal}	k_{avg}	avg W_{rel}	v.r. (%)	r.t. (s.)	o.h. (s.)	s.s.	k_{ideal}	k_{avg}	avg W_{rel}	v.r. (%)	r.t. (s.)	o.h. (s.)
SRS	30	1.82	5.3	157	0.0	6.6		30	8.24	11.0	329	0.6	17.8	
STS	30	0.94	4.3	129	0.0	6.3	0.7	30	1.92	5.4	160	0.0	10.1	1.0
MODU	6	4.51	7.7	46	0.3	2.2	0.8	6	5.88	8.9	53	0.3	3.3	1.1
CLUS	30	5.16	8.3	50	0.3	2.3	0.8	30	7.71	10.4	46	0.5	2.9	1.1
Scheme	IIR							DCT						
	s.s.	k_{ideal}	k_{avg}	avg W_{rel}	v.r. (%)	r.t. (s.)	o.h. (s.)	s.s.	k_{ideal}	k_{avg}	avg W_{rel}	v.r. (%)	r.t. (s.)	o.h. (s.)
SRS	30	1.75	5.2	156	0.0	6.5		30	0.87	4.2	127	0.0	13.2	
STS	30	0.83	4.2	126	0.0	6.0	0.8	30	0.40	3.6	108	0.0	13.0	2.0
MODU	6	4.37	7.6	45	0.3	2.3	0.9	3	4.11	7.3	22	0.2	2.1	2.1
CLUS	30	4.31	7.5	45	0.2	2.1	0.8	30	3.95	7.2	16	0.2	2.0	2.2

cell is obtained from a lookup table indexed by output loading and input signal slew. The ratio of computational speed of the predictor function vs. target simulator is about 60 to 1. The total circuit power of each circuit is first obtained by simulating the circuit over the entire sequence.

We set the error and confidence levels to 5% and 0.99, respectively. We perform 100,000 simulation runs for each sampling method. For stratified sampling, in order to reduce the overhead of stratification (including predictor calculation and sorting), we use the two-stage sampling method described in Section 6. The subpopulation size is set to 1000. That is, a different subpopulation of size 1000 is first randomly selected in each run. The stratification is performed on the subpopulation only. The sample size is set such that there are at least 30 independent random variables in a sample. Again, here we have assumed that 30 independent random variables is high enough to ensure near-normality. For SRS and STS, the sample size is simply the number of clock cycles simulated in each sample. In MODU, the numbers of strata are same for all modules. The sample size is equal to the number of strata in each module and one unit is drawn from each stratum to form a sample. For CLUS, we use two clusters. All adders and subtracters are put into one cluster whereas all multipliers in another. Each cluster is stratified into the same number of equal-size strata as the sample size and one unit is sampled from each stratum.

The results of music stream and random stream are summarized in Table 2 and 3. The 's.s.' columns list the sample size. The 'k_{ideal}' columns list the k_{ideal} values for reference purpose. The 'k_{avg}' columns list the average number of samples required in each method. The 'avg W_{rel}' columns list the average relative workload required for convergence, formulated as:

$$\frac{\text{workload per sample}}{\text{workload of simulating entire system for one clock cycle}} \cdot k_{avg}$$

The 'v.r' columns list the percentage of simulations that have greater than 5% error. Entries with value 0.0 represent values smaller than 0.01. The 'r.t' columns list the run time (excluding stratification) on a Pentium Pro 200Mhz machine. The 'o.h.' columns list the overheads of stratification.

From these tables, it shows that STS does not improve much over SRS from the Monte Carlo simulation results (with the exception of DS), although k_{ideal} shows that the improvement should have been 2x or so. In another words, the advantages of variance reduction techniques such as stratification may not be fully realized in Monte Carlo simulation if k_{ideal} is very low. Between STS and MODU, if both of them had used the same workload per sample, their k_{ideal}'s will be very close (with the exception of DS). For instance, on CHEB with music stream, if STS had used s.s. of 6, k_{ideal} would be $1.17 \times 5 = 5.85$ which is comparable to 5.17 in MODU. However, due to concerns on normality, one cannot simply use smaller sample size in SRS and STS without some analysis on the circuit power. To demonstrate that our proposed techniques achieve better normality using same workload per sample, Figure 5 shows the normality plots of all four techniques for DCT (music stream) under the workload of 2 clock cycles. Both MODU and CLUS are about 2x better than SRS and STS.

The overhead of stratification is approximately linear with subpopulation size. We varied the subpopulation size from 500 to 4000 and observed no obvious change in k_{avg} or v.r. The subpopulation size chosen in this experiment is close to twenty-five times the avg W_{rel} values of both MODU and CLUS. Again, we should emphasize that the choice of predictor function should depend on the execution speed of the target simulator to minimize the significance of stratification overhead.

MODU and CLUS are comparable except for larger benchmarks DS and DCT, where CLUS is approximately 15% to 37% better than MODU on avg W_{rel}. This follows our expectation that CLUS will be advantageous when the number of sizable modules is larger.

In this experiment, the run time improvement of MODU and CLUS over SRS and STS is about a factor of 2-3x, including the overhead of stratification.

8 Conclusion

In this paper, we propose efficient statistical sampling techniques for hierarchical circuit power estimation. We first show that Monte Carlo based statistical power estimation techniques oversample when the relative variances of samples are small. Apply existing clock-based sampling techniques to estimate total circuit power exacerbates the oversampling issues as the workload per sample is too high. To address this issue, we need to reduce the workload per sample while maintaining the normality of these samples. We propose a module-based and a cluster-based Monte Carlo simulation technique to achieve this goal. We demonstrate that the proposed techniques provide a reduction of 2-3x in simulation run time compared to existing Monte-Carlo simulation techniques.

References

[1] P. Landman. High-level power estimation. In *Proc. 1996 International Symp. on Low Power Electronics and Design*, pages 29–35, 1996.

[2] V. Tiwari, S. Malik, and A. Wolfe. Power analysis of embedded software: A first step towards software power minimization. IEEE *Transactions on VLSI Systems*, 2(4):437–445, December 1994.

[3] D. Liu and C. Svensson. Power consumption estimation in CMOS VLSI chips. IEEE *Journal of Solid State Circuits*, 29(6):663–670, 1994.

[4] F. N. Najm, R. Burch, P. Yang, and I. Hajj. Probabilistic simulation for reliability analysis of CMOS VLSI circuits. IEEE *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 9(4):439–450, April 1990.

[5] C-Y. Tsui, M. Pedram, and A. M. Despain. Efficient estimation of dynamic power dissipation under a real delay model. In *Proceedings of the* IEEE *International Conference on Computer Aided Design*, pages 224–228, November 1993.

[6] A. A. Ghosh, S. Devadas, K. Keutzer, and J. White. Estimation of average switching activity in combinational and sequential circuits. In *Proceedings of the 29th Design Automation Conference*, pages 253–259, June 1992.

[7] R. Burch, F. N. Najm, P. Yang, and T. Trick. A Monte Carlo approach for power estimation. IEEE *Transactions on VLSI Systems*, 1(1):63–71, March 1993.

[8] M. Xakellis and F. Najm. Statistical estimation of the switching activity in digital circuits. In *Proceedings of the 31st Design Automation Conference*, pages 728–733, 1994.

[9] C.-S. Ding, C.-T. Hsieh, Q. Wu, and M. Pedram. Stratified sampling for power estimation. IEEE *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(6):465–471, June 1998.

[10] L.-P. Yuan, C.-C. Teng, and S.-M. Kang. Statistical estimation of average power dissipation in CMOS vlsi circuits using nonparametric technique. In *Proc. 1996 International Symp. on Low Power Electronics and Design*, pages 73–78, 1996.

[11] F. N. Najm, S. Goel, and I. N. Hajj. Power estimation in sequential circuits. In *Proceedings of the 32nd Design Automation Conference*, pages 635–640, 1995.

[12] J. Kozhaya and F. Najm. Accurate power estimation for large sequential circuits. In *Proceedings of the* IEEE *International Conference on CAD-96*, pages 488–493, November 1996.

[13] L. Benini, G. De Micheli, E. Macii, M. Poncino, and R. Scarsi. Fast power estimation for deterministic input streams. In *Proceedings of the* IEEE *International Conference on CAD-97*, pages 494–501, November 1997.

[14] F. Najm. A survey of power estimation techniques in vlsi circuits. IEEE *Transactions on VLSI Systems*, 2(4):377–381, December 1994.

[15] E. Macii, M. Pedram, and F. Somenzi. High-level power modeling, estimation, optimization. IEEE *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(11):1061–1079, November 1998.

[16] I. R. Miller, J. E.Freund, and R. Johnson. *Probability and statistics for engineers.* Prentice Hall, 1990.

[17] L. Benini, A. Bogliolo, M. Favalli, and G. De Micheli. Regression models for behavioral power estimation. In *Proc. 1996 International Workshop on Power and Timing Modeling, Optimization and Simulation*, November 1996.

[18] J. Neyman. On the two different aspects of the representative method. *J. R. Statist. Soc.*, 97:558–606, 1934.