Regression-Based RTL Power Models for Controllers *

Luca Benini #  Alessandro Bogliolo #  Enrico Macii ¹  Massimo Poncino ¹  Mihai Surmei ¹

# Università di Bologna
Bologna, ITALY 40136

¹ Politecnico di Torino
Torino, ITALY 10129

Abstract

Power consumption of the controller of an RTL design, though typically smaller than that of the datapath, cannot be neglected. Controller power modeling is a more challenging task than that of datapath modules, because the controller is usually specified in an abstract fashion that has no structural relationship with its final implementation. This paper presents an extensive experimental study on pre-state assignment power models. Starting from a large set of controllers, we study the relation between power and static parameters, such as number of inputs and outputs, as well as dynamic parameters, such as input and output switching activity. We then quantify the loss of accuracy caused by the high level of abstraction of the models, and we show how previously published results, obtained under favorable experimental conditions, have underestimated model inaccuracy. Finally, we perform extensive exploration on a large number of alternative macro-model structures, and we select an optimal model equation.

1 Introduction

Most RTL designs contain one or more controllers that coordinate the activity of datapath components. Even though controllers have a smaller impact than the datapath, their power dissipation is not negligible. First, some designs may contain many interacting controllers. Second, datapath units are often hand-crafted, while controllers are synthesized. Controllers use silicon area less efficiently than datapath, and their relatively small gate count does not translate into a proportionally small area when compared with densely packed datapath units. Third, controllers are usually active for a large fraction of the operation time, while the datapath can be partially or completely idle. These observations motivate the need for estimating the power dissipation of controllers during RTL design.

What makes controller power modeling more challenging than that of the datapath is the fact that controllers are usually specified in a very abstract fashion even at the register-transfer level and they are specific to every design. In contrast, datapath units are usually instantiated from a library that can be pre-characterized once for all. Some controller implementation styles employ regular structures, like PLAs or ROMs, that can be pre-characterized with good accuracy. Unfortunately, many controllers are synthesized as sparse logic. Because of the irregular structure of the sparse logic implementation, controller power modeling in this case is considerably harder.

In this paper we focus on the construction of power macro-models for controllers that are synthesized into sparse logic starting from an abstract RTL specification. The abstraction level is higher for controllers than for datapath components because states are usually specified symbolically. Thus, controller synthesis goes through two steps, namely, state assignment and logic synthesis. It is a well-known fact that state assignment has a significant impact on logic synthesis results, hence it is harder to pinpoint the power dissipation of a controller before state assignment than after state codes have been assigned.

Several researchers have proposed RTL power models for controllers. A few approaches assume the availability of a gate-level implementation of the controller [1, 2]. In this case, the power macro-model can be tuned directly on the implementation. These approaches are viable when controllers are expected to be re-used many times, or when we want to build a hierarchical power model for a complex macro containing a controller. Unfortunately, when performing design space exploration for an application-specific design, it may be too time-consuming to synthesize the controller and characterize its power model for each alternative design. Hence, macro-models have been developed for estimating the power consumed by controllers before gate-level synthesis. We can distinguish between pre-state assignment [3, 4] and post-state assignment [5, 6, 7] approaches. The former are generally less accurate, and provide upper and lower bounds to power consumption. The latter try to give actual estimates of average power.

This paper presents an extensive experimental study on pre-state assignment power models. Starting from a large set of controllers (that represent the characterization sample), we investigate how power relates to static parameters, such as number of inputs and outputs, as well as to dynamic parameters, such as input and output switching activity. Then, we quantify the loss of accuracy caused by the high level of abstraction of the models, and we show how previously published results, obtained under favorable experimental conditions, have underestimated model inaccuracy. Finally, we perform extensive exploration on a large number of alternative macro-model structures, and we identify the one that guarantees optimal results.

---

*This work was supported, in part, by the European Commission under grant n. 27696 "PEOPLE".
2 Preliminary Analysis

We consider controllers specified as finite state machine (FSMs), and described in terms of their state transition graph (STG). The latter is given as a list of transitions, expressed as 4-tuples of the type:

\[< \text{input}, \text{present state}, \text{next state}, \text{output}> \]  

If state assignment has not been done yet, present and next states are symbolic names; otherwise, they are expressed as binary patterns.

The power consumption of a controller depends on several parameters, that can be grouped according to their availability within the design flow:

- **Static behavioral** parameters, i.e., parameters that can be obtained from the behavioral specification:
  1. \(N_i\), number of inputs;
  2. \(N_o\), number of outputs;
  3. \(N_s\), number of states;
  4. \(N_c\), number of cubes.

- **Dynamic behavioral** parameters, i.e., parameters that can be obtained from functional simulation:
  5. \(R_{in}\), average input signal probability;
  6. \(D_{in}\), average input transition probability;
  7. \(D_{out}\), average output transition probability;
  8. \(D_s\), average state transition probability.

- **Static structural** parameters, i.e., parameters that can be obtained from state encoding/gate-level implementation:
  9. \(N_m\), number of state variables;
  10. \(A\), area.

- **Dynamic structural** parameters, i.e., parameters that can be obtained from RTL simulation:
  11. \(D_m\), average transition probability of the state variables.

Parameter \(N_s\) actually refers to the number of (symbolic) lines of the type shown in Equation 1 in the controller description. We use the term "cube" because each transition in the (pre-state-assignment) STG can be thought of as a (symbolic) product term of a function that has primary inputs and present state as inputs, and primary outputs and next state as outputs. This parameter is an indicator of the complexity of the combinational logic, but it is a behavioral parameter, since it does not require the state assignment.

While behavioral parameters can always be extracted from the specification of a controller, structural parameters may not be available at earlier design steps. In fact, area \((A)\) depends on the gate-level implementation, while the number of state variables \((N_m)\) and their average switching activity \((D_m)\) depend on the state encoding. Since we aim at constructing a general power model for controllers to be used as early as possible during the design, we will focus only on behavioral parameters.

Before actually building the power model based on behavioral parameters, we first analyze their individual correlation with power consumption, and try to evaluate the accuracy that can be achieved by using only behavioral parameters. To this purpose, we have carried out a statistical analysis of a large set of data, collected by running extensive experiments. The experimental setup we have used to collect the data is described in the next section, while the results of the statistical analysis are shown in Section 2.2.

2.1 Experimental Setup

We used as benchmarks the finite state machines from the MCNC [8] suite. All examples were parsed, synthesized and mapped using Synopsys' DesignCompiler onto a reference gate-level library characterized for power. The gate-level implementation of each benchmark was then repeatedly simulated by Synopsys' VSS for different workloads and the corresponding power consumption evaluated by DesignPower. For each of the 46 benchmarks, 80 input streams of \(N_p = 20(N_i + N_o + 2N_m)\) patterns each were used, uniformly distributed over the space of input statistics, parameterized in terms of \(P_{in}\) and \(D_{in}\).

All experimental results were collected in a matrix \(M\) with 12 columns (the first eleven associated with the independent variables described above, and the twelfth associated with the power obtained by simulation) and 3680 rows (one row for each experiment). Entry \((i, j)\) in the matrix represents the value taken by the \(j\)-th parameter in the \(i\)-th experiment. All results reported in this paper have been obtained by processing the data in \(M\) without running further experiments.

2.2 Statistical Analysis

Table 1 shows the cross-correlation between each independent variable and the actual power consumption, i.e., the correlation between each of the first 11 columns of matrix \(M\) and the last one. Values that are closer to 1 indicate a higher correlation between a parameter and the measured power.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Correlation with Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>(N_i)</td>
<td>0.28</td>
</tr>
<tr>
<td>(N_o)</td>
<td>0.61</td>
</tr>
<tr>
<td>(N_s)</td>
<td>0.30</td>
</tr>
<tr>
<td>(N_c)</td>
<td>0.26</td>
</tr>
<tr>
<td>(P_{in})</td>
<td>0.05</td>
</tr>
<tr>
<td>(D_{in})</td>
<td>0.23</td>
</tr>
<tr>
<td>(D_{in})</td>
<td>0.12</td>
</tr>
<tr>
<td>(D_s)</td>
<td>0.27</td>
</tr>
<tr>
<td>(N_m)</td>
<td>0.35</td>
</tr>
<tr>
<td>(A)</td>
<td>0.85</td>
</tr>
<tr>
<td>(D_m)</td>
<td>0.38</td>
</tr>
</tbody>
</table>

Table 1: Correlation Between Power and Parameters.
We observe that all the parameters have positive correlation with power, but none of them provides by itself enough information about the actual power consumption (the average correlation is around 30%). This suggests that some of the parameters have to be combined to build an accurate power model. It is worth noting that area (A) is the only parameter with a correlation well above 50%. This is an intuitive result, because it is well-known that power consumption is strongly dependent on the number of components in its gate-level netlist. State encoding information seems to be less critical, since its correlation to power is just above the average. Finally, we observe that the input signal probability has almost no effect on power.

The lack of information about low-level structural information is an inherent source of error that limits the accuracy achievable at the behavioral level: two gate-level implementations of the same controller will typically exhibit different power values associated with the same 8-tuple of behavioral parameters. In general, the error induced by the abstraction level can be expressed as the variance of the power values associated with the same configuration of all behavioral parameters. This is the theoretical lower-bound error for any model based only on such parameters.

In principle, the lower-bound error should be used as a baseline to evaluate the power models we build in Section 3. We observe, however, that data collected in matrix \( M \) do not provide statistically significant estimates of the theoretical lower bound, since there are only a few rows with exactly the same configuration of behavioral parameters. On the other hand, the computation of reference values for model accuracy and complexity is mandatory.

To solve this problem, we perform a coarse-grained discretization of the behavioral parameters, by partitioning their values in four classes. More precisely, the range of values taken by each parameter is partitioned into 4 intervals and a unique value is associated with each interval. Discretization induces a partitioning of the space of parameters: Parameter configurations belonging to the same partition are represented by a unique discretized tuple and become indistinguishable after partitioning. The best conceivable power model based only on discretized behavioral parameters is a discrete function that associates to each tuple of discretized parameters the average of the power values associated with all the original configurations mapped on that tuple.

The error made by this model is now also due to the approximation induced by discretization, besides to the lack of circuit implementation details. Hence, the resulting error is actually larger than the theoretical lower bound. However, it has several features of practical interest. First, it can be estimated based on the data collected in \( M \). Second, it is a reachable bound, since the discrete function can be represented, for instance, as an 8-dimensional look-up table (LUT) and used as a behavioral power model. Third, it is possible to evaluate its cost in terms of complexity. Hence, we use the accuracy and the complexity of the discrete function, characterized and evaluated on \( M \), as a reference for evaluating the quality of any model.

We represent model accuracy in terms of its average relative error:

\[
\text{AvgRE} = \frac{1}{N} \sum_{j=1}^{N} \frac{|P_{est}^{(j)} - P_{sim}^{(j)}|}{P_{sim}^{(j)}}
\]

where \( N \) is the total number of experiments, \( P_{sim}^{(j)} \) is the power obtained from simulation, and \( P_{est}^{(j)} \) is the power estimated with the given model. The in-sample \( \text{AvgRE} \) (i.e., the \( \text{AvgRE} \) evaluated on the same sample used for characterization) of the reference model is 19.4%. The complexity of the LUT that would be required to represent such model has a size of \( 4^8 = 65536 \) entries. It is worth remarking that hundreds of thousands of simulations (possibly performed on different benchmarks) should be required to characterize such a model. The 3680 rows of \( M \) (representing all the experiments we performed) fall into less than 1000 partitions, so that they could be used to characterize only a small fraction of the entries of the look-up table.

To evaluate the sensitivity to the parameters we computed, for each parameter, the \( \text{AvgRE} \) resulting from two modified versions of the reference LUT model: i) a LUT obtained by using 100 levels (instead of 4) to represent the parameter, and ii) a 7-dimensional LUT obtained by neglecting the parameter. Results are plotted in Figure 1. Circles represent the error of the original model (19.4%), while the error bars represent the range between the lower bound error (achieved by reducing the discretization step for a parameter), and the upper bound error (achieved by removing a parameter from the model). The first 8 bars are associated with the 8 behavioral parameters, while the last two represent the improved accuracy that could be achieved by using also state encoding and area information, respectively.

![Figure 1: Sensitivity of Power to Parameters.](image)

The larger the error bar, the higher the sensitivity of the model to the corresponding parameter (reported on the X axis). From the graph we observe that sizable accuracy improvements could be achieved by improving the model resolution along the \( N_s \) and \( N_a \) axis, while neglecting \( N_i \) and \( N_d \) may determine a big accuracy loss (up to 30%).
As for post-encoding and post-synthesis information, we observe that the knowledge of the actual area would reduce the error from 19 to 5%. However, similar improvements can be achieved at the behavioral level by exploiting the knowledge of the number of cubes \( N_s \). This suggests that \( N_s \) can be used as a pre-synthesis area estimator.

3 Regression Models

We restrict our investigation to linear-regression models for several reasons. First, because linear regression models are a rich family, and they have been successfully employed for power macro-modeling [9, 10]. Second, because we are looking for a general model of practical interest, with a limited number of fitting coefficients. Third, because the errors induced by non-linearities are below the uncertainty due to the abstraction level.

Before performing a systematic exploration of the modeling space, we introduce some ad-hoc models based on intuitive considerations:

- **First order model.** Power is expressed as a weighted sum of all behavioral parameters available:
  \[
  P = c_0 + c_1 N_i + c_2 N_e + c_3 N_s + c_4 N_o + c_5 D_in + c_6 D_out + c_7 D_{out}
  \]
  The in-sample \( \text{AvgRE} \) provided by the model is 77%.

- **Intuitive model.** The model focuses mainly on the combinational part of the FSM and expresses power consumption as the linear combination of the total activity at the inputs and outputs of the combinational logic. To this purpose, \( N_m \) and \( D_m \) are used as independent variables, assuming state encoding information is available:
  \[
  P = c_0 + c_1 (N_i D_{in} + N_o D_{out}) + c_2 (N_o D_{out} + N_m D_{out}) N_s
  \]
  The in-sample \( \text{AvgRE} \) provided by the model is 55%.

- **Landman’s model.** The model proposed by Landman and Rabaey [5] focuses on the combinational logic and combines I/O activity information with complexity (i.e., area) estimates based on the number of cubes of the specification. As for the intuitive model, state encoding information is assumed to be available:
  \[
  P = c_0 (N_i D_{in} + N_m D_{out}) N_s + c_1 (N_o D_{out} + N_m D_{out}) N_s
  \]
  The \( \text{AvgRE} \) of the model is 85%. The error is sensibly higher than that provided in the original paper because the authors of [5] actually tested their model against randomly-generated controller, instead of realistic controllers and standard benchmarks. This result is not surprising: it has been shown in the past that randomly-generated benchmarks are not representative of actual human-designed logic circuits [11].

### 3.1 Model Space Exploration

We implemented a simple algorithm for the systematic construction, characterization and evaluation of regression models based on a given set of parameters. The algorithm takes as inputs matrix \( M \), the list of parameters (columns) to be used as candidate independent variables, the maximum number of terms \( T \) to be used in the linear equation, and the maximum order \( C \) of each term. The algorithm iteratively builds and tests all regression models of the form:

\[
P = c_0 + c_1 Q_1 + c_2 Q_2 + \ldots + c_T Q_T
\]

where \( Q_i \) denotes the \( i \)-th term (of order \( C \)) of the form:

\[
Q_i = v_1, v_2, \ldots, v_C
\]

and \( v_j \) is a candidate independent variable. In other words, the algorithm iterates on all regression models of up to \( T \) product terms of up to \( C \) variables each. The algorithm evaluates the in-sample accuracy (in terms of \( \text{AvgRE} \)) of each model and returns the most accurate model. We denote by \( S(T, C) \) the set of models of up to \( T \) terms of up to \( C \) variables, and by \( B(T, C) \) the most accurate model in \( S(T, C) \). It can be easily verified that the following properties hold for all pairs \((T_1, C_1)\) and \((T, C)\) such that \( T \leq T_1 \) and \( C \leq C_1 \):

\[
S(T, C) \subseteq S(T_1, C_1)
\]

\[
\text{AvgRE}_{E_{B}(x, C)} \geq \text{AvgRE}_{E_{B}(x, C_1)}
\]

Hence, we expect the algorithm to return better models for higher values of \( T \) and \( C \). On the other hand, the cardinality of \( S(T, C) \) and the exploration time grow exponentially with \( T \) and \( C \).

Based on the above observations, we explored the modeling space for increasing values of \( T \) and \( C \), starting from \((T, C) = (1, 1)\) and changing \( T \) and \( V \) independently. The following table reports the best achievable \( \text{AvgRE} \) as a function of \( T \), for \( C = 2 \):

<table>
<thead>
<tr>
<th>((T, C))</th>
<th>(1, 2)</th>
<th>(2, 2)</th>
<th>(3, 2)</th>
<th>(4, 2)</th>
<th>(5, 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{AvgRE}_{\text{in}}(%))</td>
<td>80</td>
<td>54</td>
<td>44</td>
<td>40</td>
<td>37.8</td>
</tr>
</tbody>
</table>

Looking at the model equations, we observe that the best single-term model is \( P = c_0 + c_1 N_i D_{out} \). Product term \( N_s D_{out} \) was not included in any intuitive heuristic. It is also worth mentioning that the same term appears in most of the best models returned by the exploration algorithm for higher values of \( T \).

Orthogonal results are shown in the following table, showing the best achievable \( \text{AvgRE} \) for increasing values of \( C \), with fixed \( T = 4 \):

<table>
<thead>
<tr>
<th>((T, C))</th>
<th>(4, 1)</th>
<th>(4, 2)</th>
<th>(4, 3)</th>
<th>(4, 4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{AvgRE}_{\text{in}}(%))</td>
<td>78</td>
<td>48</td>
<td>34</td>
<td>34</td>
</tr>
</tbody>
</table>
The best (4, 1) model is \( P = c_0 + c_1 N_e + c_2 N_e N_s + c_3 D_s \). Comparing the best models returned for different values of \( C \) we observe that most of the terms of a lower-order best model do appear as sub-terms in higher-order best models. In particular, the best model in \( S(4, 4) \) is a third-order model, namely, the best model in \( S(4, 3) \): \( B(4, 4) = B(4, 3) \). This means that moving from third- to fourth-order models does not improve the accuracy (at least when up to 4 terms are considered).

Figure 2 shows the distribution of the \textit{AvgRE} for the models in \( S(4, 4) \) and \( S(4, 3) \). For each value of \textit{AvgRE}, the curves report the number of models providing that error. If normalized to the total number of models in each set, the distributions represent the probability of achieving a given in-sample accuracy by using a regression model randomly taken from \( S(4, 4) \) or from \( S(4, 3) \).

The plot at the bottom of Figure 2 shows the behavior of the two distributions around the best point. As expected, \( S(4, 4) \) contains more models than \( S(4, 3) \) approaching the accuracy of the best one. However, if normalized to the cardinality of the two sets, good models have higher probability in \( S(4, 3) \) than in \( S(4, 4) \). This means that randomly selected third-order models are better, on average, than randomly selected fourth-order ones. The mean value of the \textit{AvgRE} is 80\% for \( S(4, 4) \) and 73\% for \( S(4, 3) \).

The best regression equation, found by exhaustive exploration of tens of millions solutions, has the form:

\[
P = c_0 + c_1 N_e + c_2 N_e N_s + c_3 N_e N_s D_{\text{out}} + c_4 N_e D_{\text{in}} D_s \tag{4}
\]

Its in-sample \textit{AvgRE} is 34\%.

4 Experimental Results

We compare equation 4 with the heuristic regression models presented in Section 3 using four different metrics: the in-sample \textit{AvgRE} used for exploration, the standard deviation of the relative error \textit{REStDev}, the average of root mean square relative error \textit{RMSRE} provided for each benchmark, and the cross-correlation \textit{CC} between the actual power and the power estimates provided by the model. Results are shown in Table 2.

![Figure 2: Error Distributions.](image)

Interestingly, the probability of selecting the best model is almost null (i.e., the value of the distribution corresponding to \textit{AvgRE} = 34\% is one out of several millions). In addition, there is a sizable gap between the best achievable accuracy and the values of \textit{AvgRE} with probability greater than 10^{-8}. This actually demonstrates the usefulness of the pseudo-exhaustive exploration we performed.

<table>
<thead>
<tr>
<th>Model</th>
<th>\textit{AvgRE} [%]</th>
<th>\textit{REStDev} [%]</th>
<th>\textit{RMSRE} [%]</th>
<th>\textit{CC}</th>
</tr>
</thead>
<tbody>
<tr>
<td>First-Order</td>
<td>77</td>
<td>90</td>
<td>88</td>
<td>0.75</td>
</tr>
<tr>
<td>Intuitive</td>
<td>55</td>
<td>64</td>
<td>59</td>
<td>0.83</td>
</tr>
<tr>
<td>Landman</td>
<td>85</td>
<td>95</td>
<td>94</td>
<td>0.79</td>
</tr>
<tr>
<td>Eq. 4</td>
<td>34</td>
<td>35</td>
<td>38</td>
<td>0.93</td>
</tr>
</tbody>
</table>

Table 2: Summary of Model Estimation Errors.

We remark that \textit{AvgRE}, \textit{REStDev} and \textit{CC} are evaluated on the entire set of data collected in \( M \), hence, they represent global measures of in-sample accuracy. On the contrary, the \textit{RMSRE} has been separately evaluated for each benchmark, to represent the accuracy that a designer may obtain from the models when they are used to estimate the power consumption of a given controller under different workload conditions. Values reported in the table have been obtained by averaging the \textit{RMSRE}s of several benchmarks.

5 Conclusions

We have presented an extensive experimental study on the design of power models for the controller of an RTL design. The models have been derived by statistical analysis of the relation between power and both behavioral and structural parameters of the controller.

We have performed a semi-exhaustive exploration on a large number of alternative macro-model structures, based on the correlation between power and the various parameters. The procedure identifies an optimal structure that outperforms previously presented approaches, resulting in an average error around 34\%.
References


