The Design of PRESTO: A Framework For Architecture Level Power Estimation

Chitaka Iwama, Luong Dinh Hung, Niko Demus Barli,
Shuichi Sakai and Hidehiko Tanaka

This paper presents PRESTO, a framework for architecture level power estimation. To enable flexible adaptation to various hardware designs, PRESTO separates power model of a hardware structure into three different layers: technology layer, circuit layer and dynamic behavior layer. Furthermore, to increase software reusability, circuit models are built using a hierarchical approach. This paper explains the design of PRESTO and presents validation results of analytical power models for logic gates and SRAM structure. A case study in which PRESTO is used to evaluate an architecture level power reduction technique for caches is also described.

1. Introduction

With recent trend toward higher clock frequency and larger transistor count, power and energy issues are increasingly important for modern microprocessor design. In order to estimate and optimize power at early design phases, many architecture level power estimation tools have been proposed recently [5–7, 9, 11, 14]. Among them, those that use analytical power models are especially popular [6, 11], because the models by nature can be applied to a wide range of architecture design. Unfortunately, from a software design perspective, those tools are lacking the flexibility to make changes or extensions without requiring major modifications to the existing implementations.

We have developed PRESTO, a framework for architecture level power estimation based on analytical power models. PRESTO emphasizes not only on the accuracy of power models but also on its flexibility as a software tool. PRESTO employs the following software design policies:

(1) Layering of power model : model for a hardware structure is divided into three modeling layers i.e *technology layer*, *circuit layer* and *dynamic behavior layer*.
(2) Hierarchical circuit composition : model for a large circuit is composed of models of smaller subcircuits.

The first policy makes it easier to modify or extend a model in one layer without affecting other layers. For example, in order to model different process technologies, only the technology layer needs to be modified. The second policy increases the reusability of power model and the efficiency of software development. Some components such as decoder and memory cell can be repeatedly used in multiple function blocks.

Taking advantage of the PRESTO's layered structure, we added in the technology layer accurate device models based on analytical MOS models of SPICE. Device parameters such as gate capacitance per unit area are calculated directly from SPICE model parameters. Thus, PRESTO can model different process technology provided appropriate SPICE model parameter set. We have confirmed that the estimated energy constants of simple logic gates and SRAM cell arrays are within 10% of accuracy against SPICE.

As a case study, we used PRESTO for evaluating a cache power reduction technique called dynamic zero compression [12]. The hierarchical circuit composition helped to incorporate circuit-level modifications of the proposed cache with small programming effort.

The remainder of the paper is organized as follows. Section 2 describes prior work and explains our contributions. Section 3 describes the structure of PRESTO and discusses the advantages of the design from a software design perspective. The power models implemented in PRESTO are explained and validated in Section 4. Section 5 presents a case study on dynamic zero compression for caches. Section 6 concludes the paper.

2. Prior Work

Generally, there are two approaches for power

estimation : empirical approach and analytical approach. Empirical approach uses power data measured with lower level tools. Several industrial companies have developed architecture level power estimation tools in this approach [7, 8]. SimplePower [14] and AccuPower [9] also use power constants obtained from layout implementation and circuit level simulation. Analytical approach estimates power from transistor-level schematic. CACTI [13] [10] [11] estimates cache access time, area, and energy consumption per access based on an analytical SRAM model. Wattch [6] models superscalar data path components using similar SRAM models, as well as several macro level assumptions on clocking power. Combined with SimpleScalar tool set, it estimates runtime power consumption of the simulated processor.

CACTI and Wattch are popular because of the inherent flexibility of their analytical power models. However, we see many extensions and refinements that should be made to those tools. For example, CACTI uses 0.8um process parameters and assume that both delay and energy consumption of a circuit scale linearly with technology feature size. This assumption is likely to produce large errors as technology advances, so the parameters need to be corrected according to the latest technology. Unfortunately, we found that implementing the necessary extensions on those tools is not straightforward due to the lack of consideration in their software design. Our motivation for developing PRESTO is to enhance the extendibility of these tools by carefully redesigning the software framework, and incorporate new power models to improve their accuracy or widen their use.

3. Software Design

3.1 System Overview

Figure 1 illustrates our system for architecture level power estimation, where PRESTO is combined with an instruction level performance simulator for general purpose superscalar processors. First, PRESTO builds a power model for each function block in the target processor. It defines states of the block and power consumed when the block is in each of the states. Then the performance simulator simulates execution of a given binary and sends messages to PRESTO that contain access information on each function block.

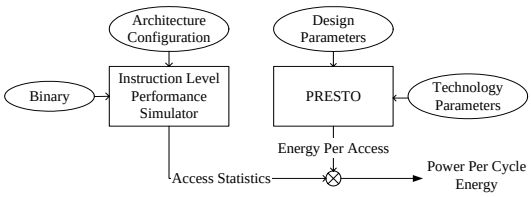

Fig. 1 Architecture level power estimation using PRESTO.

On receiving the messages, PRESTO collects access statistics and calculates power on a per-cycle basis.

PRESTO takes design parameters and technology parameters as input. Design parameters include transistor sizes and wire lengths used in modeled function blocks. To give an example, there are about 50 parameters for a cache. Technology parameters include SPICE model parameters, interconnect parameters such as capacitance per unit length, and several design rules such as minimum wire spacing. Apart from those circuit level and device level parameters, architecture level configuration parameters must be specified, such as cache size and number of ports. All the parameters are listed in an input text file so that they can be changed without recompilation.

3.2 Model Layering

PRESTO comprises three modeling layers: *technology layer*, *circuit layer*, and *dynamic behavior layer*. Figure 2 illustrates the relationship between the three modeling layers using UML notation. The technology layer is implemented as a composition of MOS FET, interconnect, and design rules classes. It calculates parasitics of the devices using technology parameters given by the user. The circuit layer manages design parameters specified by the user and describes circuit structure. It then defines states of the circuit and power consumption for each of the states. All circuits are implemented as subclasses of an abstract class named *Circuit*. The circuit classes are technology independent : only when calculating absolute amount of power consumption, they make a request for device parameters to the technology class. Dynamic behavior layer records runtime states of the circuit such as activity factors of circuit nodes (i.e. the probability that an input signal to the gate causes a transition that consumes energy). Each dynamic behavior class is linked to a circuit class. During binary simulations, it determine the current state of the circuit according

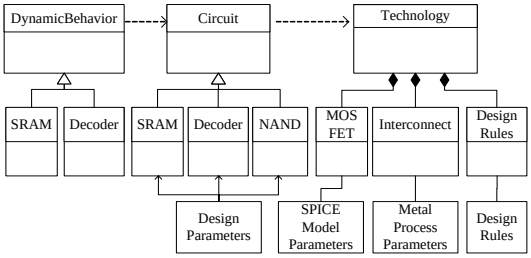

Fig. 2 Relationship between modeling layers.

to the messages from the performance simulator, and derives power consumption from the circuit class.

The layering of the models enables modification or extension of models in one layer without affecting models in the other layers. For example, all technology-dependent models are enclosed in the technology layer, so that the circuit models can be used over technology generations without any modification. It is also easy to replace or add device models in the technology layer to exploit emerging technology characteristics. Similarly, the dynamic behavior layer decouples runtime characteristics of a circuit from its static hardware structure. Changes in the dynamic behavior layer, including interface between the performance simulator, have little effect on the other two layers.

3.3 Hierarchical Circuit Composition

Circuit layer follows a hierarchical design strategy. A circuit class can be built using one or more other circuit classes that represent smaller subcircuits. Figure 3 shows the design of cache class in the circuit layer. The cache is composed of data arrays and tag arrays which are instances of SRAM subbank class. The SRAM subbank class

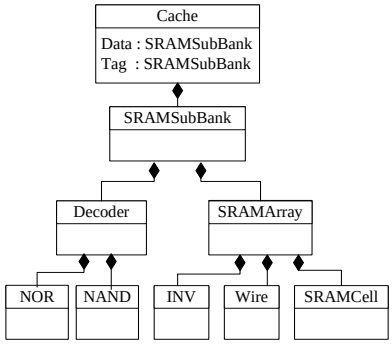

Fig. 3 Hierarchical implementation of cache class in the circuit layer.

comprises instance of SRAM macro class and decoder class, which are in turn built from instances of logic gate class, wire class, etc.

Such implementation has two advantages. First, some circuit classes can be repeatedly used in one or more classes. In this example, SRAM subbank class is used for both data array and tag array. Decoder class can also be shared by other classes that comprises array structures, such as register file. Second, any change in a subcircuit class is automatically reflected to the classes that use it. If the decoder is replaced with another of different design, the change is instantly reflected to cache data array, tag array and register file, Those composite classes do not have to be aware of the modification themselves. Thus, the hierarchical design improves software reusability, and reduces task of developing models for new circuit designs.

4. Power Models

4.1 Features

The fundamental power models currently implemented in PRESTO are similar to those used by CACTI and Wattch. PRESTO estimates dynamic power consumption based on the well known equation $P = \alpha f C V_{dd}^2$. Here α is activity factor managed by the dynamic behavior layer, and C is the equivalent capacitive load estimated by the circuit layer and technology layer.

Unlike previous tools, however, PRESTO derives C using SPICE model parameters. Most existing tools rely on empirical device parameters tailored for them, so in order to simulate different process technology, user must extract the parameters using lower level tools or predict them from the parameters of the past generations. To enable more flexible adaptation to different process technologies, we have implemented in the technology layer a class that calculates MOS capacitances directly from SPICE model cards [3]. The class uses MOS model equations of SPICE in reduced forms : all terms that depend on potential level of the MOS terminals are replaced by fixed constants. For example, drain capacitance is proportional to the reverse bias V_r at diffusion edge, so we take an average assuming $0 \leq V_r \leq V_{dd}$.

Using SPICE parameters, PRESTO can exploit process characteristics in a straightforward way : different process can be modeled by merely preparing appropriate SPICE parameter

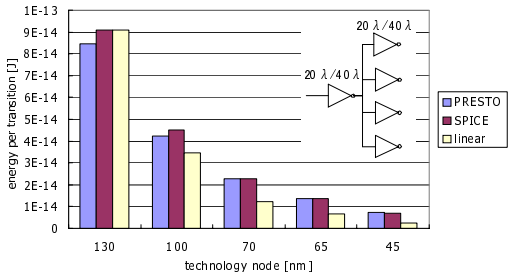

Fig. 4 Energy consumption of a FO4 inverter.

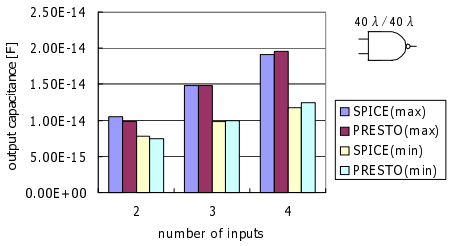

Fig. 5 Output capacitance of a NAND gate at 100nm process.

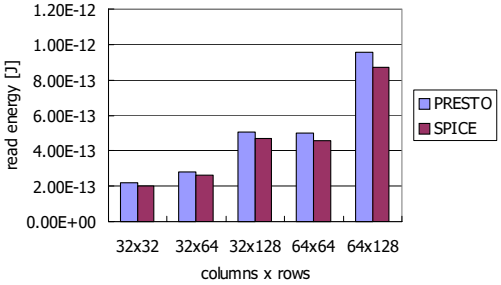

Fig. 6 SRAM read energy at 100nm process.

set which is widely recognized as a standard in LSI design. If needed, however, user can also selectively use empirical parameters instead of the SPICE model parameters. Since the technology layer is decoupled from other layers, the device models can be exchanged without disturbing other layers. It should also be noted that the calculation of the device parameters is usually done only once at simulator initialization phase, so its impact on the simulation speed is negligible.

4.2 Validation

We validated our power models based on SPICE parameters against HSPICE simulator itself. In all experiments, we used SPICE model cards and metal parameters from Berkeley Predictive Technology Model (BPTM) [1].

We first measured energy consumed by a FO4 inverter at a signal transition. Figure 4 shows the results. For 130nm to 45nm technology, our model was within 7% of error against SPICE. As a reference, the figure also shows a simple estimation similar to CACTI, which assumes that, after 130nm technology generation, the energy consumption scales linearly with the feature size. It can be seen that the linear scaling does not match well to the predicted technology roadmap.

For gates that include series-stacked transistors, the problem is more difficult because their equivalent input and output capacitances depend on the potential levels of the nodes in the stack. Figure 5 shows output capacitances of NAND gates with different input patterns. The output capacitance of a NAND is maximum when all the inputs are logical '1', because drain capacitances of all the stacked NMOS contribute to it. Conversely, the capacitance is minimum when the top NMOS in the stack is off. SPICE indicates that the maximum capacitance is larger than the minimum by as much as 60% for the 4-input NAND gate. To account for the difference, PRESTO calculates both maximum and minimum capacitance. An user can use either of them according to his/her needs. The estimations for maximum and minimum capacitances were both within 6% of accuracy.

Finally, we simulated a SRAM read operation on a circuit that consists of memory cells, wordlines, bitlines and precharge transistors. Figure 6 shows the results for 100nm process. Our model which ignores transistor's non-linear characteristics overestimated capacitances of the pass transistors in the memory cells, but the error was within 10%.

5. Case Study : Dynamic Zero Compression for Caches

We used PRESTO to evaluate a recently proposed power reduction technique called *dynamic zero compression* (DZC) for caches [12]. The idea of DZC is to compress zero byte data to reduce SRAM access energy. For every zero-valued byte, only a single bit flag is read or written instead of full eight bits. Thus, energy consumed by bitline swings can be reduced to approximately one eighth of the baseline cache. In the original paper,

Villa et al. evaluated the technique on a cache implemented physically on 0.25um process. However, such experiment requires considerably large effort. PRESTO is useful to estimate potential energy savings and explore a wider design space before proceeding to lower level design phases.

5.1 Validation

We implemented circuit layer models for a baseline cache and a DZC cache following the description of [12]. Because some details on the cache design are not given in the paper, we referred to [4] for memory cell size and sense amplifier energy constant, to SCMOS design rule [2] for wire spacings, and to [13] for the design of decoder and IO circuit. DZC requires circuit level modifications, such as adding zero-byte detection logic and wordline gating circuit. In order to take into account the overhead incurred by those additional circuits, we created their model by composing logic gate classes, and incorporated them to the baseline SRAM macro class. At dynamic behavior layer, we modified cache class so that it recognizes zero-valued bytes in the accessed data and determines cache states accordingly.

In order to validate the model, we provided PRESTO with device parameters of TSMC 0.25um process in which the cache was originally implemented and compared the estimated energy constants with those reported in [12]. Figure 7 shows energy breakdown of the baseline cache for reading a 32-bit word. It can be seen that our model shows similar characteristics as the original paper. Although we do not show the data in detail, we have also confirmed that estimation for the maximum energy saving and overhead incurred by the DZC technique are close to the data presented in the paper, 40% saving and 19% overhead per read and 67% saving and 17% overhead in maximum per write relative to the baseline cache.

5.2 Experimental Results

Using our cache model, we estimated the effect of the DZC technique under different conditions. The technology assumption is changed from 0.25um to BPTM 0.10um process. Data width is extended from 32 bits to 64 bits, and cache size is also increased accordingly.

We simulated eight applications from SPECint95 using our cycle-level simulator. The input parameters for the applications are adjusted so that the executions finish between 100-300 million instructions. Processor configuration used in the simulation is given in Table 1.

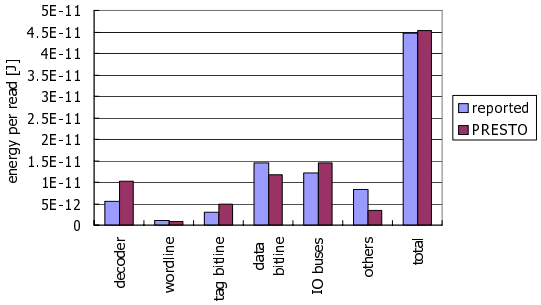

Fig. 7 Breakdown of energy consumption for 32-bit read accesses in base line cache.

Table 1 Processor configuration assumed in the cycle-level performance simulator.

Parameter	Configuration
Pipeline stages	10
Arith/Logic units	4
Address units	2
Reorder buffer size	64
Load/Store queue size	20
Fetch/Decode/Rename/Retire width	4
L1 Instruction cache	32 KB
L1 Data cache	32 KB
L2 Unified cache	ideal

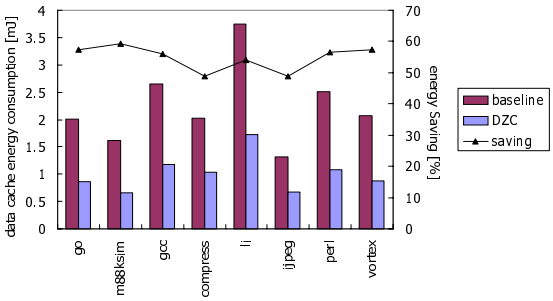

Fig. 8 Data cache energy consumption.

Figure 8 shows the effect of DZC applied on a L1 data cache. The cache is 64B line, 2-way set associative, 3-ported (2 read ports and 1 write port), and is divided into macros of 128 rows of 64 columns. The energy savings range from 49% to 59%, 55% on average. The savings are larger than the results presented in [12] because of increased percentage of zero bytes in 64-bit applications.

6. Conclusions and Future Work

This paper presented PRESTO, a framework for architecture level power estimation based on analytical power models. PRESTO emphasizes not only on the accuracy of power models but also on its flexibility as a software tool. It separates power model of a hardware structure into three different layers: technology layer, circuit layer and dynamic behavior layer. This is useful for modifying or extending a previously implemented model in one layer without affecting models in the other layers. Furthermore, circuit models are built using a hierarchical approach to increase software reusability.

In order to enable flexible adaptation to different process technologies, we implemented in the technology layer a model that calculates device parameters directly from SPICE model parameters. Our power models on simple logic gates and SRAM structure were approximately 10% of accuracy against SPICE simulation. As a case study, we used PRESTO for evaluating dynamic zero compression for caches. Taking advantage of hierarchical circuit composition policy, we effectively incorporated circuit-level design modification required for the technique into the baseline cache model. We then evaluated the technique under different conditions than the original paper, and showed its effectiveness on wider data width and latest process technology.

Currently, PRESTO estimates only dynamic power consumption. We are now working on the model of static power as well which is an increasing portion of overall processor power consumption. We also intend to carry out further validation of power models at layout level and at larger macro level. Finally, we plan to develop models for various hardware structures and explore architecture level low power design space with PRESTO.

Acknowledgement

This research is partially supported by Grant-in-Aid for Fundamental Scientific Research B(2) #13480077 from Ministry of Education, Culture, Sports, Science and Technology Japan, Semiconductor Technology Academic Research Center (STARC) Japan, CREST project of Japan Science and Technology Corporation, and by 21st century COE project of Japan Society for the Promotion of Science.

References

[1] Berkeley Predictive Technology Model. http://www-device.eecs.berkeley.edu/~ptm/introduction.html.

[2] MOSIS Scalable CMOS (SCMOS) Design Rules. http://www.mosis.org.

[3] Cadence SPICE. Reference Manual Product Version 4.4.6, 2002.

[4] B. S. Amrutur and M. A. Horowitz. Speed and Power Scaling of SRAMs. *IEEE Transactions on Solid-State Circuits*, 35(2):175–185, 2000.

[5] D. Brooks, P. Bose, S. Schuster, H. Jacobson, P. Kudva, A. Buyuktosunoglu, J.-D. Wellman, V. Zyuban, M. Gupta, and P. Cook. Power-Aware Microarchitecture: Design and Modeling Challenges for Next-Generation Microprocessors. *IEEE Micro*, 20(6):26–44, 2000.

[6] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: A Framework for Architectural-Level Power Analysis and Optimization. In *Proc. of the 27th International Symposium on Computer Architecture*, pages 83–94, 2000.

[7] A. Dhodapkar, C. H. Lim, G. Cai, and W. R. Daasch. TEM^{2}P^{2}EST : A Thermal Enabled Multi-Model Power/Performance ESTimator. In *Proc. of the 1st International Workshop on Power Aware Computer Systems*, 2000.

[8] S. H. Gunther, F. Binns, D. M. Carmean, and J. C. Hall. Managing the Impact of Increasing Microprocessor Power Consumption. Intel Technology Journal, 2001.

[9] D. Ponomarev, G. Kucuk, and K. Ghose. AccuPower: An Accurate Power Estimation Tool for Superscalar Microprocessors. In *Proc. of 2002 Design, Automation and Test in Europe Conference and Exhibition*, pages 124–129, 2002.

[10] G.Reinman and N.P. Jouppi. CACTI 2.0: An Integrated Cache Timing and Power Model. Technical Report WRL Research Report 2000/7, DEC Western Research Laboratory, 2000.

[11] P. Shivakumar and N. P. Jouppi. CACTI 3.0: An Integrated Cache Timing, Power, and Area Model. Technical Report WRL-2001-2 HP Labs Technical Reports, 2001.

[12] L. Villa, M. Zhang, and K. Asanovic. Dynamic Zero Compression for Cache Energy Reduction. In *Proc. of the 33rd International Symposium on Microarchitecture*, pages 214–220, 2000.

[13] S. J. Wilton and N. P. Jouppi. An Enhanced Access and Cycle Time Model for On-chip Caches. Technical Report WRL Research Report 93/5, DEC Western Research Laboratory, 1993.

[14] W. Ye, N. Vijaykrishnan, M. T. Kandemir, and M. J. Irwin. The Design and Use of SimplePower: A Cycle-Accurate Energy Estimation Tool. In *Proc. of the 37th Design Automation Conference*, pages 340–345, 2000.