Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 68673, 8 pages
doi:10.1155/2007/68673

Research Article

Estimation of Power Consumption at Behavioral Modeling Level Using SystemC

Robertas Damaševičius and Vytautas Štuikys

Department of Software Engineering, Faculty of Informatics, Kaunas University of Technology, Studentų 50-415, 51368 Kaunas, Lithuania

Received 22 May 2006; Revised 9 December 2006; Accepted 6 May 2007

Recommended by Sri Parameswaran

A successful embedded system design requires thorough domain analysis and design space exploration. The aim is to develop a target system, which implements the prescribed functionality and at the same time meets the design, time, and cost-related constraints. The early evaluation of design characteristics, such as power consumption, allows the user to take advantage of many architectural design options available and to modify the system architecture, if needed. Currently, SystemC is used to model the hardware and software parts of a system at the high level. However, the characteristics of the modeled system are obtained only at the late design stages during physical synthesis. Here, we present a framework for power estimation at the modeling level of a design using macromodels. The SystemC class library is modified and extended with new classes describing the computation of power characteristics of the behavioral-level hardware models.

1. INTRODUCTION

The main aim of any design is to develop a target system that implements the prescribed functionality, constrained by the given set of requirements, in time and with minimal costs. One way to achieving the goal and, at some extent, minimizing the allocated resources is to learn the design characteristics of the developed system as early as possible. Traditionally, there are three basic design characteristics: area, delay, and energy (power) consumption. With the arrival of mobile computing and battery-powered mobile appliances, energy consumption is becoming the most important design characteristic for a wide range of electronic systems, though other characteristics remain as important as ever.

Today, a mobile electronic system can be seen as an *embedded system-on-chip* (SoC). As the complexity of such systems is rapidly growing, the designers are moving towards higher levels of abstraction in the system description, modeling, and design. Different abstraction levels (such as UML diagrams [1], platforms [2], or SystemC models [3]) are used to address different design concerns at the different level of detail. The key objective of the designer is to model the system at each abstraction layer with as a little detail as possible, and to obtain the design characteristics metrics, which are further used to make sound design decisions [4]. Analysis of design characteristics is vital in the early stages of the design process with many design space exploration options for determining the SoC architecture and selecting or trading off the key components of the designed system.

Though providers of the commercial synthesis tools are showing an increased interest in SoC design, their tools usually implement a top-down design approach that requires the SoC designer to fully define the function of a developed system, repeatedly decompose coarse-grained functions into smaller subfunctions, and then map them onto the available hardware (HW) library cores [5]. Using such a design methodology, the primary design characteristics can be estimated only in the final stages of the design. The redesigning of the system, in case of any mismatch with design constraints, is very costly and hardly possible within a given design timeframe. That can be one of the reasons, why 85% of SoC design projects miss their target date [6]. Even minor modifications require large design efforts, because the entire process is time consuming. It may take two or more weeks to rebuild a moderately modified SoC to a physical implementation ready for verification [6].

Thus, with continuing system complexity growth, it is increasingly critical to address power consumption early in the design cycle, for example, at the behavioral or even at the system design level [7, 8]. On the other hand, at system level, there are significant opportunities to optimize the system architecture for meeting design constraints and improving design characteristics [9, 10].

At the system level of abstraction, SystemC [3] is becoming widely accepted as a standard tool for modeling complex embedded systems. SystemC is an extension of C++ with a library of classes for HW simulation; however, it lacks the semantics to capture energy and other HW-related design information. Nevertheless, the object-oriented nature of the SystemC library allows extending it to provide the designer with more information than just plain waveforms.

The aim of the paper is to show how, using power macromodeling, we can estimate the power consumption at the behavioral level of SystemC.

The paper is organized as follows. Section 2 overviews the related work and summarizes our contribution. Section 3 describes the estimation of power at the behavioral level of SystemC. Section 4 presents a case study demonstrating the validity of our approach for the behavioral level models in SystemC. Section 5 evaluates the results and presents a discussion. Finally, Section 6 presents the conclusions and outlines future research directions.

2. RELATED WORK AND OUR CONTRIBUTION

Recently, with the move towards system-level specifications and design methodologies in SoC design, there has been a significant research interest in power estimation and modeling. A detailed survey of the high-level power modeling and optimization techniques is presented in [11]. The consumed power can be estimated at least at five different levels of abstraction.

(1) *Transistor-level* methods simulate the circuit at the transistor or switch level and monitor the supply current [12].

(2) *Gate-level* methods simulate a design at the logic gate level and calculate power using switching activity and node capacitance [13].

(3) *Register transfer (RT) level* estimation methods model the power consumption of middle-grained components such as multiplexers, adders, multipliers, and registers [14–16]. The primary difference from the gate-level method is the complexity of analyzed components.

(4) *Behavioral-level* methods model the power consumption based on the functional or algorithmic descriptions of HW components [17–19].

(5) *System-level* methods estimate power dissipation based on the high-level system descriptions using abstract models of capacitance and switching [20, 21].

Most of the previous research has focused on the gate and transistor levels. Here, the available information on the structure and characteristics of domain entities allows obtaining accurate power estimates. However, the increasing size and complexity of developed systems and the move towards higher levels of abstraction in describing HW and embedded systems raised the need for higher-level power estimation methods, too. At the behavioral and system level of abstraction, SystemC is becoming widely accepted as a tool for modeling embedded systems. Recently, *Orinoco* [22], a commercial VHDL RT-level power modeling tool, has extended its support for SystemC models, but only using a standard design cell library.

The power estimation usually requires a variety of *power models*. Their complexity and granularity depend upon the level of abstraction the system is modeled on. Thepower modeling and estimation at the gate and transistor levels are pretty straightforward and require the application of the common mathematical formulae, which take the physical characteristics of domain entities into account [12, 13]. At the higher level of abstraction, such as RTL, behavioral and system levels, high-level power models (e.g., *power handlers* [23], *templates* [15], *power estimators* [24], and *power monitors* [25]) tend to be more complex and abstract, more relative than absolute, and less accurate. The reason for this is the lack of physical implementation details in the high-level system models and the complexity of modeling large and complex systems at a high level.

Therefore, instead of highly specific physical data such as capacitance and switching activity used to obtain the specific power consumption values at low level, high-level power models use more indirect and approximate design parameters, such as *signal entropy* [18, 26], or *abstract metrics* [27]. Note that the focus here is not on the fine-grained power estimation of specific HW components, but on the coarse-grained relative comparison of HW architectural options with respect to the estimated power consumption values.

Xanthos et al. [23] propose a modification to the SystemC library to enable power estimation of digital systems built upon a set of primitive logic gates. Minor modifications of SystemC modules enable the calculation of the dynamic power component due to logic transitions on the nodes of a digital circuit.

This paper was primarily inspired by the work of [23]. It is also an extension of our previous work [28] in estimation of design characteristics of HW systems. Our novelty is the framework combining power models for power estimation at the RTL and power macromodels for power estimation at the behavioral SystemC level, which allows obtaining design characteristic values at the early modeling stage of design.

At the behavioral modeling level, the basic idea of our approach is the overloading of behavioral operations (logic, arithmetic, etc.) to obtain their power consumption estimates during simulation. Thus, our contribution is the estimation of power consumption characteristics of HW systems modeled at the behavioral level using SystemC modeling language.

3. ESTIMATION OF POWER AT BEHAVIORAL LEVEL

3.1. Macromodeling

The estimation of power at the behavioral level of design is much more complex as compared to the estimation of power at the RT level. First, the behavioral description is not HW oriented and looks much the same as any software program. Its mapping to the HW architecture may be ambiguous, different implementation strategies can be used. Second, here we can not rely on the specific technological library components.

At the behavioral level, computation of power must be approximated in order to account for the limited knowledge of the circuit. Therefore, a number of the high-level analysis techniques such as *statistical analysis* [17], *stochastic* methods [19], and *macromodeling* [29–33] are used. These techniques are usually based on the development of abstract power models, which are used for design space exploration to evaluate the relative impact of design decisions on the quality and characteristics of the final design. The estimated power consumption values provided by such models are neither absolute nor physically accurate, because at the high level of abstraction the limited knowledge of the physical structure of the design does not allow to compute meaningful power estimates [18].

Such models can be built analytically by deriving a formula for each behavioral operation, which depends on a number of physical parameters such as switching or capacitance. Another way is to develop an empirical model or *macromodel*, which is based on the approximation of the actually measured power dissipation values. The basic idea behind power macromodeling is to generate a mapping between the power dissipation of a circuit and certain statistics of its input signals. Such macromodels can be used during modeling instead of detailed hardware models resulting in modeling speedup.

We have employed the macromodeling technique, because it allows us to apply our earlier results [28], achieved at the RT level of power modeling, for the estimation of power at the behavioral level of design.

In the analytical power macromodeling, a function maps the space of input signal properties to the power dissipation of a circuit. When the input parameters of the macromodeling function are solely determined by the input signals, the computation of power estimates is a straightforward and a fast function evaluation. The key challenges in the analytical macromodeling are the choice of the appropriate input parameters for the macromodel and the derivation of the macromodel function.

Consider a combinatorial circuit with n input signals. The variables that correspond to the input nodes are a concatenation of all input signal values as follows: $X = (x_1 x_2 \cdots x_n)$. We will use two variables (or "states"), one representing the value before the transition (X_a) and the other one representing the value after the transition (X_b). The power consumption then is expressed as a function of the previous input state and the current input state of the circuit as follows: $P = f(X_a, X_b)$.

Such function represents a macromodel of the power consumed by a specific circuit. The derivation of such macromodel may be a complex task. The complexity of the modern embedded systems is a significant challenge for the creation and use of macromodels. The designer needs to perform a thorough domain analysis to create suitable and accurate macromodels.

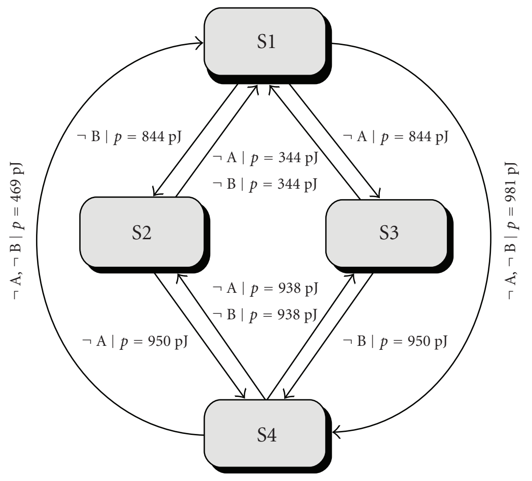

Figure 1: Exact power macromodel of 1-bit half adder.

For simple circuits, we can describe a *precise* macromodel, which models the powerconsumption of a circuit at the same level of accuracy as power estimates given by a synthesis tool for a specific manufacture technology. However, for more complex circuits, we need to introduce some simplifications, which allow to model power consumption with satisfactory accuracy, while they allow fast derivation of the power estimation results.

Therefore, power macromodels can be derived using three methods: composition, simplification, and analytical reasoning. Using *composition*, the macromodel of a circuit is derived by composing the macromodels of circuit components based on the circuit architecture. Such macromodels have the same level of accuracy as the macromodels of the circuit components. However, their complexity is significantly larger as the number of states, which the circuit has, usually increase.

An example of the exact macromodel is presented in Figure 1. Here, we have a power macromodel FSM of the 1-bit half adder with 4 states, each representing different values of the input signals (e.g., S1 means A = 0, B = 0; S2 means A = 0, B = 1). Each transition causes power to be dissipated, depending upon previous and current inputs of the half adder. Such macromodel was derived from simpler power macromodels of AND and XOR gates.

Note that here we consider only dynamic power consumption P_d, which depends upon switching activity and the size of switched capacitance. Dynamic power consumption accounts for the largest portion of the total consumption of power in digital circuits [34]. It is calculated as the sum of all switched input and output capacitancesmultiplied by power supply voltage V_{DD} as follows:

$$P_d = \sum_{m \in M} \left(c_{\text{in}} t_{\text{in}} + c_{\text{out}} t_{\text{out}} \right) V_{\text{DD}}^2, \qquad (1)$$

```
// calculate the number of bit transitions
int trans(int A, int B)
{
   if (!A && !B) return 0;
   if ((A%2) != (B%2)) return 1 + trans(A/2, B/2);
               else return trans(A/2, B/2);
}

double adder_model(int A_prev, int A, int B_prev, int B)
{
    return VDD * VDD *
              (INP_CAP * (trans(A_prev, A) +
                     trans(B_prev, B)) +
              OUT_CAP * trans(A_prev+B_prev, A+B);
}
```

Figure 2: Simple n-bit adder macromodel in C++.

where t_{in}, and t_{out} are the number of input and output transitions from logic "0" to logic "1," c_{in}, and c_{out} are the input and output capacitances that depend upon the gate type and the technology used, and m is a component of a system M.

The second method, *simplification*, can be used to reduce the complexity of a macromodel, for example, when constructing full adders from half adders, with comparatively small loss of accuracy.

However, for complex circuits with many input signals and circuit transition states we must use *analytical reasoning* to derive a power macromodel of a circuit, because macromodel derivation by composition is too complex and impractical. For example, if we derive a 16-bit adder power macromodel from a 1-bit adder power macromodel, it will have 2^{32} states, which is very impractical to represent and use. Instead, we can analytically derive a power macromodel for a 16-bit adder, which can be calculated and represented with significantly less complexity. An example of such power macromodel implemented in C++ is shown in Figure 2.

In Figure 2, the power consumption of an adder is calculated using (1), where the dynamic voltage VDD, input capacitance INP_CAP, and output capacitance OUT_CAP constant values are set for a specific implementation technology.

3.2. Modification of SystemC library

For the estimation of power at the behavioral level, we have extended the SystemC class library with an additional library of the macromodel classes that implement the modeling of basic behavioral level operations (see Table 1).

Also, we have extended SystemC *sc_signal* class, which describes SystemC model signals, with overloaded methods and class attributes for storing previous and current signal values. These methods are used to store previous signal values and call the required macromodel class (e.g., *sc_mmRegister* and *sc_mmAdder* classes are shown as an example) (see Figure 3).

Table 1: Mapping of behavioral SystemC operations into HW components.

SystemC operation	HW component
$+, -$	Adder/substractor
$=$	Trigger/register
$<, >, >=, <=, ==, !=$	Comparator
&	AND gate
$\|$	OR gate
$\wedge$	XOR gate
$\sim$	INV gate
$*$	Multiplier
$\ll, \gg$	Shift register

The class also contains methods for calculating circuit area and delay, which are not considered in this paper.

For each behavioral operation, we have implemented its own macromodel class (e.g., *sc_mmAdder* for "+" operation), which performs power modeling depending upon input signal values and computes power estimates. Since input signals can have different data types, the overloaded methods were generalized using *class templates* and specialized using the *template specialization* technique. Figure 4 gives an example of the overloaded *sc_signal* class method, which performs the addition operation and writes the result into a register.

For the behavioral power estimation, for each operation of the modeled system we need to implement the *calcPower()* method to calculate the number of 0-to-1 bit transitions at its input and output ports, and to calculate the corresponding dynamic energy consumption using (1). The values of the input and output signals are stored and used for next estimation of power consumption. The pseudocode of the *calcPower()* method is given in Figure 5.

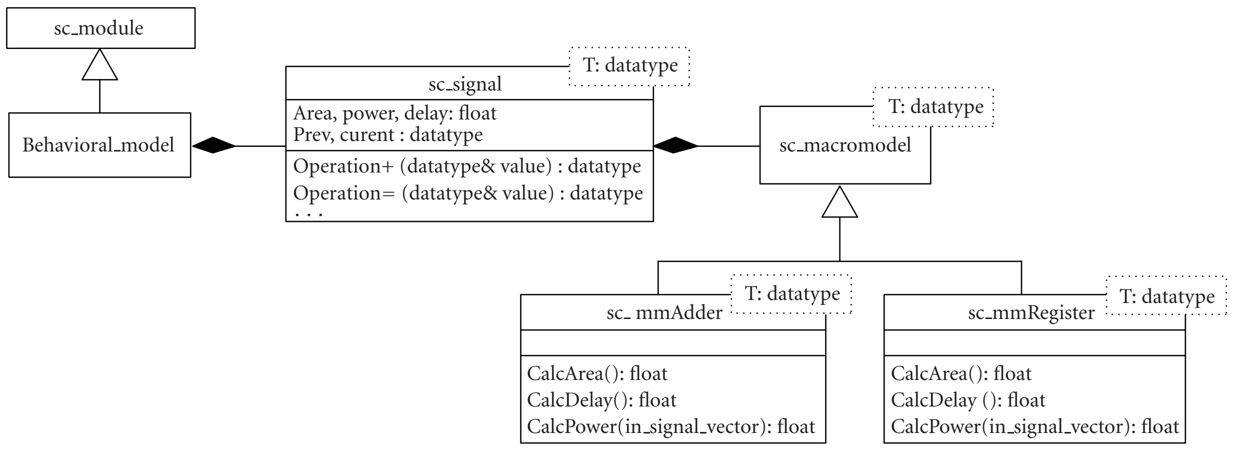

Figure 3: SystemC extension for power estimation at the behavioral level.

```
template <class T>
class sc_signal: public sc_signal_inout_if<T>,
                 public sc_prim_channel
{
public:
    void operator+( const T& value_ ) {
         prev = (*this)->read();
         (*this)->write( prev + value_ );
         current = (*this)->read();

         /* estimation of characteristics */
         power += sc_mmAdder<T>::calcPower(prev,current);
    }
}
```

Figure 4: An example of the overloaded sc_signal method for estimation of a behavioral operation.

```
calculate outputs of operation
for each input signal in operation
    increase input 0-to-1 transitions counter
end for
for each output signal in operation
    increase output 0-to-1 transitions counter
end for
calculate consumed power as
        a function of input transitions
                  and output transitions
```

Figure 5: Pseudocode of the calcPower() method.

```
SC_MODULE(counter) {
   sc_in<bool> clk;
   sc_out<int> cnt;

   void do_count() {
        cnt += 1;
        if (cnt >= 9 ) cnt = 0;
   };
   SC_CTOR(counter) {
     SC_METHOD(do_count);
     sensitive_pos ≪ clk;
   }
};
```

Figure 6: Counter model in SystemC.

4. CASE STUDY

As a case study for the estimation of power at the behavioral modeling level, we consider a 4-bit counter, which performs incremental 0-to-9 counting. The description of the SystemC model is given in Figure 6.

For this counter model, we have constructed two power macromodels. Using Table 1, from the behavioral description of the counter model we have constructed separate power models for each component of the counter (register,

Table 2: Design characteristics of 4-bit counter.

Counter	Est. power, pW		Average error
	Exact model	Analytical macromodel	
4-bit	37.369	37.369	—
8-bit	74.736	79.441	6.3%
12-bit	112.107	118.708	5.9%
16-bit	149.476	158.263	5.9%

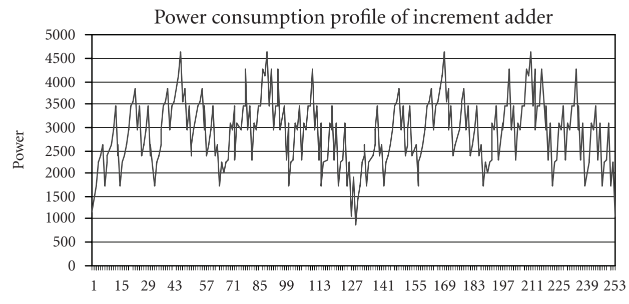

Figure 7: Power consumption of the 8-bit increment adder depending upon incremented input value.

adder, comparator, multiplexer), and used them for deriving the *exact* power macromodel of the counter (an example of such derivation is given in Figure 1).

Also, we have constructed the *analytical* macromodel of the counter, which is similar to the macromodel presented in Figure 2. The results of the estimation of the power consumption for the different bit-width counters using both power macromodels are compared in Table 2. Accuracy of the analytical macromodel as compared with the exact power macromodel is about 6%.

Our framework allows monitoring both average power consumption and power consumption depending upon the supplied input values. The power consumption profile of the 8-bit increment adder from the exact counter power macromodel is given in Figure 7. It shows how much power is dissipated during each adding operation for the input values from 0 to 255. As wecan see, even for such a simple component, the power consumption values vary by a factor of 5.2 depending upon the specific input values, which further underscores the difficulties associated with behavioral power modeling.

5. EVALUATION AND DISCUSSION

The main benefits of the presented behavioral power estimation framework are as follows.(1) The framework allows estimating power consumption at a higher level of abstraction, which means faster modeling and testing of the design. (2) The framework allows power estimation at an early design stage. It allows a designer to select more efficient hardware implementations or to modify design architecture, which does not satisfy given design constraints, with less pain and cost, thus decreasing time-to-market and increasing overall designer productivity.

The described framework also has some drawbacks. Since more computations are performed during modeling of a system, the system is modeled slower than without the power estimation. The slowdown caused by the power estimation is about 68%, which is a satisfactory result, considering that other papers report up to 8.5 slowdown factor of the modeling speed incurred by the estimation of design characteristics [25] (note that the direct comparison is not possible due to the different functionality of the power estimation frameworks and different complexity of the test cases). For example, for the 8-bit increment adder the modeling time without the power estimation is 7.5 milliseconds, and with the power estimation is 12.8 milliseconds, which is an increase of 71%. The slowdown of the modeling speed mainly depends upon the computational complexity of the developed power macromodels and the complexity of the behavioral descriptions of the modeled systems, and may vary significantly across the domain.

The power estimation at the behavioral level is much more complex than at the RT level, because it requires to perform thorough domain analysis and develop high-level power macromodels that estimate the power consumed by the specific behavioral operations. The designer must carefully choose a tradeoff between the number of power states (i.e., the complexity of calculations) and modeling speed to achieve efficient power modeling and estimation.

Validation of a power macromodel is a complicated problem. Precisely speaking, we can estimate the accuracy of the power model only when comparing it with the real-life power consumption measurements of a physically implemented system, which may depend upon many other physical factors such as environment temperature. However, in practice, the power estimation results of a macromodel are compared with other results obtained using other power models, which are considered as exact (e.g., with the power estimates given by the commercial synthesis tools). Here, we estimate accuracy of the analytical power macromodels by comparing them with the exact power models, which are developed by composing the lower-level power models down to the Boolean logic level. Our results, in terms of accuracy, are within the range of results achieved by other authors [35].

More detailed power models may increase the accuracy of the power estimation, but could slow power analysis and consequently may prevent an extensive design space exploration. The absolute accuracy of the results may not be as important for the designer as the relative accuracy, because the designer uses the results of the high-level power analysis to adopt early design decisions that have a positive impact on the final product.

6. CONCLUSIONS AND FUTURE WORK

The presented SystemC model estimation framework allows for early estimation and analysis of the power consumption characteristics. Such an analysis already at the early stage of the design process can indicate whether the designed system would match the imposed design constraints. Based on the results of the analysis, the designer can select a particular system architecture that can lead to a more efficient hardware implementation. Dynamic energy profiling helps to better understand the dynamic properties of the designed system, which may be helpful for power optimization of mobile devices.

Future work will address the design space exploration by providing the estimation of the design characteristics for different technological libraries, and further development of the power estimation macromodels in SystemC to allow more sophisticated power consumption analysis of the SystemC models.

REFERENCES

[1] L. Lavagno, G. Martin, and B. V. Selic, Eds., *UML for Real: Design of Embedded Real-Time Systems*, Springer, New York, NY, USA, 2003.

[2] G. Martin and H. Chang, Eds., *Winning the SoC Revolution: Experiences in Real Design*, Springer, New York, NY, USA, 2003.

[3] T. Grötker, S. Liao, G. Martin, and S. Swan, *System Design with SystemC*, Springer, New York, NY, USA, 2002.

[4] C. Talarico, J. W. Rozenblit, V. Malhotra, and A. Stritter, "A new framework for power estimation of embedded systems," *Computer*, vol. 38, no. 2, pp. 71–78, 2005.

[5] J. A. Darringer, R. A. Bergamaschi, S. Bhattacharya, et al., "Early analysis tools for system-on-a-chip design," *IBM Journal of Research and Development*, vol. 46, no. 6, pp. 691–708, 2002.

[6] L. Albanese, "Restoring predictability in SoC integration," *EETimes*, 2004.

[7] Y. Li and J. Henkel, "A framework for estimating and minimizing energy dissipation of embedded HW/SW systems," in *Proceedings of the 35th Design Automation Conference (DAC '98)*, pp. 188–193, San Francisco, Calif, USA, June 1998.

[8] K. Lahiri, A. Raghunathan, and S. Dey, "Efficient power profiling for battery-driven embedded system design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 6, pp. 919–932, 2004.

[9] J. M. Rabaey and M. Pedram, Eds., *Low Power Design Methodologies*, Kluwer Academic Publishers, Norwell, Mass, USA, 1996.

[10] A. Raghunathan, N. K. Jha, and S. Dey, *High-Level Power Analysis and Optimization*, Kluwer Academic Publishers, Norwell, Mass, USA, 1998.

[11] E. Macii, M. Pedram, and F. Somenzi, "High-level power modeling, estimation, and optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 11, pp. 1061–1079, 1998.

[12] S. M. Kang, "Accurate simulation of power dissipation in VLSI circuits," *IEEE Journal of Solid-State Circuits*, vol. 21, no. 5, pp. 889–891, 1986.

[13] T. H. Krodel, "PowerPlay—fast dynamic power estimation based on logic simulation," in *Proceedings of IEEE International Conference on Computer Design (ICCD '91)*, pp. 96–100, Cambridge, Mass, USA, October 1991.

[14] S. Ravi, A. Raghunathan, and S. T. Chakradhar, "Efficient RTL power estimation for large designs," in *Proceedings of the 16th International Conference on VLSI Design*, pp. 431–439, New Delhi, India, January 2003.

[15] V. Krishna and N. Ranganathan, "A methodology for high level power estimation and exploration," in *Proceedings of the 8th IEEE Great Lakes Symposium on VLSI*, pp. 420–425, Lafayette, La, USA, February 1998.

[16] A. Raghunathan, S. Dey, and N. K. Jha, "Register-transfer level estimation techniques for switching activity and power consumption," in *Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD '96)*, pp. 158–165, San Jose, Calif, USA, November 1996.

[17] B. Arts, N. Eng, M. J. M. Heijligers, et al., "Statistical power estimation of behavioral descriptions," in *Proceedings of the 13th International Workshop on Integrated Circuit and System Design, Power and Timing Modeling, Optimization and Simulation (PATMOS '03)*, vol. 2799 of *Lecture Notes in Computer Science*, pp. 197–207, Springer, Torino, Italy, September 2003.

[18] F. Ferrandi, F. Fummi, E. Macii, and M. Poncino, "Power estimation of behavioral descriptions," in *Proceedings of Design, Automation and Test in Europe (DATE '98)*, pp. 762–766, Paris, France, February 1998.

[19] R. Mehra and J. Rabaey, "Behavioral level power estimation and exploration," in *Proceedings of the 1st International Workshop on Low Power Design*, pp. 197–202, Napa Valley, Calif, USA, April 1994.

[20] A. Abril, H. Mehrez, F. Petrot, J. Gobert, and C. Miro, "Energy estimation and optimization in architectural descriptions of complex embedded systems," in *VLSI Circuits and Systems II*, vol. 5837 of *Proceedings of SPIE*, pp. 456–466, Seville, Spain, May 2005.

[21] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: a framework for architectural-level power analysis and optimizations," in *Proceedings of the 27th Annual International Symposium on Computer Architecture (ISCA '00)*, pp. 83–94, Vancouver, BC, Canada, June 2000.

[22] F. Schirrmeister, "Design for Low-Power at the Electronic System Level," ChipVision Design Systems, White paper, 2004.

[23] S. Xanthos, A. Chatzigeorgiou, and G. Stephanides, "Energy estimation with systemC: a programmer's perspective," in *Proceedings of the 7th International Conference on Systems Computational Methods in Circuits and Systems Applications*, pp. 1–6, WSEAS Press, Corfu, Greece, July 2003.

[24] M. Lajolo, A. Raghunathan, S. Dey, and L. Lavagno, "Efficient power co-estimation techniques for system-on-chip design," in *Proceedings of Design, Automation and Test in Europe (DATE '00)*, pp. 27–34, Paris, France, March 2000.

[25] N. Bansal, K. Lahiri, A. Raghunathan, and S. T. Chakradhar, "Power monitors: a framework for system-level power estimation using heterogeneous power models," in *Proceedings of the 18th IEEE International Conference on VLSI Design*, pp. 579–585, Kolkata, India, January 2005.

[26] M. Nemani and F. Najm, "Towards a high-level power estimation capability," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 6, pp. 588–598, 1996.

[27] W. Fornaciari, P. Gubian, D. Sciuto, and C. Silvano, "Power estimation of embedded systems: a hardware/software codesign approach," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 6, no. 2, pp. 266–275, 1998.

[28] R. Damaševičius, "Estimation of design characteristics at RTL modeling level using systemC," *Information Technology and Control*, vol. 35, no. 2, pp. 117–123, 2006.

[29] G. Bernacchia and M. C. Papaefthymiou, "Analytical macromodeling for high-level power estimation," in *Proceedings of IEEE/ACM International Conference on Computer-Aided*

Design (ICCAD '99), pp. 280–283, San Jose, Calif, USA, November 1999.

[30] A. Bogliolo and L. Benini, "Robust RTL power macromodels," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 6, no. 4, pp. 578–581, 1998.

[31] A. Bogliolo, L. Benini, and G. De Micheli, "Regression-based RTL power modeling," *ACM Transactions on Design Automation of Electronic Systems*, vol. 5, no. 3, pp. 337–372, 2000.

[32] S. Gupta and F. N. Najm, "Analytical models for RTL power estimation of combinational and sequential circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 7, pp. 808–814, 2000.

[33] R. Zafalon, M. Rossello, E. Macii, and M. Poncino, "Power macromodeling for a high quality RT-level power estimation," in *Proceedings of the 1st IEEE International Symposium on Quality of Electronic Design (ISQED '00)*, pp. 59–63, San Jose, Calif, USA, March 2000.

[34] L. Benini and G. De Micheli, *Dynamic Power Management: Design Techniques and CAD Tools*, Kluwer Academic Publishers, Norwell, Mass, USA, 1997.

[35] L. Shang and N. K. Jha, "High-level power modeling of CPLDs and FPGAs," in *Proceedings of the 19th IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD '01)*, pp. 46–53, Austin, Tex, USA, September 2001.