HyPE: Hybrid Power Estimation for IP-Based Programmable Systems

Xun Liu Marios C. Papaefthymiou
Advanced Computer Architecture Laboratory
Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, Michigan 48109

Abstract— This paper presents a novel power estimation scheme for programmable systems consisting of predesigned datapath and memory components. The proposed hybrid methodology yields highly accurate estimates within short runtimes by combining high-level simulation with analytical macromodeling of circuit characteristics. To assess its effectiveness in practice, we implemented our hybrid scheme into a power estimation tool, called HYPE, and applied it to explore various architectural alternatives in the design of a 256-state Viterbi decoder and a Rijndael encryptor. For designs with about 1 million transistors, our estimator terminates within seconds. Compared with industrial gate-level power estimators, our approach is up to 1,000 times faster with 5.4% deviation on average.

I. INTRODUCTION

Efficient and accurate power estimation tools are crucial for the design of low-power systems. In this paper, we present a novel hybrid power estimation methodology for IP-based systems. Our approach can be used to explore architectural alternatives of general programmable systems consisting of predesigned components. It can handle systems of arbitrary topology that execute any given program, encompassing the dissipation of memory, control, and datapath components.

To obtain fast and accurate power estimates, our scheme combines high-level simulation with analytical macromodeling that abstracts circuit-level characteristics such as switching and leakage power. Given an IP-based system and a program it executes, functional simulation is performed to derive all data signals at the datapath/memory interface as well as all control signals. For each of the datapath topologies identified from these control signals, an iterative procedure is applied in conjunction with analytical output macromodels to calculate signal statistics among the IP components of the system. These statistics are then applied to analytical power macromodels to estimate obtain the dissipation of the entire datapath.

We have implemented our hybrid scheme in a power estimation tool, called HYPE, and used it to explore several architectural alternatives in the design of 256-state Viterbi decoders and Rijndael encryptors. Our experimental results demonstrate the high effectiveness of our approach. For systems with 100k logic gates, HYPE terminates within seconds. Compared with state-of-the-art industrial gate-level power estimation tools, our methodology is 2 to 3 orders of magnitude faster with 5.4% power estimation deviation on the average.

The remainder of this paper has 4 sections. We discuss previous research on high level power estimation and analytical macromodeling in Section II. Our hybrid power estimation scheme along with a sufficient condition for the convergence of our iterative estimation scheme are presented in Section III. Our experiments are presented in Section IV. Section V summarizes our contributions.

II. PREVIOUS RESEARCH

A growing volume of research has been devoted to high-level power estimation, since early design decisions often have a large impact on system power consumption [8, 12]. Two major research directions are simulation-based *cycle-accurate* power estimation and *power macromodeling*. Cycle-accurate approaches yield fine grain information about power dissipation in each cycle. In contrast, power macromodeling relies on signal statistics to estimate average power consumption.

A plethora of cycle-accurate power estimation frameworks has been proposed [3, 7, 9, 14, 16]. In these approaches, the power consumption of each individual RTL block is characterized with power-relevant events such as instructions or input vector pairs. Given a system architecture and a sequence of input signals/instructions, architectural level simulation is applied to count these events, and total power is computed by summing up the dissipation of the events of all blocks. These research efforts are primarily targeted at specific microarchitectures and the resulting simulators and cycle-accurate power models tend to be domain-specific. In [15], a cycle-accurate power model was proposed for general circuits, but it was only applied to individual components.

Power macromodeling of predesigned components is a promising approach for accurate yet simulation-free power estimation. In this approach, macromodels are derived from circuit-level characterizations which capture physical details like parasitic capacitance. Each such macromodel contains a mapping between the power dissipation of a circuit and certain statistics of its input signals such as the average signal probability or average transition density [13]. Power macromodeling for single IP components has been investigated using various signal statistics and different mapping approaches [1, 2, 4, 5, 6]. The application of power macromodeling at system level was explored in [10, 17].

In analytical power macromodeling, a function g maps the space of input signal properties to the power dissipation of a circuit. Three widely used input parameters for the macromodel are the average input signal probability P_{in}, the average input transition density D_{in}, and the input spatial correlation S_{in} [2]. To obtain the power function g of a given IP com-

ponent, the component is first simulated under sample input streams with various P_{in}, D_{in}, and S_{in}. The set of power dissipation points $\mathcal{P}$ obtained by this procedure is then curve-fitted to derive an analytical expression g using a minimum mean-square error criterion so that

$$\mathcal{P} = g(P_{in}, D_{in}, S_{in}) \ . \tag{1}$$

In the estimation procedure, the actual signal statistics are derived and applied to g to compute the power estimate.

Similarly, in analytical output macromodeling, functions are generated to map input signal statistics to those of the output signals. In the characterization step, functional simulations of a circuit are performed with different input sequences to obtain data points for the metrics P_{out}, D_{out}, and S_{out}. Using a minimum mean-square error criterion, analytical functions $\vec{f} = (f_1, f_2, f_3)$ are derived so that

$$(P_{out}, D_{out}, S_{out}) = \vec{f}(P_{in}, D_{in}, S_{in}) \ . \tag{2}$$

Analytical output and power macromodels can be combined in a static (simulation-free) procedure as described in Fig. 1 [10]. In the sample system given in this figure, the vertices A, B, C, and D represent IP components. The signal statistics of the inter-component nodes n1, n2, and n3 are initialized to arbitrary values. These statistics and statistics of the primary input are then applied to the output macromodels of A, B, and C iteratively until convergence. The power consumption of the entire system is calculated by applying the signal statistics to the power macromodels of the corresponding components.

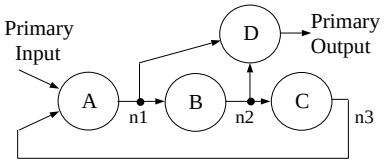

Fig. 1. A System Example

Static estimation has a number of limitations. First, it cannot handle memory components, because memory output cannot be estimated using input statistics. Second, it cannot handle control signals efficiently. Power macromodeling implicitly assumes that each input affects the power dissipation roughly in the same way. However, control signals such as clock gating signals, generally have much larger impact on power than data signals. Furthermore, control signals can alter the dataflow through components like multiplexers, reconfiguring system topology and, therefore, changing power dissipation. Not distinguishing control from data signals can potentially result in large estimation error.

III. Hybrid Power Estimation Procedure

The 3-phase flow diagram of our hybrid power estimation procedure is shown in Fig. 2. Given a functional description of a system S, an architectural implementation of S consisting of predesigned components, a sequence of input signals, and a program stored in memory, our procedure reports the total power dissipation of S when running the program.

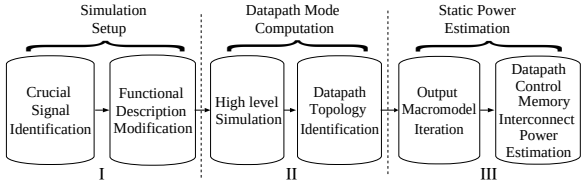

Fig. 2. Hybrid Power Estimation Procedure

A. Simulation setup

In this phase, our procedure first identifies the *crucial signals* of the system, that is, the signals that are critical for accurate system power estimation. There are two types of crucial signals: data signals at the datapath/memory interface and control signals. The data signals at the datapath/memory interface contain the memory access information and are important to estimate memory power. Furthermore, these signals are the primary input/output of the datapath. Since no output macromodels can be created for memories, these signals need to be calculated using functional simulation.

Control signals can be classified into two groups: system level and component level. *System level* controls can reconfigure datapath topology, resulting in different dataflow and therefore signal activities, such as the *Select* signal in Fig. 3(a).

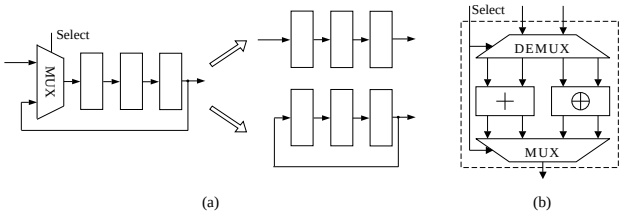

Fig. 3. Control Signals: (a) System-level (b) Component-level

Component level controls may significantly affect the power dissipation of individual circuit components. Fig. 3(b) shows a dual-function ALU in which the control *Select* decides the utilization and, therefore, power dissipation of the hardware. For such circuits, different power macromodel functions g_i are characterized for each possible control value i. Accordingly, the total power is calculated by:

$$\mathcal{P} = \sum_{i \in M} g_i(P_{in}, D_{in}, S_{in}) \cdot p_i \ , \tag{3}$$

where M is the set of all control values, and p_i is the fraction of time for which the control value is i.

It is straightforward to automate the control signal identification by requiring all IP blocks to have their control pins labeled. Moreover, controls introduced outside IP blocks should also be labeled by the designers.

After all crucial signals have been identified, our procedure modifies the functional system description to represent these signals explicitly. Fig. 4 illustrates how this step is performed. Since the architectural description is an implementation of the functional description, both descriptions have the same primary inputs (PI) and primary outputs (PO). Some of the intermediate signals are shared. The crucial signals that are missing

in the functional description are added by duplicating the corresponding architectural description of the system. The result is represented by the shaded part.

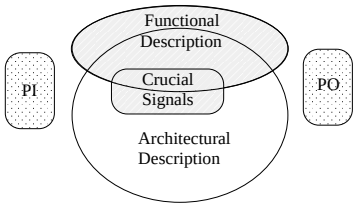

Fig. 4. Augmentation of Functional Description

B. Datapath modes computation

This step performs high-level simulation of the augmented functional description using the given input signals and/or programs. The objective is to accurately compute all control signals, memory access information, and data signals at the datapath/memory interface. After simulation, profiling of the derived control signals is performed on the architectural description. Based on the values of the system level control signals, the ensuing datapath topologies or modes are recorded. These modes can activate completely different hardware and thus give rise to significant variations in power consumption.

In addition to the datapath modes, the fraction of time when each mode is active is also computed. These fractions indicate the extent to which each mode can affect the overall system power. This step is performed by counting the frequency of every value of system-level control signals. Theoretically, there might be 2^n modes with n system controls. However, one would expect only a limited number of system modes to occur in practice, as confirmed by the designs we studied.

C. Static power estimation

In the third phase, static power estimation based on macromodeling is performed in each mode. The data signal statistics at the datapath/memory interface are first computed. Signal statistics among the circuit components are subsequently computed by iteratively evaluating output macromodel functions on each datapath topology. These signal statistics are then applied to analytical power macromodels to obtain the dissipation results. The average datapath power consumption is finally computed by combining power estimates of all modes based on the runtime percentages.

Theorem 1 gives a sufficient condition under which the iterative computation of the signal statistics in the static power estimation step of our method is guaranteed to converge to a unique fixed point. The proof is omitted due to page limitations. (See [10] for detailed symbol definitions.)

Theorem 1 *Let $\mathcal{S}$ be a system of arbitrary topology, consisting of m predesigned components. For each component n, let $\vec{F}_n$ be the output macromodel functions, and $\overline{\overline{F'_n}}$ be the output sensitivity matrix. For arbitrary initialization, the iterative estimation of signal statistics based on output macromodeling always converges to a unique solution if*

$$\left|\overline{\overline{F'_n}}\right| \cdot \vec{1}_3 \leq \delta \cdot \vec{1}_3 \ , \tag{4}$$

where δ is a real number such that $0 \leq \delta < 1$ and $\vec{1}_3$ denotes the 3-dimensional vector of 1's. $\left|\overline{\overline{F'_n}}\right|$ is obtained by taking the absolute value of each element in $\overline{\overline{F'_n}}$.

From a macromodeling standpoint, we treat global interconnect, e.g. clock nets or data buses, as a special type of component with total capacitance as an additional parameter. The capacitance value can be extracted after global system routing or estimated quickly using the size information of the predesigned components. Power dissipation can be computed using the capacitance and signal statistics of the interconnect.

The power dissipation of memories primarily depends on their access rate and does not change significantly with the data stored. It is computed using memory access information from the high level simulation and power-related parameters from IP vendors. The power dissipation of the control components is computed by applying the control signals obtained from high level simulation to the corresponding power macromodels.

IV. Experimental Results

A. 256-state Viterbi decoder

In this section, we describe the application of HYPE to investigate the impact of parallelism on power dissipation of a Viterbi decoder for the IS95 standard. Using commercial ASIC design tools and a 0.25 μm standard-cell library, we designed 6 Viterbi decoders, containing 2, 4, 8, 16, 32, and 64 ALUs. To evaluate our power estimation scheme, we applied HYPE to estimate the power dissipation of all 6 Viterbi decoders when decoding synthetic inputs. For comparison, we used *PrimePower*, a gate-level power estimation tool, to estimate power dissipation using the same inputs.

Fig. 5 shows the power estimation results of HYPE and *PrimePower*. The average estimation difference is 5.4% with a standard deviation of 0.04. This high accuracy should not be attributed to signal correlations between the characterization and estimation stages, because these two steps were decoupled in our experimental procedure. Specifically, the input signals of the circuit components during characterization are generated by a random number generator [11], while those of the evaluation procedure are computed during the decoding procedure by the various design components.

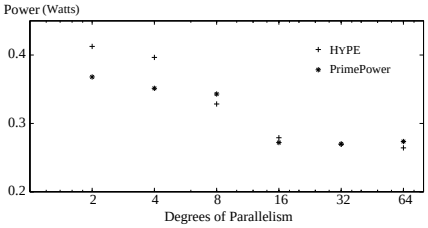

Fig. 5. Viterbi Decoder Power Estimation Results

HYPE finishes each estimation run within 13 seconds, executing up to 1,000 times faster than *PrimePower*. This significant speedup is hardly surprising, because *PrimePower* is a gate-level estimator and computes the signals of all individual gates. On the other hand, HYPE only computes the control

and data signals at the datapath/memory interface, while applying static power estimation for the datapath. Viterbi decoder designs as well as most DSP systems have relatively simple control blocks and substantial datapaths. Consequently, HYPE performs much less computation than *PrimePower* on such designs. Speedups are expected to decrease for systems with a lot of control signals and relatively small datapaths. In a worst-case scenario, the simulation performed by HYPE is close to RTL simulation, since all control signals need to be computed accurately. Macromodel characterization times, which are in the order of hours, are not included in the runtime of HYPE, because the macromodel functions are assumed to be generated by vendors once and are made available to the designers.

B. Rijndael encryptor

This section reports on the application of our power estimation scheme to analyze the power impact of loop unrolling in Rijndael encryptor designs. Fig. 6 shows the architecture of our designs. Fig. 6(a) gives the loop structure. A control unit switches the multiplexer so that new input data can only be processed until the previous loop operations are finished. Multiple loop structures can be replicated to increase the system throughput. In Fig. 6(b), the loop is *unrolled*, i.e. the hardware is repeated, and encryption steps are performed sequentially.

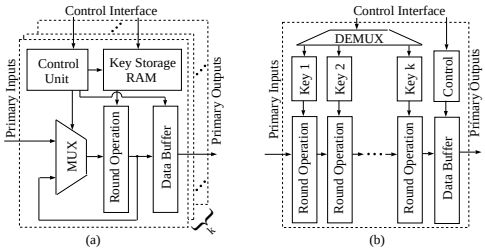

Fig. 6. Rijndael Encryptor Diagrams

To evaluate the performance of HYPE with different switching activities, for each estimation run, we generated 1,000 input vector sequences with switching activities from 0.05 to 0.85 with granularity 0.1. HYPE was applied to estimate the dissipation of the two Rijndael encryptors across the entire range. Fig. 7 shows the results of this experiment. The dotted and solid lines represent the estimates of unrolled and loop-based encryptors, respectively. HYPE gives highly accurate estimates as compared with *PrimePower*. Estimation deviation varies with the input switching activities. The average difference is 5%, with a standard deviation of 0.05. Estimation time was about 2–3 orders of magnitude faster than *PrimePower*.

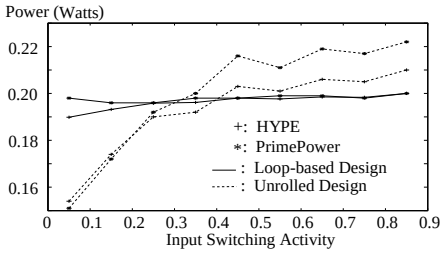

Fig. 7. Rijndael Encryptor Power Estimation Results

V. CONCLUSION

This paper presents a novel hybrid power estimation procedure for programmable systems. Our approach is applicable to general computing systems made of predesigned components including memory, control, and datapath circuits. A software implementation of our scheme has been used to explore architectural design alternatives for communication and encryption applications. Compared with state-of-the-art industrial gate-level power estimators, our method is 2 to 3 orders of magnitude faster with 5.4% deviation on average.

VI. ACKNOWLEDGMENTS

The authors wish to thank Luca Benini and Hai Zhou for their helpful suggestions and discussions. This research was supported in part by NSF under Grant No. CCR-0082876 and by ARO under Grant No. DAAD19-99-1-0304.

REFERENCES

[1] M. Barocci, L. Benini, A. Bogliolo, B. Ricco, and G. D. Micheli. "Lookup table power macro-models for behavioral library components." In *Proc. IEEE Alessandro Volta Workshop on Low Power Design*, Mar. 1999.

[2] G. Bernacchia and M. C. Papaefthymiou. "Analytical macromodeling for high-level power estimation." In *Proc. IEEE International Conference on Computer Aided Design*, Nov. 1999.

[3] D. Brooks, V. Tiwari, and M. Martonosi. "Wattch: A framework for architecture-level power analysis and optimizations." In *Proc. 27th International Symposium on Computer Architecture*, June 2000.

[4] Z. Chen, K. Roy, and T. L. Chou. "Power sensitivity–a new method to estimate power dissipation considering uncertain specifications of primary inputs." In *Proc. of IEEE International Conference on Computer Aided Design*, Nov. 1997.

[5] S. Gupta and F. N. Najm. "Power macromodeling for high level power estimation." In *Proc. 34th Design Automation Conference*, June 1997.

[6] S. Gupta and F. N. Najm. "Analytical model for high level power modeling of combinational and sequential circuits." In *Proc. IEEE Alessandro Volta Workshop on Low Power Design*, Mar. 1999.

[7] C. Hsieh, L. Chen, and M. Pedram. "Microprocessor power analysis by labeled simulation." In *Design, Automation, and Test in Europe*, pages 182–189, Mar. 2001.

[8] P. Landman. "High level power estimation." In *Proc. International Symposium on Low Power Electronics and Design*, Aug. 1996.

[9] M. T. Lee, V. Tiwari, S. Malik, and M. Fujita. "Power analysis and minimization techniques for embedded DSP software." *IEEE Trans. VLSI Systems*, pages 123–135, Mar. 1997.

[10] X. Liu and M. C. Papaefthymiou. "A static power estimation methodology for IP-based design." In *Design, Automation, and Test in Europe*, pages 280–287, Mar. 2001.

[11] X. Liu and M. C. Papaefthymiou. "A Markov chain sequence generator for power macromodeling" In *Proc. IEEE International Conference on Computer Aided Design*, Nov. 2002.

[12] E. Macii, M. Pedram, and F. Somenzi. "High-Level power modeling, estimation, and optimization." *IEEE Trans. CAD*, Nov. 1998.

[13] F. N. Najm. "Transition density: A stochastic measure of activity in digital circuits." In *Proc. 28th Design Automation Conference*, June 1991.

[14] T. Šimunić, L. Benini, and G. D. Micheli. "Cycle-accurate simulation of energy consumption in embedded systems." In *Proc. 36th Design Automation Conference*, pages 867–872, June 1999.

[15] Q. Wu, Q. Qiu, M. Pedram, and C. Ding. "Cycle-accurate macro-models for RT-level power analysis." *IEEE Trans. VLSI Systems*, Dec. 1998.

[16] W. Ye, N. Vijaykrishnan, M. Kandemir, and M. J. Irwin. "The design and use of simplepower: A cycle-accurate energy estimation tool." In *Proc. 37th Design Automation Conference*, June 2000.

[17] R. Zafalon, M. Rossello, E. Macii, and M. Poncino. "Power macromodeling for a high quality RT-level power estimation." In *Proc. International Symposium on Quality Electronic Design*, pages 59–63, 2000.