Cycle-Accurate Macro-Models for RT-Level Power Analysis*

Qinru Qiu, Qing Wu, Massoud Pedram
Department of Electrical Engineering-Systems
University of Southern California

Chih-Shun Ding
Rockwell Semiconductor Systems
Newport Beach Division

Abstract *In this paper we present a methodology and techniques for generating cycle-accurate macro-models for RT-level power analysis. The proposed macro-model predicts not only the cycle-by-cycle power consumption of a module, but the power profile of the module over time. The proposed methodology consists of three steps: module equation form generation and variable selection, variable reduction, and population stratification. First order temporal correlations and spatial correlations of up to order 3 are considered to improve the estimation accuracy. Experimental results show that, the macro-models have 15 or less variables and exhibit <5% error in average power, and <15% errors in cycle-by-cycle power compared to circuit simulation results using Powermill.*

I. INTRODUCTION

Due to rapid progress in the semiconductor manufacturing, the device density and operating frequency have greatly increased, making power consumption a major design concern. High power consumption exacerbates the reliability problem by raising the die temperature and by increasing current density on the supply rails. It also reduces battery life which is a key concern in portable devices. Therefore, low power design requirements are driving a new breed of design automation methodologies and tools which in turn rely on accurate and efficient estimation tools at various design levels.

Power estimation at RT level is crucial in achieving a short design cycle. Macro-modeling is the major technique for power estimation at RT-level. In this technique, low-level simulations of modules is replaced by power macro-model equation evaluation (which can be performed very fast).

Macro-modeling techniques use capacitance models for circuit modules and activity profiles for data or control signals [1-3]. The simplest form of the macro-model equation is given by:

$$Power = \tfrac{1}{2} V^2 \cdot f \cdot C_{eff} \cdot SW \qquad (1.1)$$

where C_{eff} is the effective capacitance, SW is the mean of the input switching activity, and f is the clock frequency. The Power Factor Approximation (PFA) technique uses an experimentally determined weighting factor, called the power factor, to model the average power consumed by a given module over a range of designs.

To improve the accuracy, more sophisticated macro-model equations have been proposed. Dual Bit Type model, proposed in [2], exploits the fact that, in the data path or memory modules, switching activities of high order bits depend on the temporal correlation of data when lower order bits behave similarly to white noise data. Thus a module is completely characterized by its capacitance models in the MSB and LSB regions. The break-point between the two regions is determined based on the signal statistics collected from simulation runs. The Activity-Based Control (ABC) model [4] is proposed to estimate the power consumption of random-logic controllers. All of the above macro-models assume some statistics or properties about the input sequence.

All of the above techniques are suitable for estimating the average-power dissipation (and are referred to as *cumulative power macro-model*). In some applications, however, estimation of average power is just one task in the broader sense of power evaluation. Other tasks include the estimation of the moving average of the power, power profiling on a cycle-by-cycle basis, and rate of current change estimation. This type of information is crucial for system reliability analysis and DC/AC noise analysis. If the macro-modeling technique does not provide such information, the circuit designers will have to resort to gate-level or circuit-level simulator again. Consequently, this kind of macro-model is considered to be less useful.

The notion of *cycle-accurate macro-models* was proposed in [5]. Let P_{jk} denote the power consumption of module j in clock cycle k, then we can write:

$$P_{jk} = F_j(V_{j,k-1}, V_{j,k}) \qquad (1.2)$$

where $V_{j,k}$ and $V_{j,k-1}$ denote the input vectors for module j at cycles k and k-1, and F_j is some function of the input vector pairs. The goal of power macro-modeling is to find function F_j given an input vector sequence V (the so called *training set*) for module j and given the corresponding power consumption values. Cycle based macro-models can be easily transformed into cumulative macro-models [5].

This paper improves results of [5] in the following directions. A new variable selection methodology is applied to capture the relation between power consumption and module inputs/outputs. The spatial correlation among inputs are considered up to order three. Because we use integer variables instead of 0-1 variables as in [5], our macro-models have fewer variables (fewer than 15 variables compared to 40~100 variables in [5]) and higher accuracy (10% error on cycle-by-cycle basis compared to 11.2% in [5]. In addition, we use population stratification to obtain a macro-model with higher fidelity.

This paper is organized as follows. Section II gives the theoretical background for regression analysis. Section III discusses a procedure of building the macro-model whereas Section IV presents the experimental results. Section V will discuss some applications of cycle-accurate macro-models.

II. BACKGROUND

Based on the theory of regression analysis, we define the relation between power and input vector pair characteristics as a statistical relation, which can be expressed as :

* This research was supported in part by DARPA under contract number F336125-95-C1627, SRC under contract number 97-DJ-559.

$$Y = f(X_1, X_2, ...X_n) + \varepsilon \qquad (2.1)$$

where ε is a random error term of normal distribution with the mean $E\{\varepsilon\} = 0$ and the variance $\sigma^2\{\varepsilon\} = \sigma^2$, and X_1, X_2, …, X_n are the characteristic variables related to the input vector pairs. (2.1) is different from a functional relation in that: 1) same variable assignment may produce different Y values because different vector pairs which result in different power consumption may produce the same set of characteristic values, and 2) Y is thus regarded as a random variable with mean $E(Y) = f(X_1, X_2, ...X_n)$.

We define the cycle-accurate macro-model as a linear function describing the statistical relationship between power dissipation of a vector pair and the characteristic values of the vector, that is, we write:

$$P = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k \qquad (2.2)$$

where P is the power (which is the power estimated from a circuit-level simulator, such as Powermill [7]), $\beta_0, \beta_1, \cdots, \beta_k$ are constants called the regression coefficients of the macro-model, and $X_1, X_2, \cdots, X_k$ are characteristic variables extracted from the input vector pair.

Assume that we have been given the equation form of the macro-model as (2.2), and have performed Powermill simulations (observations) on m randomly sampled vector pairs in the population (this set of m vector pairs is referred to as the *training set*) so that we have obtained m simulation results (observation values) of power. The linear regression model for vector pairs from the training set can be written as:

$$P_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \cdots + \beta_k x_{i,k} + \varepsilon_i, \quad i = 1,2,\cdots,m \quad (2.3)$$

or in matrix form as:

$$\mathbf{P} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \qquad (2.4)$$

where P_i's are random variates corresponding to observations: ($x_{i,1}, x_{i,2}, \cdots, x_{i,k}$) $i = 1,2,..,m$; $\beta_0, \beta_1, \cdots, \beta_k$ are the regrssion coefficients; $x_{i,1}, x_{i,2}, \cdots, x_{i,k}$ are known values derived from the input vector pair ($V_{i,1}, V_{i,2}$); and ε_i's are independent random variates representing deviation from the mean value of power with variance $\mathbf{VAR}[\varepsilon_i] = \sigma^2$, and $\mathbf{Cov}[\varepsilon_i, \varepsilon_j] = 0$, for $i \neq j$. Consequently, the random vector **P** has an expected value of $\mathbf{E}[\mathbf{P}] = \mathbf{X}\boldsymbol{\beta}$ and the variance-covariance matrix of **P** is $\mathbf{Cov}[\mathbf{P}] = \sigma^2\mathbf{I}$, where **I** is the identity matrix.

The least square estimator for the coefficients $\boldsymbol{\beta}$ is:

$$\mathbf{b} = (\mathbf{X}^{\mathrm{T}} \cdot \mathbf{X})^{-1} \cdot \mathbf{X}^{\mathrm{T}} \cdot \mathbf{P} \qquad (2.5)$$

where

$$\underset{(k+1)\times 1}{\mathbf{b}} = [b_0, b_1, \cdots, b_k]^{\mathrm{T}} \qquad (2.6)$$

It has been proved in [6] that the lest square estimator is an unbiased estimator for $\boldsymbol{\beta}$, which means $\mathrm{E}[\mathbf{b}] = \boldsymbol{\beta}$.The estimated (fitted) power from macro-model is given by:

$$\hat{\mathbf{P}} = [\hat{P}_1, \hat{P}_2, \cdots, \hat{P}_m] = \mathbf{Xb} \qquad (2.7)$$

and the residual terms (error) are defined as the difference between the fitted power and observed (actual) power:

$$\mathbf{e} = [e_1, e_2, \cdots e_m] = \mathbf{P} - \hat{\mathbf{P}} = \mathbf{P} - \mathbf{Xb} \qquad (2.8)$$

In the following, we define some relevant terms for regression analysis [6].

sum of squares error: $SSE = \sum_{i=1}^{m} e_i^2$

mean squares error: $MSE = SSE/(m-k-1)$

regrssion sum of squares: $SSR = \sum_{i=1}^{m} (\hat{P}_i - \overline{P})^2$

regrssion mean squares: $MSR = SSR/k$

coefficient of correlation: $R = \sqrt{SSR/(SSR + SSE)}$

The statistical nature of the macro-model enables us to predict the accuracy level of fitted power value as follows. Given any input vector pair, the values of its characteristic variables (x_1, x_2, …, x_k) are first computed. The fitted (predicted) power is given by $\hat{P} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_k x_k$. Given a confidence level 1-α, the confidence interval of the actual power P is defined as an interval [P_1, P_2] such that the probability that the actual power value lies inside this interval is 1-α. We can thus compute the confidence interval for P at any confidence level 1-α as:

$$[\hat{P} - t(1-\alpha/2; m-k-1) \cdot s[P], \hat{P} + t(1-\alpha/2; m-k-1) \cdot s[P]] \quad (2.9)$$

where $t(1-\alpha/2; m-k-1)$ is the $(1-\alpha/2)\times 100$ percentile point of the t distribution with degree of freedom of (m-k-1) and $s[P]$ is the standard deviation of the new observation which is given by:

$$s[P] = \sqrt{MSE \cdot (1 + X^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1} X)} \qquad (2.10)$$

where **X** and *MSE* are the variable matrix and mean squares error of the training set, respectively.

The quality of the macro-models can be evaluated in terms of the following criteria:

1. Correlation coefficients: Coefficient of multiple correlation R is a general measure of the quality of a regression model since it represents linearity of the model and the magnitude of the error. From its definition, $0 \le R \le 1$. Furthermore, the higher the R value, the better the quality of the regression model. The R value may differ from one population to next for the same macro-model. Therefore, the R values of different macro-models should be compared only when they are subjected to the same input population.
2. Errors: Error in cycle power (ECP) gives the average error when estimating power on cycle by cycle basis while error in average power (EAP) gives the average error when estimating the average power. More precisely, we can write:

$$ECP = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{\hat{P}_i - P_i}{P_i}\right|, \qquad EAP = \frac{\sum_{i=1}^{n}\hat{P}_i - \sum_{i=1}^{n}P_i}{\sum_{i=1}^{n}P_i} \tag{2.11}$$

III. GENERATING THE MACRO-MODEL

3.1 Variable Selection

3.1.1 Theoretical foundation of macro-model equation

If we ignore power consumption of the floating nodes within gates (it is <5% in practice), the power consumption of a combinational module is only a function of transitions at the primary inputs and can be written as:

$$P = f(\vec{t}_1, \vec{t}_2, \cdots, \vec{t}_k) \tag{3.1}$$

where k is the number of inputs and $\vec{t}_1, \vec{t}_2, \cdots, \vec{t}_k$ are the so called transition variables which encoded by a bit vector as following:

$$\begin{aligned} &\vec{t}_i = [a \quad b \quad c], \quad i = 1,2,\cdots,k \\ &a = b = c = 0 \text{ if } i: 0 \to 0, \quad a = 1, b = c = 0 \text{ if } i: 0 \to 1 \\ &b = 1, a = c = 0 \text{ if } i: 1 \to 0, \quad c = 1, a = b = 0 \text{ if } i: 1 \to 1 \end{aligned} \tag{3.2}$$

Note that the function is defined as a mapping from vector space to real numbers. Equation (3.1) can be expanded around $\mathbf{t}^0 = (\vec{t}_1^{\,0\to0}, \vec{t}_2^{\,0\to0}, \cdots, \vec{t}_k^{\,0\to0})$ as follows:

$$\begin{aligned} P = f(\vec{t}_1, \vec{t}_2, \cdots, \vec{t}_k) &= f(\vec{t}_1^{\,0\to0}, \vec{t}_2^{\,0\to0}, \cdots, \vec{t}_k^{\,0\to0}) \\ &+ \sum_{i=1}^{k}\left.\frac{\partial f}{\partial \vec{t}_i}\right|_{\mathbf{t}^0}\Delta\vec{t}_i + \sum_{i=1}^{k}\sum_{j=i+1}^{k}\left.\frac{\partial^2 f}{\partial \vec{t}_i \partial \vec{t}_j}\right|_{\mathbf{t}^0}\Delta\vec{t}_i \otimes \Delta\vec{t}_j \\ &+ \cdots + \left.\frac{\partial^k f}{\partial \vec{t}_1 \partial \vec{t}_2 \cdots \partial \vec{t}_k}\right|_{\mathbf{t}^0}\Delta\vec{t}_1 \otimes \Delta\vec{t}_2 \otimes \cdots \otimes \Delta\vec{t}_k \end{aligned} \tag{3.3}$$

where

$$\begin{aligned} &\Delta\vec{t}_i = \vec{t}_i \text{ xor } \vec{t}_i^{\,0\to0} = [t_{i,1} \oplus t_{i,1}^{0\to0}, t_{i,2} \oplus t_{i,2}^{0\to0}, t_{i,3} \oplus t_{i,3}^{0\to0}] \\ &= [t_{i,1} \oplus 0, t_{i,2} \oplus 0, t_{i,3} \oplus 0] = [t_{i,1}, t_{i,2}, t_{i,3}] = \vec{t}_i \end{aligned}$$

and the "⊗" operation between two vectors is defined as:

$$[u_1, u_2, \cdots, u_m] \otimes [v_1, v_2, \cdots, v_n] = [u_1v_1, u_1v_2, \cdots, u_1v_n, u_2v_1, u_2v_2, \cdots, u_2v_n, \cdots, u_mv_1, u_mv_2, \cdots, u_mv_n] = [w_1, w_2, \cdots, w_{mn}]$$

By redefining the constants in (3.3), we can write:

$$\begin{aligned} P = a_0 &+ \sum_{i=1}^{k}\vec{t}_i \cdot \begin{bmatrix} a_i^{0\to1} \\ a_i^{1\to0} \\ a_i^{1\to1} \end{bmatrix} + \sum_{i=1}^{k}\sum_{j=i+1}^{k}\vec{t}_i \otimes \vec{t}_j \cdot \begin{bmatrix} a_{i,j}^{0\to1,0\to1} \\ a_{i,j}^{0\to1,1\to0} \\ \vdots \\ a_{i,j}^{1\to1,1\to1} \end{bmatrix} \\ &+ \cdots + \vec{t}_1 \otimes \vec{t}_2 \otimes \cdots \otimes \vec{t}_k \cdot \begin{bmatrix} a_{1,2,\ldots,k}^{0\to1,0\to1,\cdots,0\to1,0\to1} \\ a_{1,2,\ldots,k}^{0\to1,0\to1,\cdots,0\to1,1\to0} \\ \vdots \\ a_{1,2,\ldots,k}^{1\to1,1\to1,\cdots,1\to1,1\to1} \end{bmatrix} \end{aligned} \tag{3.4}$$

$$\begin{aligned} &= a_0 + \sum_{i=1}^{k}\vec{t}_i \cdot \vec{a}_i^{\,\mathrm{T}} + \sum_{i=1}^{k}\sum_{j=i+1}^{k}\vec{t}_i \otimes \vec{t}_j \cdot \vec{a}_{i,j}^{\,\mathrm{T}} + \cdots \\ &+ \vec{t}_1 \otimes \vec{t}_2 \otimes \cdots \otimes \vec{t}_k \cdot \vec{a}_{1,2,\ldots,k}^{\,\mathrm{T}} \end{aligned}$$

where $\vec{t}_i$ is called *order* 1 *transition variable* of input *i*, $\vec{t}_i \otimes \vec{t}_j$ is called *order* 2 *joint transition variable* of inputs *i* and *j*, etc. Entries of these vector variables are either 0 or 1 and the sum of entries in each vector add up to 1.

Theorem 1 *Equation (3.4) gives the exact power consumption for any vector pair applied to the inputs of any combinational module with k inputs. Furthermore, coefficients in the equation are unique for given module.*

3.1.2 Relevant input correlations

It is obvious that $a_0 = 0$ since power consumption for vector pair (00…0)→(00…0) must be zero. All other coefficients in equation (3.4) can be uniquely determined from circuit-level simulation on some specific vector pairs.

Definition Inputs $i_1, i_2, \ldots, i_j$ are *transitive fanout correlated* iff their transitive fanout cones in the circuit have at least one common node, that is, there exists at least one node of the module whose logic function includes all these inputs. *j* is called the *order* of the correlation.

For sake of simplicity, we use "correlation" to mean "transitive fanout correlation" in the rest of this paper.

The coefficients in (3.4) essentially reflect the correlation between the corresponding (joint) transition probabilities and the power consumption in a circuit.

Proposition 1 If $i_1, i_2, \ldots, i_j$ are not correlated, all entries of $\vec{a}_{i_1,i_2,\cdots,i_j}$ are zero.

Corollary *If J is the highest order of correlation among inputs of a module, the first J+1 terms of equation (3.4) are sufficient to model the exact power for any input vector pair applied to the module.*

3.1.3 The macro-model equation

We have empirically observed that, on average, low order joint transition variables have higher coefficient values for most circuits. Based on this observation, we approximate (3.4) by ignoring the high order terms. Our first approximation function is written as:

$$\begin{aligned} P = a_0 &+ \sum_{i=1}^{k}\vec{t}_i \cdot \begin{bmatrix} a_i^{0\to1} \\ a_i^{1\to0} \\ a_i^{1\to1} \end{bmatrix} + \sum_{i=1}^{k}\sum_{j=i+1}^{k}\vec{t}_i \otimes \vec{t}_j \cdot \begin{bmatrix} a_{i,j}^{0\to1,0\to1} \\ a_{i,j}^{0\to1,1\to0} \\ \vdots \\ a_{i,j}^{1\to1,1\to1} \end{bmatrix} \\ &+ \sum_{i=1}^{k}\sum_{j=i+1}^{k}\sum_{l=j+1}^{k}\vec{t}_i \otimes \vec{t}_j \otimes \vec{t}_l \cdot \begin{bmatrix} a_{i,j,l}^{0\to1,0\to1,0\to1} \\ a_{i,j,l}^{0\to1,0\to1,1\to0} \\ \vdots \\ a_{i,j,l}^{1\to1,1\to1,1\to1} \end{bmatrix} + \varepsilon \end{aligned} \tag{3.5}$$

$$\begin{aligned} &= a_0 + \sum_{i=1}^{k}\vec{t}_i \cdot \vec{a}_i^{\,\mathrm{T}} + \sum_{i=1}^{k}\sum_{j=i+1}^{k}\vec{t}_i \otimes \vec{t}_j \cdot \vec{a}_{i,j}^{\,\mathrm{T}} \\ &+ \sum_{i=1}^{k}\sum_{j=i+1}^{k}\sum_{l=j+1}^{k}\vec{t}_i \otimes \vec{t}_j \otimes \vec{t}_l \cdot \vec{a}_{i,j,l}^{\,\mathrm{T}} + \varepsilon \end{aligned}$$

where ε is the error caused by approximation.

We can minimize error ε by re-computing the coefficient values doing least-square fitting for (3.5). However (3.5) is too large to be our macro-model equation because the number of variables in it is $3\cdot k + 9\cdot C_k^2 + 27\cdot C_k^3$, which is too high! The use of 0-1 variables in (3.5) makes it difficult to reduce the number of variables using regression approach.

We thus use a variable partitioning approach which offers two advantages: 1) uses integer variables, 2) has constant number of variables independent of *k*.

We define G_1 as the set of all inputs, G_2 as the set of all possible combinations of two inputs, G_3 as the set of all possible combinations of three inputs:

$$G_1 = \{1,2,\cdots,k\},$$
$$G_2 = \{(1,2),(1,3),\cdots,(1,k),(2,3),\cdots,(k-1,k)\},$$
$$G_3 = \{(1,2,3),(1,2,4),\cdots,(1,2,k),(1,3,4),\cdots,(k-2,k-1,k)\}$$

Note that G_1 consists of indices for order 1 transition variables; G_2 consists of indices for order 2 transition variables, etc. The variable partitioning technique divides G_1 into *L* subsets, G_2 into *M* subsets, and G_3 into *N* subsets such that:

$$\bigcap_{g=1}^{L} G_{1,g} = \Phi,\ \bigcup_{g=1}^{L} G_{1,g} = G_1,\quad \bigcap_{g=1}^{M} G_{2,g} = \Phi,\ \bigcup_{g=1}^{M} G_{2,g} = G_2,$$
$$\left|G_{1,g}\right| \le K \qquad \left|G_{2,g}\right| \le K$$
$$\bigcap_{g=1}^{N} G_{3,g} = \Phi,\ \bigcup_{g=1}^{N} G_{3,g} = G_3,\quad \left|G_{3,g}\right| \le K$$

where *K* is some user-specified bound. The size constraint is specified to manage the complexity of macro-model equation characterization and evaluation.

We approximate equation (3.5) by assuming that:

$$\begin{bmatrix} a_i^{0\to1} \\ a_i^{1\to0} \\ a_i^{1\to1} \end{bmatrix}_{3\times1} \equiv \begin{bmatrix} b_{1,g}^{0\to1} \\ b_{1,g}^{1\to0} \\ b_{1,g}^{1\to1} \end{bmatrix}_{3\times1} \quad \forall i \in G_{1,g}, g = 1,2,\cdots,L$$

$$\begin{bmatrix} a_{i,j}^{0\to1,0\to1} \\ a_{i,j}^{0\to1,1\to0} \\ \vdots \\ a_{i,j}^{1\to1,1\to1} \end{bmatrix}_{9\times1} \equiv \begin{bmatrix} b_{2,g}^{0\to1,0\to1} \\ b_{2,g}^{0\to1,1\to0} \\ \vdots \\ b_{2,g}^{0\to1,1\to1} \end{bmatrix}_{9\times1} \quad \forall (i,j) \in G_{2,g}, g = 1,2,\cdots,M$$

$$\begin{bmatrix} a_{i,j,l}^{0\to1,0\to1,0\to1} \\ a_{i,j,l}^{0\to1,0\to1,1\to0} \\ \vdots \\ a_{i,j,l}^{1\to1,1\to1,1\to1} \end{bmatrix}_{27\times1} \equiv \begin{bmatrix} b_{3,g}^{0\to1,0\to1,0\to1} \\ b_{3,g}^{0\to1,0\to1,1\to0} \\ \vdots \\ b_{3,g}^{0\to1,1\to1,1\to1} \end{bmatrix}_{27\times1} \quad \forall (i,j,l) \in G_{3,g}, g = 1,2,\cdots,N$$

To minimize the error introduced by the above approximation, we should do a careful variable partitioning because variables may have very different coefficients. In our approach, the partitioning criteria are based on the coefficients in (3.4) which are computed as:

$c_i = \frac{1}{3}(a_i^{0\to1} + a_i^{1\to0} + a_i^{1\to1}),\quad i = 1,2,\cdots,k$ for grouping single inputs;

$c_{i,j} = \frac{1}{9}(a_{i,j}^{0\to1,0\to1} + a_{i,j}^{0\to1,1\to0} + \cdots + a_{i,j}^{1\to1,1\to1}),\quad i,j = 1,2,\cdots,k,\ i<j$ for grouping pairs of inputs

$c_{i,j,l} = \frac{1}{27}(a_{i,j,l}^{0\to1,0\to1,0\to1} + a_{i,j,l}^{0\to1,0\to1,1\to0} + \cdots + a_{i,j,l}^{1\to1,1\to1,1\to1}),\ i,j,l = 1,2,\cdots,k,\ i<j<l$ for grouping triplets of inputs

Transition variables of the same order are sorted in increasing order of the corresponding criteria values and then divided into groups of at most *K* elements.

Let's define:

$T_{1,g}^{i\to j}$ is the total number of transitions of type $i\to j$ in group $G_{1,g}$.

$T_{2,g}^{i\to j,k\to l}$ is the total number of pair-wise joint transitions of type ($i\to j$ $k\to l$) in group $G_{2,g}$.

$T_{3,g}^{i\to j,k\to l,m\to n}$ is the total number of joint transitions of type ($i\to j$, $k\to l$, $m\to n$) in group $G_{3,g}$.

$$\vec{T}_{1,g} = \sum_{i\in G_{1,g}} \vec{t}_i = \begin{bmatrix} T_{1,g}^{0\to1} & T_{1,g}^{1\to0} & T_{1,g}^{1\to1} \end{bmatrix}_{1\times3}$$

$$\vec{T}_{2,g} = \sum_{(i,j)\in G_{2,g}} \vec{t}_i \otimes \vec{t}_j = \begin{bmatrix} T_{2,g}^{0\to1,0\to1} & T_{2,g}^{0\to1,1\to0} & \cdots & T_{2,g}^{1\to1,1\to1} \end{bmatrix}_{1\times9}$$

$$\vec{T}_{3,g} = \sum_{(i,j,l)\in G_{3,g}} \vec{t}_i \otimes \vec{t}_j \otimes \vec{t}_l = \begin{bmatrix} T_{3,g}^{0\to1,0\to1,0\to1} & T_{3,g}^{0\to1,0\to1,1\to0} & \cdots & T_{3,g}^{1\to1,1\to1,1\to1} \end{bmatrix}_{1\times27}$$

We can thus introduce our cycle-accurate macro-model as follows:

$$P = b_0 + \sum_{g=1}^{L} \begin{bmatrix} T_{1,g}^{0\to1} & T_{1,g}^{1\to0} & T_{1,g}^{1\to1} \end{bmatrix} \cdot \begin{bmatrix} b_{1,g}^{0\to1} \\ b_{1,g}^{1\to0} \\ b_{1,g}^{1\to1} \end{bmatrix}$$
$$+ \sum_{g=1}^{M} \begin{bmatrix} T_{2,g}^{0\to1,0\to1} & T_{2,g}^{0\to1,1\to0} & \cdots & T_{2,g}^{1\to1,1\to1} \end{bmatrix} \cdot \begin{bmatrix} b_{2,g}^{0\to1,0\to1} \\ b_{2,g}^{0\to0,1\to0} \\ \vdots \\ b_{2,g}^{1\to1,1\to1} \end{bmatrix}$$
$$+ \sum_{g=1}^{N} \begin{bmatrix} T_{3,g}^{0\to1,0\to1,0\to1} & T_{3,g}^{0\to1,0\to1,1\to0} & \cdots & T_{3,g}^{1\to1,1\to1,1\to1} \end{bmatrix} \cdot \begin{bmatrix} b_{3,g}^{0\to1,0\to1,0\to1} \\ b_{3,g}^{0\to0,0\to1,1\to0} \\ \vdots \\ b_{3,g}^{1\to1,1\to1,1\to1} \end{bmatrix} \quad (3.6)$$

In terms of L,M,N values, the number of variables in the macro-model is $3L+9M+27N$, which is independent of the number of circuit inputs k.

Table 1 shows the experimental results for three macro-models using different number of groups and using different grouping strategies. For Macro-model 1, L = 1, M = 1, N = 1; For Macro-model 2, L = 8, M = 8, N = 2, and the single inputs, input pairs, and input triplets are grouped randomly; For Macro-model 3, L = 8, M = 8, N = 2, and our variable partitioning heuristic is used. The input sequence are randomly generated. We did not include biased sequence into this experiment because the R values are so high that it is difficult to assess the merit of each method.

From Table 3, we can draw the following conclusions:

- In general, input grouping improves the quality of macro-models.
- A good input grouping technique is very important to obtain a high quality macro-model.

Table 1 Experimental results of input partition

Module	Macro-model 1		Macro-model 2		Macro-model 3	
	R	ECP (%)	*R*	ECP (%)	*R*	ECP (%)
C1355	0.7037	8.07	0.6041	9.2	0.7788	7.8
C1908	0.5387	15.4	0.5456	15.0	0.7987	11.2
C2670	0.3940	11.7	0.4022	11.6	0.6225	14
C3540	0.5439	17.5	0.6583	15.5	0.7599	12.2
C432	0.3169	29.1	0.3260	29.0	0.7706	20.2
C5315	0.4128	9.9	0.4813	9.4	0.8012	8.1
C6288	0.7318	8.1	0.7717	7.6	0.8011	6.8
C7552	0.1852	33.0	0.3176	31.0	0.9184	9.2
C880	0.5421	19.8	0.4854	20.8	0.6976	16.3
Mul16	0.7568	8.9	0.7813	8.3	0.8139	7.0
Adder16	0.7151	8.6	0.7268	8.4	0.8687	6.2

3.2 Population Stratification

From our experiments we have found that the regression correlation R between the estimated power and the actual power varies for different power ranges. This means that the regression model is not strictly linear over the range of possible power values. The reason for the lack of linearity is that the macro-model equation is only an approximation to the power-transition function. During the variable selection, we discard the high order terms in the power-transition function and group subsets of variables of given order together. The approximation introduces some non-linearity into the macro-model equation. This effect is more pronounced when the number of variables is small.

To improve the quality of our macro-model, we refine the macro-model to a *piece-wise linear regression model* [6]. At the first step, we stratify the training set into several disjoint subsets (strata) based on the switching activity of the vector pairs in the training set. A vector pair will fall into one and only one of these strata. Then the macro-model is trained separately for each subset of the training set. When we apply this piece-wise linear macro-model to estimate the power for a given vector pair, we first examine the switching activity range of the vector pair, and then invoke the macro-model equation which was trained using vector pairs with a similar switching activity.

Theorem 2 The regression coefficient R of the macro-model in population stratification approach is always larger than or equal to that one without population stratification, i.e.,

$$R_{str} \geq R_{nostr}$$

Experimental results in Table 2 shows the improvement on the regression coefficient R of the macro-model with the population stratification approach (Macro-model 1) and without it (Macro-model 2). The experiment sequence contains both biased and random vectors.

Table 2 Experimental results of population stratification approach

Module	Macro-model 1		Macro-model 2	
	R	ECP (%)	*R*	ECP (%)
C1355	0.9806	7.86	0.9691	8.76
C1908	0.9603	9.34	0.9507	11.19
C2670	0.9786	8.77	0.9747	10.22
C3540	0.9743	11.45	0.9566	12.88
C432	0.9196	19.07	0.9001	22.96
C5315	0.9819	7.64	0.9812	8.72
C6288	0.9892	6.03	0.9864	7.16
C7552	0.9885	6.58	0.9871	7.36
C880	0.9506	14.19	0.9509	15.32
Mul16	0.9832	6.32	0.9819	6.90
ADDER16	0.9868	5.64	0.9725	6.73

3.3 Variable Reduction

In the macro-model equation (3.6), the number of variables is about 150. Although the large number of variables will improve the quality of the macro-model, we cannot afford to evaluate a large macro-model equation for every clock cycle at RT-level. Therefore, we must reduce the number of variables in the equation.

In our approach, the search method develops a sequence of regression models. At each step, one X variable is added into or deleted from the final macro-model equation. The criterion used for adding or deleting variables is the F^* statistics of the regression theory [6]. The algorithm is described next:

Input of the algorithm: Given are a set of candidate variables { X_1, X_2, …, X_n } which is in the initial macro-model, a training set (values of variables for input vector pair and corresponding Powermill power value), a low threshold t_0 for deleting a variable, a high threshold t_1 for adding a variable, an upper bound of number of variables MAX_{var}, S is the set of selected variables.

Step 0 (Initialization) : Set $S = \Phi$ and C = { X_1, X_2, …, X_n }

Step 1 (Find the first variable) : Fit a one-variable linear regression model for each variable X_i in C. The F^* test for each model is given by:

$$F_i^* = \frac{MSR(X_i)}{MSE(X_i)}, \qquad i = 1,2,\cdots,N$$

Assume that X_j is the variable with the maximum F^* value. If $F_j^* \geq t_1$ then move X_j from C to S and denote it as X_1^*. Otherwise, no macro-model can be found for the given t_1 value (t_1 must be reduced). The algorithm terminates.

Step 2 (Add a variable) : Assume $S = \{ X_1^*, X_2^*, \cdots, X_a^* \}$, for each X_i remaining in C, fit the regression model with a+1 variables $X_1^*, X_2^*, \cdots, X_a^*$ and X_i . For each of them, the partial F test statistics is:

$$F_i^* = \frac{MSR\ (X_i \mid X_1^*, X_2^*, \cdots, X_a^*)}{MSE\ (X_i, X_1^*, X_2^*, \cdots, X_a^*)} = \left(\frac{b_i}{s\{b_i\}}\right)^2$$

where b_i is the estimated value of β_i coefficient and $s\{b_i\}$ is the standard deviation of b_i. Let X_j be the variable with the maximum F_i^* value. If $F_j^* \geq t_1$ then move X_j form C to S and denote it as X_{a+1}^*, increase a by 1, and go to Step 3; Otherwise the algorithm terminates.

Step 3 (delete a variable) : Assume S=$\{ X_1^*, X_2^*, \cdots, X_a^* \}$, and X_a^* is the latest variable added in Step 2. Compute the partial F test statistics for all other variables in S:

$$F_i^* = \frac{MSR\ (X_i^* \mid X_1^*, X_2^*, \cdots, X_{i-1}^*, X_{i+1}^*, \cdots, X_a^*)}{MSE\ (X_i^*, X_1^*, X_2^*, \cdots, X_a^*)} = \left(\frac{b_i}{s\{b_i\}}\right)^2$$

Let X_j^* be the variable with minimum F* value. If $F_j^* < t_0$ then remove X_j^* form S.

Step 4 : Repeat Steps 2 and 3 until one of following three conditions is true:

1. Algorithm terminates in Step 2
2. C=Φ.
3. The number of variables in S equals to MAX_{var}

In our approach, the number of variables in the candidate set is 162 at the beginning. We choose t_0 = t_1 = 10.0, MAX_{var} = 15. For most macro-models, the algorithm terminates at the 3rd condition at step 4 when the number of variables equals to MAX_{var}. Only for one of them the algorithm terminates at step 3 when $F_j^* < t_1$.

3.4 Other issues in macro-model generation

Another important issue in macro-model generation is the design of the training set. In our approach the training set is designed using stratified sampling techniques proposed in [5]. The population (collected or probabilistically generated sequence and the corresponding power values) covers the whole ranges of macro-model variables and actual power values, the training set design technique also ensure that these ranges are covered by the training set. The validation of macro-model is carried out based on the criteria we mentioned in Section II, which are R, ECP, and EAP.

IV. EXPERIMENTAL RESULTS

We have built our cycle-accurate macro-models for several modules, including the ISCAS-89 benchmarks. In our macro-models, we have also used information about transitions on circuit outputs, but only for two of the circuits (C432 and C880) variables related to outputs survive the variable reduction phase.

The experimental setup is as follows. For each circuit, the population size is set to 80,000 vector pairs (constructed by both biased and random sequences). We first simulate each circuit for the entire sequence using Powermill and record the cycle-by-cycle power. Size of the training set is set to 3,000 . The macro-model is then trained using the training set. After the macro-model is built, we apply it to different subsets of the population. These subsets are selected such that their power behaviors are different from that of the training set. Average ECP and EAP are computed by averaging the ECP's and EAP's of all sub-sets. The regression coefficient R is computed based on the fitted results on the entire population. Experimental results for our cycle-accurate macro-models is summarized in Table 3.

Experimental results shows that our macro-model technique are very accurate when estimating power consumption at RT-level. The average ECP and EAP are 10.2% and 2.0%, respectively.

Table 3 Experimental results of cycle-accurate macro-models

Circuit	No. of Var.	R	ECP (%)	EAP (%)
C1355	15	0.9615	9.3	2.7
C1908	15	0.9343	11.6	2.0
C2670	15	0.9744	9.6	2.0
C3540	15	0.9472	12.5	2.0
C432	14	0.8971	19.3	3.1
C5315	15	0.9816	7.8	1.6
C6288	15	0.9902	6.2	1.9
C7552	15	0.9885	6.9	1.1
C880	15	0.9405	14.3	3.2
Mul16	15	0.9853	6.5	1.6
ADDER16	15	0.9825	6.4	1.1

REFERENCES

[1] S. Powell and P. Chau, "Esitmating power dissipation of VLSI signal processing chips: The PFA techniques", *Proceeedings of IEEE Workshop on VLSI Signal Processing IV*, vol. IV, pp.250-259, 1990.

[2] P.Landman and J. Rabaey, "Power esitmation for high-level synthesis", *Proceedings of IEEE European Design Automation Conference*, pp.361-366, Feb, 1993.

[3] D. Liu and C. Svensoon, "Power consumption estimation in CMOS VLSI chips", *IEEE Journal of Solid State Circuits*, vol. 29, pp.663-670, Jun. 1994.

[4] J. Rabaey and P. Landman, "Activity-sensitive architectural power analysis for the control path", *Proceedings of International Symposiom of Low Power-Design*, pp.93-98, Apr. 1995.

[5] Q. Wu, C. Ding, C. Hsieh, and M. Pedram, "Statistical Design of Macro-models For RT-Level Power Evaluation", *Proceedings of the Asia and South Pacific Design Automation Conference*, pp.523-528, Jan. 1997.

[6] J. Neter, W. Wasseman, and M. H. Kutner, *Applied Linear Regrssion Models*, Second Edition, Richard D. Irwin, Inc, 1989.

[7] PowerMill User manual, Release 3.1, EPIC Design Technology, 1994