Energy Estimation and Optimization of Embedded VLIW Processors based on Instruction Clustering

A. Bona[§] M. Sami[‡] D. Sciuto[‡] C. Silvano[♯] V. Zaccaria[‡] R. Zafalon[†]
[§]ALaRI, Lugano, Switzerland [‡] Politecnico di Milano, Milano, Italy
[♯] Università degli Studi di Milano, Milano, Italy [†]STMicroelectronics, Agrate B. (MI), Italy

ABSTRACT

Aim of this paper is to propose a methodology for the definition of an instruction-level energy estimation framework for VLIW (Very Long Instruction Word) processors. The power modeling methodology is the key issue to define an effective energy-aware software optimisation strategy for state-of-the-art ILP (Instruction Level Parallelism) processors. The methodology is based on an energy model for VLIW processors that exploits instruction clustering to achieve an efficient and fine grained energy estimation. The approach aims at reducing the complexity of the characterization problem for VLIW processors from exponential, with respect to the number of parallel operations in the same very long instruction, to quadratic, with respect to the number of instruction clusters. Furthermore, the paper proposes a spatial scheduling algorithm based on a low-power reordering of the parallel operations within the same long instruction. Experimental results have been carried out on the Lx processor, a 4-issue VLIW core jointly designed by HPLabs and STMicroelectronics. The results have shown an average error of 1.9% between the cluster-based estimation model and the reference design, with a standard deviation of 5.8%. For the Lx architecture, the spatial instruction scheduling algorithm provides an average energy saving of 12%.

Categories and Subject Descriptors

B.7.0 [**Design Aids**]: General; B.6.0 [**Logic Design**]: General

General Terms

Experimentation

Keywords

power estimation, vliw architectures

1. INTRODUCTION

The overall goal of this work is to propose a methodology for the definition of an instruction-level energy estimation framework for VLIW processors. This methodology is the key issue to enable an effective low-power hardware and software exploration framework for state-of-the-art ILP processors. The approach presented in this paper aims at extending and integrating into an overall power estimation and optimisation framework our work previously proposed in [1–3], targeting an instruction-level energy model to evaluate the energy consumption associated with a software execution on a pipelined VLIW core. A pipelined VLIW processor executes a set of explicitly parallel *operations* (also called *syllables*) during each clock cycle; this set of operations are statically scheduled to constitute the *Very Long Instruction Word* (also called instruction or *bundle*).

The proposed strategy is based on the combination and interaction of different well known techniques such as clustering and instruction scheduling to obtain an overall hardware/software power optimization for VLIW embedded systems.

First, we aim at increasing the efficiency of the power estimation by reducing the complexity of the VLIW instruction-level model. With this purpose, we apply the instruction clustering concept to the VLIW instruction set, by characterizing only each single operation in isolation, then by clustering operations considering their average energy cost and by characterizing the inter-instruction effects on the clusters.

Second, the power optimization methodology acts at the software level through the definition of a technique to statically optimize the executable applications. The proposed technique consists of a *spatial scheduling* algorithm based on the reordering of parallel operations within the same bundle with respect to the previous bundle. The algorithm is based on the minimization of a cost function that considers inter-instruction effects and power figures carried out from the power characterization phase.

The experimental results validating the characterization and scheduling approaches have been carried out on the Lx architecture, a scalable and customizable processor technology [4] designed for multimedia and signal processing embedded applications. The Lx processor is a statically scheduled VLIW multi-clustered architecture jointly designed by Hewlett-Packard and STMicroelectronics, based on a multiple-issue VLIW core.

The paper is organized as follows. Some relevant background works on instruction-level power analysis and optimization are summarized in Section 2. An overall description of the proposed VLIW power model is given in Section 3, while Section 4 introduces the methodology to cluster operations with the same power behavior. Some experimental results are reported in Section 5, while Section 6 contains some concluding remarks and outlines some future research directions.

2. PREVIOUS WORK

Instruction-level power estimation techniques are based on the functional simulation of the application software on the target ISA (Instruction Set Architecture). During the instruction set simulation (ISS), a black-box instruction-level model correlates the power cost to each individual instruction, by considering both internal state effects, pipeline stalls and cache misses occurred at run time. One of the most significant works on instruction-level power analysis (*ILPA*, for brevity) is based on electrical measurements techniques [5–7] to characterize the *average energy per instruction* and an *average inter-instruction effect* energy. The work outlines that the spatial complexity of an instruction-level energy model that considers y inter-instruction effects (i.e., between instruction i and $i-y$) and x instructions in the ISA is $O(x^{y+1})$. The authors recognized that instruction clustering could be a feasible way to reduce this complexity: grouping the instructions in c clusters, the complexity is reduced to $O(c^{y+1})$.

The instruction-level power model proposed by Russel *et al.* [8] considers the average instruction energy as invariant for all instructions in the ISA. More specifically, this model is based on the observation that, for a given class of processors, the *energy per instruction* shows a very small variance. Sarta *et al.* [9] propose a processor power model by considering possible inter-instruction effects as well as the actual data statistics. Although the developed power model is quite accurate, it lacks general applicability, being developed only for a specific embedded processor. Klass *et al.* [10] propose a power model in which inter-instruction effects are measured by considering only the additional energy consumption observed when a generic instruction is executed after a NOP (the proposed power model is also called the *NOP model*). This model could be an effective solution to the problem of spatial complexity proper of instruction-level power models.

Concerning low-power scheduling algorithms, some approaches have been proposed by Tiwari *et al.* [5–7], but these works target either scalar or DSP processors with packed operations and their extension to VLIW code generation is not straightforward. Other authors [11,12] introduced power optimization methodologies from a software-level perspective, such as pre-processing and restructuring the source code to reduce the power consumption of the executable code. Other techniques, such as spatial and temporal scheduling, have been proposed by Lee *et al.* [13]. Spatial scheduling tries to directly minimize the switching activity on the instruction bus by choosing suitable pairs of instructions through a bipartite matching scheme. Since the temporal scheduling *algorithm* is an *NP*-hard problem, it is heuristically solved by reducing it to a bipartite matching scheme on a limited instructions window. Finally, Parikh *et al.* [14] modified list-scheduling by trading-off energy and speed simultaneously.

3. VLIW POWER MODEL

Concerning k-issue ILP processors, where each instruction is composed of k parallel operations, we have that, considering x operations in the ISA, and y inter-instruction effects, the complexity of the instruction-level energy model is $O(x^{k*(y+1)})$. In [3], we introduced the *temporal* and *spatial additive properties*, to deal with such a complexity. The temporal additive property enables the decomposition of the instruction energy into the pipeline stages of the processor. The spatial additive property enables the decomposition of the energy dissipated by the processor in the single energy dissipated by the processor parallel paths followed by the operations.

More in detail, the model considers the energy associated with the n-th instruction $\mathbf{w_n}$ as dependent on its own properties (e.g., class of the instruction, addressing mode and so on) as well as on its *execution context*, i.e., the previous instruction $\mathbf{w_{n-1}}$ and the additional stall/latency cycles introduced during the execution of the instruction. Besides, the inter-instruction effect between $\mathbf{w_n}$ and $\mathbf{w_{n-1}}$ can be decomposed into the sum of the inter-operation effects $(w_n^k|w_{n-1}^k)$ on the k-th parallel path (or *lane*) of the processor [3]:

$$E(\mathcal{W}) \approx \sum_{1\le n\le N} \sum_{\forall s\in S} \Big[U_s(\mathbf{0}|\mathbf{0}) + \sum_{\forall k\in K} \nu_s(w_n^k|w_{n-1}^k) + + m_s^n * p_s^n * S_s + l_s^n * q_s^n * M_s \Big] \quad (1)$$

where the term S is the set of pipeline stages of the processor, $U_s(\mathbf{0}|\mathbf{0})$ is the base energy cost that represents the energy consumed by stage s during an execution of a bundle constituted entirely by NOPs ($\mathbf{0} = [\texttt{NOP}\ldots\texttt{NOP}]^T$), $\nu_s(w_n^k|w_{n-1}^k)$ is the additional energy contribution due to the change of operation ($w_{n-1}^k \rightarrow w_n^k$) on the same lane k, m_s^n (l_s^n) is the average number of additional cycles due to a data (instruction) cache miss during the execution of the $\mathbf{w_n}$ in s, p_s^n (q_s^n) is the probability that this event occurs, and S_s (M_s) is the energy consumption per stage of the processor modules that are active due to a data (instruction) cache miss.

The term $\sum_s U_s(\mathbf{0}|\mathbf{0})$ corresponds to the power consumption of the core during NOPs and and we assume it can be substituted by the average base cost $U(\mathbf{0}|\mathbf{0})$ (as confirmed by the experimental results where this approach has been applied). Besides, $\sum_s \sum_k \nu_s(w_n^k|w_{n-1}^k)$ can be substituted by a cost dependent only on the pair of instructions ($\sum_k \nu(w_n^k|w_{n-1}^k)$) that corresponds to the energy consumption of the core while it is executing the same pair of instructions ($\mathbf{w_n}, \mathbf{w_{n-1}}$). For instruction and data cache misses, we assume that the probabilities per stage (p and q) and their penalties (m and l) can be averaged for each instruction of the stream:

$$\sum_s (m_s^n * p_s^n * S_s + l_s^n * q_s^n * M_s) \rightarrow (m * p * S + l * q * M) \quad (2)$$

where $m(l)$ is the average data (instruction) cache miss length, $p(q)$ is the average probability per stage and per instruction that a data (instruction) cache miss can affect one instruction and S (M) is the average energy consumption of the *entire* processor during these events.

Based on these assumptions, the equation (1) can be reduced to:

$$E(\mathcal{W}) \approx \sum_{1 \le n \le N} \Big[U(\mathbf{0}|\mathbf{0}) + \sum_{\forall k \in K} \nu(w_n^k | w_{n-1}^k) + \\ + m * p * S + l * q * M \Big] \quad (3)$$

For a k-issue VLIW core, the complexity of the model is now quadratic with respect to the number of operations into the instruction set ($O(k * |ISA|^2)$), while a black-box model would have a complexity $O(|ISA|^k)$. This implies a significant reduction of the characterization effort, since the number of experiments to be done is reduced exponentially.

In the next section, we show how this model can be further simplified by clustering the operations of the ISA with respect to their average energy consumption to achieve a faster and more effective characterization of the core's power consumption, while preserving the model accuracy.

4. CLUSTERING OF THE OPERATIONS

In this work, we consider two operations in the ISA as "different" if they differ either in terms of functionality (i.e., opcode) or in terms of addressing mode (immediate, register, indirect, etc.). Even without considering data differences (in terms of register names or immediate values), the number of operations pairs to be considered in equation (3) would be too large to be characterized with a transistor-level or even gate-level simulation engine. In the case of the Lx processor, for example, the ISA is composed of about 70 operations, but considering also the possible addressing modes for each operation, we would need to characterize 6126 pairs of operations to completely define our model (see the characterization of the matrix ν in equation (3)).

To give a rough idea of the time required for the characterization phase, a Sun Ultra Sparc 60 workstation at 450MHz with 1GB RAM performs a gate-level power simulation of the Lx core in 25 minutes. Given such simulation time, we would need approximately 108 days to perform the complete characterization phase on a single workstation.

To reduce the number of experiments to be generated during the characterization phase, we propose to apply the well known cluster analysis on the operations of the ISA. The basic idea of the cluster analysis consists of grouping into the same cluster the operations showing a power cost close to each other. The power cost of an operation is defined as the power consumed by the processor when it executes that operation.

Among the various clustering algorithms appeared in literature so far (see [15, 16] for a survey), we have chosen the k-mean clustering algorithm since it requires a lower computational cost.

Given a set of energy values $\Theta = \{e_1 \ldots e_t \ldots e_\theta\}$, where e_t is the energy consumption of instruction t measured by executing a loop composed of only t instructions, the k-mean clustering algorithm tries to partition Θ into a set of K clusters ($C_1 \ldots C_K$) to minimize the mean-square error:

$$\sum_{j=1}^{K} \sum_{i=1}^{n_j} (x_{i,j} - c_j)^2, \quad x_{i,j} \in C_j \quad (4)$$

where n_j is the number of elements of cluster C_j, $x_{i,j}$ is the i-th element of cluster j and c_j is the center of gravity of the j-th cluster. The k-mean clustering algorithm receives as input the number K of classes in which the original population must be partitioned, and it randomly splits the whole set into K subsets. Then, each element is moved into the subset j whose center of gravity is closest. This procedure iterates until the stopping criterion is met.

To provide experimental evidence on the accuracy of the clustering algorithm applied on the instructions of the Lx ISA, Figure 1 shows the mean and the variance of the energy values associated with each instruction in the ISA and the clusters in which they have been grouped. In the experimental section of this paper, we have proved that the maximum variance in a cluster for the Lx architecture can be considered very limited (within 13% for 11 clusters).

Once the instructions have been clustered into a set of $C_1 \ldots C_j$ clusters, we compute the matrix ν in equation (3) in the following way:

$$\nu(w_n^k | w_{n-1}^k) = \begin{cases} E_i & \text{if } w_n^k = w_{n-1}^k \wedge w_n^k, w_{n-1}^k \in C_i \\ D_{i,j} & \text{if } w_n^k \neq w_{n-1}^k \wedge w_n^k \in C_i, w_{n-1}^k \in C_j \end{cases} \quad (5)$$

Without considering the switching activity due to data dependency, this decomposition tries to model the fact that when operations are equal ($w_n^k = w_{n-1}^k$), they generate the least switching activity possible (first case). In the second case, when the operations are different, they generate an increased switching activity even if they are in the same cluster (accounted by the matrix $D_{i,j}$). Note that the complexity of matrix ν has been reduced from $O(|ISA^2|)$ to $O(|C^2|)$, which represents the upper bound on complexity given by $D_{i,j}$.

To successfully apply the clustering algorithm, one of the most crucial parameters to choose is the number of clusters. In general, a small number of clusters implies a high variance within them (i.e., poor accuracy of the model), though it also implies a small number of experiments during the regression. On the contrary, a large number of clusters implies good accuracy, but would result in a huge number of experiments.

In our methodology, the number of clusters is automatically determined by a tradeoff between the maximum standard deviation of the elements within the same cluster and the number of experiments that must be done to characterize the model. Section 5.2 reports the application of the tradeoff analysis performed on the Lx processor to determine the number of clusters.

The minimum number of experiments necessary to characterize the energy model is a typical cost function that must be reduced during the design of the experiments. In this work, we assume that the linear regression [17] is used to perform the characterization of the model. In this case, the minimum number of experiments is linearly dependent on the number of parameters of the model. Thus, the characterization cost has the following upper bound:

$$O\left(\frac{K \cdot (K-1)}{2}\right) \quad (6)$$

where K is the number of clusters.

5. LX CASE STUDY

In this section, we describe how the proposed methodology for power estimation and optimization has been applied to

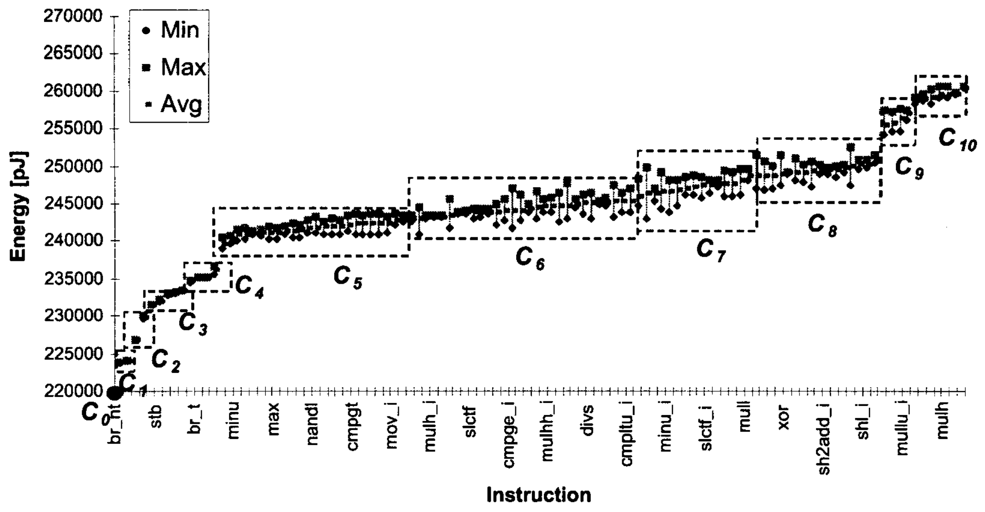

Figure 1: Energy values associated with each instruction and the corresponding clusters (where cluster C_0 corresponds to the NOP operation).

the Lx processor. The Lx architecture is a scalable and customizable processor technology [4] designed for multimedia and signal processing embedded applications. The Lx processor is a statically scheduled VLIW architecture designed by Hewlett-Packard and STMicroelectronics, that supports a multi-cluster organization based on a single PC and a unified I-cache. The basic processor is 4-issue VLIW core featuring four 32-bit integer ALUs, two 16x32 multipliers, one load/store unit and one branch unit. The cluster also includes 64 32-bit GPRs an 8 1-bit Branch Registers. Lx has an in-order 6-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 instructions issued per cycle).

Lx comes with a complete software tool-chain, where no visible changes are exposed to the programmer when the core is scaled and customized. The tool-chain includes a sophisticated *ILP* compiler technology (derived from the Multiflow compiler [18]) and GNU tools and libraries. The Multiflow compiler includes both traditional high-level optimization algorithms and aggressive code motion technology based on trace scheduling.

A mix of synthesizable RTL and gate-level netlist of the core processor has been used to perform the characterization needed for the power model presented in this work. The experiments have been carried out by using Synopsys VCS 5.2 and a set of PLI routines to elaborate toggle statistics over the whole gate-level netlist. PowerCompiler (by Synopsys) has been used to combine the toggle statistics with the power models of the standard cells library provided by STMicroelectronic, to compute the power figures of the entire core.

5.1 Lx Power Characterization

To characterize the power consumption model expressed by equation (3), we first proceed to group the operations into clusters through the k-mean clustering algorithm. For each operation o, we generated a set of assembly programs composed of repeated cycles of o operations (by varying register names and values) and we measured the energy consumption of the core at the gate-level. Then we applied, to the measured values, the k-means clustering algorithm for several values of K. In order to determine the most suitable number of clusters, we analyzed the minimum number of experiments that would be needed to characterize the energy model with K clusters and we performed a tradeoff analysis with respect to the maximum variance within the clusters. As mentioned above, the minimum number of necessary experiments depends linearly on the number and the size of the parameters involved in the model. In our case, the shape of the curve defining the number of experiments is quadratic, due to the quadratic dependence of the size of the matrix ν with respect to the number of clusters K.

Figure 2 shows the results concerning the maximum variance within the operations clusters and the number of experiments required to characterize each corresponding model. From the reported results, we can note how, for a number of clusters equal to 11, we can obtain a good tradeoff between the maximum variation (that drops to 13%) and the minimum number of experiments (that reaches 78). For this reason, we selected 11 clusters to characterize the Lx power model.

Once selected the number of clusters and, therefore, the number of coefficients in the model (i.e., the size of the ν matrix), we realized a set of experiments in which each possible pairs of clusters have been generated several times by varying register names and immediate values.

These experiments have been performed by keeping the value of p and q (the data and instruction cache miss probabilities per instruction) as lowest as possible to characterize by regression only the terms $U(\mathbf{0}|\mathbf{0})$ and $\nu(w_n^k|w_{n-1}^k)$. The values of M and S (the energy consumed per cycle during an I-cache and D-cache miss) have been characterized by generating a set of experiments with a large number of data and instruction cache misses, and by measuring the power only during these events. Finally the m and l miss penalties have been extrapolated by looking at the behavior of the microarchitecture during these events.

The model has been tested against a set of validation benchmarks that are different from the experimental setup used to perform the characterization. The validation benchmarks include a set of the Mediabench applications [19] (namely, G721 encoder and decoder, EPIC encoder and decoder, MPEG2), a set of finite impulse response filters, discrete cosine transforms and matrix elaboration algorithms. Figure 3 shows the scatter plot of the measured power values for the validation benchmarks and the power values estimated with our model. The power model has shown, on the validation benchmarks, a mean error of 1.9% and a standard deviation on the error of 5.8%. The multiple correlation coefficient, that explains the percentage of the total variation exploited by the model, has been computed as in [17] and is equal to:

$$\sqrt{\frac{SSR}{SSR+SSE}} = 90\% \tag{7}$$

As a matter of fact, there are only three benchmarks whose error on the prediction is in the neighborhood of 10% probably because of the high switching activity of the data consumed by instructions that is not captured by the model. Note that for high-level/instruction-level power macro models (whose main concern is efficiency), this can be considered an acceptable value in terms of accuracy.

5.2 Low-Power Spatial Scheduling

Once the model has been fully characterized, our basic idea consists of using it as a cost function to statically reorder instructions to minimize the power associated with the executable application.

The basic idea consists of defining an algorithm that considers each basic block of the code generated by the compiler and aims at rescheduling operations within the same bundle (*spatially*). The algorithm starts from bundle $\mathbf{w}_{n=1}$ and attempts to find a suitable reordering of each w_n^k in order to minimize the cost function:

$$\sum_k \nu(w_n^k | w_{n-1}^k), \quad \forall (w_n^k | w_{n-1}^k) \in \text{Basic block}$$

being w_{n-1}^k fixed (see equation 3). This step is iterated on all the adjacent bundles $\mathbf{w_n}$ of the basic block. All the possible permutations of operations within the bundle are evaluated, according to architectural constraints. In actual VLIW processors, the complexity of this approach is acceptable, since the number of k parallel operations ranges from 2 to 8, requiring the evaluation of $k!$ permutations (neglecting architectural constrains that decrease this number). For example, for the 4-issue VLIW Lx processor core, the total number of permutations to be checked is limited to $4! = 24$.

The choice of experimental benchmarks includes the compiled code of two FIR filters (*fir*1 and *fir*2), the fast discrete cosine transform (*fastdct*), the matrix multiply algorithm (*matrix*) and the bubble sort algorithm (*sort*). Figure 4 shows the power and energy savings obtained by apply-

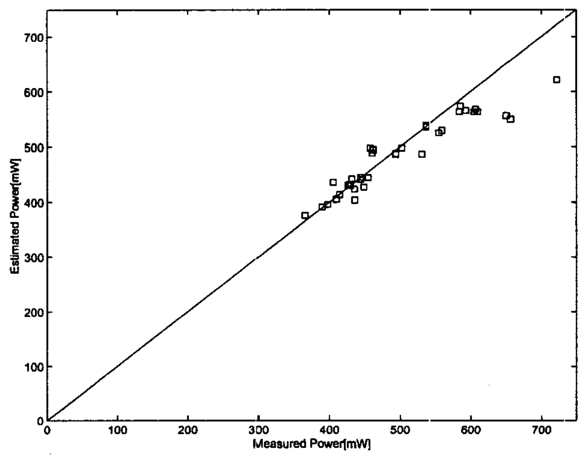

Figure 3: Scatter plot of the measured power values with respect to the estimated power values

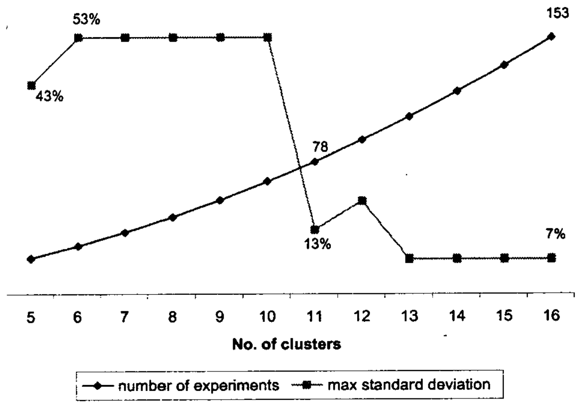

Figure 2: Tradeoff between the number of experiments required to characterize the model for a given number of clusters and the maximum variance within each cluster.

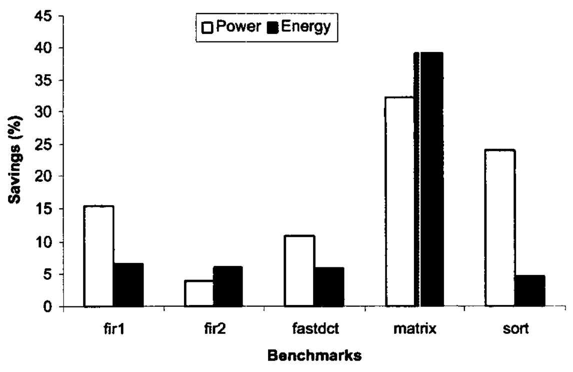

Figure 4: Power and energy savings obtained by our spatial rescheduling algorithm applied to the Lx processor executing the selected set of benchmarks.

ing the rescheduling algorithm to the selected set of benchmarks. The power savings have been computed by simulating, on the reference description of the processor core (gate-level), both the original and the rescheduled code of the benchmarks.

The average power consumption decreases by 17%, while the average energy consumption decreases only of 12%, since the rescheduling algorithm slightly increases the latency of the code. This is due to the fact that operation rescheduling could impact the efficiency of the Lx's instruction compression mechanism, leading to an increment of cache misses. However, in some cases (*fir*2 and *matrix*) the instruction cache misses are reduced since, due to the particular structure of the code, the rescheduling algorithm can lead to a more regular code in terms of instruction cache access patterns.

6. CONCLUDING REMARKS

We have presented an instruction-level methodology to estimate and to optimize the energy consumption in embedded systems based on VLIW architectures. The first goal of the proposed work is the reduction of the complexity of the energy model for VLIW cores, while preserving a good level of accuracy. The second goal of the work is to show how the proposed energy model can be further simplified by automatically clustering the operations in the *ISA*, based on the average energy behavior of the operations. Moreover, we have also shown how the general power-aware methodology has been successfully applied to the Lx VLIW embedded core. Some results have also been discussed derived from the application of the proposed low-power instruction scheduling algorithm. Future directions of our work target more complex dynamic instruction scheduling algorithms taking advantage of the accurate results derived from the proposed power characterization methodology.

7. REFERENCES

[1] L. Benini, D. Bruni, M. Chinosi, C. Silvano, V. Zaccaria, and R. Zafalon, "A power modeling and estimation framework for VLIW-based embedded systems," in *Proceedings of International Workshop-Power And Timing Modeling, Optimization and Simulation, PATMOS'01*, 26–28 2001.

[2] M. Sami, D. Sciuto, C. Silvano, and V. Zaccaria, "Instruction level power estimation for embedded VLIW cores," in *Proceedings of the 8th International Workshop on Hardware/Software Codesign (CODES-00)*, NY, May 3–5 2000, pp. 34–38, ACM.

[3] M. Sami, D. Sciuto, C. Silvano, and V. Zaccaria, "Power exploration for embedded VLIW architectures," in *Proceedings of the IEEE/ACM International Conference on Computer Aided Design (ICCAD-2000)*, Nov. 5–9 2000, pp. 498–503.

[4] P. Faraboschi, G. Brown, J. Fisher, G. Desoli, and F. Homewood, "Lx: a technology platform for customizable vliw embedded processing," in *Proceedings of the International Symposium on Computer Architecture*, June 2000, pp. 203–213.

[5] T. Lee, V. Tiwari, and S. Malik, "Power analysis and minimization techniques for embedded dsp software," *IEEE Transactions on VLSI Systems*, vol. 5, no. 1, pp. 123–135, 1997.

[6] V. Tiwari, S. Malik, and A. Wolfe, "Power analysis of embedded software: a first step towards software power minimization," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 2, no. 4, pp. 437–445, 1994.

[7] V. Tiwari, S. Malik, A. Wolfe, and M. Lee, "Instruction level power analysis and optimization of software," *J. VLSI Signal Processing*, pp. 1–18, 1996.

[8] J. Russell and M. Jacome, "Software power estimation and optimization for high performance, 32-bit embedded processors," in *International Conference on Computer Design: VLSI in Computers and Processors*, 1998, pp. 328–333.

[9] D. Trifone D. Sarta and G. Ascia, "A data dependent approach to instruction level power estimation," in *Proc. IEEE Alessandro Volta Memorial Workshop on Low Power Design*, Como, Italy, March 1999, pp. 182–190.

[10] B. Klass, D. Thomas, H. Schmit, and D. Nagle, "Modeling inter-instruction energy effects in a digital signal processor," in *Power-Driven Microarchitecture Workshop*, June 1998.

[11] N. Zervas, K. Masselos, and C. Goutis, "Code transformations for embedded multimedia applications: Impact of power and performance," in *Proceedings of the Power-Driven Microarchitecture Workshop In Conjunction With ISCA98*, June 1998.

[12] H. Mehta, R. Owens, M. Irwin, R. Chen, and D. Ghosh, "Techniques for low energy software," in *Proceedings of the International Symposium on Low Power Electronics and Design*, August 1997, pp. 72–75.

[13] C. Lee, J. K. Lee, and T. Hwang, "Compiler optimization on instruction scheduling for low power," in *Proceedings of The 13th International Symposium on System Synthesis*. Sept. 20–22 2000, pp. 55–60, IEEE Computer Society Press.

[14] A. Parikh, M. Kandemir, N. Vijaykrishnan, and M.J. Irwin, "Instruction scheduling based on energy and performance constraints," in *Proceedings of the IEEE Computer Society Workshop on VLSI*. Apr. 27–28 2000, pp. 37–42, IEEE Computer Society Press.

[15] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review.," *ACM Comp. Surveys*, vol. 31, no. 3, pp. 264–323, Sept. 1999.

[16] John H. Gennari, "A survey of clustering methods," Technical Report ICS-TR-89-38, University of California, Irvine, Department of Information and Computer Science, Oct. 1989.

[17] Q. Wu, Q. Qiu, M. Pedram, and C. Ding, "Cycle-accurate macromodels for rt-level power analysis," in *Proc. International Symposium on Low Power Electronics and Design*, 1997, pp. 125–130.

[18] P. Geoffrey Lowney, Stefan M. Freudenberger, Thomas J. Karzes, W. D. Lichtenstein, Robert P. Nix, John S. O'Donnell, and John C. Ruttenberg, "The Multiflow Trace Scheduling compiler," *The Journal of Supercomputing*, vol. 7, no. 1-2, pp. 51–142, 1993.

[19] C. Lee, M. Potkonjak, and W. H. Mangione-Smith, "Mediabench: A tool for evaluating multimedia and communication systems," in *Proceedings of Micro 30*, 1997.