Accurate Power Macro-modeling Techniques for Complex RTL Circuits *

Nachiketh R. Potlapally[†], Anand Raghunathan[‡], Ganesh Lakshminarayana[‡], Michael S. Hsiao[†], Srimat T. Chakradhar[‡]

[†] ECE Dept, Rutgers University, Piscataway, New Jersey
{*nachiket*, *mhsiao*}@ece.rutgers.edu
[‡]C & C Research Labs, NEC USA, Princeton, New Jersey
{*anand*, *ganesh*, *chak*}@ccrl.nj.nec.com

Abstract

This paper presents novel techniques for the cycle-accurate power macro-modeling of complex RTL components. The proposed techniques are based on the observation that RTL components often exhibit significantly different "power behavior" for different parts of the input space, making it difficult for a single conventional macro-model to accurately estimate the power dissipation over the entire input space. We address this problem by identifying and separating the input space into regions that display "similar" power behavior. We refer to these regions as the **power modes** of the component. We then construct separate macro-models for each region, and construct a function that, given the input trace to the component, selects an appropriate power mode (and hence macro-model) for use in each cycle.

The proposed ideas are complementary to, and improve upon, previously proposed techniques for power macro-modeling such as linear regression, table look-up, power sensitivity, *etc.* We present experimental results on several practical complex RTL components, and demonstrate that the proposed techniques result in significant reductions (up to 90 %) in the error of RTL macro-modeling compared to a gate-level power estimator.

I. Introduction

Power dissipation has become a mainstream design metric in the deep sub-micron system-on-chip age, due to the signal integrity and power delivery concerns associated with deep sub-micron technologies. The increasing complexity of system chips has led to the adoption of high-level design methodologies in order to bridge the design productivity gap. High-level power estimation lies at the the confluence of these two trends, and is critical to supporting power budgeting and tradeoffs when designing the system architecture.

The application of high-level power estimators is evolving from simple tasks like relative comparison of alternative designs with respect to their power dissipation, to sophisticated uses like chip-level power grid analysis and design, I-R drop calculation for static timing analysis, and hot-spot sensitive system floorplanning. This pervasive shift has significantly raised the requirements of the estimation accuracy. Many of the latter mentioned applications require absolute accuracy and cycle-by-cycle power estimates that can be correlated with functional and timing information.

*Acknowledgement: This work was supported in part by CCRL, NEC USA.

It is well known that power estimation is more accurate in lower-level power estimators, while higher-level power estimators display greater computational efficiency. Transistor and gate-level power estimation techniques have been well researched [1, 2, 3, 4], and several commercial tools exist that are reasonably mature. While there has also been some research on high-level power estimation techniques [5, 6, 7, 8, 9, 10, 11, 12], their limited accuracy has been one of the major challenges facing their widespread adoption.

High-level power estimators can be classified on the basis of the information they produce (*e.g.*, spatial and temporal resolution of the power report), as well as the techniques employed (*e.g.*, fast synthesis based, analytical, macro-modeling based, *etc.*). Aggregate estimators are those which, given a set of input vectors or sequences, report the *average power* dissipated in the circuit under the application of the complete set of vectors. Applications where power dissipation information needs to be correlated with functional or timing information (*e.g.*, peak power constraints, transient hot-spot analysis, and "power debugging"), require power values on a cycle-by-cycle basis. In this paper, we present a novel technique to address the problem of improving the accuracy of high-level power estimation, in the context of a cycle-accurate macro-modeling based power estimation methodology.

Macro-modeling, which we adopt and improve in this paper, is a commonly used technique for high-level power estimation. It formulates the power consumed (*dependent variable*), in terms of parameters (*independent variables*), that are easily observable at the high level of abstraction. Some examples of power macro-models are linear or non-linear equations and look-up tables. The concept of *cycle-accurate high-level macro-models* was used in [10, 11]. A cycle-accurate macro-model gives the power dissipated in each cycle of operation, *i.e.*, given a set of vectors, it gives the power dissipated by every vector pair applied to the circuit. If P_k is the power consumed by a module in cycle k, it can be defined as

$$P_k \quad = \quad F(V_{k-1}, V_k)$$

where, V_{k-1} and V_k represent the input vectors for module at cycles *k-1* and *k* respectively. Note that the above equation can be extended to account for the dependence of power consumption on any finite history of input values. In practice, the power dissipation of atomic RTL components (functional units, multiplexers/buses, and latches/registers) is amenable to being modeled by the above equation, and more complex blocks can be decomposed into these atomic components for power estimation. The function F is referred to as the *macro-model.* It is parameterized with respect to some variables, which can be derived from the input vector pair. The goal of the power macro-modeling (or characterization) process is to derive F itself. The inputs to the macro-modeling procedure are a set of characterization vectors and the corresponding cycle-by-cycle power consumption values (derived using an accurate lower-level power estimator on a gate- or transistor-level implementation). With minimal additional effort, cycle-accurate macro-models can also provide the average power dissipated (akin to the behavior of aggregate estimators).

This paper presents novel techniques for the cycle-accurate power macro-modeling of complex RTL components. The proposed techniques are based on the observation that RTL components often exhibit significantly different "power behavior" for different parts of the input space. We refer to these regions as the **power modes** of the component, since the relationship between the power dissipated and the parameters (independent variables) varies appreciably from one region to the another. We demonstrate that the use of area and performance optimized RTL components, including multi-functional units, ALUs, and functional units that perform complex operations, results in the presence of multiple power modes. Also the use of low-power design techniques such as operand isolation within the components themselves, causes this phenomenon. In the paper, we use the term "complex circuits", to refer to the class of RTL circuits which exhibit multiple power modes in their operation. Existing techniques for cycle-accurate macro-modeling [10, 11] build a single macro-model to estimate power dissipation across all the power modes of a complex circuit. We show that the large variance in power behavior from one power mode to another, significantly limits the estimation accuracy of the existing macro-models .

We address this problem by adopting the following approach:

1. Given the RTL component, its characterization vectors, and corresponding power data, we partition the input space into its constituent power modes.

2. We construct separate macro-models for each power mode (using existing macro-modeling techniques).

3. We construct a classification function that, given the input trace to the component, selects an appropriate power mode (and hence macro-model) for use in each cycle.

We need step 3 because our proposed technique builds multiple macro-models (one for each power mode) for any given complex circuit. Hence, when applying the power macro-model, for any given input vector, we need to know which one of the macro-models is to be invoked for estimating the power dissipated. Existing macro-modeling techniques do not require this step, since they build a single macro-model for the entire input space.

The proposed techniques, and the resulting improvement obtained, are illustrated in the following section through a motivating example. We also present experimental results on several practical complex RTL components, and demonstrate that the proposed techniques result in significant reductions (up to 90 %) in the error of RTL macro-modeling compared to an accurate gate-level power estimator.

The rest of the paper is organized as follows. Section II gives the motivation for our work. Section III introduces the overall setup, while Section IV describes the individual components of the flow in detail. SectionV presents experimental results that demonstrate the merit of the proposed approach.

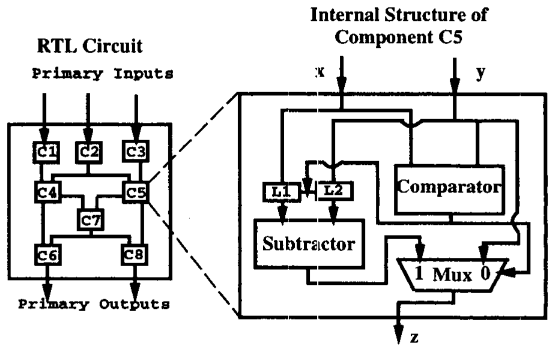


Figure 1: Example of complex circuit usage in a RTL core

II. Motivation

In this section, we provide the motivation for our work through an example. We compare the performances of our proposed macro-modeling technique and the conventional cycle-accurate macro-modeling methods on an example circuit, and show the significant enhancement in power estimation accuracy obtained by the former.

Consider circuit $C5$, which implements part of the greatest common divisor (GCD) algorithm [13]. It has two 8-bit inputs, x and y, and a 8-bit output, z. The behavioral description of the operation of circuit $C5$ is as follows:

```
If $(x > y)$
  $z = x - y$
else
  $z = y$
```

Figure 1 shows an example usage of $C5$. On the left is an RTL circuit, which uses several complex circuits (including $C5$) as basic blocks for implementing its desired functionality (say, RSA security encryption, which needs GCD calculation). The right side shows the magnified high-level view of the internal structure of $C5$. Besides some generic performance optimizations, it uses a low-power design strategy called *operand gating* [1] enforced through the use of latches, $L1$ and $L2$ (see Figure 1). In the cycles where the subtractor is not doing useful computations, *i.e.,* when x is less than or equal to y, the inputs to the subtractor are "frozen", *i.e.*, the latches $L1$ and $L2$ are disabled, thereby preventing the new input values from entering the subtractor. The freezing of the inputs to the subtractor results in power saving, by eliminating unnecessary switching activity in the subtractor.

Initially, we built a conventional cycle-accurate macro-model using a large set of randomly generated vectors known as the *profiling stimuli* [10, 11]. This macro-model is referred to as $macro-model_{conventional}$. In order to study the power consumption behavior of $C5$, we plotted its power profile using the vectors of the profiling stimuli. The profile is a two dimensional graph of the actual power dissipated (Y-axis) as a function of the macro-model power estimate (X-axis). It is shown in Figure 2. For every vector in the profiling stimuli, "actual power dissipated" is obtained from an accurate power simulator, and the "macro-model power estimate" is obtained by using the $macro-model_{conventional}$. The resulting plot unambiguously shows the presence of two homogeneous regions, *i.e.,* clusters, each having a distinct power dissipation characteristic. We can consider a cluster in the power profile to be the visual manifestation of a power mode of the circuit. Figure 2 indicates that $C5$ has two power modes. The upper and the

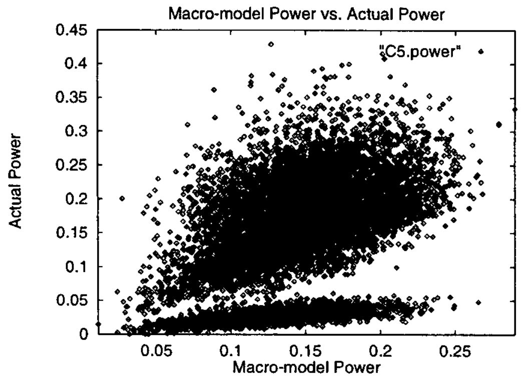


Figure 2: Two modes of operation of C5

lower clusters (in Figure 2) are said to be representative of "power mode 1" and "power mode 2", respectively.

We then applied the proposed cycle-accurate macro-modeling technique to circuit *C5*. We used an automated algorithm to separate the points (corresponding to the vectors of the profiling stimuli) in the power profile (Fig. 2) into non-overlapping sets, where each set contains points belonging to a distinct cluster (power mode) of the plot (circuit). Since there are two clusters in Fig. 2, we have two sets of vectors. Then, using the conventional approach, we built a unique cycle-accurate macro-model for each of the vector sets obtained. The two macro-models built by our approach are referred to as $macro-model_{proposed}$. This scenario is illustrated in Figure. 3. The plot of the left shows the *conventional macro-modeling approach*. The line denoted by *M* is the macro-model built. This technique does not distinguish between the two power modes and builds one macro-model (*M*) for the entire input space. This conceptual behavior is illustrated by the broken line (in the left plot), which combines the distinct clusters into one region. The plot on the right shows the proposed macro-modeling approach. Separate macro-models, *M*1 and *M*2, are built for power mode 1 and power mode 2 respectively. The macro-models (*M*, *M*1, *M*2) shown in figure 3 are linear functions of the macro-modeling variables, but in general, any macro-modeling technique including non-linear functions, table-lookup, *etc.* can be used. Finally, we need a function, which will help us decide which one of the two macro-models ($macro-model_{proposed}$) to use for any given input. We call this function as the *power mode identification function* (PIF). By examining the points in the clusters (in Figure 2), we make the following observations:

1. A very large percentage (>98 %) of the points in the upper cluster (further of the two clusters from the X-axis in Figure 2) of the plot satisfy the condition **x is greater than y**.
2. All the points in the lower cluster (which is nearer to the X-axis in Figure 2) of the plot satisfy the condition **x is less than or equal to y**.

We attribute the occurrence of a cluster (synonymous with power mode) to its *defining condition*[1]. Based on the above observations, we conclude the following about the power modes in the operation of *C5*:

- Power mode 1: The defining condition of this mode is **x is greater than y**.

[1]*Defining Condition* of a power mode is a boolean condition, the satisfiability of which causes the circuit to operate in the corresponding power mode

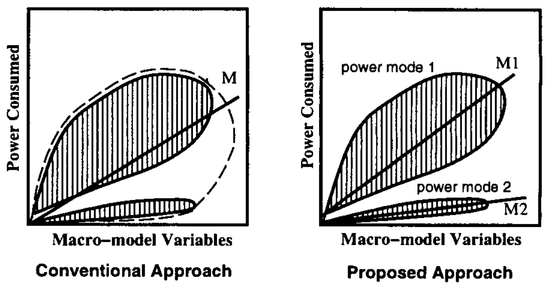


Figure 3: Conventional and Proposed Macro-models

- Power mode 2: The defining condition of this mode is **x is less than or equal to y**.

The boolean conditions representing the defining conditions of the two power modes are combined to obtain the PIF for *C5*.

In order to test the macro-models, we generated a large set of vectors (different from the profiling stimuli) and used the macro-models (conventional and proposed) to estimate the power dissipated by those vectors in *C5*. In the proposed scheme, given an input vector, we use the PIF to identify the power mode to which the input belongs to. Then, we use the macro-model corresponding to the power mode identified to estimate the power dissipated by it. The *Absolute Cycle-by-cycle Error*[2] (ACE) of $macro-model_{conventional}$ was found to be 1.304 and that of $macro-model_{proposed}$ was observed to be 0.508. This shows that $macro-model_{proposed}$ is more accurate that $macro-model_{conventional}$, and improves the estimation accuracy by **61 %**.

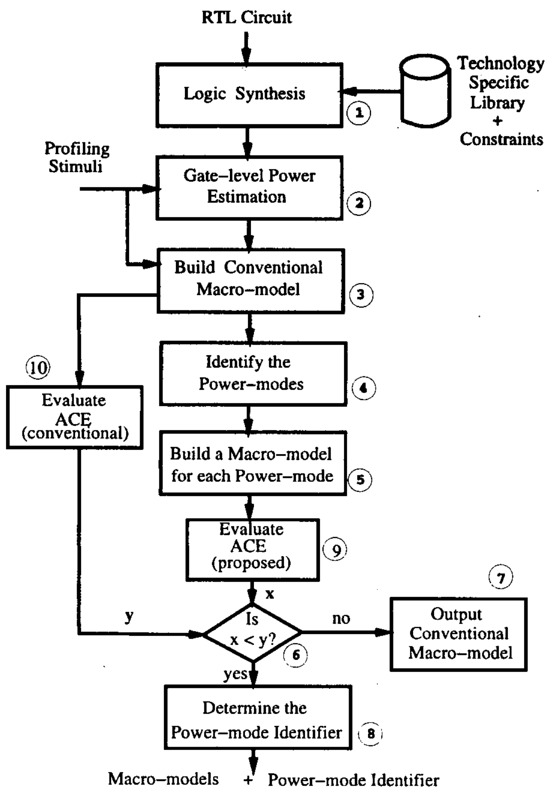
The above example indicates the potential of power mode based macro-modeling in improving estimation accuracy. The following section presents a detailed description of the proposed techniques and algorithms.

III. Overall Flow of the Process

In this section, we present an overview of the proposed cycle-accurate RTL power macro-modeling procedure. The flowchart in Figure 4 shows the high-level view of the proposed technique.

A macro-model is function which parameterizes the power dissipated in terms of some variables which can be derived from the inputs applied. The macro-model can be built using either analytical techniques [7, 8] or characterization-based techniques [9, 10, 11]. Because of its greater accuracy, we use the characterization-based power macro-modeling approach. In this technique, the macro-model is built by using power dissipation information obtained from a lower level implementation. The inputs to the macro-model building stage are a set of vectors and the actual power dissipated by them. We obtain the actual power values, by observing the power dissipated in the gate-level implementation (of the input RTL circuit) by a set of randomly generated vectors known as the *profiling stimuli*. In our experiments, we used an in-house gate-level power estimation tool, which has been calibrated with SPICE and benchmarked within 10 % of it, to measure the power dissipated. These processes are represented by steps 1 and 2. The profiling stimuli and the power values are given as inputs to step 3. The macro-model built in step 3 is referred to as $macro-model_{conventional}$. The accuracy

[2]Absolute Cycle-by-cyle Error (ACE) measures the deviation of the cycle-by-cycle power estimates given by the macro-model from the actual power dissipated (obtained from an accurate low-level power estimator). The formal defination of ACE is given in Section IV. Smaller values of ACE indicate greater is the accuracy of the macro-model estimates.



RTL Circuit
Logic Synthesis
Technology Specific Library + Constraints
Profiling Stimuli
Gate-level Power Estimation
Build Conventional Macro-model
Identify the Power-modes
Evaluate ACE (conventional)
Build a Macro-model for each Power-mode
Evaluate ACE (proposed)
x
y
Is x < y?
no
yes
Output Conventional Macro-model
Determine the Power-mode Identifier
Macro-models + Power-mode Identifier
1
2
3
4
5
6
7
8
9
10

Figure 4: Overview of our RTL power macro-modeling methodology

of the $macro-model_{conventional}$ is determined by the measure known as $ACE_{conventional}$. We compare the estimates given by $macro-model_{conventional}$ against the actual power values to obtain $ACE_{conventional}$ (step 10).

The next operation is to identify the power modes (if any) in the operation of the circuit (step 4). In order to do this, we obtain the power profile of the circuit by plotting the actual power dissipated (Y-axis) versus the macro-model power estimate (X-axis), for all the vectors in the profiling stimuli. The $macro-model_{conventional}$ is used to obtain the "macro-model power estimate" for all the vectors in the profiling stimuli. The presence of clearly demarcated clusters in the resulting plot implies the existence of power modes. Each power mode is represented by a unique cluster, *i.e.*, there is a one-to-one correspondence. In order to separate the points in each cluster, we use an automated process known as the *line sweep algorithm* described in Section IV. In this algorithm, a line of increasing slope sweeps the profile, starting from the X-axis and ending at the Y-axis. In doing so, it outputs information pertaining to the boundaries of each cluster. This information is used to cleave the profiling stimuli into non-overlapping subsets, where each subset contains vectors belonging to a distinct cluster. Using conventional macro-modeling techniques, we build a macro-model for each power-mode by using its corresponding vector set (step 5). The resulting macro-models are referred to as $macro-model_{proposed}$. $ACE_{proposed}$[3] is representative of the accuracy of the estimates given by $macro-model_{proposed}$ (step 9). If $ACE_{proposed}$ is less than $ACE_{conventional}$ (indicating that the proposed technique has better accuracy than the conventional method) (in step 6, **x** denotes $ACE_{proposed}$ and **y** denotes $ACE_{conventional}$), we proceed to the next step of building the PIF (step 8). The proposed macro-modeling scheme terminates by outputting the $macro-model_{proposed}$ and the PIF. In case, in step 6, if x is greater than y, then the macro-modeling process is terminated and $macro-model_{conventional}$ is given as the output (step 7).

In order to determine the PIF, we need to first find out the *defining conditions* of the power modes identified. This is done by the *strength classification algorithm*. The inputs to this algorithm are the clusters identified and the control conditions of the circuit. The control conditions are the obtained from the "If-then-else" statements in the behavioral specification of the complex circuit. For a given cluster, the algorithm assigns the control condition satisfied by the majority of the points in it, as the defining condition of that cluster. After the defining conditions of all the clusters are determined, the boolean conditions representing them are abstracted, to obtain the PIF for the circuit.

Finally, the usage of the proposed macro-model is shown in Figure 5. In the Figure 5, the blocks $M_1, M_2, \ldots, M_n$ indicate the macro-models which make up $macro-models_{proposed}$ of the circuit under estimation (The circuit has n power modes of operation). Thus, given an input, the PIF is used to identify the power mode to which it belongs to. Then, the macro-model corresponding to the power mode output by the PIF is used to estimate the power dissipated by the input.

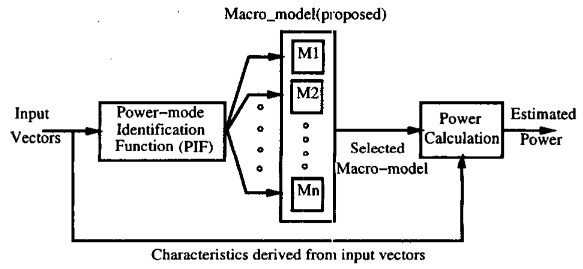


Figure 5: RTL power estimation using the proposed macro-model

IV. Implementation of the individual components

In this section, we focus on the implementation details of the constituent steps of our macro-modeling technique. This section is divided into four subsections: subsection A describes the theory used for building the cycle-accurate macro-models (blocks 1, 2 and 3 in Figure 4), subsection B describes the process of evaluating the accuracy of the conventional and the proposed macro-models (blocks 9 and 10 in Figure 4), subsection C details the identification of the power modes (blocks 4 and 5 in Figure 4) and, finally, subsection D explains the construction of the power mode identification function (block 8 in Figure 4).

A. Building the Cycle-accurate Power Macro-models

The construction of the macro-model (step 3 in Figure 4) can be implemented using a variety of techniques, such as, linear and non-linear statistical techniques, power sensitivities, tree-based regression *etc.* We chose the linear regression based approach [10, 11], since contruction and use of linear regression

[3]$ACE_{proposed}$ is function of the ACEs of the individual macro-models which make up $macro-models_{proposed}$. It is dependent on the ACE and the number of points in each power mode. The formula is provided in Section IV

based macro-models is efficient, and the tools for building linear regression models are widely available [15]. Our technique can be modified with minimal effort to incorporate any other cycle-accurate macromodeling technique.

The linear macro-model describes the power dissipated P, as a linear function of $X_1, X_2, \ldots, X_n$, which are some characteristic variables derived from the inputs. The linear statistical relationship between the power dissipated and the variables can be expressed as,

$$P = \alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \ldots + \alpha_n X_n + \theta$$

The terms $(\alpha_0, \alpha_1, \ldots, \alpha_n)$ are constants known as the *regression coefficients* of the macro-model. θ is a random quantity representative of the error of the fit. For any given input, θ is representative of the deviation in the power estimated by the macro-model from the actual power value dissipated.

B. Evaluating the Cycle-accurate Power Macro-models

The quality of the macro-model built is evaluated by using a quantity known as Absolute Cycle-by-cycle Error (ACE). It is defined as the absolute value of the percentage error of the estimated power with respect to the actual power, summed over all the vectors in the profiling stimuli. This measure is given by:

$$ACE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{P_{estm,i} - Pobs,i}{P_{obs,i}}\right|$$

where $P_{estm,i}$ is the power estimated by the macro-model for vector i, $P_{obs,i}$ is the actual power dissipated by vector i and N is the total number of vectors in the profiling stimuli. The value of ACE tells us how precisely the macro-model estimates the cycle-by-cycle power consumption of the circuit (higher values of ACE indicate lower estimation accuracies). The value of $ACE_{conventional}$ is obtained directly using the above formula. The value of $ACE_{proposed}$, which represents the accuracy of the proposed approach with multiple macro-models, is computed as follows. Consider a complex circuit that has S power modes. If $N_1, N_2, \ldots, N_S$ are the number of vectors in the S power modes and if $ACE_1, ACE_2, \ldots, ACE_S$ denote the ACEs of the corresponding macro-models built for the power modes, then the $ACE_{proposed}$ is defined as:

$$ACE_{proposed} = \frac{N_1 ACE_1 + N_2 ACE_2 + \cdots + N_S ACE_S}{N_1 + N_2 + \cdots + N_S}$$

where

$$N_1 + N_2 + \ldots + N_S = N$$

(N is the total number of vectors in the profiling stimuli). If $ACE_{proposed}$ is less than the $ACE_{conventional}$, it means that the power mode conscious macro-modeling has improved the accuracy of estimation. Otherwise, the $macro-model_{conventional}$ is used for estimation. As borne out by the experimental results presented later, $macro-model_{proposed}$ proves to be a much more accurate estimator than $macro-model_{conventional}$.

C. Power Mode Analysis

The power mode analysis step consists of identifying the power modes (if any) of the circuit and separating the vectors (profiling stimuli) into different sets based on their power modes. Highly accurate clustering information can be obtained by plotting the actual power dissipated against the independent variables (macro-modeling parameters) that are used to build the macro-models. This plot would clearly reveal the dependence of power on input variables and the combinations thereof. If the macro-model is parameterized on N variables, then the power profile would be a $N+1$ dimensional plot (1 is due to the actual power values). However, identifying clusters in $N+1$ dimensions becomes computationally infeasible for even moderate values of N. Therefore, we make an "accuracy-computational efficiency" trade-off by reducing the profile to two dimensions. In order to accomplish this trade-off, we plot the power consumption reported by the gate- or transistor-level estimator against the value estimated by $macro-model_{conventional}$ for each vector pair in the profiling stimuli. For example, the plot for the component $C5$ shown in Figure 1 was shown in Figure 2.

```
LINE SWEEP (Start Threshold $T_{start}$, End Threshold $T_{end}$,
            Step $\Delta$ ) {
  line ← 0;
  flag ← 0;
  num_cluster, start_cluster, end_cluster ← 0;
  structure CLUSTER (num_cluster,start_cluster,end_cluster);
  while (line $\leq 90^o$) {
    line ← line + $\Delta$ ;
    line_popl ← number of points on the line;
    if (line_popl $\geq T_{start}$) and (flag = 0) {
      num_cluster ← num_cluster + 1;
      start_cluster ← line;
      flag ← 1;
    }
    if (line_popl $\leq T_{end}$) {
      end_cluster ← line;
      flag ← 0;
      return CLUSTER;
    }
  }
}
```

Figure 6: Line sweep algorithm

In order to identify the clusters and separate them, we use the *line sweep algorithm*, whose pseudo-code is shown in Figure 6. A "sweep line" detects the clusters by sweeping the power profile plot, starting from the X-axis and stopping when it reaches the Y-axis (the angle between the line and the X-axis is incremented in steps of Δ starting from 0^o until it reaches 90^o). At each step, we compute a quantity called *line_popl*, which is the number of points on the line. When *line_popl* exceeds a user-specified threshold T_{start} and the *flag* is not set, the start of a cluster is detected. The algorithm notes the starting value of the cluster (*start_cluster*) and the cluster number (*num_cluster*). It keeps moving in the detected cluster, until the cluster termination condition is satisfied. The termination condition is satisfied when *line_popl* falls below T_{end}. At termination, the boundary of the cluster is marked (*end_cluster*). The *flag* is reset and the cluster information CLUSTER(*num_clstr, start_clstr, end_clstr*) is returned. Based on the angles *start_cluster* and *end_cluster*, returned by the algorithm, the vectors belonging to the corresponding cluster (*num_cluster*) are identified.

Any points that do not fall into any of the identified clusters at the end of the sweep process are assigned to the cluster nearest to them (based on Euclidean distance). The values of T_{start} and T_{end} are specified by the user, based on the point density in the clusters and the regions separating the clusters, respectively. Note that the algorithm presented above can be easily modified to also vary the

```
STRENGTH CLASSIFICATION (cntrl_cond[1→N],
                         clusters[1→M]) {
  for ($j \leftarrow 1; j \le M; j \leftarrow j+1$) {
    for ($k \leftarrow 1; k \le N; k \leftarrow k+1$) {
      $num\_op_k \leftarrow 0$;
    }
    $num\_vectors_j \leftarrow$ number of points in $cluster_j$;
    for ($i \leftarrow 1; i \le num\_vectors_j; i \leftarrow i+1$) {
      for ($k \leftarrow 1; k \le N; k \leftarrow k+1$) {
        if ($vector_i$ satisfies $cntrl\_cond_k$) {
          $num\_op_k \leftarrow num\_op_k + 1$;
        }
      }
    $defn\_cond_j \leftarrow cntrl\_cond[$max$(num\_op)]$;
    return $defn\_cond_j$;
    }
}
```

Figure 7: Strength classification algorithm

line intercept in addition to the slope.

D. Building the Power Mode Identification Function

The first step in formulating the power mode identification function (PIF) is to identify the defining conditions corresponding to each power mode. As mentioned before, the control conditions, which are the inputs to the algorithm, are obtained from the "If-then-else" statements in the behavioral specification of the given RTL component. We use *strength classification algorithm* (Figure 7) to select one of the control conditions to be used as the defining conditions for each power mode. In effect, the control condition satisfied by the largest number of points in a power mode is designated as the defining condition of the power mode.

The defining conditions are represented as a set of boolean conditions. The satisfaction a defining condition implies that the circuit is operating in the corresponding power mode. We use these boolean conditions to formulate the PIF. Since the classification is associated with the circuit behavior rather than the input statistics, it is robust, as demonstrated by the experimental results presented in Section V.

Since our macro-model is cycle-accurate, the PIF needs to be invoked at every cycle of operation. This implies that the evaluation of PIF has to be fast, inorder to avoid imposing a performance penalty. Therefore, we represent the PIF as a binary tree structure known as the power mode identification tree (PIT). The internal nodes of the PIT are boolean conditions derived from the PIF and the leaf nodes are the power modes. The PIT is traversed using the input vectors as arguments to the boolean conditions at the nodes. Depending on whether the result of a node boolean condition is true or false, the tree will branch right or left, to another node, respectively. This process continues until a leaf node is reached. The power mode denoted by the leaf node is the power mode to which the given input belongs. The macro-model corresponding to that power mode is used to estimate the power dissipated. Since the PIT is an efficient data structure, the performance overhead is minimal. This fact is illustrated in the results section.

In to order to build the PIT, we need to find out the PIF. We illustrate the formulation of the PIF for $C5$ (the example circuit shown in Figure 1). $C5$ has two 8-bit inputs, x and y. Its power profile showed the existence of two power modes. It has two control conditions: *x greater than y* (say, cntrcond-1) and *x less than or equal to y* (say, cntrcond-2). Through strength classification, we find that cntrcond-1 is the defining condition of power mode 1 and cntrcond-2 is that of power mode 2. Let $(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8)$ denote the bit positions of x, with x_1 and x_8 denoting the MSB and LSB respectively. Input y can be defined in a similar manner. The PIF for detecting the power mode 1 of $C5$ is the OR of the following boolean conditions (assuming x and y represent unsigned integers):

- $\{x_1 > y_1\}$
- $\{x_1 == y_1\} \&\& \{x_2 > y_2\}$
- $\ldots$
- $\{x_1 == y_1\} \&\& \{x_2 == y_2\} \&\& \{x_3 == y_3\} \&\& \{x_4 == y_4\} \&\& \{x_5 == y_5\} \&\& \{x_6 == y_6\} \&\& \{x_7 == y_7\} \&\& \{x_8 > y_8\}$

If any given input vector to component $C5$ satisfies at least one of the above boolean equations, then $C5$ is said to operate in power mode 1. In a similar fashion, PIF can be obtained for power mode 2 of $C5$. In the present example, the PIF for power mode 2 is simply the negation of the PIF for power mode 1. The next step consists of translating the PIF into the PIT for $C5$. The resulting PIT is shown in Figure 8.

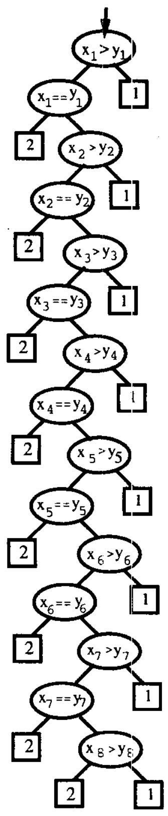


Figure 8: Power mode Identification Tree

In Figure 8, the nodes (ovals) contain the boolean conditions and the leaf nodes (squares) represent the power modes. For example, suppose the inputs given to $C5$ are: $x = \{11010101\}$ and $y = \{01011010\}$. The first node of the PIT (Figure 8) results in a true, thereby the tree will branch to the right. The right child is a leaf node containing 1. This shows that the given inputs belong to power mode 1 of $C5$. Subsequently, the macro-model corresponding to that power mode can be used for estimating the power consumed.

V. Experimental Results

In this section, we present the results of our proposed cycle-accurate macro-modeling technique on different complex RTL components.

Table 1 describes the characteristics of the RTL circuits used in the experiments. Each circuit is a combination of several atomic RTL components. The second and third columns specify the number of inputs (I) and outputs (O) respectively. The fourth column gives the bit-width (BW) of the circuit inputs. The last column specifies the atomic components which constitute the circuit: comparator (*Comp*), 2-to-1 multiplexer (*Mux*), subtracter (*Sub*), adder (*Add*), multiplier (*Mult*), bit-vector AND (*And*), bit-vector OR (*Or*), and *Latches*.

Table 1: Characteristics of the circuits

Circuit	I	O	BW	Components
C5	2	1	8	2 Latches, Comp, Mux, Sub
C6	2	1	8	2 Latches, Comp, Mux, Mult
C7	2	1	8	4 Latches, Comp, Mux, Mult, Add
C8	2	1	8	4 Latches, Comp, Mux, Mult, And
C9	2	1	8	4 Latches, Comp, Mux, Add, Or
C10	2	1	8	4 Latches, Comp, Mux, Or, And
C11	2	1	16	2 Latches, Comp, Mux, Sub

For each of the 7 circuits, we built the macro-models using the proposed techniques, which involve identifying the power modes, building multiple macro-models, and constructing the power mode identification function. For building the macro-models, we used a set of 10000 vectors as the profiling stimuli. For the purpose of comparison, we also built the $macro-model_{conventional}$ for all the circuits. Then, we generated an independent set of 10000 vectors and used it to simulate all the 7 circuits. For each circuit, we measured the following three quantities for each vector: the actual power dissipated (estimated using the NEC OpenCAD cell-based power estimator [16]), the power value estimated by $macro-model_{conventional}$, and the power estimated by $macro-model_{proposed}$. By substituting these three quantities in the formula for *Absolute Cycle-by-cycle Error* (ACE), we computed the estimation error of the $macro-model_{conventional}$ and the $macro-model_{proposed}$ for each circuit. The results of this analysis are shown in Table 2. The first column gives the circuit name. The second column gives the error in the estimates of $macro-model_{conventional}$ (shown as *Error(conv)*). The third column specifies the estimation error of $macro-model_{proposed}$ (shown as *Error(prop)*). The last column gives the improvement in estimation accuracy obtained by our technique (shown as *Imprv*).

The results indicated that our technique significantly improves the accuracy of cycle-by-cycle power estimation compared to conventional macro-models (the improvements were as high as 90.56 %, while the average improvement was 50 %). We also examined the performance penalty incurred by our technique. This overhead is largely due to the invoking of the power mode identification function (PIF) at every cycle of operation. For the purpose of comparative study, we modeled the PIF in two ways: as boolean equations (brute force approach), and as the power mode identification tree (PIT). For simulation runs involving 10000 vectors, brute force approach had an average overhead of **20.7** seconds over the conventional approach over all the 7 example circuits.

Table 2: Experimental results of our RTL power macro-modeling approach

Circuit	Error(conv)	Error(prop)	Imprv.
C5	1.304	0.508	**61.1 %**
C6	4.42	0.45	**90.56 %**
C7	0.82	0.389	**52.5 %**
C8	1.65	0.447	**72.9 %**
C9	0.846	0.483	**42.9 %**
C10	0.77	0.56	**27.7 %**
C11	0.987	0.543	**45 %**

The PIT model incurred an average overhead of only **6.04** seconds over the conventional method, indicating a speed-up of about 3X approach over the brute force approach for power mode identification. For example, in the case of $C7$, the estimation with the conventional macro-model took 13.95 seconds, whereas, the estimation using the proposed macro-model, in which the PIF was modeled as PIT, took 20.2 seconds. In the same example, if the PIF was modeled as plain boolean equations (brute force approach), the estimation took in the excess of 40 seconds.

References

[1] J. Rabaey and M. Pedram, *Low Power Design Methodologies*. Kluwer Academic Publishers, Norwell, MA, 1996.

[2] A. R. Chandrakasan and R. W. Brodersen, *Low Power Digital CMOS Design*. Kluwer Academic Publishers, Norwell, MA, 1994.

[3] J. Monteiro and S. Devdas, *Computer-Aided Design Techniques for Low Power Sequential Logic Circuits*. Kluwer Academic Publishers, Norwell, MA, 1996.

[4] M. Pedram, "Power minimization in IC design: principles and applications," in *ACM Trans. Design Automation Electronic Systems*, vol. 4, no. 1, pp. 3 – 56, Jan. 1996.

[5] S. Powell and P. Chau, "Estimating power dissipation of VLSI signal processing chips: The PFA techniques," in *Proc. IEEE Workshop on VLSI Signal Processing IV*, vol. IV, pp. 250–259, 1990.

[6] P. Landman and J. Rabaey, "Power estimation for high-level synthesis," in *Proc. European Design Automation Conf.*, pp. 361–366, Feb. 1993.

[7] D. Marculescu, R. Marculescu and M. Pedram, "Information theoretic measures for energy consumption at RTL," in *Proc. Intl. Symp. Low Power Electronics*, pp. 81–86, April 1995.

[8] M. Nemani and F. Najm, "High-level power estimation and area complexity of boolean applications," in *Proc. Intl. Symp. Low Power Electronics*, pp. 329–334, Aug. 1996.

[9] A. Raghunathan, S. Dey and N. K. Jha, "RTL estimation techniques for switching activity and power consumption," in *Proc. Intl. Conf. Computer-Aided Design*, pp. 583–588, Nov. 1996.

[10] L. Benini, A. Bogliolo, M. Favalli, and G. DeMicheli, "Regression models for behavioral power estimation," in *Proc. of Intl. Workshop - Power and Timing, Modeling, Optimization and Simulation*, 1996.

[11] Q. Wu, C. Ding, C. Hsieh and M. Pedram; "Statistical design of macro-models for RT-level power estimation," in *Proc. Asia and South Pacific Design Automation Conf.*, pp. 523–528, Jan. 97.

[12] Z. Chen, K. Roy and K. P. Chong, "Estimation of power sensitivity in sequential circuits with power macromodeling application," in Proc. Intl. Conf. Computer-Aided Design, pp. 468–472, 1998.

[13] High-level synthesis benchmarks, CAD Benchmarking Laboratory, Research Triangle Park, NC. (http://www.cbl.ncsu.edu).

[14] S. M. Wiess and C. A. Kulikowski, *Computer Systems that Learn*. Morgan Kaufmann 1991.

[15] W. N. Venables and B. D. Ripley, *Modern Applied Statistics with S-PLUS*. Springer-Verlag 1998.

[16] *OpenCAD V 5 Users Manual*. NEC Electronics, Inc. 1997.