

# Power Estimation of Embedded Systems: A Hardware/Software Codesign Approach

William Fornaciari, *Member, IEEE*, Paolo Gubian, *Member, IEEE*, Donatella Sciuto, *Member, IEEE*,  
and Cristina Silvano

**Abstract**— The need for low-power embedded systems has become very significant within the microelectronics scenario in the most recent years. A power-driven methodology is mandatory during embedded systems design to meet system-level requirements while fulfilling time-to-market. The aim of this paper is to introduce accurate and efficient power metrics included in a hardware/software (HW/SW) codesign environment to guide the system-level partitioning. Power evaluation metrics have been defined to widely explore the architectural design space at high abstraction level. This is one of the first approaches that considers globally HW and SW contributions to power in a system-level design flow for control dominated embedded systems.

**Index Terms**— Embedded systems, hardware/software codesign, low-power design, power estimation.

## I. INTRODUCTION

EMBEDDED systems are those computing and control systems designed for dedicated applications [1] where *ad hoc* software routines are provided to respond to specific requirements. The diffusion on the semiconductor market of standard processors characterized by high performance and reasonable prices contributed to increase the importance of embedded systems. The typical embedded system architecture is constituted by one or more dedicated hardware units such as application specific integrated circuits (ASIC's) to implement the hardware part and a set of software routines running on a dedicated processor or application specific instruction processor (ASIP) for the software part. Exploiting the advantages offered by submicron complementary metal-oxide-semiconductor (CMOS) technologies, the entire embedded system can be implemented on a single ASIC, including the processor core, the on-chip memory, the input/output (I/O) interface and the custom hardware part.

Innovative codesign techniques emerged as a new computer-aided design (CAD) discipline in the recent past, to cope with the complexity of a comprehensive exploration of the design alternatives in the hardware/software design space. Codesign aims at meeting the system-level requirements by using a concurrent design and validation methodology, thus exploiting the synergism of the hardware and the software parts.

Several design tasks are covered during the codesign process, mainly system-level modeling, capture of the functional cospecification, analysis and validation of the cospecification, system-level partitioning, exploration and evaluation of several architectures with respect to given design metrics, cosynthesis and cosimulation. The availability of a codesign methodology, covering all these design phases, is mandatory during embedded systems design to meet the system-level requirements.

The overall system costs and performance are greatly impacted by the effects of the partitioning task, that targets the assignment of operations to the hardware (HW) or software (SW) parts. To guide the partitioning process, design metrics should be defined to compare alternative partitionings and to evaluate their conformance with respect to the system requirements, typically defined in terms of performances, area, power, costs, etc. Moreover, the design of embedded systems is often over-constrained. Thus, a solution satisfying all those constraints is difficult to be identified in a reasonable design time. As a result, only a partial exploration of the architectural design space can be usually carried out, to get to an acceptable solution, far from the optimal one.

The importance of the power constraints during the design of embedded systems has continuously increased in the past years, due to technological trends toward high-level integration and increasing operating frequencies, combined with the growing demand of portable systems. Despite of the increasing importance of power consumption in most of the embedded applications, only a few codesign approaches take into account such a goal at the higher levels of abstraction.

While several power estimation techniques have been proposed in literature at the gate, circuit and layout levels [2], a few papers have been published addressing the power estimation problem at high-level until recently [3], [4], despite the increasing interest in the system and behavioral levels. According to [3], high-level power estimation techniques can be classified depending on their abstraction level.

The average power is strongly related to the switching activity of the circuit nodes, hence power estimation can be considered a pattern-dependent process. In particular, the input pattern-dependency of the power estimation approaches can be classified as strong or weak pattern-dependency [4]. Main advantages of the strongly pattern-dependent process, based on extensive simulations, derive from their accuracy and wide applicability. However, to obtain a complete and accurate power estimation, the designer should provide a

Manuscript received March 15, 1997; revised July 1, 1997.

W. Fornaciari and D. Sciuto are with Politecnico di Milano, Dipartimento di Elettronica e Informazione, Milano 20133 Italy.

P. Gubian and C. Silvano are with the Università di Brescia, Dipartimento di Elettronica per l'Automazione, Brescia 25123 Italy.

Publisher Item Identifier S 1063-8210(98)02948-5.

comprehensive amount of input patterns to be simulated, thus making this approach very time consuming and computationally very costly. To avoid the need of a large amount of input patterns, the weakly pattern-dependent approaches require input probabilities, reflecting the typical input behavior, but the estimated results will depend on the user-supplied input probabilities.

High-level power estimation is a key issue in the early determination of the power budget for embedded systems. However, high-level power estimation methods [5] have not yet achieved the maturity necessary to enable their use within current industrial CAD environments. Our work is an attempt to fill such a gap, by providing a set of metrics based on a high-level power model, to cover the different parts composing the basic architecture of embedded systems. The goal is to widely explore the architectural design space during the system-level partitioning and to early retarget architectural design choices. Accuracy and efficiency should be the driving forces to meet the power requirements, avoiding redesign processes. In general, the relative accuracy in high-level power estimation is much more important than the absolute accuracy, the main objective being the comparison of different design alternatives [3].

The aim of this paper is to define a power evaluation codesign methodology. The method is part of a more general HW/SW codesign approach for control dominated embedded systems. The related CAD environment, called TOSCA (TOols for System Codesign Automation) [6], among other design quality estimation techniques, provides accurate and efficient power metrics to guide the system-level partitioning. Metrics suitable for power evaluation of both the hardware and software parts are defined.

The availability of a high-level power analysis is of paramount importance to obtain early estimation results, while maintaining an acceptable accuracy and a competitive global design time. Based on these results, tradeoff considerations can be carried out in a reasonable time, by avoiding to follow the entire design flow to get power comparison results. Our approach can be considered as one of the first attempts to cover power estimation issues from a HW/SW comprehensive perspective, mainly focusing on the hardware part and considering a general architecture adopted by most industrial synthesis systems.

The paper is organized as follows. Foundations and notations constituting the background of our analysis are shown in Section II. Power metrics to guide the system-level partitioning are derived in Section III, while the proposed power models for the HW and SW parts are addressed in Section IV and V, respectively. Simulation results are also provided in Section VI, to demonstrate the advantages offered by the proposed methodology during the development of control dominated embedded systems. Finally, concluding remarks are drawn in Section VII.

## II. BACKGROUND OF THE ANALYSIS

Let us introduce the general formalism to express power dissipation, the TOSCA codesign framework and the target system architecture.

### A. Power Dissipation in CMOS Circuits

Power dissipation in CMOS devices is composed of both a static and a dynamic component. Anyway, the dominant part [7] is the dynamic part, expressed by the switching activity power  $P = V_{DD}^2 f_{CLK} C_{EFF}$  where  $V_{DD}$  is the supply voltage,  $f_{CLK}$  is the system clock frequency and  $C_{EFF}$  is the effective switched capacitance (that is the product of the total physical capacitance  $C_{Li}$  of each node in the circuit and the switching activity factor  $\alpha_i$  of each node summed over all the  $N$  nodes in the circuit).

The switching activity of each signal is fully characterized by a *static* and a *dynamic* component. The static component can be expressed in terms of the *static signal probability* ( $p_n^1$ ) of each node  $n$ , that is the probability of the node to be at one (therefore,  $p_n^1 \leq 1$  and  $p_n^0 = 1 - p_n^1$ ). A signal is called *equiprobable* when  $p_n^1 = p_n^0 = 0.5$ . The *transition probability* ( $p_n^{01}$ ) is the probability of a zero to one transition at node  $n$ . In the spatial and temporal independence assumption [4],  $p_n^{01}$  is given by the probability that the current state is zero times the probability that the next state is one  $p_n^{01} = p_n^0 p_n^1 = (1 - p_n^1) p_n^1$ . Under the same assumption, the *switching activity* of a node  $n$  ( $\alpha_n$ ) is  $\alpha_n = p_n^{01} + p_n^{10} = 2p_n^1(1 - p_n^1)$ , while the *toggle rate* ( $TR_n$ ) is  $TR_n = \alpha_n f_{CLK}$ .

### B. The TOSCA Codesign Flow

The design flow of the TOSCA codesign environment, where the present work is going to be integrated, is shown in Fig. 1. Main goal is to reduce the impact of the system integration and design constraints verification bottlenecks on the global design time, thus allowing a cost-effective evaluation of alternative designs.

The design capture is performed via a mixed textual/graphical editor based on a OCCAMII customization [6] improving the user friendliness and gathering in the same design database timing constraints, design requirements, design goals and possibly an initial HW versus SW allocation of the modules composing the system. If the latter information is left unspecified by the user, an initial allocation is decided based on the results of an heuristic, by statically inspecting the properties of the system description. The main part of the codesign flow is represented by the design space exploration, i.e., a “what if” analysis of alternative architectural solutions to discover an acceptable final system modularization and HW versus SW allocation fulfilling the initial requirements and goals. This is obtained by evaluating system properties through a set of metrics, by applying system-level transformations, producing new modularization of the system specification semantically equivalent to the original one. When an acceptable partitioning is found, synthesis of the HW and SW parts can be performed. The SW synthesis passes through an intermediate uncommitted format, called virtual instruction set (VIS) [8], allowing the designer to consider the timing performance when different CPU cores are employed and to make possible a flexible simulation of the cooperating HW and SW based on the same VHDL simulator engine. HW-bound modules and interfaces are automatically converted into suitable VHDL templates. Finally, simulation

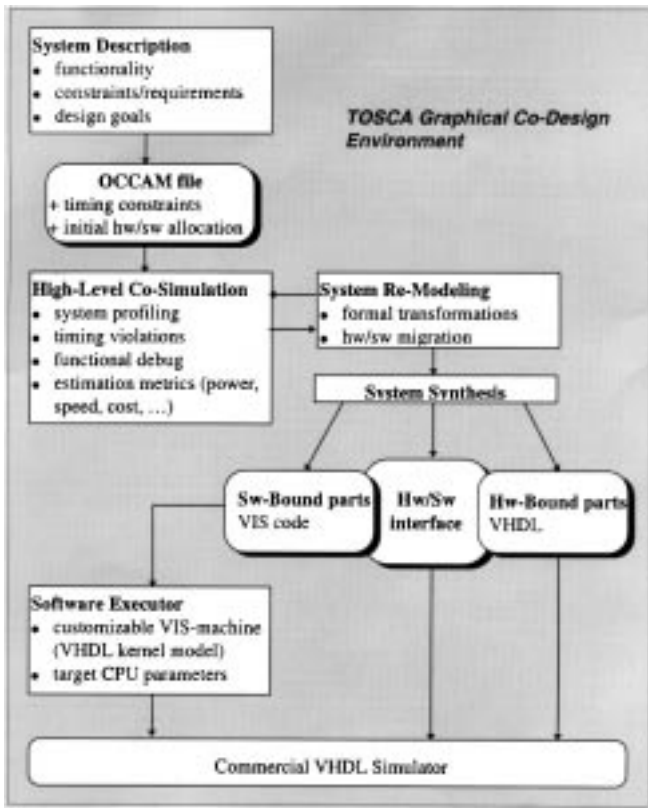


Fig. 1. The TOSCA codesign roadmap.

of the HW/SW system is performed, considering the side-effects due to the HW/SW based communication and the different performance of HW and SW technologies.

The task of system-level partitioning should provide alternative solutions in terms of the cost/performance ratio. To afford the partitioning process with respect to the design constraints, it is necessary to define a *cost function*, based on some metrics. Thus, a preliminary and iterated phase is a metric-based analysis of the system-level description. Design metrics, considering the contribution of both the HW and SW parts, can be conceived to evaluate the quality of a partitioning solution in terms of fulfillment of several design optimization criteria [6], such as performance, cost, resource exploitation, communication and power consumption. The current version of TOSCA evaluates a set of static and dynamic metrics, based on the analysis of the object oriented representation of the specification, high-level simulation and profiling. Metrics to evaluate area and performances are described in [6], while metrics for power analysis are the subject of such paper.

### C. The Target System Architecture

The system-level architecture of the embedded system is implemented within a single ASIC, including both the HW and SW parts. The target architecture is presented in Fig. 2.

The single ASIC architecture is defined at the RT-level and it is composed of the following parts.

- 1) *Data Path*—including storage units, functional units, and multiplexers. A two-level multiplexer structure is

considered for the interconnection among registers and functional units and the typical operations imply a register-to-register transfer;

- 2) *Main Memory*—to be accessed through input/output registers;
- 3) *Control Unit*—implemented as a set of finite state machines (FSM's);
- 4) *Embedded Core Processor*—such as a general-purpose standard processor, a microcontroller, a DSP, etc., with its memory (even if part of the memory can be external) implementing the SW part;
- 5) *Clock Distribution Logic*—including the buffers of the clock distribution network;
- 6) *Crossbar Network*—to interface the architectural units by using a communication protocol at the system-level;
- 7) *Primary I/O's*—to interface with the external environment.

### III. HIGH-LEVEL POWER ESTIMATION METRICS

Our goal is to define power metrics to be applied at the system-level to measure and to compare the power consumption of several design alternatives. In general, it is quite difficult to define a single metric suitable for accurate and efficient power assessment for all the embedded systems applications. Thus, first we classify the embedded systems depending on their constraints and computational modes, then we propose a set of metrics for each class of systems.

We can divide the embedded systems in *timing-constrained* systems, if the speed is the most important design constraint, and *area-constrained* systems, if the area is the most important constraint. Several computational modes characterize the *timing-constrained* systems, depending on the *system throughput T* defined as the number of operations performed in a given time [7]. For microprocessor-based embedded systems, we can define three main modes of computation: *fixed* throughput mode, *maximum* throughput mode, and *burst* throughput mode, the latter characterized by a fraction of time performing useful computations, during which the maximum throughput is required, while during the rest of the time the system is in idle state, such as among user requests. Since the power budget strictly depends of the computational mode for which the embedded system is dedicated, a specific power metric can be defined for each one of the above defined operating modes [7].

For fixed throughput systems, a suitable metric is represented by the power/throughput ratio or equivalently the energy per operation. Since the throughput is fixed, if a partitioning solution leads to a reduction of such metric with respect to an initial partitioning, the corresponding power dissipation is reduced.

For maximum throughput systems, the most appropriate metric should account for both the low power and high performance needs. A suitable metric is thus the energy to throughput ratio (ETR) defined as in [7]  $ETR = E_{MAX}/T_{MAX}$  where  $E_{MAX}$  is the energy per operation or equivalently the power per throughput and  $T_{MAX}$  is the maximum throughput. Hence, the ETR metric can also be expressed as  $ETR = Power/T^2$ ,

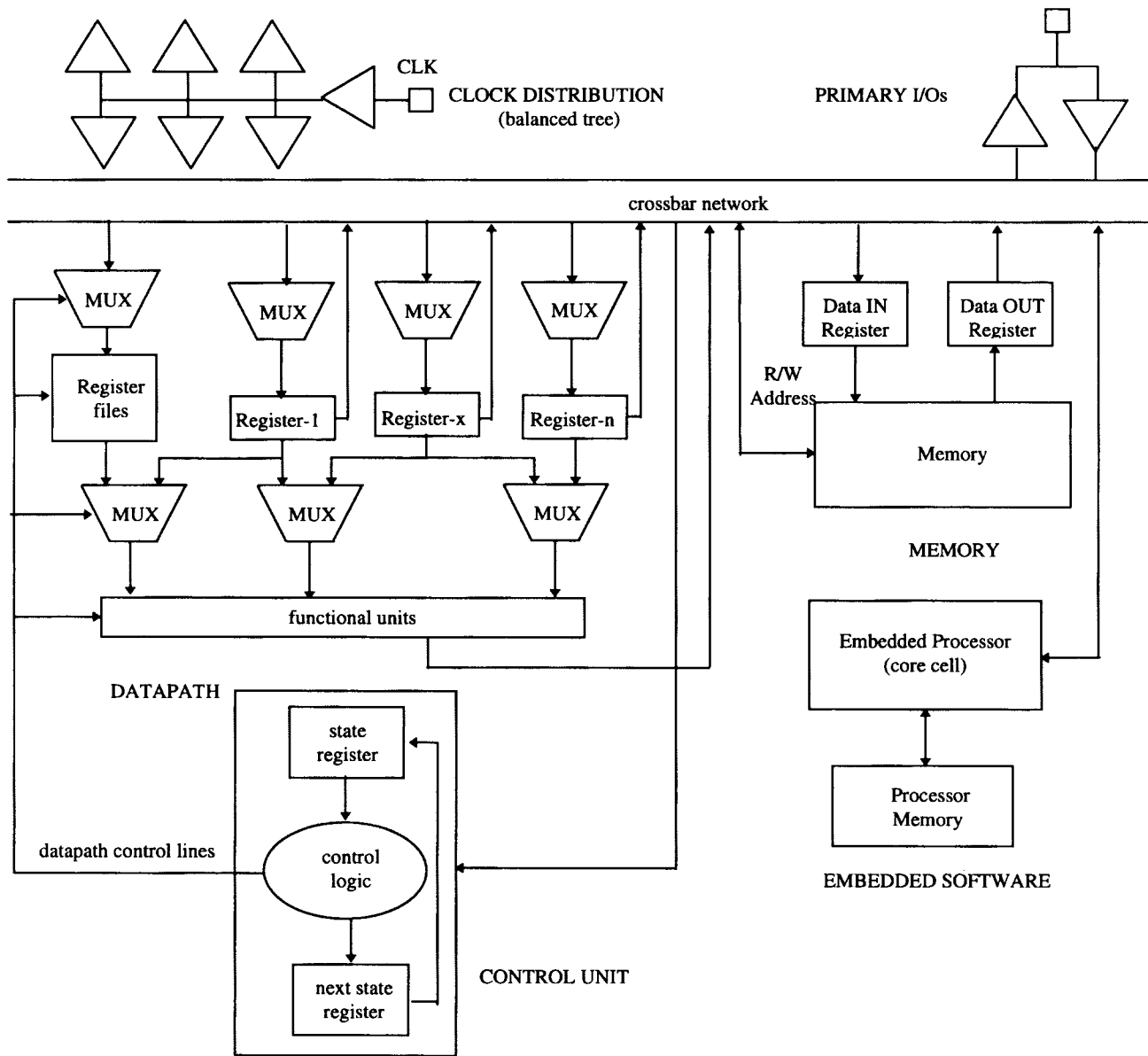


Fig. 2. The target system architecture at the RT-level.

The ETR metric expresses the concept of optimization of both the throughput and the power dissipation. A partitioning corresponding to a lower value of ETR represents a solution with lower energy per operation for equal throughput as well as a solution with greater throughput for the same amount of energy per operation.

For systems operating in the burst throughput mode, the power metric should provide power reduction, during both the idle and computing time, and throughput optimization when computing. For those systems applying power shut down techniques during idle cycles, an efficient metric is just ETR, since the power dissipation has been completely eliminated when idling. For those systems not supporting power saving modes, a most effective metric is [7]

$$M_{BURST} = (E_{MAX} + E_{IDLE})/T_{MAX} \quad (1)$$

where  $E_{MAX}/E_{IDLE}$  is the total energy dissipated when computing/idling per total operations and  $T_{MAX}$  is the maximum throughput.

For those area-constrained systems for which the target area is fixed, a valid metric  $M_A$  is represented by the power by area product (or equivalently the power/area ratio). Since the area is fixed, a reduction in the value of  $M_A$  corresponds to a minimization in the power consumption.

In general, for those area-constrained systems aiming at both power and area reduction, a good metric is given by the product of the energy per operation by area  $EAP = E_{MAX} * A$  where  $E_{MAX}$  is the energy per operation or equivalently the power per throughput and  $A$  is the area. Hence, the EAP metric can also be expressed as  $EAP = (Power * A)/T$ . The EAP metric expresses the concept of optimization of both the area and the power dissipation. A partitioning with a lower value

of EAP represents a solution with lower energy per operation for the same area as well as a solution with lower area for the same energy per operation.

The models used to estimate the *power* terms (for both the SW and HW parts) contained in the above equations are detailed in the next sections. The power assessment of the SW side is based on the system-level specification described at the VIS level, while the analysis of the HW side is related to the VHDL description of the ASIC model at the behavioral/RT level. The methodology proposed in [6] can be used to evaluate the area and throughput terms in the above metrics.

#### IV. POWER ESTIMATION FOR THE HW PART

The power model for the HW-bound part is based on the VHDL description of the ASIC at the behavioral/RT levels and the probabilistic estimation of the internal switching activity. The proposed approach is based on the following general assumptions:

- 1) the supply and ground voltage levels in the ASIC are fixed, although it is worth noting the impact of supply voltage reduction on power;
- 2) the design style is based on synchronous sequential circuits;
- 3) the data transfer occurs at the register-to-register level;
- 4) a zero delay model (ZDM) has been adopted, thus ignoring the contribution of glitches and hazards to power.

The inputs for the estimation are as follows:

- 1) *the ASIC specification*—consisting of a hierarchical VHDL description of the target system architecture;
- 2) *the allocation library*—composed of the available components implementing the macro-modules (such as adders, multipliers, etc.) and the basic modules (such as registers, multiplexers, logic gates, I/O pads, etc.).
- 3) *the technological parameters*—such as frequency, power supply, derating factors, etc.;
- 4) *the switching activity*—of the ASIC primary I/O's.

The power model is an analytical model, where the average power of the VHDL descriptions is related to the physical capacitance and the switching activity of the nets. The estimation approach is hierarchical: at the highest hierarchical level, *ad hoc* analytical power models for each part of the target system architecture are proposed; these models are in turn based on a macro-module library, at the lowest hierarchical levels. Furthermore, to avoid a large amount of input patterns to be simulated, our approach is weakly pattern-dependent. User-supplied input probabilities are required, reflecting the typical input behavior and derived from the system-level specification.

In the proposed single ASIC architecture, the total average power dissipated  $P_{AVE}$  is given by

$$P_{AVE} = P_{IO} + P_{CORE} \quad (2)$$

where  $P_{IO}$  and  $P_{CORE}$  are the average power dissipated by the I/O nets and the core internal nets, respectively. The power model of the core logic is based on the models of the different components of the target system architecture, therefore the

$P_{CORE}$  term can be in turn expressed as

$$P_{CORE} = P_{DP} + P_{MEM} + P_{CNTR} + P_{PROC} \quad (3)$$

where the single terms represent the average power dissipated by the data-path, the memory, the control logic and the embedded core processor. The power models related to the single terms in the above equations will be detailed in the following subsections, except for the  $P_{PROC}$  term, that is considered to be part of the power dissipated by the SW-bound part, detailed in Section V.

##### A. $P_{IO}$ Estimation

Although a presynthesis analysis is performed, we assume the knowledge of the ASIC interface in terms of primary I/O pads characteristics and related switching activity from the system-level specifications. The set  $S$  of input, output and bidirectional nets of the ASIC can be partitioned into  $N$  sets, such as  $S = \{s_1, s_2, \dots, s_k, \dots, s_N\}$  where the  $k$ th set  $s_k$  is composed of the same type  $t_k$  of I/O pads. Considering for example a set of output pads, the average power of the set  $s_k$  can be estimated as

$$P_{s_k} = \sum_{i=1}^{n_k} P_i(C_i)TR_i \quad (4)$$

where  $n_k$  is the number of output pads in the set  $s_k$ ;  $TR_i$  is the toggle rate of the  $i$ th output pad, derived from the system-level specifications and  $P_i(C_i)$  is the average power consumption per MHz of the  $i$ th output pad in  $s_k$  as a function of the output load  $C_i$  at a given reference frequency  $f_0$ .

##### B. $P_{DP}$ Estimation

The average power dissipated by the data-path can be expressed as

$$P_{DP} = P_{REG} + P_{MUX} + P_{FU} \quad (5)$$

where the single terms represent the average power dissipated by the registers, the multiplexers and the functional units.

Concerning the  $P_{REG}$  term, the live variable analysis has been applied to the behavioral-level VHDL code to estimate the number of required registers and the maximum switching activity of each register. The preliminary step is the estimation of the number of required registers and, consequently, the values of the toggle rate  $TR_i$  for each of them. According to the abstraction level, such data are directly available from the RT-level description or the live variable analysis can be applied to the behavioral-level specifications.

The algorithm [9] examines the life of a variable over a set of VHDL code statements to derive information concerning the registers switching activity and it can be summarized as follows.

- 1) Compute the lifetimes of all the variables in the given VHDL code, composed of  $S$  statements. A variable  $v_j$  is said to live over a set of sequential code statements  $\{i, i+1, i+2, \dots, i+n\}$  when the variable is written in statement  $i$  and it is last accessed in statement  $(i+n)$ . When a variable is written in a statement  $(i+k)$  in the

set, but last used in the same statement  $(i+k)$  of the next iteration, it is assumed to live over the entire set.

- 2) Represent the lifetime of each variable as a vertical line from statement  $i$  through statement  $(i+n)$  in the column  $j$  reserved for the corresponding variable  $v_j$ .
- 3) Determine the maximum number  $N$  of overlapping lifetimes, computing the maximum number of vertical lines intersecting with any horizontal cut-line.
- 4) Estimate the minimum number  $N$  of set of registers necessary to implement the code by using register sharing, that has to be applied whenever a group of variables, with the same bit-width  $b_i$ , can be mapped to the same register. The total number of registers is given by the sum of all  $b_i$ .
- 5) Select a possible mapping of variables into registers by using registers sharing.
- 6) Compute the number  $w_i$  of write to the variables mapped to the same set of registers.
- 7) Estimate  $\alpha_i$  of each set of registers dividing  $w_i$  by  $S$ :  $\alpha_i = w_i/S$ ; hence,  $TR_i = \alpha_i f_{CLK}$ .

The value of  $P_{REG}$ , considers that the power of latches and flip/flops is consumed not only during output transitions, but also during all clock edges by the internal clock buffers, even though the data stored in the register does not change. Thus, our analytical model of registers takes into account both the *switching* and *nonswitching* power. Let the set of registers  $S$  be composed of  $N$  sets, such as  $S = \{s_1, s_2, \dots, s_k, \dots, s_N\}$ , where the  $k$ th set  $s_k$  is composed of the same type  $t_k$  of registers, the average register power can be estimated as

$$P_{REG} = \sum_{k=1}^N (P_{s_k} + P_{N s_k}) \quad (6)$$

where  $P_{s_k}$  is the average power of each set  $s_k$  and  $P_{N s_k}$  is the corresponding average nonswitching power, that is the average power dissipated by the internal clock buffers when there are no output transitions. The estimated value of  $P_{s_k}$  accounts for  $TR_{s_k}$ , while the estimated values of the  $P_{N s_k}$  should consider a toggle rate of  $(f_{CLK} - TR_{s_k})$ . The estimated values of  $P_{s_k}$  and  $P_{N s_k}$ , for the  $k$ th set  $s_k$  (constituted by an estimated number of registers  $n_k$ ) are respectively given by

$$P_{s_k} = \sum_{i=1}^{n_k} P_i(C_i) TR_i \quad P_{N s_k} = P_{0k} \sum_{i=1}^{n_k} (f_{CLK} - TR_i) \quad (7)$$

where  $P_i(C_i)$  is the average power consumption per MHz of the  $i$ th register in  $s_k$ , and  $P_{0k}$  is the nonswitching power consumption per MHz of a single register of type  $t_k$ , that is load-independent.

Let us consider the estimation of the power related to multiplexers. First, to estimate the size and number of multiplexers from the VHDL code, it is necessary to determine the number of paths in the data-path. Then, the approach is based on the definition of the power model of a two-input noninverting multiplexer, based on both static signal probability of the selection net and the switching activities of the input nets. Given the pass-gate model of the two-input noninverting multiplexer, a simplified model for the maximum

switching activity of the output  $Z$  of a two-input noninverting multiplexer is

$$\alpha_Z = \alpha_A(1 - p_s^1) + \alpha_B p_s^1 \quad (8)$$

where  $\alpha_A$  and  $\alpha_B$  are the switching activity of inputs  $A$  and  $B$ , respectively, while  $p_s^1$  is the static signal probability of the selection net  $S$ . Globally, the average power dissipated by the multiplexers can be estimated as the sum of the average power of the single multiplexer contributions.

For the estimation of the average power of the functional units, we use complexity-based analytical models [3], where the complexity of each functional unit is described, in a library of macromodules, in terms of equivalent gates. Then, the estimated power dissipated by the functional units can be expressed as the sum of the contributions of the average power consumption  $P_i$  of the  $i$ th macromodule given by

$$P_i = n_i P_{TECH} TR_i \quad (9)$$

where  $P_{TECH}$  is a technological parameter expressed in  $(\mu W / (\text{gate MHz}))$ ;  $n_i$  is the estimated number of logic gates in the  $i$ th macrofunction;  $TR_i$  is the toggle rate of the output net of the  $i$ th macromodule.

### C. $P_{MEM}$ Estimation

Considering a fully CMOS single port static RAM, at a high-level of abstraction, we assume to have in the target library the information related to the power consumption of a single memory cell  $P_{cell}$  and of a single memory output buffer.

The average power dissipation during a read access to a single row of the array, composed of  $n$  rows and  $m$  columns, is proportional to the inverse of the read access time  $t_a$  and to the sum of the average power dissipated by the following blocks: the row decoder, the  $m$  memory cells composing the  $i$ th row and the output buffers. In particular, the power dissipated by the row decoder can be estimated with a complexity-based model, where the number of equivalent gates is proportional to the product  $(n \cdot Xlg_2 n)$  and the load capacitance is the word line capacitance.

### D. $P_{CNTR}$ Estimation

This section describes the contribution to the power consumption due to the control part of the target system architecture, described as a set of finite-state machines (FSM's) represented by state transition graphs (STG's). The proposed FSM power model is a probabilistic model, where we approximate the average switching activities of the FSM nodes by using the switching probabilities (or transition probabilities) derived by modeling the FSM as a Markov chain. Given a typical implementation of a FSM, composed of a combinational circuit and a set of state registers, we consider the different contributions to the global average power

$$P_{CNTR} = P_{IN} + P_{STATE\_REG} + P_{COMB} + P_{OUT} \quad (10)$$

where  $P_{IN}$  is the average power dissipated by the primary inputs  $P_{STATE\_REG}$  is the average power dissipated by the state registers,  $P_{COMB}$  is the average power dissipated by the

combinational logic and finally  $P_{\text{OUT}}$  is the average power dissipated by the primary outputs.

The input static signal probabilities and the input switching activity factors are obtained from the system-level specifications, being derived by either simulating the FSM at a high abstraction level or by direct knowledge of the typical input behavior. Furthermore, we assume a ZDM for the logic gates and synchronous primary inputs. Under these assumptions, we can ignore the effects of glitches and hazards on the state bit lines, therefore the switching activity of the present and next state bit lines are equal.

Let the FSM, composed of  $n_s$  states, described by using a STG composed of  $n_s$  vertices, corresponding to the states in the set  $S = \{s_1, s_2, \dots, s_{n_s}\}$ , and the related directed edges. The edges are labeled with the set of input configurations that cause a transition from the source state to the destination state. Considering a transition from state  $s_i$  to state  $s_j$ , we can compute the factor  $p_{ij}$ , called *conditional state transition probability*, that represents the conditional probability of the transition from state  $s_i$  to state  $s_j$ , given that the FSM was in state  $s_i$ :  $p_{ij} = \text{Prob}(\text{Next} = s_j | \text{Present} = s_i)$ . The computation of the  $p_{ij}$ 's can be carried out as in [10], assuming totally independent primary inputs  $P_I = \{x_1, x_2, \dots, x_k, \dots, x_{n_I}\}$  and being  $p_{xk}$  the static signal probability of input  $x_k$ . The *steady-state probability*  $P_i$  of a state  $s_i$  is defined as the probability to be in the state  $s_i$  in an arbitrarily long random sequence [11]. Computing the  $P_i$ 's implies solving the system composed of the Chapman–Kolmogorov equations and the equation representing the normality condition:

$$P^T = P^T p \sum_{i=1}^{n_s} P_i = 1 \quad (11)$$

where  $P^T = (P_1, \dots, P_k, \dots, P_{n_s})$  is the row vector of the steady-state probabilities and  $p$  is the matrix of the conditional state transition probabilities  $p_{ij}$ . Note that the above system has  $(n_s + 1)$  equations and  $n_s$  unknowns, thus one of the Chapman–Kolmogorov equations can be dropped [10]. Given the state probabilities  $P_i$ 's and the conditional state transition probabilities  $p_{ij}$ 's, the *total state transition probabilities*  $P_{ij}$  between the two states  $s_i$  and  $s_j$  can be expressed as  $P_{ij} = p_{ij}P_i$ .

Given a state encoding, the next steps are represented by the estimation of the switching activity of the state bit lines and the primary outputs. The switching activity of the state bit lines depends on both the state encoding and the total state transition probabilities between each pair of states in the STG. Let us generalize the concept of state transition probability to transitions occurring between two distinct subsets of disjoint states,  $S_i$  and  $S_j$ , contained in the set of states  $S = \{s_1, s_2, \dots, s_{n_s}\}$ , as defined in [11]

$$TP(S_i \leftrightarrow S_j) = \sum_{s_i \in S_i} \sum_{s_j \in S_j} (P_{ij} + P_{ji}) \quad (12)$$

Being  $b^i$  the  $i$ th bit ( $1 \leq i \leq n_{\text{var}}$ ) of the state code (called state bit) and  $n_{\text{var}}$  the number of state bits ( $\lceil \lg_2 n_s \rceil \leq n_{\text{var}} \leq n_s$ ), we consider the two sets of substates in which the  $i$ th

state bit assumes the value one and zero, respectively. The switching activity  $\alpha_b^i$  of the state bit line  $b_i$  is given by [11]

$$\alpha_b^i = TP(\text{States}(b^i = 1) \leftrightarrow \text{States}(b^i = 0)).$$

In a Moore-type FSM, the total state transition probabilities  $P_{ij}$  between the two states  $s_i$  and  $s_j$  are equal to the total transition probabilities between the corresponding outputs  $o_i$  and  $o_j$  where the output row vector  $o_i$  ( $i = 1, 2, \dots, n_s$ ) is composed of the  $n_O$  primary outputs ( $y_i^1, \dots, y_i^k, \dots, y_i^{n_O}$ ). Let us define the transition probability of the transitions occurring between two distinct subsets of disjoint outputs  $O_i$  and  $O_j$  contained in the set of the outputs  $O = \{o_1, o_2, \dots, o_{n_s}\}$ , as

$$TP(O_i \leftrightarrow O_j) = \sum_{o_i \in O_i} \sum_{o_j \in O_j} (P_{ij} + P_{ji}) \quad (13)$$

Being  $y^m$  the  $m$ th output bit ( $1 \leq m \leq n_O$ ) and  $n_O$  the number of primary outputs, we consider the two sets of outputs in which the  $m$ th output bit assumes the value one and zero, respectively. The switching activity  $\alpha_{ym}$  of primary outputs  $y_m$  is given by  $\alpha_{ym}^m = TP(\text{Outputs}(y^m = 1) \leftrightarrow \text{Outputs}(y^m = 0))$ .

At this point of the analysis, we can detail the different power terms contained in the expression of  $P_{\text{CNTR}}$ .

The average power dissipated by the  $k$ th primary input belonging to the set  $P_I = \{x_1, x_2, \dots, x_k, \dots, x_{n_I}\}$  depends on the switching activity factors  $\alpha_{xk}$  and the input load capacitance  $C_{xk}$ , the latter being proportional to the number of literals,  $n_{litxk}$ , that the  $k$ th primary input is driving in the combinational part, and the estimated capacitance  $C_{lit}$  due to each literal [11]. Therefore, the average power PIN can be estimated as

$$P_{\text{IN}} = \sum_{x_k \in P_I} P_{xk}(C_{xk})TR_{xk} \quad (14)$$

where  $C_{xk} = n_{litxk}C_{lit}$ ;  $TR_{xk} = \alpha_{xk}f_{\text{CLK}}$  and  $P_{xk}(C_{xk})$  is the average power consumption per MHz of the cell driving the  $k$ th input.

The average power dissipated by the state registers  $P_{\text{STATE\_REG}}$  can be derived by using the switching activity  $\alpha_{b_i}$  of the  $i$ th state bit line  $b_i$  where  $1 \leq i \leq n_{\text{var}}$  and the corresponding toggle rate is  $TR_{b_i} = \alpha_{b_i}f_{\text{CLK}}$ . The term  $P_{\text{STATE\_REG}}$  accounts for the switching and nonswitching power of the state registers

$$P_{\text{STATE\_REG}} = \sum_{i=1}^{n_{\text{var}}} (P_i + P_{NSi}) \quad (15)$$

where  $n_{\text{var}}$  is the number of state registers and  $P_i$  and  $P_{NSi}$  are the average switching and nonswitching power dissipated by each state register. The terms  $P_i$  should account for a toggle rate given by  $TR_{b_i}$ , while the terms  $P_{NSi}$  should consider a toggle rate of  $(f_{\text{CLK}} - TR_{b_i})$ .

The average power dissipated by the combinational logic  $P_{\text{COMB}}$  has been estimated by considering a two-level logic implementation, before the minimization step. The  $i$ th state bit line  $b_i$  (where  $1 \leq i \leq n_{\text{var}}$ ) can be expressed by using the canonical form as the sum of  $N_{b_i}$  minterms ( $N_{b_i} \leq 2^{n_{lit}}$  where  $n_{lit}$  is the number of literals and  $2^{n_{lit}}$  is the maximum number

of minterms). Similarly, the  $m$ th output bit  $y^m$  ( $1 \leq m \leq n_O$ ) can be expressed in the canonical form as the sum of  $N_{ym}$  minterms ( $N_{ym} \leq 2^{n_{lit}}$ ).

Let us assume to use a single AND gate to represent the generic minterm, hence the maximum number of AND gates in the AND-plane is  $2^{n_{lit}}$ , while in general  $n_{AND} \leq 2^{n_{lit}}$ . Given the probabilistic model of the switching activity of the generic  $n_{lit}$ -input AND gate, we can derive an upper bound for the estimated power of the AND-plane

$$P_{COMB} = \sum_{i=1}^{n_{AND}} P_i(C_i)TR_i \quad (16)$$

where  $P_i(C_i)$  is the average power consumption per MHz of the  $i$ th  $n_{lit}$ -input AND gate;  $C_i$  is the capacitance driven by the  $i$ th  $n_{lit}$ -input AND gate and  $TR_i = \alpha_i f_{CLK}$  is the toggle rate of the  $i$ th  $n_{lit}$ -input AND gate (derived by using the switching activity model of the  $n_{lit}$ -input AND gate).

$P_{OUT}$  is the average power dissipated by the OR-plane, that is composed of  $n_{var}$   $N_{bi}$ -input OR gates corresponding to the state bit lines, driving the input capacitance of the state registers, and  $n_O$   $N_{ym}$ -input OR gates corresponding to the primary outputs, driving the output load capacitances.

Therefore, the upper bound for the power of the OR-plane is composed of two terms. The first term is thus proportional to the switching activity factors  $\alpha_{bi}$  of the state bit line  $b_i$ , while the second term is proportional to the switching activity factors  $\alpha_{yi}$  of the primary outputs:

$$P_{OUT} = \sum_{i=1}^{n_{var}} P_i(C_{IN\_REG})TR_{bi} + \sum_{i=1}^{n_O} P_i(C_{yi})TR_{yi} \quad (17)$$

where  $P_i(C_{IN\_REG})$  is the average power consumption per MHz of the  $i$ th  $N_{bi}$ -input OR gate driving the  $i$ th state bit line,  $C_{IN\_REG}$  is the input capacitance of each state register;  $TR_{bi} = \alpha_{bi} f_{CLK}$  is the toggle rate of the  $i$ th state bit line  $b_i$ ,  $P_i(C_{yi})$  is the average power consumption per MHz of the  $i$ th  $N_{yi}$ -input OR gate driving the  $i$ th primary output,  $C_{yi}$  is the output load capacitance of the  $i$ th primary output and finally  $TR_{yi} = \alpha_{yi} f_{CLK}$  is the toggle rate of the  $i$ th primary output.

## V. POWER ESTIMATION FOR THE SW PART

The software power assessment in TOSCA is performed by following a bottom-up approach. Each software-bound part of the OCCAM2 specification is considered in terms of basic blocks and it is compiled in the VIS. Hence, the power analysis has been performed at the VIS-level, by considering the average power consumption of each VIS instruction during the execution of a given program. The choice to work at the VIS-level is motivated by the goal to make our analysis processor-independent.

In general, the average power dissipated by a processor while running a program is  $P_{SW} = I_{AVE} * V_{DD}$ , where  $I_{AVE}$  is the average current and  $V_{DD}$  is the supply voltage. The associated energy is given by  $E_{SW} = P_{SW} * t_{SW}$ , where  $t_{SW}$  is the execution time of the software program, that can be expressed as:  $t_{SW} = N_{CLK} * \tau_{CLK}$ , being  $N_{CLK}$  the number of clock cycles to execute the program and  $\tau_{CLK}$  the

clock period. To compute the average current drawn during the execution of each instruction, it is necessary to perform some measurements on the energy cost of each instruction, such those proposed in [12], [13], or to have detailed power information provided by the processor supplier, in terms of the energy dissipated by each type of instruction in the instruction-set. This latter power information can be derived by the processor supplier by simulating the execution of instruction sequences on a lower level (circuit or layout) or gate level model of the processor, to obtain an estimate of the current drawn. Based on this information, a power table can be derived for each processor, reporting the energy consumption for each instruction in the instruction-set and for all the possible addressing modes associated with each instruction type.

Additional contributions to the global energy derive from to interinstruction effects, not considered computing the base cost of each instruction. The possible interinstruction effects are mainly related to the previous state of the processor, the limited number of resources leading to pipeline and write buffer stalls and the rate of cache misses [12], [13]. The condition of the processor in the previous clock cycle may cause an energy overhead due to the different switching activities on data and address busses and the different processors internal behavior. In general, the previous state of the circuit is different during program execution, since there is a switching from an instruction to another, with respect to the execution of the program used for the measurements of the base energy, where the same instruction was executed many times. The circuit state overhead has been measured in [12] by considering all the possible instruction pairs and it results approximately less than 5% of the base energy per instruction. This overhead has been considered in [12], as an average constant value to be added to the base cost, without a significant loss of precision. The effects of resource constraints and cache misses on the power budget have been measured in [12]. However, these effects can be usually neglected in embedded software based on both simple microcontrollers (e.g., M68000, Intel 8051, Z80, ...) where such advanced features can be absent, and advanced processors, achieving cache hit-rate over the 98% and providing a fully exploitation of the pipeline stages.

Once the power analysis is completed for all the basic VIS-level instructions, the analysis is extended to upper-level software modules, by weighting the power consumption of each basic block according to the execution frequencies.

## VI. SIMULATION RESULTS

Since we are focusing on control dominated embedded systems, we report some results derived from the application of the proposed power model to a set of 35 FSM's derived from the MCNC-91 benchmark suite. The measures have been derived by using the HCMOS6 technology, featuring 0.35  $\mu$ m and 3.3 V, supplied by SGS-Thomson Microelectronics at the target operating frequency of 100 MHz. First we applied the area-oriented state assignment program NOVA to the selected benchmarks, then the encoded FSM's have been synthesized by the Synopsys Design Compiler tool targeting the HCMOS6 technology. The estimation results obtained by the proposed

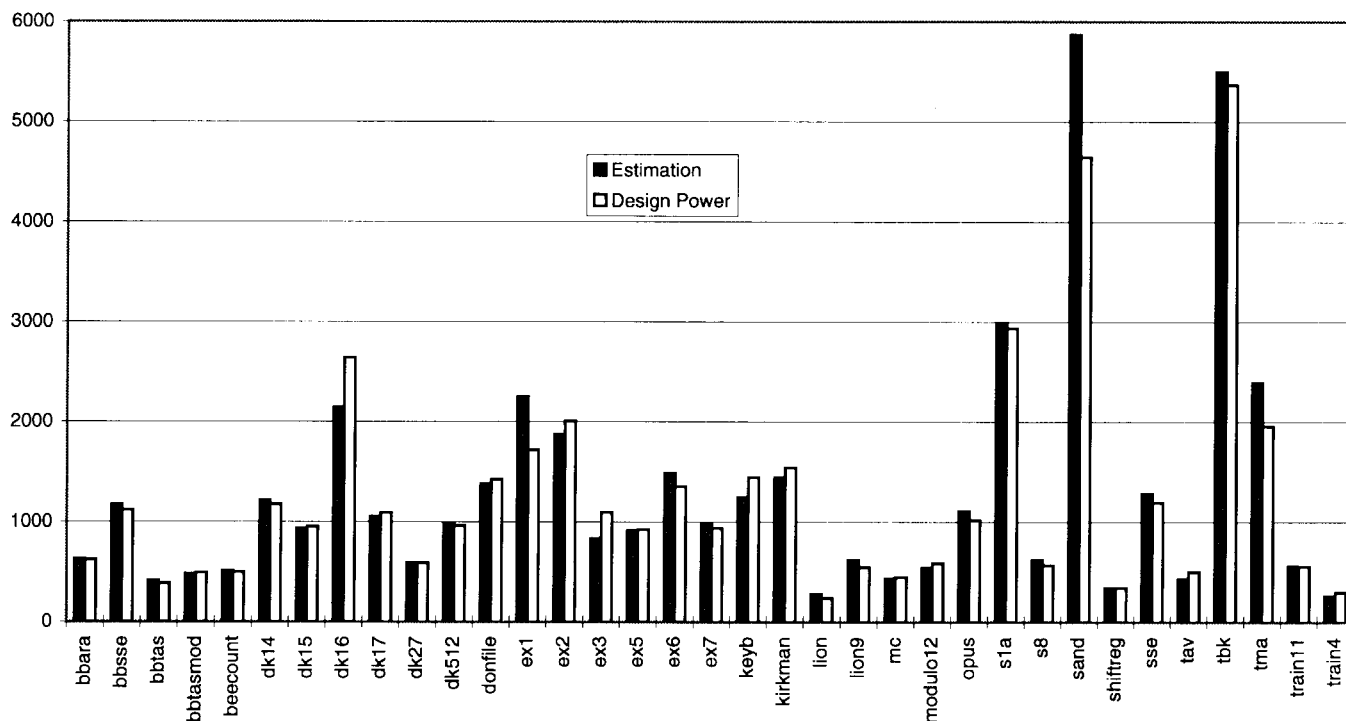


Fig. 3. Total power: estimated versus Design Power results.

methodology at presynthesis level have been compared with the results derived by using the Synopsys Design Power tool, based on the synthesized gate-level netlist. Note that both methods are based on a ZDM.

Fig. 3 summarizes the results. Considering the sequential power, the proposed model shows an average percentage error of 9.52% (ranging from 0.01 to 25.8%) with respect to the design power estimates. Concerning the combinational and total power, the average percentage errors is equal to 9.21 and 8.17%, respectively. Globally, the relative accuracy of our results compared with the design power results is considered satisfactory at this level of abstraction.

## VII. CONCLUSIONS

The proposed analysis affords the problem of power estimation for control oriented embedded systems, implemented into a single ASIC. The main goal has been to offer a power-oriented codesign methodology, with particular emphasis on power metrics, to compare different design solutions described at high abstraction levels. Power models for both the HW and SW parts have been presented. The paper covers in more detail the HW part, since it is usually the more complicated part to be estimated with an acceptable precision, due to its heterogeneous nature. As it has been shown, the proposed approach is quite general, since it considers both the implementation domains as well as all the subparts, which typically constitute the HW side of an embedded system. The value-added has been to introduce a third dimension, power, to the speed versus area space, where the architectural design exploration is usually carried out. Finally, experimental results on benchmark circuits have shown a sufficient relative accuracy with respect to gate-level power estimates.

The approach is limited by the fact that at present the proposed power model is tailored to the target system architecture shown in Fig. 2 and that only the average power consumption is considered. However, the inclusion of the peak power could be performed by considering maximum switching activity values at input/output nodes. Moreover, work is in progress aiming at defining a power model suitable for the HW/SW communication part.

## REFERENCES

- [1] G. De Micheli and M. G. Sami, Eds., *Hardware/Software Co-Design*. New York: Kluwer Academic, NATO ASI Series, 1996.
- [2] D. Singh, J. Rabaey, M. Pedram, F. Catthoor, S. Rajgopal, N. Sehgal, and T. Mozden, "Power conscious CAD tools and methodologies: A perspective," *Proc. IEEE*, vol. 83, pp. 570–594, Apr. 1995.
- [3] P. Landman, "High-level power estimation," in *Proc. ISLPED-96: Int. Symp. Low Power Electron. Design*, Monterey, CA, 1996, pp. 29–35.
- [4] F. N. Najm, "A survey of power estimation techniques in VLSI circuits," *IEEE Trans. VLSI Syst.*, vol. 2, pp. 446–455, Dec. 1994.
- [5] P. E. Landman and J. M. Rabaey, "Activity-sensitive architectural power analysis," *IEEE Trans. Computer-Aided Design*, vol. 15, pp. 571–587, June 1996.
- [6] A. Balboni, W. Fornaciari, and D. Sciuto, "Partitioning of HW-SW embedded systems: A metrics-based approach," in *Integrated Computer-Aided Engineering*. IOS Press, 1998, vol. 5, no. 1, pp. 39–55.
- [7] T. D. Burd and R. W. Brodersen, "Energy efficient CMOS microprocessor design," in *Proc. 28th Hawaii Int. Conf. System Sci.*, HI, 1995.
- [8] A. Balboni, W. Fornaciari, and D. Sciuto, "Co-synthesis and co-simulation of control dominated embedded systems," in *International Journal Design Automation for Embedded Systems*, vol. 1, no. 3, July 1996.
- [9] W. Fornaciari, P. Gubian, D. Sciuto, and C. Silvano, "A conceptual analysis framework for low power design of embedded systems," in *Proc. ISIS-96: IEEE 8th Int. Conf. Innovative Syst. Silicon*, Austin, TX, 1996, pp. 170–179.
- [10] E. Macii, "Sequential synthesis and optimization for low power," in *Low Power Design in Deep Submicron Electronics*. New York: Kluwer Academic, NATO ASI Series, 1997.

- [11] C. Y. Tsui, M. Pedram, C. A. Chen, and A. M. Despain, "Low power state assignment targeting two- and multi-level logic implementations," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, 1994, pp. 82–87.
- [12] V. Tiwari, S. Malik, and A. Wolfe, "Power analysis of embedded software: A first step toward software power minimization," *IEEE Trans. VLSI Syst.*, vol. 2, pp. 437–445, Dec. 1994.
- [13] V. Tiwari, S. Malik, A. Wolfe, M. T.-C. Lee, "Instruction level power analysis and optimization of software," in *Journal of VLSI Signal Processing*. New York: Kluwer Academic, 1996, pp. 1–18.



**William Fornaciari** (S'92–M'95) received the Laurea (cum laude) in electronic engineering and the Ph.D. degree in automation engineering and computer sciences from the Politecnico di Milano, Milano, Italy.

In 1993, he joined the CEFRIEL Research Center in Milano, where he currently supervises the Electronic Design Automation (EDA) area. Since 1995, he has been an Assistant Professor at the Politecnico di Milano, Department of Electronic Engineering and Information Sciences. His research

interests covered algorithms for electrical circuit simulation, VLSI design with particular emphasis on the problems related to the digital implementation of artificial neural networks. Currently, his main research is in the field of the design automation for embedded systems, hardware/software codesign and low-power system-level analysis/design.

Dr. Fornaciari is member of the IEEE Computer Society. He has organized a special session on hardware/software codesign for the ICRAM'95 and CESA'96 conferences. During the IEEE-ICONIP'95 and IEEE-IJCNN'92 conferences, he received the Best Paper Award. In 1996, he received the Certification of Appreciation from the IEEE Circuits and Systems Society.



**Paolo Gubian** (M'88) received the Dr.Eng. degree (summa cum laude) from Politecnico di Milano, Italy, in 1980.

After an initial period as a Research Associate at the Department of Electronics of the Politecnico di Milano, he started consulting for SGS-Thomson Microelectronics (then SGS-Microelectronics) in the areas of electronic circuit simulation and CAD system architectures. During this period, he worked at the design and implementation of ST-SPICE, the company proprietary circuit simulator. He also

worked in European initiatives to define a standard framework for integrated circuit CAD systems. From 1984 to 1986, he was a Visiting Professor at the University of Bari, Italy, teaching a course on circuit simulation. In 1987, he joined the Department of Electronics at the University of Brescia, Italy as an Assistant Professor in the Department of Electrical Engineering, where he is now an Associate Professor. His research interests are in statistical design and optimization, modeling of frameworks for IC CAD environments and low-power design of IC's.

**Donatella Sciuto** (S'84–M'87) received the Laurea in electronic engineering in 1984 and the Ph.D. degree in electrical and computer engineering from University of Colorado, Boulder, in 1988.

She has been an Assistant Professor at the Dipartimento di Elettronica per l'Automazione, University of Brescia, Italy, until 1992. She is currently an Associate Professor at the Dipartimento di Elettronica e Informazione of the Politecnico di Milano, Italy, and she is Secretary of the Special Interest Group in VHDL. Her research interests include VLSI synthesis and testing, VHDL system specification and design, and hardware/software codesign.



**Cristina Silvano** received the Dr.Eng. degree in electronic engineering from the Politecnico di Milano, Italy, in 1987. She is currently working towards the Ph.D. degree at the Università di Brescia, Italy, where her dissertation is on advanced design and estimation techniques for low-power circuits.

From 1987 to 1995, she held the position of Senior Design Engineer in the ASIC Development and Validation Group, PowerPC Platform Department, Bull Research and Development Laboratories, Pregnana M., Italy. In 1996, she joined the Department

of Electronics of the Università di Brescia, Italy. Her current research interests are in the area of computer-aided design of integrated circuits and systems, with particular emphasis on low power and codesign techniques for embedded systems.

Ms. Silvano is a member of the IEEE Computer Society.