The Petrol Approach to High-Level Power Estimation

Rafael Peset Llopis and Kees Goossens

Philips Research Laboratories

Prof. Holstlaan 4, WAY 4-1

5656 AA Eindhoven, The Netherlands

e-mail: {peset,goossenk}@natlab.research.philips.com

Abstract

High-level power estimation is essential for designing complex low-power ICs. However, the lack of flexibility, or restriction to synthesizable code of previously presented high-level power estimation approaches limits their use. In this paper we present a novel, more general and flexible high-level power estimation approach, that avoids these limitations. Petrol, as we call it, is not limited to specialized application domains, synthesizable VHDL, or data path parts of a design. We show that glitches can be usefully modeled at higher levels of abstraction. The Petrol approach shows good correlation with gate-level power estimates. It is currently used for commercial designs.

1 Introduction

The huge integration capability of modern technologies (several millions of transistors) allows very complex systems on a single IC, such as MPEG2 encoders and digital audio channel decoders. For these systems power dissipation is a critical issue, since it is often the bottleneck for further integration. Limited battery capacity imposes an upper bound on the power dissipation of ICs for mobile applications; the dissipation of "non-mobile" ICs is often limited by the cost of specialized "high-power" packages. Furthermore, high power dissipation can cause reliability problems in ICs. Therefore, it is necessary to take power dissipation into account during the design process, besides the "classical" performance parameters (delay and area). This requires good power estimation tools in order to guide the designer to power optimal implementations. Several types of high-level power estimation approaches have been published in literature. They are described in the following three paragraphs:

The first type is based on macro modeling, for both the RT [2] and behavioral [4] levels. Macro modeling usually addresses data-dominated circuits, and requires a library of precharacterized and parameterized building blocks, such as arithmetical logic units (ALUs), multipliers, memories, and controllers. Put simply, this library contains a power model for each of the building blocks, derived from many layout exercises. These models relate the power dissipation to the switching activities of the block inputs and to some block specific parameters, such as signal widths, area, and number of states. The switching activities of the block inputs and outputs are obtained from RT simulations [2], or from the data flow graph [4]. Although macro modeling seems very promising, it is limited its applications. It targets specific applications based on fixed building blocks and cannot cope with "flexible" architectural synthesis tools [5] or handwritten VHDL code. Furthermore, each change in technology, synthesis or layout tool, or in the structure of the building blocks requires a considerable recharacterization of the power models.

A different approach in high-level power estimation is currently being developed [3, 6]. This approach estimates the circuit area and an average value for the switching activity of all nodes, based on the entropy of the inputs and outputs of the circuit. This approach targets the behavioral level. The objective is to obtain power estimates of, for example, operators without any information about the implementation, using exclusively the behavior. It is therefore not possible to determine the impact of RT transformations (replace a fast adder by a slower one, add pipelines, add clock gating, and so on) on the power dissipation. Still in its initial stages, this approach requires additional fine-tuning before its application becomes a reality.

The third approach is based on a (fast) synthesis of the behavioral or RT description into a netlist description and applying netlist-level power estimation techniques ([1, 8]). This method cannot be extended beyond the RT level because the circuit descriptions must be synthesizable. Moreover, design iterations are slowed down by the synthesis step.

The main motivation for our work is to develop the techniques to estimate the power dissipation at the higher levels of design abstraction. These levels are in some cases behavioral VHDL, but mostly C. (Although these techniques can be applied to both languages in this paper we will concentrate on VHDL.) Since no tools exist to synthesize behavioral VHDL or C descriptions directly into gates, it is not possible to use power estimation techniques based on fast synthesis. The only commonly used approach to high-level power estimation is therefore macro modeling. However, macro modeling has several disadvantages. First, due to the elaborate task of developing a macro model (synthesis, layout, simulation and multi-dimensional parameter fitting) it is necessary to limit the number of basic blocks. Second, the complex interaction of a macro block with its environment (such as correlation and arrival time of inputs) must be abstracted to a small number of parameters. The former limits the applicability and flexibility; the latter reduces its accuracy.

2 Petrol Approach

The Petrol approach addresses these issues by decreasing the size of the power dissipating primitives. Instead of using a coarse-grained macro model, we use an expansion into single bit operators (called power primitives). The activity of these power primitives is monitored during simulation. The total number of power primitives is limited, and power models of these primitives can easily be found in a standard cell library. Consider an n-bit comparator. Figure 1 shows the expansion of this comparator into two-input XNOR and AND Boolean functions. Each time a comparator is executed using simulation, the expansion function is used, and the number of transitions at the inputs and output of the Boolean functions is recorded. This, together with the power models of (the smallest) AND2 and XNOR2 standard cells, is used to compute the power estimates.

```
function "=" (l,r: std_logic_vector) return boolean is
  variable res: boolean := true;
begin
  for i in l'range loop
    res := res and (l(i) xnor r(i));
  end loop;
  return res;
end;
```

Figure 1: Expansion function of n-bits comparator.

This has the following advantages:

- All VHDL that can be written is ultimately defined in terms of power primitives and expansion functions. In other words, using smaller grained macro blocks, rather than a limitation, is a liberation from a fixed set of operators.
- Only power models of the power primitives are required, and these models can be found in a standard cell library. Therefore no additional library of power models needs to be maintained.
- Alternative expansion functions correspond to different implementation of macro blocks. For example, adders can be implemented by carry-look-ahead or carry-ripple structures. Both are defined only as expansion functions in simple power primitives; no macro model is required.
- The environment of a macro block need not be abstracted, resulting in a significant improvement in accuracy. Simulation of the expanded function automatically takes care of correlation and input arrival times at the level of power primitives instead of the whole macro block.

The main disadvantage of the Petrol approach is increased simulation times compared to macro modeling. However, in future we plan to include in situ macro model generation [1]. After simulating the expanded operators many times, a macro model is built and used to speed up simulation.

A consequence of the Petrol approach is that a designer is able to see the impact of design decision on the power dissipation. For example adding clock gating to a design does not require any recharacterization of power models. Only the VHDL code has to be modified in order to incorporate clock gating.

3 Implementation

Petrol is built on top of a commercial VHDL simulator. It consists of two parts: a parser that adds profiling functionality to the VHDL descriptions, and a VHDL library containing the expansion functions. The design is simulated, while monitoring the transitions at the inputs and outputs of the power primitives. The amount of switched capacitance of the power primitives is used to compute a hierarchical power breakdown of the circuit.

Petrol uses the following power primitives: 2 input Boolean (N)AND, (N)OR, and X(N)OR, the inverter, three input full adder and subtractor, the 2-1 multiplexer, the flip-flop, and ROM/RAM memory.

Hazards are modeled by introducing artificial delays, which are taken into account by Petrol. Each power primitive is assumed to have a unit delay, and these delays are propagated during simulation. Filtering is also used during this propagation. The main advantage of modeling hazards is that the designer is able to see the impact of pipelining on his design.

Due to the large impact of interconnect on the performance of deep submicron ICs it is very important to model the capacitance of the interconnect. This is done by means of a wire load model similar to that used in logic synthesis. The capacitance of a wire is approximated by a constant plus the weighed fanout.

The accuracy of Petrol is limited by the following factors.

- Boolean optimization due to synthesis constraints. These have a direct impact on the performance of the synthesized design, and can cause variations upto 50% in power dissipation. This is clearly the largest error source of Petrol, and imposes a limit to the obtainable accuracy of power estimates at the RT level.
- Redundancy removal by logic synthesis. This is partially taken into account by Petrol by detecting constants.
- Approximation of real delays by unit delays clearly introduces an error. However, this error is smaller than those previously mentioned (Figures 2 and 3).
- Modeling the interconnect by wire load models introduces an additional error source. However, since the same wire load model is used as synthesis, the results are consistent with those of logic synthesis.

For more technical details about the theory and implementation of Petrol we refer to [7]. The following section shows the results obtained by Petrol.

4 Results

In order to get an impression of the overall accuracy, we have analyzed a large number of circuits, such as adders (Add), subtractors (Sub), multipliers (Mult), pipelined multipliers (Mult pipe) and IIR filters of order 2 (IIR), for various wordwidths (2 to 32 bits), for both carry look ahead and carry ripple adder implementations. The size of the largest circuit was (after synthesis) 10500 standard cells. The power estimates have been computed for both the cases without and with hazards (Figure 2 and 3). All input vectors were randomly chosen.For each circuit three energy dissipation values have been computed: the first two are based on gate-level simulations after synthesizing for cost and timing and the third on RT simulations. The average of the two gate-level power estimates is plotted on the horizontal axis and the Petrol power estimate on the vertical axis. The least squares method has been used to determine the best linear fit to these points. The slopes of these approximations are equal to 0.98 and 0.99 for the cases without and with hazards, respectively. These linear approximations are shown as solid lines in the figures. The dashed lines represent the case of a deviation of a factor of two from these approximations.The average deviation of the RTL from the gate-level power estimates is 15% and 20% for the cases without and with hazards, respectively. These results show that the unit delay approximation of all power primitives introduces only an error of 5%.

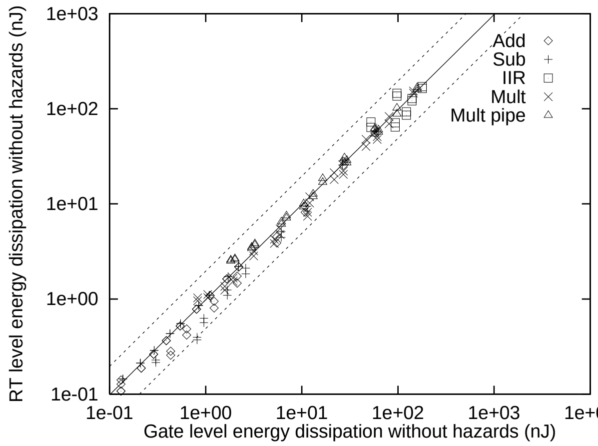

Figure 2: Overall accuracy scatter diagram without hazards.

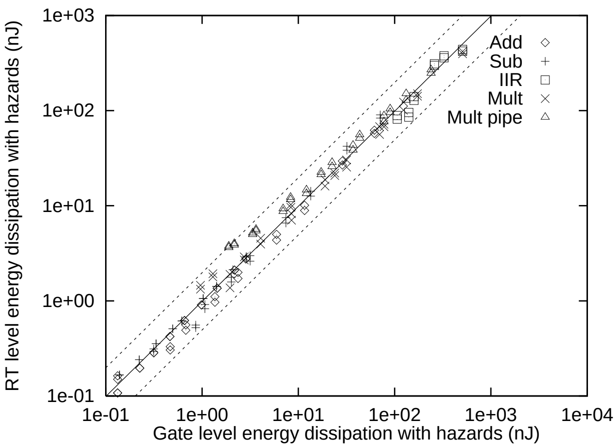

Figure 3: Overall accuracy scatter diagram with hazards.

Here we demonstrate that Petrol is useful for design space exploration in order to minimize power dissipation. The circuit under consideration is a 24 taps finite impulse response (FIR) filter, containing 24 multiplications and 24 additions. Several variants have been made of this architecture, by reducing the number of resources (multipliers and adders) and hence introducing time multiplexing. Variants with 1, 2, 3, 4, 6, 8, 12, and 24 multipliers and adders have been generated by Phideo [5], an in-house architectural synthesis tool. The size of the largest variant (24 resources) was 1.7 mm2 in a 0.5u technology. Figure 4 shows the energy dissipation for 200 random input samples for the different variants of this filter.

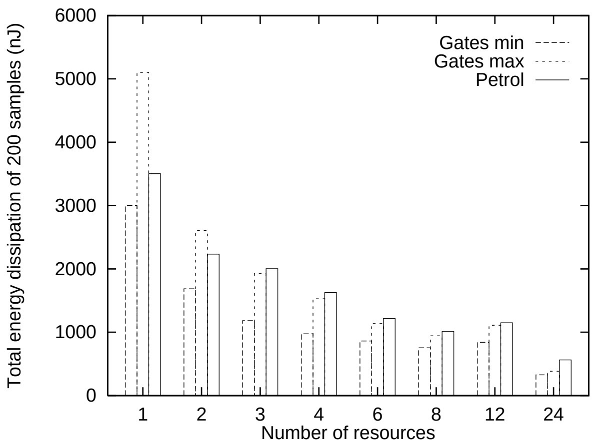

Figure 4: Results of design space exploration for FIR filter.

For each variant we show the gate-level power estimate after synthesizing for cost (Gates min), the gate-level power estimate after synthesizing for timing (Gates max), and the Petrol estimate. First, the absolute accuracy: we see that the average error of Petrol is 19%. Second, the relative accuracy is perhaps more important: given the Petrol figures only, would the implementation with least power have been selected? Here the answer is yes: Petrol correctly predicts the trend that the variant with 24 resources dissipates least, as well as showing the counter-intuitive fact that 12 resources consume more power than 8 resources.

The CPU times of Petrol for these filters was approximately equal to those of the gate level power estimation tool. The synthesis CPU times were one order of magnitude larger. This shows that Petrol saves a considerable amount of computation time, since a synthesis step is not needed in order to get power estimates. As stated before, Petrol does not require synthesizable code in order to provide the power estimates. Therefore it is possible to use the input description of the architectural synthesis tool for Petrol.

5 Conclusions

Petrol, a new approach in RT power estimation has been presented. An RT VHDL description is simulated and dissipating hardware structures are detected. Together with the signal activity, deduced from the simulation, this gives a power figure.

Petrol avoids synthesis and macro modeling by using a limited number of small grain power primitives in combination with expansion functions. This has the following advantages. No precharacterization is needed because the power information related to these power primitives can simply be found in industrial standard cell libraries. Petrol is independent of the application domain and avoids the inflexibility of macro modeling. Moreover, no synthesizable VHDL descriptions are required, since no synthesis is performed. Petrol can easily be extended to higher and lower levels of abstraction.

The main advantage of Petrol is that it uses all structural information available at the RT level in order to estimate the power dissipation. It is therefore possible to determine the impact of e.g. clock gating or pipelining on the power dissipation. Furthermore, the impact of different implementation structures of operators (e.g. carry-ripple versus carry-look-ahead for adders) can be taken into account by using several expansion functions.

Our results show that there is a good correlation between the Petrol and gate level power estimates; the average error is approximately 20%. These results also show that the simple delay approximation used for power primitives is accurate enough to model hazards.

The CPU times of Petrol are comparable to those of gate-level power estimation.

6 Acknowledgments

The authors wish to recognize useful discussions with Sabih Gerez, Ed Huijbregts, and Manoj Sachdev.

7 References

[1] A. Bogliolo et. al., “Adaptive Least Mean Square Behavioral Power Modeling”, Proc ED&TC 97, pp. 404-410.

[2] P. Landman et al., “An Integrated CAD Environment for Low Power Design”, IEEE Design and Test, pp. 72-82, Summer 1996.

[3] D. Marculescu et. al., “Information Theoretic Measures for Power Analysis”, IEEE Trans. on CAD, Jun. 96, pp. 559-610.

[4] R. Mehra and J. Rabaey, “Behavioral Level Power Estimation and Exploration”, Proc. Int. Workshop on Low Power Des., Apr.94, pp. 197-202.

[5] J.L. v. Meerbergen et al., “PHIDEO: High-Level Synthesis for High Throughput Applications”, J. of VLSI Signal Processing, 9, 95, pp. 89-104.

[6] M. Nemani and F.N. Najm, “Towards a High-Level Power Estimation Capability”, IEEE Trans. on CAD, Jun. 96, pp. 588-598.

[7] R. Peset Llopis, “A New Approach in High-Level Power Estimation”, Proc. DATE 98 Des. Track, pp. 31-35.

[8] Power Compiler of Synopsys.