A Power Modeling and Estimation Framework for VLIW-based Embedded Systems

L. Benini[§]   D. Bruni[§]   M. Chinosi[†]   C. Silvano[♯]   V. Zaccaria[‡]   R. Zafalon[†]

[§]Università degli Studi di Bologna, Bologna, ITALY
[†]STMicroelectronics, Agrate B. (MI), ITALY
[♯]Università degli Studi di Milano, Milano, ITALY
[‡] Politecnico di Milano, Milano, ITALY

Abstract. In this paper, a framework for modeling and estimating the system-level power consumption for embedded VLIW (Very Long Instruction Word) architectures is proposed. Power macro-models have been developed for the main components of the system: namely the core, the register file, the instruction and data caches. The models have been integrated within a hierarchy of dynamic power estimation engines, at several abstraction levels. The main goal is to define a system-level simulation framework *(i)* to profile dynamically the power behavior during software execution and *(ii)* to provide a break-out of the power contributions due to the single components of the system. The proposed methodology has been applied to an industrial case study: the Lx family of scalable embedded VLIW processors, designed for multimedia and signal processing applications. Extensive validation of the proposed methodology has been carried out over a set of multimedia benchmarks for embedded applications. The experimental results have demonstrated an average accuracy of 5% of the instruction-level estimation engine with respect to the RTL engine, with an average speed-up of four orders of magnitude.

1   Introduction

High-level power/energy estimation techniques [1, 2] based on macro-modeling can be used for software power estimation, by leveraging fast cycle-accurate HDL simulators. HDL simulation speed for complex processor cores and memory systems is still insufficient to estimate the power/energy consumed by realistic applications. For this reason, higher-abstraction approaches for software-level power estimation have been proposed in the last few years. Probably, the best-known technique in this class is *instruction-level power analysis* [3]. This approach defines a power consumption value for each instruction (or instruction pair) in the instruction set, and computes average power by weighted averaging of power costs with instruction execution frequency (obtained by instruction-level simulation).

Instruction-level power analysis (ILPA, for brevity) has been successful in estimating power for relatively simple embedded cores (SPARC, ARM), as well

as off-the-shelf processors. The main limitations of ILPA are: *(i)* it does not provide any insight on the causes of power consumption within the processor core, which is seen as a black box; *(ii)* it does not account for the power consumed in the memory system, which is often dominant. To address the second limitation, researchers have developed power estimation frameworks which integrate processor and memory models [4–7] and are built around instruction set simulators. Instruction set simulators produce both the instruction profiles for ILPA, and address traces to drive memory system simulators, augmented by memory power models [8–12]. These integrated core-and-memory simulators are fast enough to run complex applications for millions of cycles. Their accuracy has not been fully validated for system-on-chip designs, but it has been shown to be satisfactory for board-level designs built with commercial off-the-shelf components [5].

The lack of insight on the sources of power consumption within the processor core has been recently addressed by a new generation of microarchitectural power estimation tools, targeting high-end processors with complex microarchitectures [13–15] and on-chip caches. The main purpose of these tools is to support exploration of micro-architectural tradeoffs in processor design, when energy is one of the metrics of interest. Similarly to ILPA, these estimators are built around an instruction set simulator, but they feature a detailed micro-architectural model of the processor, with separate power models for its main functional units. Analytic energy models for caches are also provided [11], where energy per access is automatically scaled depending on cache organization (e.g., number of cache lines, associativity, etc.). Micro-architectural power modeling is a tool for processor architects, aiming at exploring the design space of processor and cache organizations. The target of power modeling is a good relative accuracy over a wide range of hardware configurations.

Our work focuses on software power estimation for embedded applications, where an embedded core (with a memory hierarchy) is integrated in complex system-on-chip solutions. In contrast with micro-architectural power modeling for hardware design exploration, the main purpose of the estimation engine is to provide power consumption estimates for *software running on a given hardware architecture*, and to help optimizing the target application for energy efficiency. While relative accuracy is certainly useful, absolute accuracy is the ultimate target.

The main difference with respect to ILPA-based approaches is that our approach gives better insight on the power bottlenecks during software execution, because it is based on a detailed micro-architectural model of the core. The proposed power estimation methodology is based on a complex system-level simulation framework to profile dynamically the power behavior during software execution and to provide a break-out of the power contributions due to the single components of the system. This approach can be adopted in an industrial environment, where detailed description on the processor's hardware architecture is available.

The main contributions of our work are: *(i)* the development of novel power macro-models for the main components of the system, namely the VLIW core,

the register file and the caches; *(ii)* the validation methodology to evaluate the accuracy of the macro-models against post-layout circuit and gate-level simulation; *(iii)* the integration of the power macro-models within a hierarchy of simulators, from RT-level (cycle-accurate) to the instruction-level. Our work demonstrates the viability of high-level power estimation for processor cores both from the *efficiency* and from the *absolute accuracy* standpoints.

As a case study, we describe the application of the proposed modeling and estimation framework to support the system-level power analysis for the Lx core, a high-performance embedded VLIW processor for multimedia and signal processing applications, jointly developed by Hewlett-Packard and STMicroelectronics [16]. In the Lx processor, a very long instruction (*bundle*) is composed of four explicitly parallel instructions (*syllables*). A complete software environment is being developed concurrently with the hardware [16]. Software development support includes an aggressive ILP compiler, instruction-set simulators and the power estimation environment described in this paper.

The rest of the paper is organized as follows. In Section 2, we describe the overall power estimation framework based on an instruction-level engine characterized by using an RT-level engine. The energy models for the VLIW core, the register file and the instruction and data caches are discussed in Section 3, while experimental results derived from the application of the proposed methodology to a case study are described in Section 4.

2   Power Estimation Framework

The proposed power analysis environment is built as a hierarchy of estimators, at several levels of abstraction. At the lowest level, the baseline power estimation is provided by commercial power estimation tools, working at the circuit and gate level. Using the characterization data from low-level simulators, we built a complete Register Transfer Level (RTL) power model for the system. The RTL model is embedded within a functional RTL description of the core, written in Verilog. The RTL power estimation engine has then been used as the reference for building the instruction-level (IL) engine, which is coupled with an Instruction Set Simulator (ISS). Design size prevents full-core estimation at the lowest level, hence the RTL macro-models have been built by low-level simulation of one target RTL at a time. Full-core estimation at the RTL is feasible, but fairly slow (160 bundles per second on average), while IL estimation is much faster (1.7 millions of bundles per second on average).

2.1   RT-Level Engine

The RTL engine is based on power macro-models characterized by either gate-level analysis (with back-annotation of wiring capacitances extracted from layout) for synthesized modules, or transistor-level power analysis for post-layout full-custom modules, such as cache memory banks and RF. The macro-models for all synthesized units are based on lookup tables [2], while the macro-models for memory banks and RF are designed ad-hoc. All macro-models are linked to a cycle-accurate RTL simulation model of the core through the standard PLI

interface. Thus, power estimation is obtained as a by-product of RTL functional simulation.

2.2   Instruction-Level Engine

The IL power estimation engine is based on the Instruction Set Simulator (ISS) of the target machine. The ISS interprets an executable program by simulating and profiling the effects of each instruction on the main components of the architectural state of the system (e.g., the RF, the memory hierarchy and the program counter). In our framework, the ISS is used to derive a very fast estimate of the actual RTL architectural state. These estimates are then combined with the RTL power models to infer the power consumption of the entire system. The IL engine is characterized by two modes of operation:

Instantaneous Power Report: Instantaneous power consumption values $P(t)$ are dynamically traced during bundles execution.
Time-Averaged Power Report: Power consumption is computed at the end of the simulation time as an average value of the function $P(t)$.

The accuracy of the IL power estimation engine depends on how well the ISS infers the correct RT-state and must be traded off with the ISS speed. Experimental results have shown an average accuracy of approximately 5% of the IL engine with respect to the RTL engine.

3   Power Macro-Modeling

In this section, we describe the macro-models developed to describe the power behavior of the main resources of the target system architecture, namely the VLIW core, the RF, and the separated I- and D-caches. The main issues of the proposed power macro-models are: *(i)* they are tightly related to the micro-architectural details of each system module; *(ii)* they accurately consider the processor-to-memory communication in terms of read/write accesses to each level of the memory hierarchy; *(iii)* they can be used at both RTL and IL to estimate the power consumption.

3.1   VLIW Core Model

For VLIW architectures, an instruction-level energy model should account for all possible combinations of instructions (syllables) in a very long instruction (bundle), thus the problem complexity is $O(N^{2K})$ where N is the number of syllables in the ISA and K is the number of syllables in a bundle. The analytical energy model proposed in this section aims at reducing the complexity of the instruction-level energy model proposed in [17], while preserving a good level of accuracy in the estimates with respect to energy estimates derived from gate-level description of the core. The main simplification consists of indirectly taking into account of complex inter-instruction effects, mainly in terms of additional *stall/NOP* cycles introduced during the execution of an instruction. In the target VLIW architecture, we assume that, when an I-cache miss occurs, a sequence

of NOPs is generated, while when a D-cache miss occurs, the core stalls the pipelines until the miss is served. Let us consider a stream W composed of N bundles w_n where $n \in [1 \ldots N]$. The average energy consumption per instruction can be decomposed as:

$$E(w_n) = B + \alpha_n * c_{syl} + m * p * S + l * q * M$$

where:

- B is an average energy base cost;
- α_n is the number of syllables different from NOPs within the bundle w_n, and c_{syl} is the average energy consumption associated to the execution of a syllable;
- The third term in the summation is the additive average energy consumption due to a miss event on the D-cache, where m is the average number of additional stall cycles per bundle occurred due to a D-cache miss, p is the probability per bundle that a D-cache miss occurs, and S is the core energy consumption during a pipeline stall.
- The fourth term is related to the I-cache misses, where l is the average number of additional instruction cache NOP operations per bundle introduced during an I-cache miss, q is the probability per bundle that this event occurs after the execution of w_n, and M is the energy consumption of the core during an I-cache miss.

Globally, the average power associated with the stream W can be expressed as:

$$P(W) = (1 - f_S - f_M)\frac{(B + \overline{\alpha} * c_{syl})}{T_c} + f_S\frac{S}{T_c} + f_M\frac{M}{T_c}$$

where T_c is the clock period, f_S is the fraction of time spent by the processor stalling the pipeline, f_M is the fraction of time spent by the processor during an I-cache miss, and $\overline{\alpha}$ is the average number of syllables per bundle different from NOPs. The average power is therefore linear with respect to three power contributions: the one-cycle-per-instruction ideal power consumption, the power due to a pipeline stall and the power due to an I-cache miss.

3.2   Register File Model

The general problem of evaluating the power consumption of RFs has recently been addressed in [18]. The paper compares various RF design techniques in terms of energy consumption, as a function of architectural parameters such as the number of registers and the number of ports.

In our work, we propose a parametric power model of a multi-ported RF: the power behavior is linear with respect to the number of simultaneous read/write accesses performed on the different ports:

$$P_{RF} = P_i + \frac{1}{T}\sum_{1 \leq n \leq N}(E_{r,n} + E_{w,n})$$

where P_i is the RF base power cost measured when neither read nor write accesses are performed, T is the total simulation time, $E_{r,n}$ $(E_{w,n})$ is the energy consumption of a read (write) access occurred during bundle w_n, and f_S has been defined above.

The energy contribution $E_{r,n}$ is defined as:

$$E_{r,n} = \sum_{1 \le i \le N_{rp}} H(RR_{i,n}, RR_{i,n-1}) * E_{rb}$$

where N_{rp} is the number of read ports of the RF, H is the Hamming distance function, $RR_{i,k}$ is the data value read from the RF output port i by the k-th bundle and E_{rb} is the energy consumption associated with a single bit change on a read port.

The energy contribution $E_{w,n}$ is defined as:

$$E_{w,n} = \sum_{1 \le i \le N_{wp}} H(RW_{i,n}, old_{i,n}) * E_{wb}$$

where N_{wp} is the number of write ports of the RF, H is the Hamming distance function, $RW_{i,n}$ is the new data value written by the n-th bundle on input port i, $old_{i,n}$ is the previous data value contained in the same RF location and E_{wb} is the energy consumption associated with a single bit change on a write port.

3.3   Cache Model

Most published analytic cache models [11] deal with relatively simple cache organizations, and they are not suitable for modeling complex caches based on multiple SRAM memory banks, with a significant amount of control logic. The multi-banked structure is dictated mainly by performance constraints, because cache access time is critical for overall processor performance.

The modeling approach proposed in this paper is hierarchical: We first built power macro-models for all the various types of SRAM banks contained in the caches, and then we compose these models in a single logical model that creates the correct access patterns for every bank according to the cache organization.

The macro-models for the atomic SRAM modules are *mode-based:* power consumption depends on the mode of operation (i.e., `read`, `write`, `idle`). More precisely, since the SRAM modules are synchronous, the energy consumed in a given clock cycle is mainly a function of the *mode transition* between the previous and the current cycle. Thus, we characterized energy as a function of the nine possible mode transitions (e.g., `read-read`, `read-write`, etc.). For a given mode transition, energy is weakly dependent on the number of transitions on the address lines. Accounting for this dependency leads to a macro-models with $9 \cdot (N_{addr} + 1)$ characterization coefficients, where N_{addr} is the number of address lines. The coefficients were characterized by simulating the layout-extracted transistor-level netlist of the SRAM modules with the MACH-PA circuit simulator by Mentor Graphics. Average accuracy of the SRAM macro-models was satisfactory (percentage average errors are within 5%).

Composition of the atomic macro-models in the complete cache model is trivial at the RT level, because the RTL description of the cache subsystem does contain the behavioral description of every SRAM module. Hence, we simply linked the power macro-models with each SRAM module instance in the RTL description via PLI calls. Building the cache model for instruction simulation was not as straightforward, because the ISS simply provides cumulative counts cache accesses, but it does not contain any knowledge of internal cache organization. Hence, we re-constructed the pattern of accesses to the various SRAM sub-modules in response to every type of cache access. Unfortunately, it is not possible to fully reconstruct SRAM access patterns from cumulative counts, and some loss of accuracy is incurred in the process. Nevertheless, accuracy is still satisfactory. In the worst case (the data cache) percentage error with respect to RTL simulation is below 17%. Accuracy could be improved by modifying the instruction simulator, adding more detailed access cache reporting features.

4   Experimental Results

In this section, we describe how the proposed power modeling and estimation techniques have been successfully applied to an industrial case study. The experimental results have been carried out over a set of selected benchmark applications including C language implementation of digital filters, discrete cosine transforms, etc., especially tuned for embedded processors.

4.1   Target System Architecture

The target architecture is based on the scalable and customizable Lx processor technology [16] designed for multimedia and signal processing embedded applications. The Lx processor is a statically scheduled VLIW architecture designed by Hewlett-Packard and STMicroelectronics to supports a multi-cluster organization based on a single PC and a unified I-cache. The single-cluster is 4-issue VLIW core composed of four 32-bit integer ALUs, two 16x32 multipliers, one load/store unit and one branch unit. The cluster also includes 64 32-bit GPRs an 8 1-bit Branch Registers. Lx supports an in-order six-stage pipeline and a very simple integer RISC ISA. For the first generation, the scalable Lx architecture is planned to span from one to four clusters (i.e., from 4 to 16 issued instructions per cycle).

Lx comes with a commercial software toolchain, where no visible changes are exposed to the programmer when the core is scaled and customized. The toolchain includes a sophisticated ILP compiler technology (derived from the Multiflow compiler [19]) coupled with GNU tools and libraries. The Multiflow compiler includes most traditional high-level optimization algorithms and aggressive code motion technology based on trace scheduling.

4.2   Characterization and Validation of the Power Macro-Model

The RTL power estimator links the power macro-models to the RTL functional description of the processor. The RTL macro-model for the core has been validated against gate-level simulation on the selected set of benchmarks. The agreement between predicted and measured power values is shown in Fig. 1. The plot

clearly illustrates three different regions where power consumption is dominated by D-cache misses, I-cache misses and ideal execution.

$System Module$	$Ref.$	$Max.Err.$	$Avg.Err.$	Rms
Core	GL	10%	-0.74%	4.1%
RF	TL	8%	-0.17%	1.75%
Cache Tags	TL	-2.6%	0.23%	1.82%
I-Cache Bank	TL	1.6%	-0.12%	0.9%
D-Cache Bank	TL	1.8%	0.08%	0.8%

Table 1. Comparison results of RTL vs. reference power models for each system module. GL (TL) stands for gate-level (transistor-level) description of reference models.

The RTL models of RF and caches have been validated against transistor-level simulation of the circuit extracted from layout including parasitics. Table 1 summarizes the accuracy of the characterization of all the system modules. The maximum and the average error are reported in the second and third column respectively, while the corresponding rms is reported in the fourth column. The maximum error is within $\pm 10\%$. The maximum values of the average error and the rms are -0.74% and 4.1% respectively.

Fig. 2 shows the total average power consumption for each benchmark as well as the break-out into the contributions due to the single system modules. The total power is dominated by the I-cache and D-cache contributions.

4.3   Characterization and Validation at the Instruction-Level

The IL engine is based on the ISS available in Lx toolchain. The Lx ISS has been purposely modified to gather a fast estimate of the RTL status and parameters. These values are then linked to the power macro-models to get power estimates. The proposed IL estimation engine has been applied to the selected set of benchmarks to demonstrate the viability of our approach both from the absolute accuracy and the efficiency standpoints. Comparison results between IL power estimates with respect to RTL estimates are shown in Fig. 3. For the benchmark set, the average error is 5.2%, while the maximum error is 7.9%. The RTL engine simulates 160 bundles per second on average, while the IL engine simulates 1.7 millions of bundles per second on average, thus providing a speed-up of 10^4 approximately.

5   Conclusions

In this paper, we have presented an efficient and accurate framework for embedded core power modeling and estimation. The proposed methodology has been validated against a complete post-layout implementation on an industrial embedded core. Future directions of our work aim at defining: *(i)* power efficient instruction scheduling opportunities, *(ii)* a more general exploration methodology to evaluate power-performance tradeoffs at the system-level, and *(iii)* techniques to optimize the software code from the power standpoint.

2.1.9   L. Benini et al.
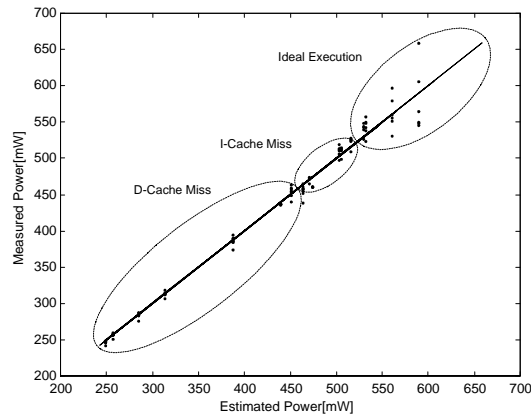
Fig. 1. Agreement between measured Gate-Level power values and estimated RT-Level power values for the Lx core (Maximum error within ±10%).
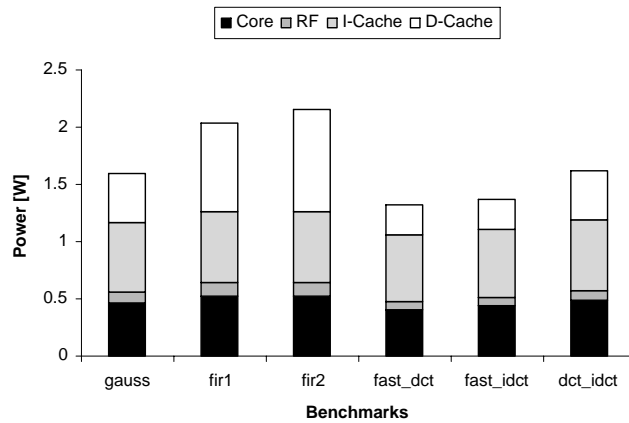
Fig. 2. RTL power break-out due to core, RF, I-cache, and D-cache for the benchmark set.

References

1. P. Landman, "High-Level Power Estimation," in *Proceedings of ISLPED*, pp. 29–35, 1996.
2. E. Macii, M. Pedram, F. Somenzi, "High-Level Power Modeling, Estimation and Optimization," *IEEE Transactions on CAD*, vol. 17, no. 11, pp. 1061–1079, 1998.
3. V. Tiwari, S. Malik, A. Wolfe, "Power Analysis of Embedded Software: A First Step Towards Software Power Minimization," *IEEE Transactions on VLSI Systems*, vol. 2, no.4, pp.437–445, Dec. 1994.
4. Y. Li, J. Henkel, "A Framework for Estimating and Minimizing Energy Dissipation of Embedded HW/SW Systems," *Design Automation Conference*, pp.188–193, June 1998.
5. T. Simunic, L. Benini, G. De Micheli, "Cycle-accurate simulation of energy consumption in embedded systems," *Design Automation Conference*, pp. 867-872, June 1999.
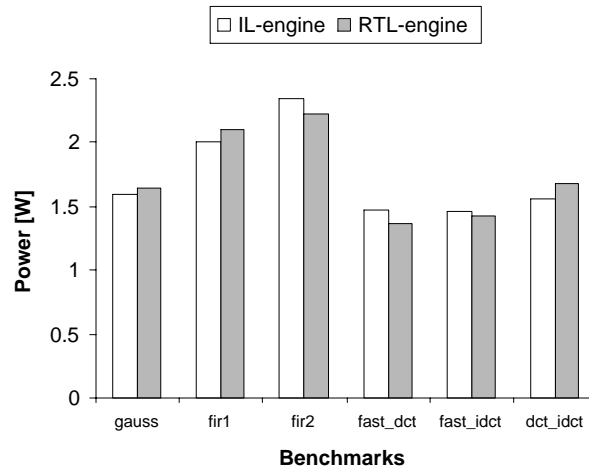
Fig. 3. Comparison between Instruction-Level power estimates and RTL power estimates for the benchmark set. (Average Error 5.2% - Maximum Error 7.9%).

6. M. Lajolo, A. Raghunathan, S. Dey, L. Lavagno, "Efficient power co-estimation techniques for system-on-chip design," *Design, Automation and Test in Europe Conference*, pp. 27-34, March 2000.
7. T. Givargis, F. Vahid, J. Henkel, "Fast cache and bus power estimation for parameterized system-on-chip design," *Design Automation and Test in Europe Conference*, pp. 333-338, March 2000.
8. D. Liu, C. Svensson, "Power consumption estimation in CMOS VLSI chips," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 6, pp. 663-670, June 1994.
9. R. Evans, P. Franzon, "Energy consumption modeling and optimization for SRAM's," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 5, pp. 571-579, May 1995.
10. P. Landman, J. Rabaey, "Architectural power analysis: the dual bit type method," *IEEE Transactions on VLSI*, vol. 3, no. 2, pp. 173-187, June 1995.
11. M. Kamble, K. Ghose, "Analytical Energy Dissipation Models for Low Power Caches," *International Symposium on Low Power Electronics and Design*, pp. 143–148, August 1997.
12. S. Coumeri, D. Thomas, "Memory Modeling for System Synthesis," *IEEE Transactions on VLSI*, vol. 8, no. 3, pp. 327-334, June 2000.
13. W. Ye, N. Vijaykrishna, M. Kandemir, M. Irwin, "The Design and Use of SimplePower: A Cycle-Accurate Energy Estimation Tool" *Design Automation Conference*, pp. 340-345, June 2000.
14. D. Brooks, V. Tiwari, M. Martonosi "Wattch: A Framework for Architectural-Level Power Analysis and Optimization, *International Symposium on Computer Architecture*, pp. 83-94, June 2000.
15. D. Brooks, et al. "Power-microarchitecture: design and modeling challenges for next-generation microprocessors," *IEEE Micro*, vol. 20, n. 6, pp. 26-44, Nov/Dec 2000.
16. P. Faraboschi, G. Brown, J. Fisher, G. Desoli, F. Homewood, "Lx: a technology platform for customizable VLIW embedded processing," *International Symposium on Computer Architecture*, pp. 203-213, June 2000
17. M. Sami, D. Sciuto, C. Silvano and V. Zaccaria, "Power Exploration for Embedded VLIW Architectures," *ICCAD-2000: IEEE/ACM Int. Conference on Compuer Aided Design,* pp. 498-503, San Jose, CA, Nov. 5-9, 2000.
18. V. Zyuban, P. Kogge, "The energy complexity of register files," *International Symposium on Low Power Electronics and Design*, pp. 305-310, August. 1998.
19. P. G. Lowney *et al.* "The Multiflow Trace Scheduling Compiler," *Journal of Supercomputing,* 7 (1/2), pp. 51-142.