

Power Estimation of System-Level Buses for Microprocessor-Based Architectures: A Case Study

William Fornaciari
Politecnico di Milano, DEI
Milano (Italy)
fornacia@elet.polimi.it

Donatella Sciuto
Politecnico di Milano, DEI
Milano (Italy)
sciuto@elet.polimi.it

Cristina Silvano
CEFRIEL
Milano (Italy)
silvano@cefriel.it

Abstract

The processor-to-memory communication on system-level buses dissipates a significant amount of the overall power in microprocessor-based architectures. A methodology has been set up to evaluate the effects of both encoding schemes and multi-level cache memories on the power consumption associated with the system-level address and data buses of a high-end computer system based on the PowerPC604e architecture. The main goal is to evaluate how different values of cache parameters (cache size, block size, associativity, write strategy, and block replacement policy) and the introduction of bus encoding techniques, at the different levels of the memory hierarchy, affect the system-level power dissipation.

1. Introduction

The problem of evaluating the impact of design choices on the overall power budget in a digital system can be afforded at different abstraction levels. The tight interaction between the power estimation and optimization phases, at different levels, is crucial to define a power balance during the first design stages, to early determine the power budget and to avoid design re-cycles. Recently, the main research efforts have been directed toward the investigation of *high-level* power modeling and optimization techniques.

In this perspective, it is mandatory to provide a model to access the power consumption due to the *processor-to-memory communication* on system-level buses, which represents one of the major contributions to the overall power budget, in particular for high-end computer architectures characterized by wide word lengths. In general, a multi-level cache implies a strong impact on the overall power budget due to the communication on heavily loaded buses. The power cost of an external memory access is at least one order of magnitude higher than that of an on-processor access, due to capacitance overhead of I/O pads, on-board traces and input loads of DRAM components.

However, up to now, most cache studies are related to performance evaluation in terms of cache access time and miss rate with respect to different cache parameter values. Only recently, cache models have focused on power evaluation ([1], [2]). Nevertheless, to the best of our knowledge, none of them considers the impact of caches and main memory on the power consumption due to the processor-to-memory communication in the system-level context.

The approach we proposed in [3] can be considered as an attempt to fill such a gap. Previous literature has been mainly concentrated either on power estimation models for systems including memory hierarchies or on bus encoding for low-power consumption.

Su and Despain [2] proposed a model to evaluate the power/performance trade-offs in cache design and the effectiveness of novel cache design techniques targeted for low power (such as vertical and horizontal cache partitioning). Another model of energy consumption for the memory hierarchy has been provided in [4]. The adopted model is the same analytical model for *on-chip* caches proposed in [2], which has been generalized to include the main memory, so as to consider the amount of power required by cache misses. In [1], the *Avalanche* framework is presented, to simultaneously evaluate the energy-performance trade-offs for *SW*, memory and *HW* for embedded systems based on a *system-on-a-chip* architecture.

Encoding techniques for reducing the switching activity on the bus lines have recently been investigated and compared ([5], [6], [7], [8], [9], [10]). Most of them rely on the well-known *spatial locality* principle [11].

Aim of our research [3] is to evaluate the HW/SW communication contribution to the overall system-level power due to the simultaneous effects of both bus encoding and memory hierarchy. The power model we proposed in [3] consists of three main cooperating sub-models: the memory hierarchy, the bus encoder and the address/data stream generator. First, the model of the *memory hierarchy* enables us to consider multi-level unified or split caches offering several configurations in

terms of cache size, block size, associativity, write strategy and block replacement policy. Second, the *bus encoding* model implements the most widely adopted power-oriented encoding schemes for both data and address buses. The model enables the insertion of the encoder/decoder logic at the different levels of the memory hierarchy. Third, the *stream generator* models typical address and data generated from the processor to the memory sub-system, taking into account spatial and temporal locality.

The primary goal of our model is to perform a trade-off analysis in defining the most suitable cache parameters and system-level bus architectures from the power standpoint. To preserve the generality of the approach, we do not include power figures related to the contribution of the memory arrays. However, the model can be easily combined with power data related to the memory arrays technology [2].

A power estimation methodology has been set up to evaluate the switching activity on system-level buses for a wide range of processor-to-memory configurations. This paper presents a case study aiming at analyzing power-performance trade-offs derived by applying the proposed model to a high-end computer system based on the 64-bit *PowerPC604e* processor. The simulation results show how the use of multi-level caches with variable parameters can be combined with different bus encoding techniques to modify miss rates and to obtain power figures at the system-level.

The original contributions introduced by our model with respect to the previous ones can be summarized as follows. The model includes different levels in the memory hierarchy, such as *on-* and *off-*processor caches as well as the main memory, where each level of the hierarchy can be customized in terms of cache parameters (such as cache size, block size, associativity, etc.). The model is independent of both the internal cache or memory organization and implementation technology. The most recent low-power bus encoding techniques for both data and address buses have been considered and the bus encoder/decoder logic can be placed between the different levels of the memory hierarchy. The bus parameters can be varied in terms of width, frequency, and capacitive load. Real bus tracings (derived from different application programs and executed by different processors) as well as dedicated bus tracings (generated to analyze the effects of different spatio-temporal correlations of the bus streams) can be analyzed.

The paper is organized as follows. Section 2 describes the proposed system-level power model, while Section 3 presents the simulation methodology used to profile the power trends of the selected case study, and discusses the experimental results. Finally, Section 4 summarizes the most significant contributions of the paper.

2. The System-Level Power Model

The proposed power estimation model [3] is composed of *three* main cooperating sub-modules: (i) the *memory hierarchy*, (ii) the *bus encoder*, and (iii) the *address/data stream generator*, which have been integrated in an analysis tool, written in C++. The *memory hierarchy model* consists of a multi-level memory sub-system composed of on-processor and off-processor caches and the main memory. A generic cache level can be organized either as single *unified* cache or *split* cache for instructions and data. The cache model enables the designer to consider several configurations in terms of cache size, block size, associativity, write strategy and replacement policy [11]. The memory address space can be arbitrarily large; the size of the *block* or *cache line* can be set arbitrarily; the cache size can be any multiple of the block size; the degree of associativity (or number of way *n*) can vary from direct-mapped to fully associative caches. The write strategy can be *write-through* or *write-back* and, in the case of a write miss, either the *write-allocate* or the *no-write-allocate* option can be chosen. For set or fully associative caches, the block replacement policy can be *random* or *LRU (Least Recently Used)*.

The *bus encoder model* can be inserted either on the interface from the processor to the first level of the memory hierarchy or between any adjacent levels of the hierarchy to evaluate the bus encoding effects on power consumption. The model implements the main power-oriented bus encoding techniques, namely Gray, Bus-Invert [9], *T0*, *T0_BI*, *Dual_T0* and *Dual_T0_BI* [6]. The encoding schemes can be applied to both data and address buses.

The generator outputs are tightly dependent on the processor architecture. The current version of the *stream generator model* includes a generic load/store *RISC* architecture. For our analysis, we considered a sub-set of a generic *RISC* instruction set, which is composed of *three* basic classes of instructions: Conditional Branch Instructions (*B*); Arithmetic-Logic or Data Processing Instructions (*DP*); Load/Store or Data Transfer Instructions (*DT*). The memory address spaces for data and instructions are *separated*. Basically, the sequence of memory addresses is generated by assigning the percentage of the different classes of instructions with respect to the total number of generated addresses. The address sequence is generated by the processor by varying: the format and the execution frequency for each instruction class; the possible addressing modes for each instruction and the related execution frequency; the frequency to satisfy a conditional branch. All these parameters contribute to modify the spatial and temporal locality of memory references.

The address bus from the processor to the memory sub-system contains a memory address corresponding to a

datum or an *instruction*. The address stream characteristics can be assigned depending on the desired level of the spatial and temporal locality. The bi-directional data bus can carry *two* different types of information: *instructions* and *data*. The type of instruction contained at a given memory address depends on the parameters set for the address bus model. Meanwhile, the datum contained in the memory address can be generated either probabilistically or pseudo-randomly. In the first case, the model is based on a medium average model of the first order, $MA(1)$, to take into consideration the correlation between two consecutive data words, responsible for the switching activity on the system-bus.

3. Power Analysis of the Case Study

The model described in [3] has been set up to profile the power consumption due to the system-level buses communication from the processor to the memory sub-system in a high-end computer system based on the 64-bit *PowerPC604e* processor. The memory sub-system includes the L1 and L2 caches and the main memory. The L1 cache is composed of a 32 KB 4-way set-associative *on*-processor cache, while the L2 is based on a configurable *off*-processor cache.

The behavior of the different architecture configurations has been compared with the behavior of the *reference architecture* composed of the *PowerPC604e* processor interfacing the main memory directly. Based on the reference architecture, two system configurations have been studied in 3.1 and 3.2 respectively. The reported power figures are intended as the average power dissipated due to the communication on the system-level buses, thus we disregarded the power contributions of hardware resources to the overall power budget. Previously published power models can be adopted to evaluate the power associated with the memory arrays and the processor, while for the encoding logic we can use the data reported in ([6]). Under this assumption, the average power consumption on the system bus is:

$$P_{ave} = \frac{1}{2} C_{load} V_{dd}^2 n_{trans} f$$

where C_{load} is the load capacitance, V_{dd} is the power supply voltage, n_{trans} is the average number of transitions on the bus lines, and f is the operating frequency of the system bus. Let us assume $B^{(t)}$ be the value of the encoded word on the system bus at time t , and L be the total length of the generated stream, the average number of transitions on the bus lines, n_{trans} , can be given by:

$$n_{trans} = \frac{\sum_{t=0}^{L-1} H(B^{(t)}, B^{(t+1)})}{L-1}$$

where $H(B^{(t)}, B^{(t+1)})$ is the Hamming distance between the encoded word on the bus at time t and $(t+1)$.

Since to the best of our knowledge, no commercial or academic estimation tool exists to compare the results obtained in this paper, we try to find an analytical explanation to the obtained power trends.

The reference architecture for the high-end system consists of the 64-bit *PowerPC604e* processor and the main memory. The processor runs at 200 MHz *on*-chip and 100 MHz *off*-chip, while the main memory interfaces the processor through 32-bit address and data buses operating at 100 MHz and loaded by 60 pF. For the simulation, we generated an instruction stream composed of 1 M instructions constituted by 30% *B* instructions (20% *NSB*), 25 *DP* instructions and 45 *DT* instructions. The average number of consecutive addresses is 0.05, the ratio for the data addresses among the number of write and read address is 0.5, the average length of the basic streams is equal to 600 memory addresses, the 32-bit data are generated probabilistically (the one-step correlation coefficient is equal to 0.1 on the 28 least significant bits and 0.9 for the remaining bits). The memory word is 32-bit. The values for the generated stream have been chosen to obtain a significant number of accesses for both the first and second level caches. Hence, the spatial and temporal locality indexes for the high-end system are those expected for a system whose resources are shared among different tasks [11].

The memory hierarchy we analyzed for power profiling is composed of two cache levels. The *PowerPC604e* includes the *on*-processor first level 4-way set-associative cache providing 16KB for instructions and 16KB for data, and 256-bit blocks. Concerning the *off*-processor second level cache, we considered the effects of the cache parameters variation in terms of cache size (512 KB, 1024 KB, 2048 KB and 4096 KB), block size (256-bit and 512-bit) and degree of associativity (one-two-four-eight way). Both cache levels apply a write through strategy with no write allocate and *LRU* substitution policies.

3.1. Second-Level Cache Hierarchy without Bus Encoding

This case analyzes the effects of the memory hierarchy, where the first level cache parameters are fixed, and the second level cache parameters vary. In this case no bus encoding is provided and the system architecture is depicted in Figure 1.

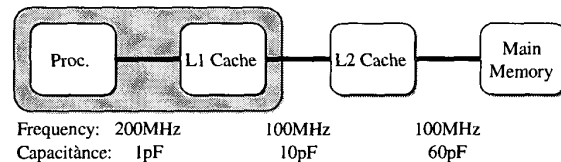


Figure 1: Architecture to evaluate the cache effects (3.1)

The miss rate for the L1 cache has been fixed to 32.8%, while the miss rate trend for the L2 cache has been analyzed versus the cache size for two different block sizes (256- and 512-bit) and four degrees of associativity. In particular, the L2 miss rate of a two-way set-associative cache versus cache size for two different block sizes is given in Figure 2. Assuming the degree of associativity as fixed, two behaviors can be observed. First, given the block size, the miss rate decreases as the cache size increases. Second, given the cache size, the miss rate decreases as the block size increases, in fact larger blocks can take advantage of the spatial locality [11].

Other results have been obtained by reporting the L2 miss rate with respect to the degree of associativity for different cache sizes and block sizes. Two basic trends can be noted, given a fixed block size. First, for the same cache size, the miss rate is reduced when passing from direct mapped to two- and four-way, while it is constant or slightly increases from four-way to eight-way. Second, given the degree of associativity, the miss rate is reduced when the cache capacity grows.

In general, the advantage of increasing the associativity is that it decreases the miss rate. The improvement in miss rate comes from reducing misses that compete for the same location. The largest gains are obtained in going from direct-mapped to two-way set-associative, while smaller benefits can be obtained by further increasing associativity. Smaller caches obtain a significantly larger absolute benefit from associativity (especially for larger block sizes) because the base miss rate of a small cache is larger. As cache size grows, the relative improvement from associativity is constant or increases slightly; since the overall miss rate of a large cache is lower, however, the opportunity for improving the miss rate decreases and the absolute improvement in the miss rate from associativity shrinks significantly. The potential disadvantages of associativity are, in general, increased costs and slower access time.

The power trend behaves similarly to the miss rate trend. The corresponding power profiling data are shown in Figure 3 and Figure 4, for address and data buses, respectively. The figures report the bus power dissipation of the system architecture including a two way set-associative L2 cache with respect to the cache size for two different block sizes. The bus power data include the three different bus sub-sections: processor-to-L1, L1-to-L2, and L2-to-memory.

By assuming fixed the associativity and the block sizes, a power reduction occurs for both address and data buses for larger cache sizes. As a matter of fact, as the L2 cache capacity increases, the number of memory requests directly satisfied by the cache grows, whilst the number of references to the main memory decreases. Consequently, a considerable reduction of the traffic occurs on the L2-to-

memory bus, which has to switch a larger bus capacitance (60 pF) with respect to the other buses.

Given the associativity and the cache size, a power reduction for larger block sizes can be observed only for the address bus, whilst for the data bus the power is almost invariant for any block size. In fact, for larger block sizes, the number of consecutive memory locations loaded in cache increases and the miss rate decreases. Therefore, the average number of transitions (i.e. the power) of the address bus decreases for larger block sizes. The data bus behavior is quite different, since in general the data values corresponding to consecutive memory locations are distributed randomly in the same way as the data values corresponding to non-consecutive memory locations. Thus the power trend is almost the same as the block size varies.

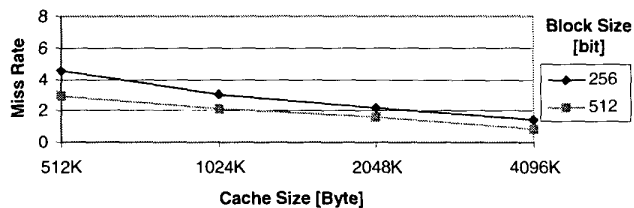


Figure 2: Miss rate of a 2-way set-associative L2 cache vs. cache size for two different block sizes.

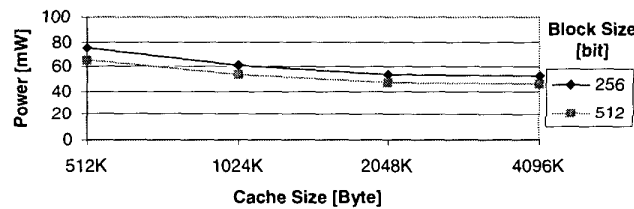


Figure 3: Power dissipation for address bus of a 2-way set-associative L2 cache vs. cache size for two different block sizes.

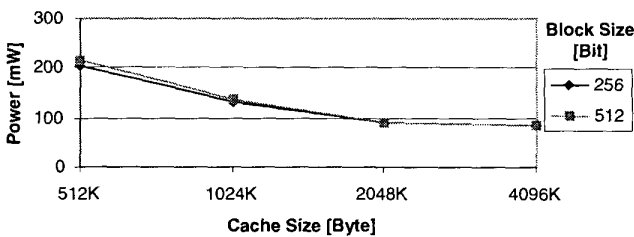


Figure 4: Power dissipation for data bus of a 2-way set-associative L2 cache vs. cache size for two different block sizes.

A comparison with the bus power dissipated by the reference architecture (380.97 mW and 511.40 mW for address and data buses respectively) has shown the power savings related to the memory hierarchy. Table 1 reports

the percentage of power saved on address and data buses with respect to the power related to the reference architecture for four different L2 sizes. The presence of the multi-level memories guarantees an average power saving of 84.72% for the address bus (higher than 79.34% in all cases), while the average power saved on the data bus is 63.66%.

Table 2 reports the percentage of power saved on address and data buses with respect to the power due to the presence of the L1 cache for four different sizes of the L2 cache. The average power saved on the address bus is 49.75%, whereas the average power saved on the data bus is 74.38%. Concerning the data bus, a comparison between Table 1 and Table 2 shows how the power savings due to the L2 cache with respect to the reference architecture (63.66%) are smaller than the savings due to the L2 cache with respect to the L1 cache (74.38%). On the contrary, for the address bus, higher savings are obtained with respect to the reference architecture (84.72% instead of 49.75%).

L2 Cache Size	% Power Saved on Address Bus vs Reference			% Power Saved on Data Bus vs Reference		
	Min.	Ave.	Max.	Min.	Ave.	Max.
512KBs	79.34	81.03	82.72	29.47	41.44	46.22
1024KB	83.08	84.58	85.77	55.85	62.96	65.44
2048KB	84.82	86.49	87.65	64.95	74.34	76.92
4096KB	85.41	86.76	87.78	68.44	75.88	77.65

Table 1: Percentage of power saved on address and data buses with respect to the reference architecture for four L2 cache sizes.

L2 Cache Size	% Power Saved on Address Bus vs L1 Cache			% Power Saved on Data Bus vs L1 Cache		
	Min.	Ave.	Max.	Min.	Ave.	Max.
512KBs	32.09	37.65	43.21	50.29	58.73	62.09
1024KB	44.39	49.30	53.21	68.88	73.89	75.64
2048KB	50.09	55.60	59.40	75.30	81.91	83.74
4096KB	52.03	56.46	59.83	77.76	83.00	84.25

Table 2: Percentage of power saved on address and data buses with respect to the L1 cache for four L2 cache sizes.

3.2. Combined Effects of On-Processor Bus Encoder and 2-Level Cache

This case targets the simultaneous influence of the on-processor bus encoder and the 2-level memory hierarchy on the system-level bus power. The corresponding architecture is shown in Figure 5, where the on-processor encoder interfaces the processor and the L1 cache. The L1 parameters are maintained fixed as in the previous cache, whereas the L2 cache size varies (512 KB, 1024 KB, 2048 KB and 4096 KB) as well as the associativity (1-2-4-8 ways), and the block size is fixed to 256-bit. Figure 6 and Figure 7 show the global bus power dissipated by a 2-way set-associative cache on the address and data buses,

respectively, versus the L2 cache size for several bus encoding techniques.

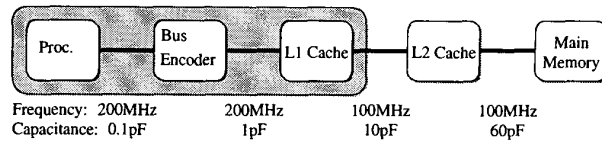


Figure 5: Architecture to evaluate the bus encoding and multi-level caching effects (3.2)

Concerning the address bus, the power dissipation is reduced by adopting the Gray, Dual_TO and Dual_TO_BI schemes, whereas the power saving due to encoding schemes are not remarkable for data buses. The average percentage of power saved by several encodings with respect to the reference architecture are reported in Table 3 and Table 4 for four L2 cache sizes and for address and data buses, respectively. As before, for each encoding scheme, larger savings are obtained for larger caches.

Table 5 and Table 6 report the average percentage of power saved by several encodings with respect to the binary one for four L2 cache sizes. Some limited advantages (max 7.88%) can be obtained by adopting the Dual_TO and Dual_TO_BI techniques for address buses. In this case, the introduction of the bus encoder provides a limited benefit, not only for data buses but also for address buses, especially if we consider the additional power required by the encoding logic.

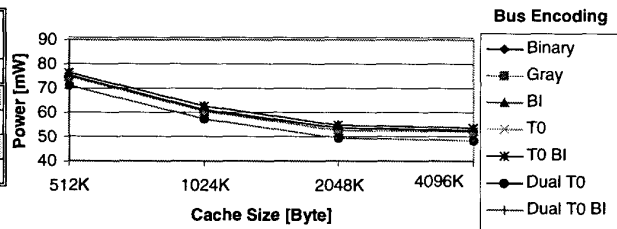


Figure 6: Power dissipation for address bus of a 2-way set-associative L2 cache vs. cache size for several bus encodings.

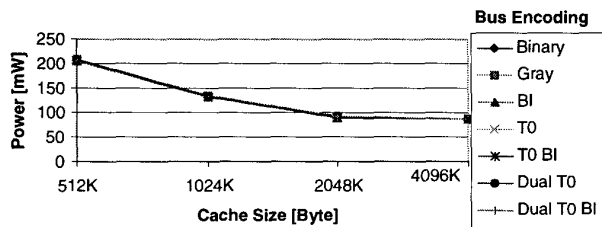


Figure 7: Power dissipation for data bus of a 2-way set-associative L2 cache vs. cache size for several bus encodings.

L2 Cache Size	Bin.	Gray	BI	T0	T0_BI	Dual T0	Dual T0_BI
512KB	79.80	79.95	79.49	79.47	79.48	80.90	80.90
1024KB	83.66	83.81	83.35	83.33	83.34	84.76	84.76
2048KB	85.75	85.90	85.44	85.42	85.44	86.86	86.86
4096KB	86.03	86.18	85.72	85.70	85.71	87.13	87.13

Table 3: Average percentage of power saved on address bus by several bus encoding techniques with respect to the reference architecture for four L2 cache sizes.

L2 Cache Size	Bin.	Gray	BI	T0	T0_BI	Dual T0	Dual T0_BI
512KB	57.96	57.55	58.00	57.63	57.97	57.63	57.70
1024KB	73.16	72.75	73.20	72.83	73.17	72.83	72.90
2048KB	81.45	81.04	81.49	81.12	81.46	81.12	81.19
4096KB	82.49	82.08	82.53	82.16	82.50	82.16	82.23

Table 4: Average percentage of power saved on data bus by several bus encoding techniques with respect to the reference architecture for four L2 cache sizes.

L2 Cache Size	Gray	BI	T0	T0_BI	Dual T0	Dual T0_BI
512KB	0.74	-1.55	-1.63	-1.56	5.45	5.45
1024KB	0.92	-1.91	-2.02	-1.94	6.74	6.74
2048KB	1.06	-2.20	-2.32	-2.22	7.74	7.74
4096KB	1.08	-2.24	-2.36	-2.26	7.88	7.88

Table 5: Average percentage of power saved on address bus by several bus encoding techniques with respect to the binary encoding for four L2 cache sizes.

L2 Cache Size	Gray	BI	T0	T0_BI	Dual T0	Dual T0_BI
512KB	-0.98	0.09	-0.79	0.03	-0.79	-0.62
1024KB	-1.54	0.14	-1.24	0.04	-1.24	-0.98
2048KB	-2.24	0.21	-1.82	0.06	-1.82	-1.43
4096KB	-2.36	0.22	-1.91	0.06	-1.91	-1.51

Table 6: Average percentage of power saved on data bus by several bus encoding techniques with respect to the binary encoding for four L2 cache sizes.

4. Conclusions

The paper analyzes the application of a model to estimate the power dissipation associated with the system-level address and data buses due to the processor-to-memory communication in a system based on the *PowerPC604e* processor and a multi-level memory sub-system. A simulation framework has been set up to demonstrate how the simultaneous presence of the bus encoding techniques and the memory hierarchy can impact the overall system power budget. A power reduction has been observed for both address and data buses for larger cache sizes, given the associativity and the block size. On the other hand, given the associativity and the cache size, a power reduction for larger block sizes has been observed only for the address bus, while for

the data bus the power is almost invariant for any block sizes. The presence of a multi-level memory sub-system implies higher power savings on the address buses with respect to data buses. The power dissipation is reduced by adopting encoding schemes for address buses, whereas the power savings due to encoding schemes are not remarkable for data buses. Globally, the reported results have shown how the model can be effectively used to configure the memory hierarchy and the system-bus architecture from the power standpoint.

5. References

- [1] Y. Li and J. Henkel, "A Framework for Estimating and Minimizing Energy Dissipation of Embedded HW/SW Systems," *DAC-35: ACM/IEEE Design Automation Conference*, June 1998.
- [2] C.L. Su and A.M. Despain, "Cache Design Trade-offs for Power and Performance Optimization: A Case Study," *ISLPED-95: ACM/IEEE Int. Symposium on Low Power Electronics and Design*, 1995.
- [3] W. Fornaciari, D. Sciuto, and C. Silvano, "Power Estimation for Architectural Exploration of HW/SW Communication on System-Level Buses," *Int. Conf. on HW/SW Codesign 1999*, Roma, Italy, April 1999, pp. 152-156.
- [4] P. Hicks, M. Walnock, and R. M. Owens, "Analysis of Power Consumption in Memory Hierarchies," *ISLPED-97: ACM/IEEE Int. Symposium on Low Power Electronics and Design*, Monterey, CA, August 1997, pp. 239-242.
- [5] L. Benini, G. De Micheli, E. Macii, M. Poncino, and S. Quer, "Power Optimization of Core-Based Systems by Address Bus Encoding," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, Vol. 6, No. 4, Dec. 1998, pp. 554-562.
- [6] L. Benini, G. De Micheli, E. Macii, D. Sciuto, and C. Silvano, "Address Bus Encoding Techniques for System-Level Power Optimization," *DATE-98: IEEE Design Automation and Test Conference in Europe*, Paris, France, Feb. 1998, pp. 861-866.
- [7] H. Mehta, R. M. Owens, and M. J. Irwin, "Some Issues in Gray Code Addressing," *GLS-VLSI-96: IEEE 6th Great Lakes Symposium on VLSI*, Ames, IA, March 1996, pp. 178-180.
- [8] E. Musoll, T. Lang, and J. Cortadella, "Working-Zone Encoding for Reducing the Energy in Microprocessor Address Buses," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, Vol. 6, No. 4, Dec. 98, pp. 568-572.
- [9] M.R. Stan and W.P. Burlison, "Low-Power Encodings for Global Communication in CMOS VLSI," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, Vol. 5, No. 4, December 1997, pp. 444-455.
- [10] C.L. Su, C. Y. Tsui, and A. M. Despain, "Saving Power in the Control Path of Embedded Processors," *IEEE Design and Test of Computers*, Vol. 11, No. 4, Winter 1994, pp. 24-30.
- [11] J.L. Hennessy, D. A. Patterson, *Computer Architecture - A Quantitative Approach*, Second Edition, Morgan Kaufmann Publishers, 1996.