High-Level Power Modeling of CPLDs and FPGAs[‡]

Li Shang and Niraj K. Jha
Department of Electrical Engineering
Princeton University
{lshang, jha}@ee.princeton.edu

Abstract

In this paper, we present a high-level power modeling technique to estimate the power consumption of re-configurable devices such as complex programmable logic devices (CPLDs) and field-programmable gate arrays (FPGAs). For simplicity of reference, we simply refer to these devices as FPGAs. First, we capture the relationship between FPGA power dissipation and I/O signal statistics. We then use an adaptive regression method to model the FPGA power consumption. Such a high-level model can be used in the inner loop of a system-level synthesis tool to estimate the power consumed by different FPGA resources for different potential system-level synthesis solutions. It can also be used to verify the power budgets during embedded system design. With our high-level power model, the FPGA power consumption can be obtained very quickly. Experimental results indicate that the average relative error is only 3.1% compared to low-level FPGA power simulation methods.

1. Introduction

Reconfigurable devices [1-5], such as CPLDs and FPGAs, are parallel and general hardware platforms. For embedded systems produced in limited quantities, FPGAs are much more flexible and cost-effective alternatives to application-specific integrated circuits (ASICs). More importantly, modern applications, such as multimedia and wireless communication, have different objectives under different operating modes. A multi-mode system architecture can be supported by FPGAs in a natural way with the help of dynamic reconfiguration.

With the success of battery-based personal computing devices and wireless communication systems, low power has become a key issue in embedded system design. Although its flexibility makes FPGA a good solution for portable applications, the power consumption problem cannot be neglected. First, less efficient utilization of available resources makes an FPGA less power-efficient than an ASIC. Second, since routing resources in an FGPA include many switches, the interconnect load capacitance of an FPGA is 10X to 100X compared to the capacitance in an ASIC [12]. In order to effectively use FPGAs in low power systems, efficient FPGA power estimation methods are needed. However, the problem of FPGA power estimation is not that well-studied.

The rest of this paper is organized as follows. In Section 2, we discuss previous works on FPGA and ASIC power estimation. In Section 3, we introduce our high-level power model. In Section 4, we discuss high-level FPGA power modeling in detail. In Section 5, we present an adaptive regression-based high-level power modeling technique. We demonstrate the feasibility of our methods experimentally in Section 6. Finally, we conclude in Section 7.

2. Related Work

Some power estimation techniques for ASICs have been presented in [6-10]. In [7], techniques are presented for estimating switching activity and power consumption in register-transfer level (RTL) circuits. A framework for exact and approximate switching activity estimation in a sequential circuit is provided in [8]. A lookup table based ASIC power estimation method is proposed in [9,10]. To circumvent the problem of an exponentially increasing storage for table lookup, a four-dimensional table is used to model power consumption.

Works in [11-14] address the power estimation/ optimization problem in FPGAs. A power estimation method, which is based on power modeling of different components in an FPGA, is proposed in [12]. In this approach, the internal structure of the FPGA needs to be fully explored. Due to the periphery effect and estimation error in switching activity and interconnect capacitance, the total power estimation error may be up to 30%. The method in [14] characterizes the power dissipation of FPGAs using Manhattan distances between configurable logic blocks (CLBs) to model the interconnect power consumption. The error is usually less than 5% when compared to actual power measurement. However, since an iterative approach is used for updating the values of unknown signal parameters, the iteration process may not converge. Both the above methods need internal FPGA configuration information, and use internal signals to model power consumption. However, since many new macro modules, such as Block RAM [1], embedded

[‡]Acknowledgements: This work was supported by DARPA under contract no. DAAB07-00-C-L516.

system block (ESB) [2] and standard interfaces etc., have been integrated into FPGAs, fully exploring the internal architecture of an FPGA has become very challenging. With these "internal" signal modeling approaches, different modules may need their own description models. Hence, it becomes difficult to construct a single fixed high-level power model template.

3. Overview of Our Work

In this paper, we propose a high-level FPGA power modeling approach with the following characteristics.

- The model is based on input and output signal statistics to estimate the internal power consumption of FPGAs.
- Models for differently-configured circuits are based on the same power macromodel template. Thus, the same modeling procedure is applied to different circuits, whether combinational or sequential.
- An adaptive regression method is used to tackle the problem of biased training sequences. A good trade-off between accuracy and efficiency is achieved.

There are two important concerns in high-level FPGA power modeling. First, high-level power estimators need to be more efficient than gate-level power estimators. The high-level power model we propose can be used in the inner loop of system-level synthesis, putting power estimation in the critical path of synthesis. Thus, estimation efficiency is a must. We use a regression function to estimate the FPGA power consumption. In our approach, spatial correlation information is utilized to partition input signals into different subsets. Statistics for three input signal parameters for each subset and one output signal parameter are used in our regression method to construct the high-level power model.

A shortcoming of high-level power estimation methods in general is that a training set of input vectors with given statistical distributions is typically used in developing the power macromodels. Hence, the macromodel is biased towards the training set. If the input trace observed in practice has a different statistical distribution than the training set, the power estimation result may not be accurate. During normal operation, there may be hundreds of different typical input traces for different applications. We cannot use a static off-line method to build a power macromodel for each of these input traces. In our approach, we use adaptive regression methods to address this problem. An adaptive regression method has previously been used for ASICs [19], but not for FPGAs.

A low power CPLD, called Coolrunner [15], is used for experimentally validating our technique. Coolrunner is an ideal choice for battery-operated portable devices. Training sequences with different statistical distributions are generated for timing and power simulation. In order to capture the timing information for different configurations, we chose Scirocco [16], a real-delay timing simulator. A low-level power simulator [17] is invoked to estimate the power consumption. Since this low-level power simulator captures all the internal switching information for this device, its accuracy is comparable to physical power measurements. Then with all the collected information, we use an adaptive regression method to construct a high-level power model. Although our experiments are based on the Coolrunner, our method is also suitable for other reconfigurable devices, including other CPLDs and FPGAs. There are two reasons for this. First, our method does not need to explore the internal structure of the reconfigurable devices. Only input and output statistical parameters are used to construct the high-level power model. Second, although the detailed internal structures of different reconfigurable devices are different, most reconfigurable devices, including CPLDs and FPGAs, have the same macro structure: embedded logic blocks connected with global and local interconnection resources.

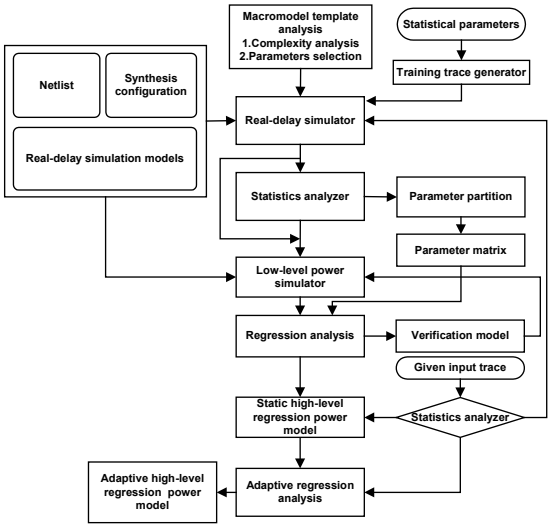

Fig.1: High-level power modeling for FPGAs

The adaptive regression-based high-level power modeling flow is shown in Fig. 1. First, the macromodel template analyzer is invoked to perform complexity analysis and other preprocessing steps. Then, training sequences with different statistical characteristics are generated. These sequences are used for real-delay timing simulation. With these results, the statistics analyzer is invoked to partition the input signals into subsets, and create a parameter matrix. A low-level power simulator tool is also invoked to estimate FPGA power consumption. The power results and parameter matrix form the inputs of the regression analysis step. Using feedback from the verification model, a static high-level regression power model is constructed. During the power estimation period, the input trace is analyzed to decide whether adaptive regression analysis should be used. If

so, the input trace is sampled, and the sampled data are sent to the low-level power estimator, while the static high-level regression power model estimates the FPGA power consumption with the whole input sequence. Adaptive regression analysis is invoked finally to construct the adaptive high-level regression power model.

4. High-Level Power Modeling

In this section, we give details of the constituent parameters used in our power model.

4.1 Partitioning of input signals

In [10], similar to our method, four kinds of parameters are used to model the input and output signal statistics of ASICs. However, the method in [10] ignores the nature of the signals. In practical circuits, different input signals may or may not be correlated. For example, if the circuit inputs include both data and control signals, each may have a very different statistical distribution. Thus, one should avoid using one average statistical parameter to cover both types of signals. In our input signal partitioning approach, we study the spatial correlation behavior between different input signals based on the input traces with different statistical distributions. If the correlation between two input signals is less than a pre-defined correlation threshold, we partition them to different input subsets, as shown in Fig. 2. Then, we use statistical parameters to model the signals in each subset, as discussed next.

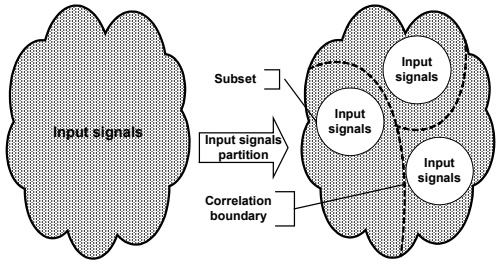

Fig. 2: Input signals partition

4.1 Input subset signal probability

The FPGA power consumption can vary significantly with the statistical distribution of the input data. Hence, "white noise" input sequence cannot be applied to model the power consumption of FPGAs. In our approach, after spatial correlation partitioning is done using typical input traces, the average signal probability for each input signal subset is calculated and used in high-level power modeling.

4.2 Input subset signal transition density

For each input subset, the average signal transition density [10] is calculated to model the switching activity for all the signals in the subset, which is defined as:

$$D_k = \frac{\sum_{in_i \in subset_k} \lim_{T\to\infty} \frac{n_{in_i}(T)}{T}}{N_{subset_k}} \quad (1)$$

where $n_{in_i}(T)$ is the number of times input signal in_i switches during time T. N_{subset_k} is the number of input signals in subset k.

4.3 Input subset signal spatial correlation

The average input subset pairwise correlation parameter [8] is defined as:

$$SC_k = \frac{\sum_{(in_i, in_{i+1}) \in subset_k} prob\{in_i \wedge in_{i+1} = 1\}}{N_{subset_k}} \quad (2)$$

where the parameters are self-evident.

4.4 Output signal transition density

The last parameter is the average output signal transition density. In order to construct a more accurate power macromodel, we use real-delay timing simulation and capture the transition density of the output signals. This allows us to capture output glitching information, which also reflects the internal FPGA glitching behavior.

4.5 Regression macromodeling function

Our high-level power model is based on the average input subset signal probability (P_{in}), average input subset signal transition density (D_{in}), average input subset signal spatial correlation (SC_{in}), and average output signal transition density (D_{out}). First-order, quadratic and cubic regression templates are three possible approaches to model the FPGA power, which have different execution complexity and accuracy. In the simplest model, a first-order polynomial template is expressed as:

$$Power_{FPGA} = \sum_{i=0}^{n-1} (C_{P_{in_i}} P_{in_i} + C_{Din_i} D_{in_i} + C_{SC_{in_i}} SC_{in_i}) + C_{Dout} D_{out} + C_c \quad (3)$$

where $C_{P_{in_i}}, C_{Din_i}, C_{SC_{in_i}}, C_{Dout}, C_c$ are regression coefficients, and n is the total number of input subsets.

In order to achieve a good trade-off between efficiency and accuracy, we design a hybrid model in our approach, which is based on the sensitivity of different parameters. First, we evaluate the sensitivity of FPGA power consumption to the different parameters. A low order term is chosen for parameters to which the FPGA power is less sensitive, while a high order term is used for parameters to which the FPGA power is more sensitive.

The pseudo-code of our algorithm is shown in Fig. 3. In this algorithm, p denotes the total number of terms and q the total number of first-order terms (e.g., D_{in_i} is a first-order term in Equation (3)). $\varepsilon_{threshold}$ is a predefined error threshold. The error minimization ratio (EMR) is defined as the error minimized per term divided by the error threshold:

$$EMR = \frac{\varepsilon_{n-1} - \varepsilon_n}{\varepsilon_{threshold}} \quad (4)$$

1. $Create\_candidate\_term\_pool(term_1, term_2, ..., term_p)$
2. $Reg\_fun\_tem = Create\_first\_order\_template(term_1, ..., term_q)$
3. $Power_{low\_level} = Low\_level\_power\_est(real\_delay\_simulation)$
4. $Power_{high\_level} = Regression\_power\_est(Reg\_fun\_tem)$
5. $Error\_calculation(Power_{low\_level}, Power_{high\_level})$
6. $While(\neg(error \le \varepsilon_{threshold} \vee terms \ge term_{upperbound} \vee EMR \le EMR_{lowerbound}))\{$
7. $\quad Term\_pri\_sen\_sorting(term\_pool)$
8. $\quad new\_term = term\_pool.begin()$
9. $\quad Reg\_fun\_tem = Reg\_template(Reg\_fun\_tem, new\_term)$
10. $\quad Power_{high\_level} = Regression\_power\_est(Reg\_fun\_tem)$
11. $\quad Error\_calculation(Power_{low\_level}, Power_{high\_level})$
12. $\quad terms++$
13. $\quad EMR\_calculation()$
14. $\}\ output(Reg\_fun\_tem, error)\}$

Fig. 3: The power macromodeling algorithm

In the first step of the algorithm, a pool of terms is created, and the sensitivity of FPGA power to each term is evaluated dynamically in the following iterations. Then, in Steps 2-5, the initial regression model is constructed and the pertinent error compared to low-level power estimation results is calculated. If the error is larger than $\varepsilon_{threshold}$, a heuristic procedure is invoked in Step 6. The priority of terms in the pool is decided dynamically according to their sensitivity (Step 7), and the term (quadratic or cubic) with the highest priority is chosen and included in the macromodel (Steps 8 and 9). The error and EMR are evaluated (Steps 11 and 13). At the end of each iteration, three criteria are used to make a decision on whether the heuristic procedure should be stopped. The three criteria are: first, the error is less than a predefined error threshold, $\varepsilon_{threshold}$, or second, the total number of terms has exceeded a predefined upper bound on the number of terms, or third, the EMR is less than a predefined threshold, $EMR_{lowerbound}$. Since different circuits mapped to an FPGA have different power consumption behavior, using our algorithm, we can flexibly construct high-level power estimation models with different complexity for them, achieving a good trade-off between efficiency and accuracy. For our example circuits, a typical high-level power estimation function was found to be:

$$Power_{FPGA} = \sum_{i=0}^{n-1}(C_{P_{in_i},Din_i} P_{in_i} D_{in_i} + C_{P_{in_i},SC_{in_i}} P_{in_i} SC_{in_i} + C_{P_{in_i}} P_{in_i} + C_{Din_i} D_{in_i} + C_{SC_{in_i}} SC_{in_i}) + C_{Dout} D_{out} + C_c \quad (5)$$

Having derived a high-level power estimation function, we need to estimate the regression coefficients C_i. Hence, we generate different random input sequences with different values of P_{in}, D_{in}, and SC_{in}, covering a wide range of input statistics. For each input sequence, a lower-level power simulator and real-delay timing simulator is invoked to estimate the average FPGA power consumption and average output signal transition density D_{out}. These results are used to calculate the regression coefficients as follows. We first set up a matrix equation based on the inputs:

$$\begin{pmatrix} P_0 \\ P_1 \\ \dots \\ P_{r-2} \\ P_{r-1} \end{pmatrix} = \begin{pmatrix} 1 & \dots & x_{0,k} & \dots & x_{0,s} \\ 1 & \dots & x_{1,k} & \dots & x_{1,s} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \dots & x_{r-2,k} & \dots & x_{r-2,s} \\ 1 & \dots & x_{r-1,k} & \dots & x_{r-1,s} \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ \dots \\ C_{s-1} \\ C_s \end{pmatrix} \quad (6)$$

where $P_l, l = 0,1,...,r-1,$ denotes the power simulation results for the l^{th} input sequence, $x_{j,k}$, $j = 0,...,r-1, k = 1,...,s$ represents the k^{th} parameter's j^{th} statistical value, and C_t, $t = 0,1,...,s$, is the regression coefficient we need to estimate. Denote the matrix by $[X]$. If $[X]^T[X]$ is of full rank, then [18]:

$$\begin{pmatrix} C_o \\ C_1 \\ \dots \\ C_{s-1} \\ C_s \end{pmatrix} = \left[[X]^T[X]\right]^{-1}[X]^T \begin{pmatrix} P_0 \\ P_1 \\ \dots \\ P_{r-2} \\ P_{r-1} \end{pmatrix} \quad (7)$$

To evaluate the accuracy of the macro-model, we define relative error as:

$$\varepsilon = \sqrt{\frac{\sum_{i=0}^{r-1}((\hat{P}_i - P_i)/P_i)^2}{r}} \quad (8)$$

where P_i's are the simulated power consumption, and $\hat{P}_i$'s are the estimated values.

5. Adaptive Regression-Based Power Estimation

The method and algorithm described so far are based on static macromodels, which do not change from one design that uses the macro logic block to another. However, such a static macromodel is biased towards the set of training patterns used. It is most accurate when the training patterns reflect well the typical input sequences in a specific design implementation. However, this may not always be true. The macro logic block and its high-level power model are likely to be stored in a design library, which is used for different applications, in which the data statistics can vary dramatically. On the other hand, since low-level power estimation is time-consuming, as we mentioned earlier, if the high-level power models are to be used in the inner loop of system-level synthesis, only fast and relatively accurate power estimation is acceptable.

We use an adaptive regression modeling approach to tackle the above problem, as shown in Fig. 4. First, the real-delay timing simulator is invoked for the given input trace. We compute the relevant statistics of the input sequence in Step 2 to determine if the input sequence under consideration has similar statistical characteristics as the training set used to derive the static high-level power model (Step 3). If so, static regression modeling is used (Step 4). Otherwise, we invoke the adaptive regression modeling engine in Step 5, where the input sequence is sampled. A low-level power estimator is invoked to estimate the FPGA power consumption with

the sampled sequence (Step 6). The static high-level power model is also used to estimate the power consumed by the FPGA for the whole input trace (Step 7). Finally, in Step 8, an adaptive regression engine is invoked to generate the adaptive high-level power model. Since the length of the sampled sequences is much shorter than the whole typical input trace, the time consumed in running the low-level power simulator is acceptable. By adapting the static high-level power model using sampled vectors, we can achieve a good trade-off between accuracy and efficiency.

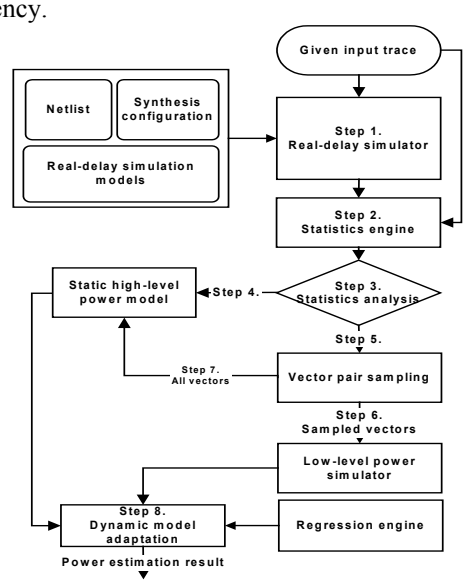

Fig. 4: Adaptive regression method flow

The method we use is similar to the one used in [19] for ASICs. Let S be the set of all input vector pairs for a macro logic block such that $|S|=N$. We sample a subset $s \subset S$, such that $|s|=n$. Let x_i be the power estimation result obtained from the static high-level power model, and y_i be the power estimation result obtained from the low-level power simulator. A linear function of high-level power estimation results, $\sum_{i\in s} x_i$, is used to estimate the low-level power simulation results, $\sum_{i\in s} y_i$, as follows.

$$\sum_{i\in s} y_i = n\alpha + \beta \sum_{i\in s} x_i \qquad (9)$$

where α and β are constants.

Thus, minimizing $\sum_{i\in s}(y_i - \alpha - \beta x_i)^2$ least-square-error, we get $\tilde{\alpha} = \bar{y} - \tilde{\beta}\bar{x}$ and

$$\tilde{\beta} = \frac{\sum_{i\in s}(y_i - \bar{y})(x_i - \bar{x})}{\sum_{i\in s}(x_i - \bar{x})^2} \qquad (10)$$

where $\bar{y}$ and $\bar{x}$ are the average of y_i and x_i. Then the modified power estimate, P, is given by [19]

$$P = \frac{1}{N}\left(\sum_{i\in s} y_i + (N-n)\tilde{\alpha} + \tilde{\beta}\sum_{i\notin s} x_i\right) \qquad (11)$$

6. Experimental Results

In this section, we present the experimental results to demonstrate the efficacy of our high-level power modeling approach.

We should first keep in mind that FPGA and ASIC libraries are different. In an ASIC library, if a component is presented as a hard core, it means that the layout and technology are fixed. When such a component is used in a system, with the same input sequence, the timing and power behavior will be the same. This may not be true for a component from the FPGA library. Since FPGAs constitute a parallel hardware platform, multiple tasks may execute concurrently on the same device. Combinations of different tasks change the task configuration. For the same macro logic block, the power consumption under different configurations may be different, i.e., power estimation cannot be strictly modeled with the same parameters or functions. Therefore, for high-level power modeling, multiple tasks need to be considered together, and macro logic block combinations, which occur with a high probability, need to be explored. The second point is that the high-level power models we construct are suitable for both combinational and sequential logic. Although in sequential logic, the initial states are difficult to ascertain, the estimation error introduced by unknown initial states is minimized in a long simulation.

A series of experiments was performed to verify our high-level power estimation model on a Sun Enterprise 4500 server with 4 gigabyte memory. The example circuits are (i) 8-bit ALU with control logic, (ii) 32-bit adder, (iii) FIR filter, (iv) IIR filter, (v) Huffman encoder, (vi) ALU, controller and encoder, and (vii) ALU, controller and FIR filter. The results for these examples are shown in Tables I and II. The circuits are referred to as Circuit 1-7, respectively.

Table I: Accuracy of the high-level power models

Ex.	Terms	Relative error	Max. error	Offline CPU time(hrs)	Online CPU time(sec)
Circuit1	14	0.9%	1.7%	5.6	3.3
Circuit2	10	1.1%	4.4%	10.5	5.1
Circuit3	12	1.4%	3.4%	7.5	3.1
Circuit4	12	1.0%	4.1%	8.5	4.8
Circuit5	10	2.4%	7.9%	5.1	4.1
Circuit6	16	2.3%	6.7%	5.7	7.4
Circuit7	17	2.7%	5.5%	11.9	5.3

In Table I, for each example, we show five results for static regression-based high-level power models: number of terms we use in the model, relative error, maximum error, time used to construct the model, and time used to estimate power consumption. For these examples, the average relative error is 1.7%.

Table II: Accuracy of the adaptive regression-based high-level power models

Ex.	Terms	Relative error	Sampling ratio	Online CPU time(sec)
Circuit1	14	1.2%	5%	5.8
Circuit2	10	3.3%	5%	8.4
Circuit3	12	2.6%	5%	5.5
Circuit4	12	3.3%	5%	8.0
Circuit5	10	5.0%	5%	6.9
Circuit6	16	3.4%	5%	11.7
Circuit7	17	3.1%	5%	8.7

In Table II, we show results for adaptive regression-based power models for input traces with very different statistical characteristics than those used to derive the static macromodel. The table includes the number of terms, relative error, and the vector-pairs sampling ratio (i.e., the percentage of vector pairs from the input trace that are sampled). Comparing these results with those in Table I, we note that adaptive modeling leads to relative errors that are quite comparable to those from the static macromodel. Thus, it is indeed able to adapt to different input traces without much loss in accuracy. For these examples, the average relative error is only 3.1%. Figure 5 demonstrates the agreement between high-level and low-level power estimation for part of the experimental results.

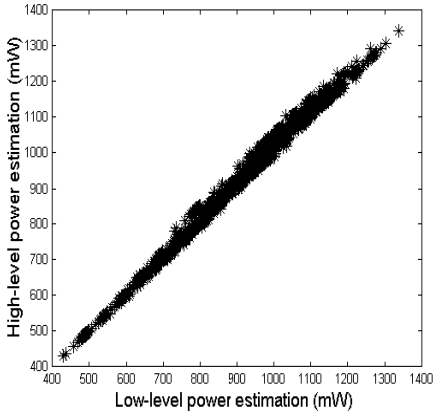

Fig. 5: High-level vs. low-level power estimation

7. Conclusions

We have presented efficient adaptive regression-based high-level power models to estimate FPGA power consumption. For each FPGA configuration, only four types of input/output statistics are needed, and only one compact function is needed to estimate the power consumption. Thus, the memory space required to store the high-level power model is also small. The methods and algorithms we use are general: (i) the same power macromodel template can be used for all modeled circuits. (ii) They do not need to explore the internal structure of the reconfigurable devices. (iii) In order to improve on-line power estimation accuracy, we use adaptive regression methods to lessen the problem of biased training sequences, and achieve a good trade-off between efficiency and accuracy.

References

[1] Virtex data sheet, http://www.xilinx.com.

[2] APEX data sheet, http://www.altera.com.

[3] FPSLIC, http://www.atmel.com.

[4] S. Trimberger, D. Carberry, A. Johnson, and J. Wong, "A time-multiplexed FPGA," in *Proc. Field-Programmable Custom Computing Machines*, pp. 22-28, Apr. 1997.

[5] K. Compton and S. Hauck, "Configurable computing: A survey of systems and software," submitted to *ACM Computing Surveys,* 2000.

[6] P. E. Landman and J. M. Rabaey, "Architectural power analysis: The dual bit type method," *IEEE Trans. VLSI Systems,* vol. 3, pp. 173-187, June 1995.

[7] A. Raghunathan, S. Dey, and N. K. Jha, "Register-transfer level estimation techniques for switching activity and power consumption," in *Proc. Int. Conf. Computer-Aided Design*, pp. 158-165, Nov. 1996.

[8] C-Y. Tsui, J. Monteiro, M. Pedram, S. Devadas, A. M. Despain, and B. Lin, "Power estimation in sequential logic circuits," *IEEE Trans. VLSI Systems,* vol. 3, no. 3, pp. 404-416, Sept. 1995.

[9] S. Gupta and F. N. Najm, "Analytical model for high level power modeling of combinational and sequential circuits," in *Proc. Alessandro Volta Memorial Wksp. Low Power Design*, pp. 164-172, Mar. 1999.

[10] S. Gupta and F. Najm, "Power modeling for high-level power estimation," *IEEE Trans. VLSI systems*, vol. 8, no. 1, pp. 18-29, Feb. 2000.

[11] K. Roy, "Power-dissipation driven FPGA place and route under timing constraints," *IEEE Trans. Circuits & Systems*, vol. 46, no. 5, pp. 634-637, May 1999.

[12] E. A. Kusse, "Analysis and circuit design for low power programmable logic modules," Master's thesis, Dept. of Electrical Engineering and Computer Science, University of California at Berkeley, 1998.

[13] V. George, H. Zhang and J. Rabaey, "Design of a low energy FPGA," in *Proc. Int. Symp. Low Power Electronics & Design*, pp. 188-193, Aug. 1999.

[14] T. Osmulski, J. T. Muehring, B. Veale, J. M. West, H. Li, S. Vanichayobon, S. -H. Ko, J. K. Antonio, and S. K. Dhall, "A probabilistic power prediction tool for the Xilinx 4000-series FPGA," in *Proc. 5th Int. Wksp. Embedded/Distributed HPC Systems and Applications (EHPC 2000), IPDPS 2000 Workshops,* pp. 776-783, May 2000.

[15] CoolRunner, http://www.xilinx.com.

[16] Scirocco simulator, http://www.synopsys.com.

[17] Power estimator for CoolRunner, http://www.xilinx.com.

[18] R.H. Myers, *Classical and Modern Regression with Application*, Duxbury Press, Belmont, CA, 2nd edition, 1989.

[19] C.-T. Hsieh, Q. Wu, C.-S. Ding, and M. Pedram, "Statistical sampling and regression analysis for RT-level power evaluation," in *Proc. Int. Conf. Computer-Aided Design*, pp. 583-588, Nov. 1996.