Analytical Models for RTL Power Estimation of Combinational and Sequential Circuits[†]

Subodh Gupta and Farid N. Najm

ECE Dept. and Coordinated Science Lab.
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801, USA

ECE Department
University of Toronto
Toronto, Ontario M5S-3G4, Canada

Abstract— In this paper, we propose a modeling technique that captures the dependence of the power dissipation of a (combinational or sequential) logic circuit on its input/output signal switching statistics. The resulting *power macromodel* consists of a quadratic or cubic equation in four variables, that can be used to estimate the power consumed in the circuit for any given input/output signal statistics. Given a low-level (typically gate-level) description of the circuit, we describe a characterization process that uses a recursive least squares (RLS) algorithm by which such an equation-based model can be automatically built. This approach has been implemented and models have been built and tested for many combinational and sequential benchmark circuits.

1. Introduction

With the advent of portable and high-density micro-electronic devices, the power dissipation of very large scale integrated (VLSI) circuits has become a critical concern. Modern microprocessors are hot, and their power consumption can exceed 30 or 50 Watts. Due to limited battery life, reliability issues, and packaging/cooling costs, power consumption has become a more critical design concern than speed and area in some applications. In order to avoid problems associated with excessive power consumption, there is a need for CAD tools to help in estimating the power dissipation of VLSI designs.

A number of CAD techniques have been proposed for gate-level power estimation. However, by the time the design has been specified down to the gate level, it may be too late or too expensive to go back and fix high power problems. Hence, in order to avoid costly redesign steps, power estimation tools are required that can estimate the power consumption at a high level of abstraction, such as when the circuit is represented only by Boolean equations, say using a register-transfer-level (RTL) description. This will provide the designer with more flexibility to explore design trade-offs early in the design process, reducing the design cost and time.

In response to this need, a number of high-level power estimation techniques have been recently proposed. Two styles of techniques have been proposed, which we refer to as top-down and bottom-up. In the top-down techniques, a combinational circuit is specified only as a Boolean function, with no information on the circuit structure, number of gates/nodes, etc.. Top-down methods are useful when one is designing a logic block that was not previously designed, so that its internal details are unknown.

In contrast, bottom-up methods [1—8] are useful when one is reusing a previously designed logic block, so that all the internal structural details of the circuit are known. In this case, one develops a *power macromodel* for this block which can be used during high-level power estimation (of the overall system in which this block is used), in order to estimate the power dissipation of this block without performing a more expensive gate-level power estimation on it. Naturally, this is useful only in case one plans to reuse previously designed blocks.

The method in [1] uses the power factor approximation technique, which treats all the circuit input bits as digital "white noise" and due to this assumption can give errors of up to 80% in comparison to gate-level tools. Although [2] gives more accurate result, it has the disadvantage that it treats different modules differently, requiring specialized analytical expressions for the power to be provided by the user. Thus, depending upon the functionality of the module, a different type of macromodel (analytical equation) may have to be used.

The method in [3] characterizes the power dissipation of circuits based on input transitions rather than input statistics. In [4], the authors present a technique to estimate switching activity and power consumption at the RTL for data path and control circuits, in the presence of glitching activity. In [5], the authors present a macromodel for estimating the cycle-by-cycle power at the RTL. They show good accuracy in estimating average and cycle-by-cycle

† This research was supported in part by the National Science Foundation (NSF MIP 97-10235) and by the Semiconductor Research Corporation (SRC 97-DJ-484), with technical mentorship from Texas Instruments and from Intel.

power. The macromodels are dependent on a training vector set, so that the accuracy is compromised if the training set is not similar to the vector set to be applied.

In [6], the authors present a macromodel for estimating the average power based on power sensitivity. In [7], the authors present a macromodeling approach in which the model is a linear function of every input and output switching activity. Moreover, they present an adaptive algorithm based on least-mean-square (LMS) to characterize the model on-line. They characterize the model only at a specific value of probability and activity and hence may not get good accuracy if the statistics of the test vector set are different from those of the characterization vector set.

Most of the approaches discussed above are limited to only combinational circuits. In this paper, we propose a power macromodeling approach for both combinational and sequential circuits that (1) takes into account the effect of the circuit input switching activity and does not treat the circuit inputs as white noise, (2) takes into account input correlation, both spatial and temporal and (3) is based on a single fixed macromodel template which does not depend on the type of circuit being analyzed. Our model is equation-based. Specifically, we construct a quadratic or cubic equation in the following four variables: *average input signal probability* (P_{in}), *average input switching activity* (D_{in}), *average input spatial correlation coefficient* (SC_{in}), and *average output zero delay switching activity* (D_{out}). For a logic node, the switching activity, also called the transition density, is defined as the average number of logic transitions per unit time. The zero delay switching activity refers to the case when the circuit gates are considered to have zero delay, so that only truly required logic transitions (and no hazards or glitches) are observed. From a high-level view, it is reasonable to assume that fast functional simulation will be applied to measure signal switching statistics, so that only the zero delay output activity (and not the real delay output activity) will be computed. The main advantage of our approach is that all types of circuits are treated in the same way, i.e., we do not use different model equation types for different modules. As a result, the method is very easy to use, and requires no user intervention. Indeed, we will present an automatic characterization procedure, based on the method of Recursive Least Squares (RLS), by which the macromodel can be built for a given circuit.

This paper is organized as follows. In section II, we will give some background regarding our original 4-dimensional (4D) table-based macromodel and extend it to sequential circuits. In section III, we discuss the new analytical-equation-based formulation of the 4D model and describe the characterization procedure in section IV. In section V, we give empirical results that show the effectiveness of the model, and we summarize and conclude in section VI.

II. Extension Of 4D Tabular Macromodel To Sequential Circuits

We have previously [8] presented a 4-dimensional (4D) look-up-table (LUT) based macromodel for combinational circuits, which is given by:

$$P_{avg} = f(P_{in}, D_{in}, SC_{in}, D_{out}) \tag{1}$$

Note that while deriving the above model we have assumed no glitching activity at the primary inputs. The variables P_{in}, D_{in} and SC_{in} satisfy the following constraints [8]:

$$\frac{D_{in}}{2} \leq P_{in} \leq 1 - \frac{D_{in}}{2} \tag{2}$$

$$\frac{nP_{in}^2 - P_{in}}{(n-1)} \leq SC_{in} \leq P_{in} \tag{3}$$

where n is the number of primary inputs. Notice that (2) and (3) give only the lower and upper bounds on P_{in} and SC_{in}. The variables P_{in} and SC_{in} can take any value between these bounds but, otherwise, no specific dependence exists between them. Furthermore, while D_{out} is affected by the values of P_{in}, D_{in} and SC_{in}, it is not completely determined by them because its value depends on the logic of the circuit. The detailed justification of the 4D model can be found in [8].

The results for ISCAS-85 circuits, for correlated input vector streams, are summarized in Table I under the column marked "4D(LUT)". The average error in all cases is less than 10%.

The macromodel (1) was restricted to combinational circuits. Upon first consideration, it would seem that primary inputs information is not sufficient to model the power of sequential circuits, and that some information on the state bits would have to be required. However, when the objective is to estimate the *average* power, the fact that the state over a long time period becomes independent of the initial state leads to significant simplification of the problem. Indeed, it was shown in [9], that if the sequence of inputs to a sequential machine is of order k, then a lag-k Markov chain correctly models the input sequence as well as the k-step conditional probabilities of the primary primary inputs *and* internal states.

As a result, if the input signal temporal correlations die down with time, which is a reasonable assumption in practice, then under steady state the state bits distribution is completely determined by the primary inputs distribution, and the statistics of the primary inputs should be sufficient to model the power of a sequential circuits. Indeed, we have experimentally found out that the same 4D macromodel also works for sequential circuits, as will be shown in the experimental results section below.

However, in some cases one or two key vectors may be crucial to setting the circuit "state" or "mode of operation" forever after. These cases will require special case treatment and our modeling approach may need to be modified or extended in such cases.

III. Equation-based Macromodel

The table-based macromodel requires a lot of memory for storing the table and also requires a lot of time (see column under "Time LUT" in Table II.) to build the whole table. We have developed an alternate approach by which we can fit a general non-linear equation to the function $f(\cdot)$ in (1) without user intervention and with much less time than it takes to fill the table. This general equation is fixed and is used as the starting point for all circuits - no user intervention is required in the choice of equation. We will refer to this equation as the *template*. This works because, even though the function $f(\cdot)$ is non-linear, it turns out that in practice it is "not too non-linear" to defy fitting, and a general polynomial template turns out to be sufficient.

For efficiency reasons, one would like to use the lowest order polynomial template that works. One option is the linear function:

$$\hat{P}_{avg} = c_0 + c_1 P_{in} + c_2 D_{in} + c_3 SC_{in} + c_4 D_{out} \quad (4)$$

where the coefficients c_i are unknown and are to be determined during the characterization using regression analysis. To estimate the regression variables c_i, we generated 1000 blocks of correlated vector streams for different values of P_{in}, D_{in}, and SC_{in}, covering a wide range of input statistics. For each block of vectors, Monte Carlo simulation [10] was used to estimate the average power and to compute the value of D_{out}. Using this set of data and the standard linear least squares method, the regression variables c_i were estimated.

To test the accuracy of the fit, we again generated 1000 blocks of correlated input vectors for different values of P_{in}, D_{in}, and SC_{in}. Using Monte Carlo simulation [10], the average power (P_{avg}) and D_{out} were estimated. For the given P_{in}, D_{in}, SC_{in}, and D_{out} values, the average power $\hat{P}_{avg}$ was then found using (4) and the relative error between P_{avg} and $\hat{P}_{avg}$ was computed and is shown in Table I, under the column marked "L". It is evident from the table that a linear function is not good enough for estimating the average power for most of the circuits.

To improve the accuracy, another option is the quadratic function:

$$\begin{aligned}\hat{P}_{avg} &= c_0 + c_1 P_{in} + c_2 D_{in} + c_3 SC_{in} + c_4 D_{out} \\ &+ c_5 P_{in} D_{in} + c_6 P_{in} SC_{in} + c_7 P_{in} D_{out} \\ &+ c_8 D_{in} SC_{in} + c_9 D_{in} D_{out} + c_{10} SC_{in} D_{out} \\ &+ c_{11} P_{in}^2 + c_{12} D_{in}^2 + c_{13} SC_{in}^2 + c_{14} D_{out}^2\end{aligned} \quad (5)$$

Using the same approach as above, the regression variables were estimated and the accuracy of the results was tested and is shown in Table I, under the column marked "Q". It is evident from the table that the quadratic function is better for estimating the average power for most of the circuits except c499 and c1355 for which the average error was very high, above 15%.

We also investigated the general cubic form. Due to space limitations, we are not showing the cubic equation here as it consists of 35 coefficients. Table I shows the average error and maximum error for the case of a cubic, under the column marked "C". It is clear that the improvement in the error is not much for most of the circuits, except for c499 and c1355 for which the quadratic function did not do well. This observation was found to hold in general, that for many circuits the quadratic model is enough, but the cubic model can still be superior in some cases. As a result, we use a hybrid approach by which we start with the quadratic model by default and increase the order of the model to a cubic only if the measured error during characterization is too big. It was observed that the cubic function was the highest order function required by all the ISCAS-85 circuits. Similar experiments were performed on sequential circuits, except that the power was estimated using [11]. It was observed that the quadratic function is sufficient for all the sequential circuits that we tested.

IV. Characterization

In the standard non-iterative linear least squares method the number of data points used for fitting the regression coefficients has to be predefined. For some circuits fewer data points may be sufficient, while for other circuits more data points may be needed. It is difficult to tell before-hand how many data points will be needed for each circuit. One possibility is to use a fixed number of data points irrespective of the circuits, but in this case we may be doing more work than required (i.e., more gate-level power estimation runs) which leads to larger run time. Therefore, we have developed an automatic characterization process using the recursive least squares (RLS) [12] algorithm. RLS is used in adaptive filtering for on-line estimation of filter parameters. The following summary of the

Table 1. Average and maximum error when total power was estimated using different models.

Circuit	Avg.Error				Max.Error			
	4D(LUT)	L	Q	C	4D(LUT)	L	Q	C
c499	5.96%	36.6%	25.64%	8.3%	46.3%	1407.2%	1319.5%	56.06%
c880	6.7%	19.7%	5.9%	5.2%	50.61%	257.16%	48.74%	45.41%
c1355	6.2%	34.7%	18.08%	12.07%	33.56%	993.9%	107.9%	50.78%
c1908	3.85%	11.6%	4.9%	4.12%	31.17%	68.18%	57.74%	34.31%
c432	5.56%	11.01%	4.19%	3.13%	-27.8%	30.45%	36.8%	24.16%
c5315	3.48%	11.4%	3.6%	3.8%	29.03%	89.84%	32.56%	40.42%
c2670	7.8%	20.36%	10.25%	6.25%	37.53%	304.6%	42.25%	39.45%
c3540	7.5%	13.07%	5.86%	3.7%	44.19%	258.6%	50.49%	46.38%
c7552	8.95%	15.5%	8.8%	5.6%	-45.79%	85.8%	50.48%	47.96%
c6288	9.6%	31.9%	9.6%	8.3%	43.15%	257.16%	55.74%	54.6%

A. RLS Algorithm

Let $y = f(x_1, x_2, \ldots, x_p)$ be a real valued function of real variables. We will use **bold font** to denote vector or matrix quantities. Let the (column) vector $\mathbf{x}$ be the vector of the p variables, so that the transpose of $\mathbf{x}$ is the (row) vector $\mathbf{x}^T = [x_1\ x_2\ \cdots\ x_p]$ and we write $y = f(\mathbf{x})$. We are interested in approximating $f(\cdot)$ with a closed form analytical expression $\hat{y} = \mathbf{c}^T\mathbf{u}$, where $\mathbf{c}^T = [c_0\ c_1\ \cdots\ c_{m-1}]$ is a vector of constant coefficients whose values are to be determined and $\mathbf{u}^T = [u_0\ u_1\ \cdots\ u_{m-1}]$, where each u_i is some function of the variables $x_1, x_2, \ldots, x_p$. We would like to find a vector $\mathbf{c}$ so that $\hat{y}$ is a good approximation to y. In our case $y = P_{avg}$, $\mathbf{u}^T = \left[P_{in}^k, D_{in}^k, \cdots, 1\right]$ and $\mathbf{c}^T = [c_{m-1}, \cdots, c_0]$, where k is the order of the analytical function and m is the number of coefficients to be estimated.

To this end, suppose we generate n randomly chosen samples $\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(n)$, from which we also compute the corresponding $y(i)$, for $i = 1, 2, \ldots, n$. Consider the error $e(i) = y(i) - \hat{y}(i)$ and the cumulative error $\zeta(n) = \sum_{i=1}^{n} |e(i)|^2$. One way of finding an appropriate $\mathbf{c}$ is to find one that minimizes the error $\zeta(n)$. Typically, the solution will depend on n, and we denote it by $\hat{\mathbf{c}}(n)$. Finding such a $\hat{\mathbf{c}}(n)$ is the traditional problem of least-squares fitting which is solved using standard linear regression techniques and is given by:

$$\hat{\mathbf{c}}(n) = \mathbf{\Phi}^{-1}(n)\mathbf{z}(n) \tag{6}$$

where $\mathbf{\Phi}(n)$ is a $m \times m$ matrix, and $\mathbf{z}(n)$ is a $m \times 1$ vector, given by:

$$\mathbf{\Phi}(n) = \sum_{i=1}^{n} \mathbf{u}(i)\mathbf{u}^T(i) \quad \text{and} \quad \mathbf{z}(n) = \sum_{i=1}^{n} \mathbf{u}(i)y(i) \tag{7}$$

In order to avoid using a large number of samples (6) should be solved iteratively, but it requires inverting $\mathbf{\Phi}(n)$ every time, a new sample is added. In the RLS algorithm, this problem is overcome by using the *matrix inversion lemma* which leads to an iterative update mechanism for $\mathbf{\Phi}^{-1}(n)$ that does not require any matrix inversions [12], as follows:

1. Initialize $\mathbf{\Phi}^{-1}(0)$ and $\hat{\mathbf{c}}(0)$
2. For $n = 1, 2, \ldots$, until converged, do:
   a. Compute the $m \times 1$ gain vector $\mathbf{k}(n)$ as follows:

$$\mathbf{k}(n) = \frac{\mathbf{\Phi}^{-1}(n-1)\mathbf{u}(n)}{1 + \mathbf{u}^T(n)\mathbf{\Phi}^{-1}(n-1)\mathbf{u}(n)} \tag{8}$$

   b. Update the coefficient vector $\hat{\mathbf{c}}(n) = \hat{\mathbf{c}}(n-1) + \mathbf{k}(n)\left[y(n) - \hat{\mathbf{c}}^T(n-1)\mathbf{u}(n)\right]$
   c. Update the correlation matrix $\mathbf{\Phi}^{-1}(n) = \left[\mathbf{I} - \mathbf{k}(n)\mathbf{u}^T(n)\right]\mathbf{\Phi}^{-1}(n-1)$
3. End.

To use the above algorithm, the initial values $\mathbf{\Phi}(0)$ and $\hat{\mathbf{c}}(0)$ are required. A common method of initialization is to use:

$$\mathbf{\Phi}(0) = \sum_{i=1}^{n_0} \mathbf{u}(i)\mathbf{u}^T(i) \tag{9}$$

$$\hat{\mathbf{c}}(0) = \mathbf{\Phi}^{-1}(0) \sum_{i=1}^{n_0} \mathbf{u}(i)y(i) \tag{10}$$

where n_0 data points have been accumulated before starting the RLS algorithm.

B. Convergence

The RLS algorithm is standard textbook material, which we have applied to the power macromodeling problem. However, the convergence criterion to be used to stop the iterative updates depends on the particular application. In this section, we describe a novel method of stopping the RLS iterations that is useful for power macromodeling.

Since all we care about is the accuracy of the predicted value $\hat{y}$ and how well it approximates y, we need not wait for all the components of the coefficient vector $\hat{\mathbf{c}}(n)$ to converge. Instead, some norm of $\hat{\mathbf{c}}(n)$ may suffice. Instead of using an arbitrary norm (which we have found can require a large number of iterations), a more efficient and more meaningful method of checking convergence, which indirectly monitors some aggregate measure of convergence of $\hat{\mathbf{c}}(n)$, is to do the following. Consider the following relative error terms, for $i = 1, 2, \ldots, n$:

$$r^{(n)}(i) = \frac{|y(i) - \hat{\mathbf{c}}^T(n)\mathbf{u}(i)|}{y(i)} \tag{11}$$

Notice that $r^{(n)}(i)$ represents the relative error for the ith sample using the regression coefficients $\hat{c}(n)$ generated after the nth iteration. If the $\mathbf{x}(i)$ are iid (independent and identically distributed) random vectors, then the $\mathbf{u}(i)$ and $y(i)$ are also iid. For a given fixed n, it also follows that the $r^{(n)}(i)$ are also iid, so that the mean of each $r^{(n)}(i)$ is independent of i, and depends only on n, so that we denote it by:

$$\mu_n = E\left[r^{(n)}(i)\right] \tag{12}$$

where $E[\cdot]$ is the expected value (or mean) operator. If we now define:

$$r_n = \frac{1}{n}\sum_{i=1}^{n} r^{(n)}(i) \tag{13}$$

then it follows, under very general conditions of ergodicity, that:

$$\lim_{n\to\infty} |r_n - \mu_n| = 0 \tag{14}$$

so that, at some point, the *known* (measured) sample mean r_n converges to the *unknown* true mean μ_n. Standard methods of mean estimation in statistics (Monte Carlo Mean Estimation [10]) can be used to check if this convergence has been achieved to within some (user-specified) accuracy, with a certain amount of (user-specified) confidence. Once this has been achieved, we start to use r_n as an estimate of μ_n, which we consider to be a measure of the error of the model. We consider the model to be "good enough" when μ_n is small enough.

In our implementation, we start with the quadratic model and apply RLS while monitoring the convergence of r_n to μ_n. Once that has been achieved, then if μ_n is below some user-specified error threshold $\mathcal{E}$, we stop the algorithm. Otherwise, we continue to iterate while monitoring μ_n and declare convergence if it goes below $\mathcal{E}$. If we reach a point where μ_n has leveled off at some value larger than $\mathcal{E}$, we switch over to the cubic model and re-evaluate the error and continue to iterate if needed. The process is terminated once either $\mu_n < \mathcal{E}$ or if it levels off again at some value larger than $\mathcal{E}$. (If convergence is not achieved with the cubic model, we revert to the table-based approach and construct the LUT for the circuit as described in [8]. This, however, did not happen for any of the circuits that we tested.) The experimental results to be given in the next section are based on a setting of $\mathcal{E} = 10\%$.

V. Results

All the results below are based on an error threshold setting of $\mathcal{E} = 10\%$ (for convergence of (14)), an error tolerance of $\epsilon = 5\%$ and a confidence of $(1-\alpha) = 95\%$ (for the Monte Carlo mean estimation). The execution times are on a SUN Ultra Sparc 1 with 64MB of RAM. For generating the input vectors used during characterization we use a heuristic technique which is explained in [8]. We will describe the generation of input vectors used for testing the macromodels, in order to explain different extreme ways in which the model was tested.

A. Input Vector Generation, for Testing the Model

In order to test our macromodels in extreme ways, we performed 2500 experiments for each circuit. The experiment differ in the *scheme* that was used to generate the input vectors, as follows:

1. In 100 experiments, we applied correlated input vectors resulting from the output of an n-bit

counter. This test case checks the accuracy of the model in the presence of signal correlation at the primary inputs. The 100 experiments differ in the choice of initial state for the counter.

2. In 700 experiments, for given values of D_{in} and P_{in}, half of the input nodes were given high switching activity (d_i) and signal probability (p_i) and the other half low switching activity and signal probability. By "low", we mean that d_i was varied between 0 and 0.3 (keeping p_i between its feasibility bounds). By "high", we mean that d_i was varied between 0.7 and 1.0 (keeping p_i between its feasibility bounds). The vectors were then randomly generated for the chosen input signal probability (p_i) and switching activity (d_i) values. The values of D_{in} and P_{in} used were different in each experiment. This case tests the sensitivity of power to specific values of the statistics at the primary inputs.
3. In another 700 experiments, for given values of P_{in} and D_{in}, p_i and d_i at every primary input were randomly assigned. The input vectors were then generated randomly for the given p_i and d_i values. The values of D_{in} and P_{in} used were different in each experiment. Here, we test how the model behaves when the inputs have any arbitrary assignment of p_i and d_i.
4. Finally, in 1000 experiments, we randomly chose a number of primary inputs to be kept constant at 0 or 1. At the remaining inputs, p_i and d_i were randomly assigned and the vectors were generated randomly. This test case checks the model for biased input vectors.

It can be seen that the test vectors generated by the above procedure stress the model heavily. Also, the vectors are different in character from our characterization vectors [8]. In each experiment, the number of total vectors generated was not fixed a priori - it was determined by the Monte Carlo power estimation techniques that we used, as described below.

B. Results for Combinational and Sequential Circuits

In this section, we report the results of our equation-based power macromodeling approach on the ISCAS-85, MCNC and ISCAS-89 circuits. The ISCAS-85 and MCNC circuits are combinational, and the ISCAS-89 circuits are sequential. We have implemented this approach and built the power macromodels for a number of combinational and sequential circuits. First, we will discuss the accuracy for the case of combinational circuits, and then for sequential circuits.

The macromodels were constructed by using the characterization process explained in section 4 and the vectors generated using the heuristic algorithm [8]. For testing the macromodels, we performed 2500 experiments, varying the vector generation scheme as explained in section 5.A, and accurate power estimation was performed using Monte Carlo simulation [10], based on a gate-level simulation with a scalable-delay gate timing model. The Monte Carlo simulation also provides accurate estimation of D_{out}. The power values predicted by the macromodel were compared to those from the gate-level Monte Carlo simulation, and the results are shown in Fig. 1. The fit is good and shows that it is indeed possible to do high-level power modeling across the whole range of input switching statistics.

For a more detailed comparison, three different error measures were computed for every circuit: the average absolute value of the relative error, the maximum absolute value of the relative error, and the standard deviation of the absolute value of the relative error. The results are summarized in Table II for ISCAS-85 circuits which shows for each circuit, the number of inputs (#I), outputs (#O), and gates (#G), the model finally used by RLS ("Q" means quadratic and "C" means cubic), the time taken by RLS to build the model, the time taken to build the LUT, the different error measures, and the number of RLS iterations (#RLS) required to build the model. It is clear that the time taken by the RLS method is 3–5 times less than the LUT approach. Moreover, the quadratic model suffices for most circuits. Also the average error for most of the circuits is less than 20%. Even though the maximum error is high, it was observed that the number of sample points with those error values was very small, as can be seen from the small values of standard deviation and from the good fit in Fig. 1. The results for MCNC circuits are shown in the Table III. It can be seen that this technique gives good results for these circuits as well.

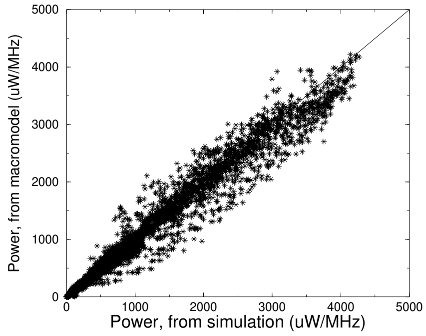

Figure 1. Power comparisons for the ISCAS-85 and MCNC circuits.

Similar experiments were carried out for the ISCAS-89 sequential circuits, except that the average power in this case was estimated using [11]. The comparison with gate level power estimation is shown in Fig. 2. Here again, the fit is seen to be very good, even better than in the combinational circuits case. Table IV reports the detailed results and contains the same columns as table II, except that the column marked #FF gives the number of flip-flops in each circuit. The quadratic model turned out to be sufficient for all these circuits. The average error is less than 15% for most circuits, and the standard deviation of the error is very small.

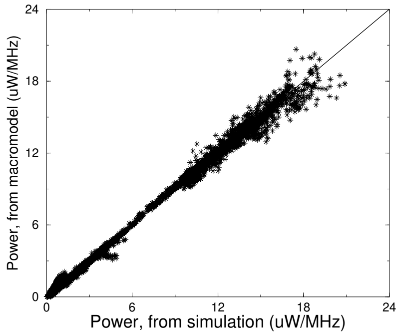

Figure 2. Power comparisons for the ISCAS-89

Table II. Accuracy of the equation-based macromodel on the ISCAS-85 circuits.

Circuit	#I	#O	#G	Model	Time		Error		Std. Dev.	#RLS
					RLS	LUT	Avg.	Max.		
c499	41	32	202	C	16.31min	2.3hr	18.11%	87.20%	15.17%	258
c880	60	26	383	Q	29.10min	6.24hr	10.4%	65.22%	9.78%	162
c1355	41	32	546	C	22.2min	2.09hr	16.54%	92.74%	13.56%	230
c1908	33	25	880	Q	36.14min	4.59hr	8.98%	74.11%	8.76%	120
c432	36	7	160	Q	22.52min	11.79hr	4.8%	34.42%	4.41%	163
c5315	178	123	2307	Q	3.74hr	16.2hr	6.02%	59.41%	5.38%	286
c2670	233	140	1193	C	2.24hr	14.23hr	11.11%	79.5%	11.97%	207
c3540	50	22	1669	Q	5.08hr	21.32hr	7.85%	80.6%	7.38%	228
c7552	207	108	3512	Q	19.75hr	58.4hr	12.03%	81.1%	11.05%	601
c6288	32	32	2406	C	9.23hr	34.4hr	15.4%	88.8%	13.4%	286

Finally, we checked the accuracy of our approach in a case when one input signal has a much stronger influence on the power than all the rest. To do this, we considered an 8×8 bit multiplier and added a control input to it. If the control input is '1', multiplication is performed, otherwise the output is 0. It is clear that the probability and switching activity of the control input have minimal effect on P_{in} and D_{in}, but the power is highly influenced by its probability and activity. The power macromodel was constructed using the characterization process explained above. For testing the model, we varied the probability and activity of the control input from 0 to 1 and the vectors for the remaining inputs were generated using the procedure explained above. The results are shown in Fig. 4. The average error, maximum error and standard deviation in estimating power were found to be 18.09%, 84.88%, and 13.53%, respectively. Even though the maximum error is high, the number of sample points with those error values is very small, as can be seen from the small value of standard deviation. In more general cases, where more complicated control schemes may be employed, it is possible that more involved modeling may be useful, perhaps making use of our 4-variable macromodel as a building block.

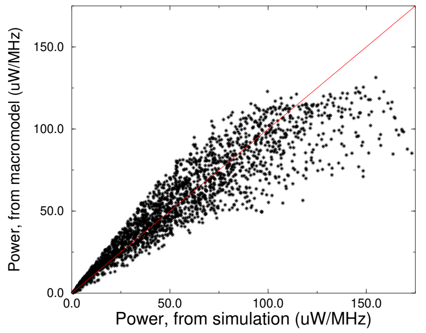

Figure 3. Power comparisons for 8×8 bit multiplier with control input.

Table III. Accuracy of the equation-based macromodel on the MCNC circuits.

Circuit	#I	#O	#G	Model	Time	Error		Std. Dev.	#RLS
						Avg.	Max.		
alu2	10	4	368	Q	59.26min	6.23%	51.47%	7.63%	260
vda	17	27	341	Q	39.43min	6.7%	63.18%	7.64%	248
random8	8	1	158	C	12.37min	10.8%	75.21%	14.33%	150
frg2	152	109	451	Q	1.14hr	5.96%	62.33%	5.25%	272
random10	10	1	487	Q	1.39hr	4.23%	28.69%	3.97%	286
apex6	135	85	775	Q	1.72hr	9.66%	76.47%	11.72%	253
vdx3	135	89	792	Q	1.81hr	6.85%	67.17%	7.61%	266
x3	136	89	745	Q	1.48hr	5.4%	60.56%	5.6%	242
alu4	14	6	604	Q	1.26hr	8.5%	45.48%	7.17%	296
i8	133	81	869	Q	2.53hr	7.47%	51.73%	7.90%	310

VI. Conclusion

Since gate-level power estimation can be time-consuming and because power estimation from a high level of abstraction is desirable so as to reduce design time and cost, we have proposed an equation-based power macromodeling approach for combinational and sequential circuits. Our macromodel consists of an analytical function with four variables: average input signal probability, average input switching activity, average input spatial correlation coefficient, and average output (zero-delay) switching activity. We also presented a Recursive Least Squares (RLS) algorithm by which such an analytical expression can be generated. The proposed model works for all possible signal switching statistics and no user intervention is needed for the model characterization. The only thing that the user has to specify is how much accuracy is desired. The macromodel has been built and tested for many combinational and sequential benchmark circuits.

References

[1] S. R. Powell and P. M. Chau, " Estimating Power Dissipation of VLSI signal Processing Chips: The PFA technique," *VLSI Signal Processing IV*, pp. 250-259, 1990.

[2] P. E. Landman and J. M. Rabaey, "Architectural Power Analysis: The Dual Bit Type Method," *IEEE Transactions on VLSI*, vol. 3 pp. 173-187 June 1995.

[3] H. Mehta, R. M. Owens and M. J. Irwin, "Energy Characterization based on Clustering," *33rd ACM/IEEE Design Automation Conference*, pp. 702-707, June 1996.

[4] A. Raghunathan, S. Dey and N. K. Jha, "Register-Transfer Level Estimation Techniques for Switching Activity and Power Consumption," *IEEE International Conference on Computer-Aided Design*, pp. 158-165, November 1996.

[5] Q. Qiu, Q. Wu, Chih-S. Ding, and M. Pedram, "Cycle-accurate macro-models for RT-level power analysis," *Proc. International Symposium on Low Power Electronics and Design*, pp. 125–130, 1997.

[6] Z. Chen and K. Roy, "A Power Macromodeling Technique based on Power Sensitivity," *35th ACM/IEEE Design Automation Conference*, pp. 678-683, June 1998.

[7] A. Bogliolo, L. Benini, and G. D. Micheli, "Adaptive Least Mean Square Behavioral Power Modeling," *European Design and Test Conference*, pp. 404-410, 1997.

[8] S. Gupta and F. N. Najm, "Power Macromodeling for High Level Power Estimation," *IEEE Transactions on VLSI Systems*, vol. 8, no. 1, pp. 18-29, 2000.

[9] D. Marculescu, R. Marculescu, and M. Pedram,

"Sequence Compaction for Probabilistic Analysis of Finite-State Machines," *34th ACM/IEEE Design Automation Conference*, pp. 12-15, June 1997.

[10] M. Xakellis and F. N. Najm, "Statistical Estimation of the Switching Activity in Digital Circuits," *31st ACM/IEEE Design Automation Conference*, pp. 728-733, June 1994.

[11] J. Kozhaya and F. N. Najm, "Accurate power estimation for large sequential circuits," *IEEE International Conference on Computer-Aided Design*, pp. 448-493, November 1997.

[12] L. Ljung, *System Identification: theory for the user.* Engelwood Cliffs, NJ: Prentice Hall, 1987.

Table IV. Accuracy of the equation-based macromodel on the ISCAS-89 circuits.

Circuit	#I	#O	#G	#FF	Model	Time	Error		Std. Dev.	#RLS
							Avg.	Max.		
s349	9	11	161	15	Q	22.36min	13.4%	78.7%	14.7%	201
s344	9	11	160	15	Q	19.87min	10.7%	67.4%	12.2%	120
s400	3	6	164	21	Q	25.8min	7.67%	62.9%	8.03%	151
s713	35	23	393	19	Q	49.88min	9.23%	68.3%	8.21%	129
s832	18	19	287	6	Q	1.85hr	7.04%	72.53%	8.82%	185
s526	3	6	193	21	Q	31.3min	5.24%	42.95%	5.95%	171
s1494	8	19	647	6	Q	1.56hr	6.25%	62.38%	6.88%	189
s1488	8	19	653	6	Q	33.21min	5.47%	62.87%	5.71%	123
s1423	17	5	657	74	Q	1.09hr	12.5%	79.6%	10.38%	160
s386	7	7	159	6	Q	28.43min	4.1%	39.72%	4.1%	235
s382	3	6	158	21	Q	28.64min	3.23%	29.52%	3.7%	160
s420	18	1	218	16	Q	56.96min	1.47%	13.47%	1.23%	164
s641	35	24	379	19	Q	1.33hr	8.84%	63.98%	7.57%	179
s1196	14	14	529	18	Q	1.69hr	5.9%	65.13%	7.5%	2716
s510	19	7	211	6	Q	1.45hr	0.14%	0.58%	0.12%	147
s953	16	23	395	29	Q	5.5hr	0.38%	2.1%	0.32%	227
s298	3	6	119	14	Q	38.46min	5.17%	32.3%	3.4%	288
s1238	14	14	508	18	Q	25.6min	5.95%	62.85%	8.15%	456
s444	3	6	181	21	Q	22.2min	3.1%	28.15%	3.23%	137
s820	18	19	289	5	Q	1.47hr	5.6%	65.66%	5.96%	141
s838	34	1	446	32	Q	2.86hr	1.6%	9.67%	1.02%	179
s5378	35	49	2779	179	Q	5.5hr	3.8%	36.2%	1.65%	89
s9234	36	39	5597	211	Q	14.64hr	7.6%	68.5%	8.4%	127