

A POWER MODELING AND ESTIMATION FRAMEWORK FOR VLIW-BASED EMBEDDED SYSTEMS

L. Benini⁽¹⁾
D. Bruni⁽¹⁾
M. Chinosi⁽²⁾
R. Zafalon⁽²⁾
C. Silvano⁽³⁾
V. Zaccaria⁽⁴⁾

(1) Università degli Studi di Bologna, Bologna, Italy

(2) STMicroelectronics

(3) Università degli Studi di Milano, Milano, Italy

(4) Politecnico di Milano, Milano, Italy

In this paper, we propose a framework for modeling and estimation of the system-level power consumption for embedded VLIW (Very Long Instruction Word) architectures. We have developed power macro-models for the main components of the system, namely the core, the register file, instruction and data caches, all of which we have integrated, at several abstraction levels, within a hierarchy of dynamic power estimation engines. Our 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.

We have applied the proposed methodology to an industrial case study: the Lx family of scalable embedded VLIW processors, designed for multimedia and signal processing applications.

We have carried out an extensive

validation of the proposed methodology over a set of multimedia benchmarks for embedded applications. Our 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

Most current systems-on-chip contain one or more processors and a significant amount of memory organized hierarchically. Processors and memories are large hardware blocks, and their power consumption during software execution cannot be assessed by circuit-level or gate-level tools for obvious efficiency reasons.

Landman and Macii et al provide high-level power/energy estimation techniques [1, 2] based on macro-modeling for software power estimation, which leverage fast cycle-accurate HDL simulators. Unfortunately, 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].

A new generation of microarchitectural power estimation tools, targeting high-end processors with complex microarchitecture [13-15] and on-chip caches, have recently addressed the lack of insight into the sources of power consumption. The main purpose of these tools is to support exploration of microarchitectural tradeoffs in processor design, where energy (power) 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 (e.g., execution units, prefetch buffers, register files, etc.). 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, aimed 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.

1.1 Main contributions of this work

Our work focuses on the development of a general framework for software level power estimation for embedded solutions, 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 optimize the target application for energy efficiency.

The main difference with respect to ILPA-based approaches is that our approach gives better insight into the power bottlenecks during software execution because it is based on a macro-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 adapted to an industrial environment, where detailed description on the processor's hardware architecture is available.

The main contributions of our work are:

- The definition of a system-level integrated framework for software power estimation;
- The development of novel power macro-models for the main components of the system, namely the VLIW core, the register file and the caches;

- The validation methodology to evaluate the relative accuracy of the macro-models against post-layout circuit and gate-level simulation;
- The integration of the power macro-models within a hierarchy of simulators from RT-level (cycle-accurate) to the instruction-level.

Our work analyzes the viability of high-level power estimation for processor cores both from the efficiency and from the *relative 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*). Faraboschi et al are concurrently developing a complete software environment 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. We discuss the energy models for the VLIW core, the register file, and the instruction and data caches in Section 3 and describe the experimental results derived from the application of the proposed methodology to a case study in Section 4.

2 POWER ESTIMATION FRAMEWORK

We have built the proposed power analysis environment as a hierarchy of estimators at several levels of abstraction. At the lowest level, commercial power estimation tools, working at the circuit and gate level, provide the baseline power estimation. Using the characterization data from low-level simulators, we

built a complete Register Transfer Level (RTL) power model for the system. We have embedded the RTL model within a functional RTL description of the core, which we wrote in Verilog. Using the RTL power estimation engine as the reference for building the instruction-level (IL) engine, we have coupled the IL engine 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

Zafalon et al base the RTL engine on the RTPow tool [17], an RT-level power macro-modeling and estimation tool relying on a top-down estimation engine with no need to perform any type of on-the-fly logic synthesis when analyzing the HDL description [17]. We built our first version of the RTL when the layout of the core was not yet available. At this stage we leveraged pre-synthesis RT-level power estimation technology. As the logic and physical design of the various units became available, we progressively replaced the power macro-models provided by RTPow with new ones, characterized by either gate-level analysis (with back-annotation of wiring capacitances extracted from layout) for synthesized modules, or by transistor-level power analysis for post-layout full-custom modules, such as cache memory banks and RF. Macii et al base the macro-models for all synthesized units on lookup tables [2] and design the macro-models for memory banks and RF ad-hoc. They link all macro-models to a cycle-accurate RTL simulation model of the core through the standard PLI interface, thus, obtaining power estimation as a by-product of RTL functional simulation.

2.2 Instruction-Level Engine

We have based the IL power estimation engine 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, we have used the ISS to derive a very fast estimate of the actual RTL architectural state and combined these estimates 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: The dynamic tracing of instantaneous power consumption values $P(t)$ during bundles execution. To give the flavor of such a feature, Fig. 1 shows an example of ISS-based dynamic power profiling applied to the Lx architecture.

Time-Averaged Power Report: The computation of power consumption 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) the tight relationship of each to the macro-architectural details; (ii) the accurate consideration of the processor-to-memory communication in terms of read/write accesses to each level of the memory hierarchy; (iii) their use 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

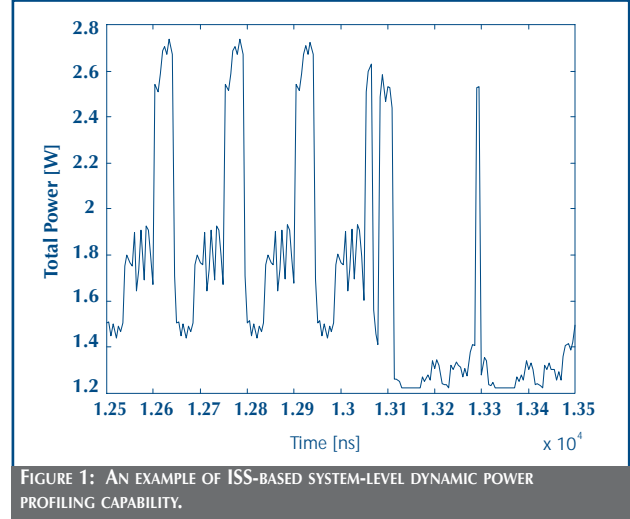


FIGURE 1: AN EXAMPLE OF ISS-BASED SYSTEM-LEVEL DYNAMIC POWER PROFILING CAPABILITY.

syllables in the ISA and K is the number of syllables in a bundle. This means that, if we were to apply a traditional ILPA-based model to a 4-issue VLIW core with an ISA composed of 50 syllables, assuming an average gate-level simulation time of 1 minute to characterize each pair of long instructions, the total simulation time would require millions of years! Sami et al present an attempt to reduce this complexity in [18], preserving a good level of accuracy in the estimates with respect to energy estimates derived from gate-level description of the core. In the present work, we further simplify the model in [18] to reduce simulation time, while indirectly taking into account 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 $n \in [1 \dots N]$. The average energy consumption per instruction decomposes 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 occurring 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 relates 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, we express the average power associated with the stream W as:

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

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 $\bar{\alpha}$ 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.

As we show in the experimental results section, even if this model seems very simplified (since we assume as constants many factors such as B , M and S), the reported characterization and validation errors confirms the validity of the basic assumptions.

3.2 Register File Model

Zyuban and Kogge address the general problem of evaluating the power consumption of RFs, comparing the 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 as defined above.

We define the energy contribution $E_{r,n}$ as:

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

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. We define the energy contribution $E_{w,n}$ as:

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

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. Performance

constraints mainly dictate the multi-banked structure 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 of the various types of SRAM banks contained in the caches, and then we composed these models in a single logical model that creates the correct access patterns for every bank according to the cache organization.

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.

Consequently, we bank the data and instruction cache topology of the LX processor, described as follows. The instruction cache is a 32Kb direct mapped architecture composed of two tags and eight SRAM memory blocks.

This topology enables the extraction of an entire bundle (4 syllables) from the cache memory in the same clock cycle, in case of an I-cache hit. The 32Kb data cache is a 4 way set associative structure with FIFO replacement policy, composed of four tags and eight SRAM memory blocks. Memory blocks used in D-cache enable data write of 64-bit blocks and data read of 32-bit blocks at a time. Designers have chosen this architecture to increase block replacement speed by exploiting the burst access capability of the main memory.

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 have 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. Thus, we can model the energy consumed by cache modules with the following equations:

$$E_{tag} = E_s N_s + \sum_{n=0}^8 \sum_{x=s,r,w} \sum_{y=r,w} E_{xy}[n] N_{xy}[n] \quad (1)$$

$$E_{Imem} = E_s N_s + \sum_{n=0}^{10} \sum_{x=s,r,w} \sum_{y=r,w} E_{xy}[n] N_{xy}[n] \quad (2)$$

$$E_{Dmem} = E_s N_s + \sum_{n=0}^{10} \sum_{x=s,r,w} \{ E_{xw}[n] N_{xr}[n] + \sum_{B=0}^8 E_{wr}[B,n] N_{xw}[B,n] \} \quad (3)$$

where E_s is the energy consumed during a sleep cycle for the selected module and N_s is the number of sleep cycles during the entire RTL simulation. The term $E_{xy}[n]$ is the energy consumed during one access operation, depending on the number $[n]$ of address bits that change from the previous access, the current access operation y , and the access operation x performed during the previous clock cycle.

For the D-cache, E_{Dmem} has an additional term $\sum_{B=0}^8 E_{xw}[B,n] N_{xw}[B,n]$ because the data cache line can be written byte per byte with a bus composed of byte selection signals. In such a case, terms $E_{xw}[B,n]$ and $N_{xw}[B,n]$, depend on the current byte selection $[B]$ as well as the number $[n]$ of address bits changed from the previous to the current access. We can efficiently implement this type of model into the Verilog description with a Look Up Energy table, selecting each row by means of an access code that depends on the parameters x , y , $[n]$ and, if required, $[B]$. Simulating the back-annotated transistor-level netlist of the SRAM modules with Mentor Graphics' MACH-PA circuit simulator characterizes the coefficients.

The average accuracy of the SRAM macro-models used at the RT level are satisfactory (percentage average errors within 5%) compared to gate level, as shown in the next section.

As to IL power estimation, the we propose reconstructing

access patterns to the various SRAM sub-modules in response to every type of cache access. This enables the reuse of the same Look Up Energy models used at RT level. Unfortunately it is not possible to reconstruct SRAM access patterns fully from cumulative counts, and we incur some loss of accuracy in the process because we can only consider the average values associated with the given type of access (read, write, idle) gathered from the Look Up Energy Table. Notwithstanding that, the accuracy is still satisfactory: the worst case (data cache) percentage error, with respect to RTL simulations, is near 18 %. This value is due to the fact that the D-cache has a very complex behavior compared to the I-cache and it is more difficult to gather the parameters needed for the model from global counts.

4 EXPERIMENTAL RESULTS

In this section, we describe the successful application of the proposed power modeling and estimation techniques to an industrial case study. We have carried out the experimental results 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

We based the target architecture 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 support 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 and 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 [20]) 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 appears in Fig. 2. The plot clearly illustrates three different regions where power consumption is dominated by D-cache misses, I-cache misses and ideal execution. A larger spread with respect to the other two regions characterizes the ideal execution region because the ideal one-cycle execution part of the power model (namely the terms $B + a_n * c_{syj}$) does not consider the type of the single operations or the data dependency. In the other two regions, this effect is less significant due to the fact that the cache misses

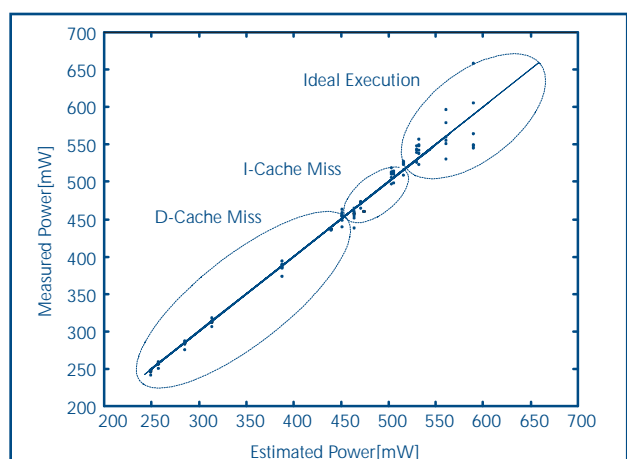


FIGURE 2: AGREEMENT BETWEEN MEASURED GATE-LEVEL POWER VALUES AND ESTIMATED RT-LEVEL POWER VALUES FOR THE Lx CORE (MAXIMUM ERROR WITHIN $\pm 10\%$).

are dominant and only marginally influenced by operation type and data dependency.

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 second and third column report the maximum and the average error, respectively, while the fourth column reports the corresponding rms. The maximum error is within $\pm 10\%$. The maximum values of the average error and the rms are -0.74% and 4.1% respectively.

SYSTEMMODULE	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.

Fig. 3 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 I-cache and D-cache contributions dominate the total power.

4.3 Characterization and Validation at the Instruction-Level

The IL engine is based on the ISS available in Lx toolchain. We have purposely modified the Lx ISS to gather a fast estimate of the RTL status and parameters; we then link these values to the power macro-models to obtain power estimates. We analyze the viability of our approach from both the relative accuracy and efficiency standpoints by applying the proposed IL estimation engine to the selected set of benchmarks. Fig. 4 shows the comparison results between IL power estimates with respect to RTL estimates.

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,

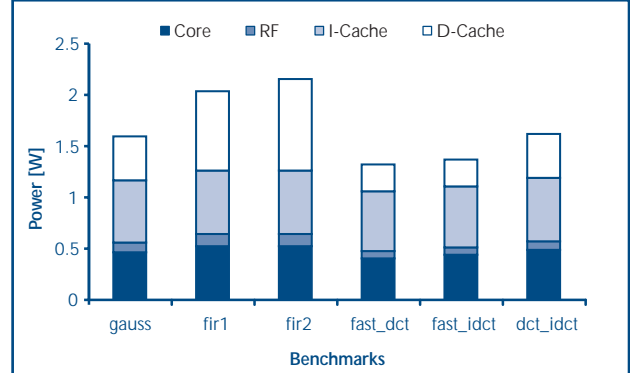


FIGURE 3: BREAK-OUT OF THE RTL POWER CONTRIBUTIONS DUE TO CORE, RF, I-CACHE, AND D-CACHE FOR THE BENCHMARK SET.

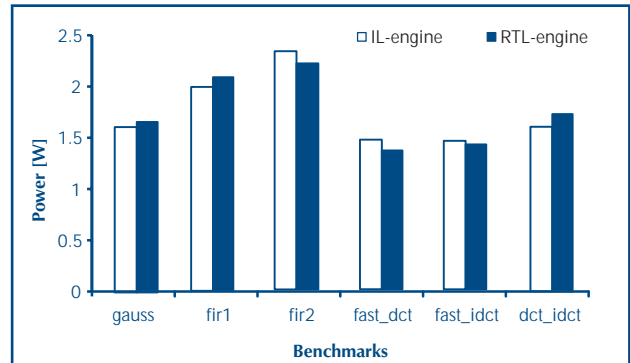


FIGURE 4: COMPARISON BETWEEN INSTRUCTION-LEVEL POWER ESTIMATES AND RTL POWER ESTIMATES FOR THE BENCHMARK SET. (AVERAGE ERROR 5.2% - MAXIMUM ERROR 7.9%).

while the IL engine simulates 1.7 millions of bundles per second on average, thus providing a speed-up of 104 approximately.

5 CONCLUSIONS

In this paper, we have presented an efficient and accurate framework for embedded core power modeling and estimation. We have validated the proposed methodology 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 trade-offs at the system-level, and (iii) techniques optimizing the software code from the power standpoint. Other efforts will address the application of the proposed methodology to superscalar processors.

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.
- [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] R. Zafalon, M. Rossello, E. Macii, M. Poncino, "POWER MACROMODELING FOR A HIGH QUALITY RT-LEVEL POWER ESTIMATION," ISQED 2000: IEEE International Symposium on Quality Electronic Design, pp. 59-63, S.Jose, CA, March 2000.
- [18] M. Sami, D. Sciuto, C. Silvano and V. Zaccaria, "POWER EXPLORATION FOR EMBEDDED VLIW ARCHITECTURES," ICCAD-2000: IEEE/ACM Int. Conference on Computer Aided Design, pp. 498-503, San Jose, CA, Nov. 5-9, 2000.
- [19] V. Zyuban, P. Kogge, "THE ENERGY COMPLEXITY OF REGISTER FILES," International Symposium on Low Power Electronics and Design, pp. 305-310, August 1998.
- [20] P. G. Lowney et al. "THE MULTIFLOW TRACE SCHEDULING COMPILER," Journal of Supercomputing, 7 (1/2), pp. 51-142.

■ CONTACT: ST.JOURNAL@ST.COM ■