Circuit Partitioning Techniques For Power Estimation Using The Full Set of Input Correlations

Ana T. Freitas*  
aff@inesc.pt  
IST-INESC  
Lisbon, Portugal

Arlindo L. Oliveira  
aml@inesc.pt  
IST-INESC/CEL  
Lisbon, Portugal

Abstract

Exact power estimation, at logic level, is only possible if all the input correlations are taken into account. Recently, a probabilistic approach that uses a simple but powerful formalism for power estimation taking into account all the input correlations has been proposed. With this probabilistic approach it is possible to compute exactly the power dissipation of combinational modules using input statistics that would require extremely large traces if simulation based methods were to be used. However, the applicability of the method is limited to very small circuits.

This paper describes a circuit partitioning technique that speeds up that method. By using partitioning techniques we avoid the computation of global BDD representations for node functions, thereby extending considerably the range of applicability of the algorithm. Moreover, the partitioning maintains the full set of correlations and, therefore, does not induce any loss of accuracy.

1 Introduction and related work

This work addresses the complex problem of estimating the average power dissipation in combinational modules under general input correlation statistics. The basic model used by most power estimation techniques for a functional module is of the form:

\[ P = C_{DD} V_{DD} f \]

where \( V_{DD} \) is the voltage supply of the module and \( f \) is the clock frequency at which it operates. The factor \( C_{DD} \) is the average switched capacitance of the functional unit.

Existing approaches for power estimation can be divided into two categories: simulation based methods and probabilistic methods.

Simulation based methods can model very accurately the spatial and temporal correlations present at the input and internal signals, but require the existence of a trace that accurately matches the conditions of operation of the system. These methods can be very inefficient if the trace is very long.

In order to speed up this computation, a number of methods have been proposed which aim at reproducing accurately the input correlations with traces that are much smaller than the initial trace. One such possibility is random sampling where only a subset of the transitions present in the trace is used. More sophisticated methods include sequence compaction [7] and sequence synthesis [6].

Entropy based methods [5, 9] can also be included in this class since they use an input sequence to extract input and output entropies of a circuit. These methods use the concept of computational work (which is obtained from the entropy of the input and output signals) as a measure of the average activity in a circuit. Najm observes that average power is proportional to the product of the circuit area and the average node transition activity [9]. However, these methods may incur in a significant loss of accuracy since they do not take into account the details of the circuit structure.

Probabilistic methods do not require the existence of a trace and can be used to characterize circuits in a manner that is independent of the inputs presented. However, the most efficient approaches presented to date allow only for very simplified models of temporal and spatial correlations, and, therefore, obtain estimates of limited accuracy. The most common simplification considers the primary inputs uncorrelated in time and space [4], but this represents, in general, a very crude approximation. If one models exactly the spatial and temporal correlations, probabilistic methods have the potential to estimate exactly the power in conditions that would require exponentially large traces if simulation based methods are used.

However, it is known that efficient (i.e., guaranteed polynomial time) algorithms do not exist for power estimation problems that involve exponentially large input sequences. In fact, recent work has shown that the problem of average power estimation in combinational and sequential circuits is inherently complex. For instance, the problem of average power estimation for combinational circuits under the uniform input distribution and the binary code input sequence is \#P-complete [2]. This complexity analysis is also applicable to a large number of operating conditions that involve exponentially large input traces. Since any \#P (number-P) complete problem is known to be no simpler than any problem in the polynomial hierarchy (PH) [12], this results implies that no efficient algorithms exist for these problems unless \#PH=\#P.

Despite this fact, several methods that aim at modeling spatial and temporal correlations more accurately have been presented. One such method [8] models the pairwise spatial correlations of the primary inputs and propagates them through the circuit. This process only involves local calculations, and therefore can be applied to larger circuits, but does not control the total error introduced in successive computations.

Another approximation that has the potential to model accurately spatial and temporal correlations is based on the use of algebraic decision diagrams (ADDs) to characterize the dissipation incurred by each module under each particular input transition [1]. The main problem of this method is that ADD representations can easily become unmanageably large. The circuit partitioning technique introduced in this work can also be applied to ADD based approaches, with the objective of reducing the overall computational complexity.

A method that is able to compute the power dissipated taking into account the temporal and spatial correlations at the inputs has been presented recently [3]. However, as presented there, it requires the computation of the BDD representation of all internal nodes, specified as a function of the primary inputs of the circuit. For large circuits, and for some special families of circuits like, for instance, multipliers, it is impossible to compute this global BDD representation. Therefore, the applicability of this method is restricted to relatively small circuits.

In this work, we propose a circuit partitioning technique that effectively replaces the one global BDD representation by an ap-
Table 1: Specification of the input word statistics for Gray and Binary codes.

<table>
<thead>
<tr>
<th>Gray</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.500</td>
</tr>
<tr>
<td>. . .</td>
<td>0 . . 1</td>
</tr>
<tr>
<td>. . .</td>
<td>0 . . 0</td>
</tr>
<tr>
<td>. . .</td>
<td>1 . . 0</td>
</tr>
<tr>
<td>1</td>
<td>0.111</td>
</tr>
<tr>
<td>0</td>
<td>0.000</td>
</tr>
<tr>
<td>1</td>
<td>0.125</td>
</tr>
<tr>
<td>0</td>
<td>0.125</td>
</tr>
</tbody>
</table>

The appropriate number of partial BDDs of limited size. This technique provides a natural extension of the methodology proposed in [3] while keeping the full set of correlations required to maintain the accuracy of the estimation. Due to the inherent complexity of the problem the method proposed in [3] was exact but not very efficient. The circuit partitioning technique proposed in this work extends significantly the range of applicability of the former method by making it applicable to larger circuits, without any loss of accuracy.

The remainder of this paper is organized as follows. Section 2 briefly describes the method proposed in [3] which represents the base of the current work. Section 3 contains the main contribution of this work and shows how the computation described in the previous section can be performed much more efficiently using circuit partitioning. Section 4 presents the experimental results obtained and section 5 presents the preliminary conclusions and proposes future work.

2 Input correlation modeling and power estimation

2.1 Input correlation modeling

Consider a combinational circuit with $n$ input nodes and $m$ internal nodes. The variables that correspond to the input nodes are $\{x_1, \ldots, x_n\}$, while the circuit internal nodes are represented by variables $\{x_{n+1}, \ldots, x_{n+m}\}$. A subset of the internal nodes are also output nodes of the circuit. For each node $i$, we will use two variables, one representing the value before the transition ($x_i^b$) and the other one representing the value after the transition ($x_i^a$).

The input correlation modeling proposed in [3], based on the definition of a set of functions $F_i$ and associated probabilities $P_i$, describes a discrete Markov process at the inputs. Each $F_i$ defines over the variables $X = \{x_1, \ldots, x_n\}$ all transitions that take place with probability $P_i$ (a real number between 0 and 1, and satisfying $\sum P_i = 1$).

By specifying a set of pairs $(F_i, P_i)$ complex temporal correlations between input words are accurately modeled. Each $F_i$ is specified using the following formalism: for each possible transition, a description of the input word before and after a transition is given; individual bits in the word can be represented by:

- 0: The bit takes the value 0
- 1: The bit takes the value 1
- . : The bit takes any value.
- #: The bit changes its value after the transition.

As an example, the following segment of a specification:

0.10 . . 1.0 . 0.2

means that:

- With probability 0.2, the first bit stays the same, the second and third bits are fixed at 10 and the last bit can take any value.

Vectors representing transitions with the same probability are modeled by an unique $F_i$ function. As an example, a 4 bit Gray code and a 4 bit binary code are modeled by the specifications of the $F_i$ shown in table 1. In these cases, each specification describes a Markov process with 16 states. In general it is possible to describe Markov processes with a number of states that is exponential on the number of inputs.

2.2 Computation of dissipated power

The computation of the power dissipated in the circuit using the input statistics specified as above can be viewed as a generalization of the work by Ghosh [4]. His approach is based on the computation, for each node, of a function (represented by a BDD) that describes the conditions under which that node switches. With this function it is possible to evaluate the dissipated power by computing the number of transitions in internal nodes. However this computation is straightforward only if one assumes that all transitions at the inputs have equal probability. If the input transitions have different probabilities one needs to compute a sum over all possible paths in the BDD.

Our approach avoids the need to explicitly sum over all the paths in the BDD by using the functions $F_i$ and the associated probabilities $P_i$, defined in the previous section. To use this information in an effective way, a single function that characterizes the circuit is computed. This function, the Transition Consistency Function (TCF) depends on the input and internal signals at time $t$ and time $t+1$ and represents all possible transitions of the circuit under analysis.

We are interested in the computation of the dissipated power when the input variables change from $\{x_1^b, x_2^b, \ldots, x_n^b\}$ to $\{x_1^a, x_2^a, \ldots, x_n^a\}$. For simplicity, we will consider only the zero delay model.

Since there exists a total of $2(n+m)$ variables, the TCF specifies a probability distribution defined over a space of dimension $2^{n+m}$. This function is defined over the space $X \cup Y$, where $Y = \{x_1^b, \ldots, x_n^b, x_{n+1}^a, \ldots, x_{n+m}^a\}$.

The TCF is a function $T(X, Y)$ that describes all possible transitions. We define $T(x_1^b, \ldots, x_{n+m}^b)$ to be 1 if the values of the variables in the circuit $\{x_1, \ldots, x_{n+m}\}$ may experience a transition from $x_1^b, x_2^b, \ldots, x_{n+m}^b$ to $x_1^a, x_2^a, \ldots, x_{n+m}^a$.

Consider the circuit with two inputs and two outputs, shown in figure 1, and the corresponding list of all possible transitions.

![Figure 1: Example of a two input/two output circuit and the list of minterms in the Transition Consistency Function.](image)

Let the function at internal node $i$ be $f_i(x_1, \ldots, x_n)$. The function $T(X, Y)$ is computed using:

$$T(X, Y) = \prod_{j=n+1}^{n+m} x_j^a \equiv f_i(x_1^b, \ldots, x_n^b)$$

Note that the computation of expression 2 requires the computation of a global BDD representation for each node function. This represents the critical step of the original method. The use of a partitioning algorithm overcomes the limitations imposed by this global computation.

Once the TCF is known, the total number of transitions in node $k$, $T_k$, is given by:

$$T_k = \sum_{i} (x_i^b \oplus x_i^a) T(x_1^b, \ldots, x_{n+k}^b) F_i \times \frac{P_i}{|F_i|}$$

where $|F_i|$ denotes the number of minterms in $F_i$. 904
The dissipated power can now be computed using the following expression:

\[ P = \left( \sum \alpha_k T_k \right) f_{\text{total}} \]

The method described takes into account only functional transitions, i.e., transitions between steady states of the signals. This is equivalent to the use of a zero-delay model for all the gates in the circuit. However, power dissipation due to spurious transitions is also important and may account for a significant fraction of the total dissipated power.

A generalization of this method to handle non-zero-delay models has been described in [3]. The mechanism proposed to handle generic delay models does not change the structure of the method.

3 Power computation using circuit partitioning

This section describes a circuit partition technique that avoids the need to compute global BDDs and keeps the full set of correlations between sub-circuits. Therefore, the application of this technique considerably extends the range of applicability of the method described in the previous section, while maintaining its high level of accuracy.

Consider an arbitrary combinational module \( M \) with \( n \) inputs and \( p \) outputs, as depicted in figure 2. In this figure, the nodes

\[
\begin{align*}
1 & \quad \cdots \quad 2 \quad \cdots \quad M_1 \\
1 & \quad \cdots \quad 2 \quad \cdots \quad M_2
\end{align*}
\]

Figure 2: Combinational circuit module and cutset corresponding to nodes \( \{n_1, \ldots, n_k\} \).

\( \{n_1, \ldots, n_k\} \) constitute a cutset of the circuit, i.e., a set of internal nodes that break every path from input to output. Under these conditions, the computation of the dissipated power can be performed by computing the TCF of two simpler combinational modules:

- \( M_1 \) with inputs \( \{i_1, \ldots, i_k\} \) and outputs \( \{n_1, \ldots, n_k\} \)
- \( M_2 \) with inputs \( \{n_1, \ldots, n_k\} \) and outputs \( \{i_1, \ldots, i_p\} \)

For both \( M_1 \) and \( M_2 \), the BDD of each node is computed as a function of the sub-circuit inputs, thereby avoiding the need to compute the BDD for each node in \( M \) as a function of the primary inputs of the circuit.

Given the TCF \( T \) for block \( M \), it is possible to compute the functions \( F_i \) for block \( M_2 \) that are used to compute the power dissipated in block \( M_2 \). Recall that the functions \( F_i \) describe the correlations at the primary inputs of the circuit, and therefore the correlations at the inputs of block \( M_2 \). To compute the correlations at the inputs of block \( M_2 \), one needs to compute the functions \( F_i \) that depend on the variables \( \{n_1, \ldots, n_k\} \). Now, each \( F_i \) is transformed by block \( M_1 \) in accordance with the permissible transitions specified in its TCF. Therefore, the functions \( F_i \) are obtained by computing the conjunction of the TCF of \( M \) with functions \( F_i \), and smoothing the internal variables of block \( M_1 \).

To formalize the procedure, we introduce some additional notation. Let the original circuit be partitioned in a sequence of circuits \( M_k \), such that the outputs of \( M_{k-1} \) are connected to the inputs of \( M_k \). Let \( X_j \) be the set of input variables \( \{x_{j1}, \ldots, x_{jk}\} \) for circuit \( M_k \), and \( Y_j \) be the set of internal variables \( \{x_{j1}, \ldots, x_{jk}\} \) and \( Z_j \) be the subset of \( Y_j \) that corresponds to output nodes of \( M_k \). Finally let \( W_j = Y_j \setminus Z_j \) denote the set of variables that correspond to purely internal nodes of circuit \( M_k \).

The set of functions \( F_i \) that is required to compute function 3 can now be computed using the following expressions:

\[
\begin{align*}
F_i^1 & (X_1) = F_i(X_1) \\
F_i^2 & (X_1, X_2) = (X_2 \leftarrow Z_i \exists z \in W_j) T^j(X_1, Y_j) F_i^1 (X_1) \\
F_i^{i+1} & (X_1, X_{i+1}) = (X_{i+1} \leftarrow Z_i \exists z \in W_j) T^j(X_j, Y_j) F_i^{i} (X_1, X_j)
\end{align*}
\]

Note that functions \( F_i^j \) depend on both the primary input variables and the input variables of circuit \( M_j \). This, however, does not represent a strong limitation of the method since only the dependence on the primary input variables needs to be kept.

With the partition strategy proposed it is possible to define exactly the size of each sub-circuit in a way that the computation of the TCF no longer represents a limiting factor for this method. Note, however, that the size of the BDDs that can be obtained when expression 5 is applied, is not guaranteed limited since the computation of the \( F_i \), for an arbitrary \( i \), may generate BDDs with a large number of nodes. By using a known result [10] that the size of a BDD with \( k \) minterms and \( n \) input variables cannot exceed \( \log_2 n \) nodes, it is easy to obtain the following bound on the size of the largest BDD generated in the process of computing the values of \( F_i^j \):

\[
\text{BDD Size}(F_i^j) \leq |X_j||F_i^1|
\]

where \( |X_j| \) represents the number of variables in \( X_j \) and \( |F_i^j| \) represents the number of minterms in the functions specifying the primary input correlations. This bound implies that the maximum size of the BDDs reached depends only on the size of the cutsets obtained and the number of minterms in the functions specifying the original input correlations.

3.1 Algorithm for circuit partitioning

As illustrated in figure 2, the objective is to partition the network in a number of smaller networks such that each network has a user-defined maximum number of nodes and, as a whole, the circuit functionality is equivalent to the original. Given a maximum number of nodes for each circuit, \( N_{\max} \), and a circuit \( M_0 \) to be partitioned, we obtain a list of circuits \( L \) by the following algorithm:

1. \( S \leftarrow \{\} \)
2. \( M \leftarrow M_0 \)
3. While \( \text{Size}(M) > N_{\max} \) do:
   - Partition circuit \( M \) in two circuits \( (M_j, M_k) \), such that \( \text{Size}(M_j) \leq N_{\max} \).
   - \( S \leftarrow S \cup \{M_j\} \)
   - \( M \leftarrow M_k \)

The goal of this partition algorithm is to get sub-circuits with a maximum of \( N_{\max} \) internal nodes. First the nodes in the original circuit are ordered by level and then the algorithm starts the cut procedure from the primary inputs. In the first step all nodes in level 1 are inserted in the cutset. If the number of nodes is still smaller than \( N_{\max} \) then nodes from the following level are added to the cutset, substituting nodes from the previous level, until the sub-circuit has \( N_{\max} \) internal nodes. If the number of nodes in level 1 is greater than \( N_{\max} \), then only \( N_{\max} \) internal nodes are considered in the cutset. In this case it is necessary to cut using primary inputs.

As described, the partition algorithm recursively breaks the circuit in two parts by finding a set of nodes that is a cutset of the circuit. Although preference is given to cuts that include only internal nodes, in some cases primary inputs may also have to be included. In these cases buffers are inserted after the primary inputs. Buffers are also inserted if there is a connection from the first sub-circuit to the second that doesn't go through a cut node. To illustrate the partition algorithm let us consider, as an example, the combinational circuit with ten logic gates (each black circle represents a logic gate) shown in figure 3.
Figure 3: Cut example using a simple circuit.

The original circuit, presented in figure 3, shows the logic gates ordered by levels. The first iteration of the partition algorithm breaks the original circuit into two circuits. A buffer is inserted in node a2 because there is a direct connection from a2 to node n6, in the second sub-circuit.

Figure 4: Circuits obtained after first cut.

Since the second sub-circuit has more than 4 internal nodes, it is necessary to cut it again. After the second cut, the partition algorithm stops and obtains the final partition shown in figure 5. It is possible to see that two buffers were inserted since there were some connections from one sub-circuit to another that didn’t go through cut nodes.

Figure 5: Circuits obtained after second cut.

In some cases a large number of buffers may have to be inserted, leading to a significant increase on the actual number of nodes in the circuit. However, a simple analysis shows that this increase on the circuit size has only a moderate impact on the size of the BDD required to represent the TCF.

Consider a buffer with an input \( x_i \) and an output \( x_o \). The portion of the TCF that represents that buffer is given by \( (x'_i + x'_o) \land (x'_o \equiv x'_o) \). Since we use the optimal variable order \( (x'_1, x'_2, x'_3, x'_4) \), only 6 nodes are inserted in the BDD to represent the buffer function. Apart from those extra nodes, i.e., 6 BDD nodes per inserted buffer, there is no other impact on the size of the TCF required to describe the partitioned circuit, since these new variables are inserted before all other circuit variables.

4 Results

This section presents preliminary results obtained using the approach described in section 3. To evaluate the applicability of the method the algorithm was integrated within the SIS logic synthesis system [11].

As test cases, we used a set of circuits from the MCNC 91 benchmark and some additional ALU type circuits. These circuits were mapped to the MSU library before power estimation. All tests were performed on a 300MHz Pentium running Linux with a CPU time limit of 4 hours and memory usage restricted to 90 MB.

Before running the partition algorithm a study was made in order to find the best value for the partition parameter \( N_{\text{max}} \). The TCF for 36 circuits was computed using 8 different values for this parameter. Figure 6 shows the relation between the average CPU time for the 36 circuits and the value of \( N_{\text{max}} \). This figure shows that the partition algorithm gives best results for \( N_{\text{max}} = 20 \), i.e. 20 nodes per sub-circuit. Based on this result the partition algorithm was instructed to create partitions with a maximum of 20 nodes for all test circuits.

Figure 6: Find the best \( N_{\text{max}} \) for the partition algorithm.

Table 2 lists the set of test circuits for which we tried to compute the TCF. In this table, columns 2, 3 and 4 show the statistics of the circuits, namely the number of primary inputs, the number of primary outputs and the number of literals. Column 5 shows the CPU time necessary to compute the TCF when no partitions are used, as proposed in [3], and column 6 the CPU time necessary to compute the TCF of the circuits using the partition algorithm. This table shows that the partition algorithm effectively extends the size of the circuits that can be handled. Note that the algorithm returned CPU time limit for the 4 circuits where it was not possible to compute the TCF.

Table 3 shows the results obtained when the power estimation method was applied to these circuits using four different input statistics specified by four different sets of \( E_i \)'s. In some cases it is still not possible to compute the dissipated power even though it is possible to compute the TCF for each sub-circuit. The reason is that, as we show in expression 6, the maximum size of the BDDs required to compute the TCF only grows with the number of minterms. As the results show, with a small number of minterms, it is possible to compute the TCF for each sub-circuit. The second set (OneHot) specifies an One Hot code where only one bit at a time takes the value one. Finally the last set (Binary) of \( E_i \)'s specifies a binary encoding at the inputs of the circuits. The HighLow and Binary input specifications describe a Markov chain with a number of states that is exponential on the number of inputs.

The results in table 3 show that, in some cases, it is possible to compute exactly the switching activity in conditions that would require a prohibitively large trace if simulation based methods were used. As an example, it was possible to estimate power for test circuits with more that 30 inputs taking into account the full set of correlations introduced by a binary input code. Note that
Table 2: Circuit statistics, number of partitions, maximum TCF size with partition and TCF size of the original method.

for this case the $F_i$s, that describe the binary input code correspond to a Markov chain with $2^N$ states. Furthermore, the algorithm is currently limited in its performance by a critical step in the BDD manipulation process, that is consuming more than 90% of the total CPU time even though it does not correspond to any fundamental BDD operation. We are working in removing this bottleneck.

5 Conclusions and future work

In this paper we describe a circuit partitioning technique used to speed up a probabilistic method that performs power estimation considering both spatial and temporal correlations of the primary input signals. By using this partitioning technique, the method does not require the computation of global BDD representations for node functions, thereby extending considerably the range of applicability of the original method. This method requires only the computation of the BDDs for nodes in the sub-circuits in terms of the local inputs. The proposed partition algorithm recursively breaks the network in a number of smaller networks such that each network has a user defined maximum number of nodes while preserving the functionality of the whole circuit. The experimental results have shown that this partition algorithm gives best results for a maximum of 20 nodes per sub-circuit.

As the results show, the partitioning algorithm proposed effectively extends the size of the circuits that can be handled. In fact, the algorithm was able to compute exactly the switching activity in conditions that would require a prohibitive large trace if simulation based methods were used.

References


Table 3: Results for HighAct, Onehot and Binary encodings.


