Power-Aware Compilation for Embedded Processors with Dynamic Voltage Scaling and Adaptive Body Biasing Capabilities

Po-Kuan Huang
University of California, Davis
pohuang@ece.ucdavis.edu

Soheil Ghiasi
University of California, Davis
soheil@ece.ucdavis.edu

Abstract

Traditionally, active power has been the primary source of power dissipation in CMOS designs. Although, leakage power is becoming increasingly more important as technology feature sizes continue to shrink, traditioinal power optimization techniques often neglect its contribution to total system power. In this paper, we present a power-aware compilation methodology that targets an embedded processor with both dynamic voltage scaling (DVS) and adaptive body biasing (ABB) capabilities. Our technique has the unique advantage of optimizing design power by jointly optimizing dynamic and leakage power dissipation. Considering the delay and energy penalty of swithching between processor modes, our compiler generates code with minimum power consumption under deadline constraints. Compared to not performing any optimization, or using DVS alone, our technique improves the power consumption of a number of embedded application kernels by 26%*, and* 14%*, respectively.*

1. Introduction

Power consumption has become one of the most critical embedded system design considerations due to its significant impact on system density, battery life, reliable operation, packaging and cooling costs. Traditionally, active (dynamic) power has been the primary source of power consumption of CMOS designs. However, as we move down the technology nodes leakage power is increases exponentially, and becomes comparable to (or even dominates) the active power. Hence, joint optimization of active and leakage power is essential for efficient power optimization methodologies.

Quadratic dependence of active power to supply voltage, and exponential effect of transistor threshold voltage on its leakage current has motivated the idea of dynamically controling the two parameters to minimize the overall power consumption under timing constraints. These two techniques have been tried out on several designs, and significant power improvements have been reported [10, 13, 11]. However, the potentials of such architectural enhancements would not be fully exploited without an effective mode-switching strategy.

In this paper, we present a power-aware compilation methodology that targets an embedded processor with joint dynamic supply voltage and adaptive body bias (threshold voltage adjustment mechanism) capabilities. Our compiler assigns different basic blocks of the code to different supply voltage and body bias modes such that the total power consumption is minimized and the application deadline is met. We show that simultaneous consideration of active and leakage power results in improved power savings, compared to traditional dynamic voltage scaling techniques.

2. Power-Aware Compilation

The compiler optimization goal is to minimize the application energy consumption by assigning different basic blocks to different operation modes subject to meeting the deadline of the application, while considering the penalty of switching modes between basic blocks. A similar approach has been used by Xie et al. [4] for DVS-enabled processor, however, their technique neglects the leakage contribution to total power consumption. We formulate the problem to consider both dynamic voltage scaling and adaptive body biasing.

We analyze the structure of the application and profile the typical execution traces of the application to extract the required statistical information such as the latency of each basic block and the frequency of traversing edges. We utilize the extracted information to perform mode assignment to execution traces of the ap-

plication, under deadline constraint. We proceed to present this problem as an MILP formulation, which can be solved by utilization of commercially available solvers. Once mode assignment for control flow edges of the application is decided, appropriate mode switching instructions are inserted to the code.

Once application basic blocks are assigned to operating modes, total energy consumption of the application can be represented as energy consumption of all basic blocks plus the energy penalties incurred by mode switching instructions. The energy consumption of one basic block is determined by its latency under its assigned operating mode times the processor power consumption under that operating mode.

The latency of the application includes the latency of all basic blocks and the switching delay of the processor. The latency of one basic block is equal to the product of the number of execution cycles and the latency for one iteration under the operating mode.

The optimization object of MILP is to minimize the energy consumption for all basic blocks, and the operating mode switching energy. One of the constraint should be the summation of the latency of all basic blocks and the operating mode switching delay.

3. Experimental Results

We experimented our technique on compute-intensive kernels of eight applications selected from Mediabench [3] and Mibench [7] benchmarks. The selected applications are from multimedia, networking and automotive domains, which represent typical applications running on embedded processors. We extracted the control-data flow graph (CDFG) of the testbenches using SUIF compiler infrastructure [9] and Machine-Suif [8]. We used SimpleScalar simulator [12] to obtain accurate and realistic cycle count for each basic block, and frequency of traversing control edges over typical input data space. The simulated data, along with the CDFG structure, processor and its power model parameters were utilized to generate the MILP problem instance. We solved the MILP problems using the commercial CPLEX solver [1] and obtained a set of mode switching instructions for control flow edges of the applications. The MILP solution were used to estimate the total power consumption of the application running on the target processor. Total power consumption was calculated as sum total of leakage and active power for execution of application code, plus mode switch instructions, and their corresponding power penalty. For each operating mode, the power consumption and clock frequency of the processor were determined using the models and parameters adopted from [13, 11, 2, 5, 6]. The energy and delay penalty incurred by mode switching were also taken into account.

Our optimization obtained average of 13.69% and 25.81% energy improvement over DVS optimized and original code, while the improvement were as high as 18.26% and 27.73%, respectively.

References

[1] http://www.ilog.com/products/cplex.

[2] http://www.intel.com/design/intelxscale.

[3] C. Lee, M. Potkonjak, and W.H. Mangione-Smith. "MediaBench: A Tool for Evaluating and Synthesizing Multimedia and Communications Systems". In *International Symposium on Microarchitecture*, pages 34–41, 1997.

[4] F. Xie,M. Martonosi,S. Malik. "Intraprogram Dynamic Voltage Scaling:Bounding Opportunities with Analytic Modeling". *ACM Transactions on Architecture and Code Optimization*, 1(3):1–45, September 2004.

[5] Intel Corporation. "Enhanced Intel SpeedStep Technology for the Intel Pentium M Processor". March 2004.

[6] J.T. Kao, M. Miyazaki, A.P. Chandrakasan. "A 175-mv Multiply-Accumulate Unit Using an Adaptive Supply Voltage and Body Bias Architecture". *Journal of Solid-State Circuits*, 37(11):1545–1554, November 2002.

[7] M. Guthaus, J. Ringenberg, D. Ernst, T. Austin, and T. Mudge. "Mibench: a free, commercially representative embedded benchmark suite". In *Proceeding of the IEEE 4th Annual Workshop on Workload Characterization*, pages 3–14, December 2001.

[8] M.D. Smith, and G. Holloway. "An introduction to machine SUIF and its portable libraries for analysis and optimization". Technical report, Division of Engineering and Applied Sciences, Harvard University, 2002.

[9] M.W. Hall, J.M. Anderson, S.P. Amarasinghe, B.R. Murphy, L. Shih-Wei, E. Bugnion, and M.S. Lam. "Maximizing Multiprocessor Performance with the SUIF Compiler". *Computer*, 29(12):84–89, 1996.

[10] R. Gonzales, B.M. Gordon, M.A. Horowitz. "Supply and Threshold Voltage Scaling for Low Power CMOS". *Journal of Solid-State Circuits*, 32(8):1210–1216, August 1997.

[11] S.M. Martin, K. Flautner, T. Mudge, D. Blaauw. "Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads". In *Proceedings of the 2002 IEEE/ACM international conference on Computer-aided design*, pages 721–725, 2002.

[12] T. Austin, E. Larson, D. Ernst. "SimpleScalar: an infrastructure for computer system modeling". *Computer*, 35(2):59–67, February 2002.

[13] T.D. Burd, T.A. Pering, A.J. Stratakos, R.W. Brodersen. "A dynamic voltage scaled microprocessor system". *IEEE Journal of Solid-State Circuits*, 35(11):1571–1580, November 2000.