ENERGY ESTIMATION IN HIGH LEVEL CYCLE-ACCURATE DESCRIPTIONS OF EMBEDDED SYSTEMS

Ana Belén Abril García, Jean Gobert, Thomas Dombek,
Habib Mehrez* and Frédéric Pétrot*
Philips Research France &
*ASIM/LIP6 Lab, Université Paris VI, France
Ana.Abril@philips.com

Abstract. *Simulating performance and energy consumption of embedded systems using a high-level description is a challenging task in VLSI CAD. However such simulations are needed to select the best hardware architecture and software organization for a particular application. In this paper, we present an approach for cycle-accurate hardware/software co-simulations of energy consumption in embedded systems. The aim of our work is to provide a simulation framework enabling power estimations for high level descriptions (behavioral C models) of embedded systems that include new hardware components. A System-level cycle-accurate simulator and a processor Instruction Set Simulator are extended with energy models that take into account the state of the component.*

1 Introduction

The portable segment of digital consumer market is growing very fast due to the exploding demand for devices like mobile telephones, laptops or PDAs. These products become far more complex with the introduction of new multimedia functionalities, e.g. audio or video. They make use of highly integrated components, like embedded systems (System-On-a-Chip) that have to be optimized in performance, size, power consumption and cost. These embedded systems are a mix of specialized hardware (for performance and power) and general purpose processors or DSP (for flexibility). Autonomy is an important feature but improvements on batteries are smaller than the increase of processing complexity. For this reason energy consumption is a critical factor in embedded system design.

Multimedia applications typically require very large data storage and transfer capacities, which are the main sources of power consumption. To reduce this consumption, exploration techniques and optimizations for memory management must be performed at system level [1]. One has to note that most of the power can be saved at highest levels: architectural, system or algorithmic. Various architectures can be explored giving different performances, sizes and power consumption, so performing trades-off early in the design process is very interesting and their application is still rather easy. This early exploration can also help to anticipate necessary structural changes in the system in case that it is not able to comply with the specifications.

There exists a lot of CAD tools able to estimate power consumption in circuits at lower levels of design description, e.g. SPICE, but fewer at higher levels. Typical Systems-On-a-Chip composed of processors, memories and custom hardware are firstly modeled at high level in system functional simulators. The aim of our work is to provide a simulation framework able to get power estimation for high level descriptions of embedded systems including

new hardware blocks. The simulator we use is TSS (*Tool for System Simulation*) [2], a cycle-accurate simulator for system level descriptions in C language, developed by Philips, working with the Armulator [3], an Instruction Set Simulator developed by ARM Ltd.. Armulator is linked to TSS simulator. In order to obtain the power estimation, we create energy models for every hardware block and we introduce these in the TSS simulator functional models. Then we simulate the system with a set of representative input data and we obtain the power estimation. We propose an approach to create the energy models for new hardware blocks for which we do not have any gate structural information. It consists in estimating the energy consumption per cycle depending of the internal state of the component.

A brief discussion of the existing power estimation approaches of embedded systems is given in the section 2. Section 3 explains the approach we propose and introduces the simulation tools we use. In section 4 we describe the system model detailing how we model the energy for each category of component. The last section concludes, makes some suggestions for the validation of the approach and presents the future work.

2 Related Research

A common approach to obtain fairly accurate power estimations of large circuits with a conventional RTL simulator, consists in first modeling the basic cells power consumption with parameters extracted from circuit level simulations (SPICE, PowerMill) or from the characteristics of the standard cell library [5, 6, 7, 8]. This information is then embedded in the VHDL description of every basic cell so that the logic simulation allows to obtain the power consumption of the circuit. DIESEL, the power estimation tool developed in Philips, uses this approach [5]. This tool keeps track of the instantaneous signal transitions and combines them with a library characterization, to determine the power figures from a gate level design. This method has some drawbacks: it takes place in an advanced phase in the design process and the simulation time for large circuits becomes very important making rather impracticable power estimations using hardware/software co-simulation.

A higher level embedded hardware/software system co-simulation is proposed by Li [9]. It consists of a framework that simultaneously evaluates the tradeoffs of energy consumption of software and hardware. The goal is to optimize the software in a way to reduce energy consumption in memory accesses under performance constraints. A processor simulator running together with a memory tracer model gives the number of memory accesses for a particular application program. At the end of the simulation this number is combined with processor and memory power models giving the total power consumption of the system. The drawback of this method is that the power models do not depend on the internal states of the components and therefore lack accuracy.

A more accurate approach is developed by Simunic [4]. It consists of a methodology for power estimations at higher level using an instruction-level cycle-accurate simulator. It is extended with energy models for processor, bus and memories. The goal is to find an optimal memory size and hierarchy, for minimal energy consumption. Power models compute the energy consumed at every cycle for every hardware element of the system. They have been completely inferred from data-sheet information. This is possible in systems based on components that have been already built, e.g. memories and processors, like in Simunic's example. Unfortunately the methodology does not cover the case of systems containing new hardware blocks. In this case, a simulator of the global system has to be used and it should be extended with power models for all blocks.

Power consumption estimations of new designs at higher levels, i.e. behavioral C models, are very difficult. Modeling physical parameters into C functions needs knowledge on how these functions will be implemented in hardware. Nevertheless at this level of the design the gate structure of the design is not yet defined. A solution could be to analyze all the operations (additions, multiplications, etc) in the new block, and to estimate an energy consumption for each type. During the functional simulation, the corresponding energy could be accumulated every time we use an operation. The problem is too that we still do not know how operations will be implemented in hardware so we can not associate it to an energy estimation. The solution we propose is to analyze the basic functional states of the block, estimating an energy consumption for each one and accumulate this state energy every cycle during the simulation. Next paragraphs will present this approach and the simulation tools we use to implement it.

3 Proposed Approach

Embedded system design starts with the partitioning of the application. Some parts will be implemented in hardware and others in software. Building highly complex single chip systems requires models of the used hardware building blocks which are more abstract than RTL or gate-level. These models are used to develop the system architecture and to support the software development of low level drivers, because billions of simulation cycles are required.

TSS is a cycle-accurate C-based simulation framework developed to simulate complex hardware/software architectures [2]. TSS also enables simulations with other tools, allowing to integrate Instruction Set Simulators (ISS), VHDL/Verilog and TSS models.

We take an example system including a processor, memories and custom hardware. In the current set-up we use:
- An Instruction Set Simulator (Armulator), giving information about the processor behavior.
- A simulation of TSS hardware blocks descriptions and the interconnect.

The TSS simulator is normally used for cycle-accurate functional simulation. In this paper we present how the TSS simulation models can be enriched with energy models allowing to get power estimations by simulation. The simulation provides the energy consumption of each element and the total. This is not a completely accurate estimation but it is sufficient to find the best HW architecture scenario in terms of power consumption. The energy per cycle of each element is calculated from an energy model, according to the current state of the element. The memory and processor energy models are built using an approach similar to Simunic's work [4], based on data-sheet information. For the new target hardware we define energy models that take into account the estimated number of gates in RTL implementation, the gate's activity and the implementation technology. For the interconnect, the model needs an estimation of the wire length and the capacitance per unit length.

4 System and Components Models

The power, i.e. energy consumption, for executing a task, can be observed at several levels. We will analyze these from higher to lower level. At the first level we can consider the total energy consumption in one second or for the whole task. If n is the number of total processing cycles per second (frequency), or by task, the total energy consumption is obtained by accumulating energies of all processing cycles and is written as in (1).

$$E_{Total} = \sum_{i=1}^{n} E_{cycle_i} \tag{1}$$

At a second level we can consider the total energy consumption per cycle. This is the energy consumed during a specific processing cycle, obtained by adding the energies consumed during this cycle by the m hardware components of the system, represented in (2).

$$E_{cycle} = \sum_{i=1}^{m} E_{component_i} \tag{2}$$

Since a hardware component has different states or ways of working, the energy consumption per cycle is different depending on the state of the component: active, idle, refresh, sleep, etc. At this third level, we distinguish between the energy for each internal state in every hardware component of the system. The TSS functional model of each element must be accurate enough to associate a state to each cycle. The model is described like a state machine, separating states with significantly different power consumption. During the simulation, the current state is known every cycle and therefore the corresponding energy can be added to the total energy according to the Equation (2). The set of energies per cycle for a component with j different states is called the energy model of the component and is given in (3).

$$E_{comp} = (E_{comp,state1}, E_{comp,state2}, E_{comp,state3}, ..., E_{comp,statej}) \tag{3}$$

The energy associated to each state of a component, $E_{comp,state}$, corresponds to the lowest level of our model.

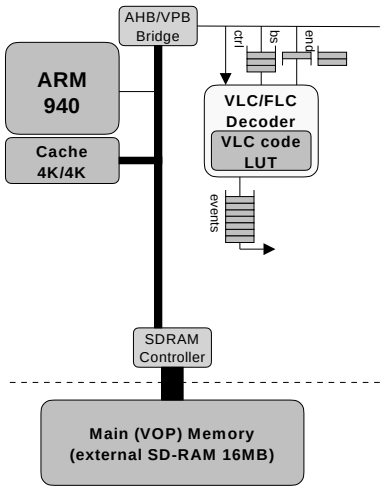

Figure 1: Architecture of the system.

The approach is tested on a typical example of embedded system for portable application: a codec video MPEG-4 [10], implementing the simple profile. A preliminary HW/SW partitioning has been already made. The system consist of a processor with I+D caches, memories, hardware accelerators and interconnect.

In a first step we will consider only a part of the system including the processor, memory and the Variable Length Decoder (VLD) [11] hardware accelerator (see Fig. 1). The architecture is used to validate the methodology proposed for power estimation, before including the other hardware blocks. In this case the E_{cycle} can be written as in (4).

$$E_{cycle} = E_{VLD} + E_{mem} + E_{proc} + E_{interc} \tag{4}$$

Next subsections explain in detail those energy models and the energy value per state of each element.

4.1 VLD Energy Model

The energy consumption in a digital CMOS circuit can be expressed as in (5).

$$E = \mu * N_{gates} * E_{gate} \tag{5}$$

where μ is the circuit activity (the percentage of the number of switching gates over the total number of gates), N_{gates} the total number of gates in the circuit and E_{gate} the energy consumption per switching gate for a particular technology. These are the parameters introduced

into the VLD TSS energy model. Obviously, the number of gates and their activity are not known *a priori* if an RTL description of the circuit is not available. But estimations of these values can be sufficient to build an energy model for architecture exploration.

In the case of the VLD, we know that the architecture will consist of three parts: interface, barrel-shifter and Look-Up-Table (LUT). The interface and the barrel-shifter will always be used in active cycles. The LUT will be composed of six tables: three for DC coefficients and the other three for AC coefficients. In every active cycle only one table will be used therefore only part of gates are active. The complete VLD will be around 4000 gates. The gates estimation per part is 1000 gates for the interface, 1300 gates for the barrel-shifter and 1700 gates for the LUT tables.

During active cycles, the VLD analyzes a bitstream and extracts coefficients. The number of switching gates is roughly: $1000 + 1300 + (1700 \div 6) \simeq 2600$ gates, which corresponds to 65 % of the gates. At worst, all these gates will switch every cycle. Since only half of the transitions (0-0, 0-1, 1-0, 1-1) corresponds to a switch activity, we suppose that the estimate switching activity is $65\% \div 2 = 33\%$. In non active cycles (idle or fifo write/read waiting cycles), we estimate roughly an activity of 1 %, corresponding to the wake-up logic. The VLD state machine has six states: *dc*, *ac* (actives states) and *idle, dc_w, ac_w* and *end_vld* (non actives states) (see Fig. 2).

With a physical implementation on a 0.18 micron technology, the typical energy consumption per gate is $E_{gate} = 0.240$ pJ. Taking into account the estimated activity values and the expression (5), we obtain the value of the energy consumption per cycle, showed in the Table 1. The values according to the current state are accumulated in a variable associated to the instance of the model.

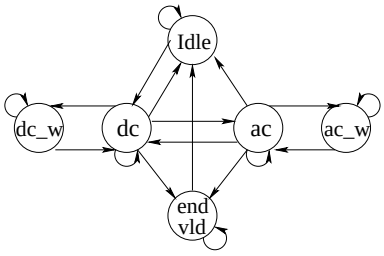

Figure 2: VLD state machine.

Table 1: VLD energy consumption.

VLD state	Energy per cycle (pJ)
idle	9.6
dc	317
dc_w	9.6
ac	317
ac_w	9.6
end_vld	9.6

4.2 Memory Energy Model

The memory energy model reflects the fact that the energy consumption per cycle is usually different in every operating mode: standby, RAS cycle, refresh, etc. Some memories allow even the power modes to be externally controlled from a power management unit.

The memory we use is a NEC SDRAM device with several power states: operating, idle, power down, pre-charge and refresh [12]. Besides the modes that are entered by the intrinsic behavior of the state machine, a power down mode exists, that is entered using an external signal, CKE (CK Enable), and can be thus controlled by a power management unit. A TSS

model that implements all these states is not yet available. For the moment the memory model implements only the following states: *idle*, *read_pend*, *read*, *write* and *write_pend* (see Fig. 3). The model has an adittional wait parameter to introduce wait states (*write_pend* and *read_pend*).

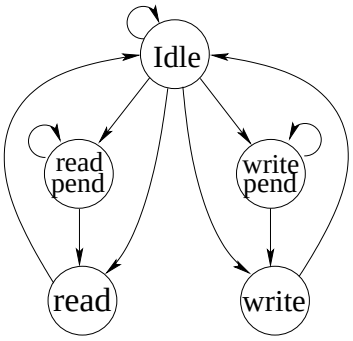

Figure 3: Memory state machine.

The size of the used memory is 64 Mbits. The memory is divided into four banks of 2^{12} rows with 256 words of 16 bits. Two memories in parallel will be used to obtain 32-bit accesses. When a word is not in the output memory buffer a new row has to be fetched from the corresponding bank (page miss). A page miss adds a latency of 3 cycles. Since this latency is not represented in our model, we introduce a scaling factor τ that takes into account the average page hit/miss ratio according to the equation: $\tau = (\%page\_miss * 3 + \%page\_hit * 1)cycles$.

For a system consisting of N memories, simultaneously running at the operating current I, voltage V and frequency f (these three last values are specified in the data-sheet), the energy consumption per cycle is given in (6).

$$E = \tau * \frac{I * V}{f} * N_{mem} \tag{6}$$

The page hit/miss ratio in MPEG-2 decoding was about 3/4 (25% pages miss). We consider a similar value for MPEG-4. So the active states have $\tau = 1.5$. Idle states do not need this parameter in the equation ($\tau = 1$) and pending states are not used.

The operating current of the memory at 3.3 V and 100 MHz is 100 mA, and the idle current is 20 mA. Introducing these values in expression (6) leads to the energy consumption per cycle listed in the Table 2.

Table 2: Memory energy consumption.

Memory state	**Energy per cycle (pJ)**
idle	1320
read	9900
write	9900

4.3 Processor Energy Model

The processor energy model corresponds to the energy consumption per cycle during the execution of a program in the processor. Sinha [13] demonstrated that the energy consumption in ARM processors varies only about 8% between the different instructions of a program. Therefore we can consider an average energy consumption for all the instructions of our application program, depicted in ARM data-sheet. When there is a cache miss or when the processor is waiting for the result from the VLD, it does not execute instructions. The processor stalls and executes NOP cycles that consume less energy. So we distingue two types of cycles: active cycles, when the processor is normally running and NOP cycles, when the processor is waiting for data or instructions.

Our processor is an ARM920T (ARM9 with I + D caches) whose energy consumption in active cycles is 900 pJ and in NOP cycles 400 pJ [3, 4]. The type of cycle can be detected during the simulation by analyzing the bus transfers. The energy consumption for each type of cycle is shown in the Table 3.

Table 3: Processor energy consumption.

Processor state	Energy per cycle (pJ)
active	900
NOP	400

4.4 Interconnect Energy Model

The system performs two types of transfers, one between the processor and the VLD, and the other one between the processor and the memory. Therefore there are two kinds of energy consumption for the interconnect, both using the same energy model. If we have N_{switch} switching lines during a cycle, with a capacitance C_{line} per interconnect line and a voltage swing of V_{dd}, the energy used in the model is represented in (7).

$$E_{interconnect} = N_{switch} * C_{line} * V_{dd}^2 \tag{7}$$

A 1.1 pF on-chip line capacitance is derived for CPU-VLD transfers. The voltage is 1.8 V for 0.18 micron technology. N_{switch} is obtained during the simulation by monitoring the wire activity with a spy module, that detects for every cycle the number of switching lines in data and address buses.

The memory transfers consume more power than the VLD ones, because the memory is off-chip and the capacitances involved are bigger. They correspond to the PCB track, pin and driver capacitances. The total is 10 pF and the PCB voltage is 3.3 V. N_{switch} is obtained during the simulation by monitoring with another spy module. The resulting energies per switched line are shown in the Table 4.

The total energy of the system and the instantaneous energy consumption of the current cycle are available through simulation variables.

Table 4: Interconnect energy consumption.

Transfer	Energy per cycle (pJ)
CPU–VLD	3.5
CPU–Mem	108.9

4.5 System Simulation

The total energy of the system and the instantaneous energy consumption of the current cycle are available through simulation variables. This information allows to know the system activity and consumption evolution in the time. Our system is highly fluctuating and the blocks have very irregular behaviors, so knowing this evolution is a very interesting characteristic

to study the application of low power techniques like dynamic power management, performing the dynamic control of power states transitions, or dynamic voltage scaling, allowing dynamic modifications of frequency and voltage.

5 Conclusions and Future Work

We have presented a simulation framework to estimate the energy consumption at high description level of embedded systems containing new hardware blocks. We apply this methodology to a System-level cycle-accurate simulator and a Instruction-Set Simulator. They have been extended with energy models allowing to obtain power estimations by simulation at system-level. The energy model provides the typical energy for each state of the component. The blocks and system power consumption are being measured with a lower level power estimation tool. This will allow to validate the approximation and to give information about the error involved. That will also help to calibrate the energy models and to extend them for different blocks architectures. The system will be completed with the rest of hardware blocks of the codec video MPEG-4. This approach allows the application of differents dynamic low power techniques and their utility can be evaluated. This is a very interesting characteristic because it lets to model and optimize the power consumption in highly fluctuating systems, composed of blocks with very irregular behaviors, e.g. communications networks or interactive systems. This method is intended to help designers to develop complex systems. It allows the exploration of the energy consumption at earlier stages of the architecture definition. The impact of low power techniques can be analyzed at system level.

References

[1] IMEC: http://www.imec.be/desics/design_technology_top.html

[2] Theeuwen, F.: TSS, System Simulation at Philips Research. Talk at the MEDEA Workshop on System Simulation, May 1998.

[3] ARM, Advanced RISC Machines Ltd.: ARM Software Development Toolkit Version 2.11. 1996.

[4] Simunic, T.,Benini, L., and De Micheli, G.: Cycle-Accurate Simulation of Energy Consumption in Embedded Systems. Proceedings of the 36th IEEE DAC'99, June 1999, pp. 867-872.

[5] DIESEL 2.5 User Manual, Electronic Design & Tools Group, Philips Research, June 2001.

[6] Schimpfle, C., V.,Simon, S. and Nossek, J., A.: High-Level Circuit Modeling for Power Estimation. Proceedings of the 6th IEEE ICECS'99, Sept 1999, Vol. 2, pp. 807-810.

[7] Peset Llopis, R. and Goossens ,K.: The Petrol Approach to High-Level Power Estimation. Proceedings of the IEEE ISLPED'98, Aug 1998, pp. 130-132.

[8] Ye, W., Vijaykrishnan, N., Kandemir, M. and Irwin, M., J.: The Design and Use of SimplePower: A Cycle-Accurate Energy Estimation Tool. Proceedings of the IEEE DAC'00, Jun 2000, pp. 340-345.

[9] Li, Y. and Henkel, J.:A Framework for Estimating and Minimizing Energy Dissipation of Embedded HW/SW Systems. Proceedings of the IEEE DAC'98, Jun 1998, pp. 188-193.

[10] MPEG-4, Motion Picture Expert Group: http://mpeg.telecomitalialab.com/

[11] Huffman, D., A.: A method for the construction of minimun redundancy codes. Proceedings of the IRE, Sept 1952, vol. 40, pp. 1098-1101.

[12] NEC uPD4564163 Datasheets: 64Mbit Synchronous DRAM, 4 bank, LVTTL. Oct 1998.

[13] Sinha, A. and Chandrakasan, A., P.: JouleTrack - A web based tool for software energy profiling. Proceddings of the IEEE DAC'01, 2001, pp. 220-225.