Abstract—In this paper we present an in depth analysis of the power consumption in a key element of processor based systems: the cache hierarchy. This study has been carried out using a power simulator specially developed to analyze energy dissipation in memories for different target processors. We have selected an image processing example as base of the study to perform the analysis of important influences on the power consumption, as they are the compiler optimizations or the effect of data correlation. Experimental results show how this information can be very helpful for the designer in the design exploration process of power-aware embedded systems.

I. INTRODUCTION

Nowadays, power consumption has become a key parameter in the design of embedded systems. This is specially due to the kind of applications that usually run in these systems: multimedia, network processing, mobile systems, etc. In such context, the designer should consider as soon as possible energy information. For this reason, it is very important to implement power estimation tools that provide this kind of information at every stage of the design cycle. These tools should be flexible (supporting different processor platforms), robust (including architectural and technical parameters) and complete (providing information at different levels, at various structural components, when performing different operations, etc.). The designer can take advantage of such tools to evaluate the impact of design choices on the overall power budget (by experimenting with different architectural and algorithmic tradeoffs). The goal is to meet the design time deadlines, while exploring the space of possible design alternatives. Accuracy and efficiency of a high level analysis approach should be the “booster” to meet the power requirements, avoiding a costly redesign process and decreasing a key parameter, the time-to-market.

Moreover, power estimation tools can be used to simulate different implementations of an application, characterize their power consumption or analyze the influence of the variation of data or compilation optimizations [5].

Other case studies on power consumption have been presented previously, for instance, the DCT benchmark [6]. However, this work only studied the effect of the memory architecture and memory access patterns on energy consumption in an ARM based system. In [7] an approach is presented that evaluates different cache hierarchies in a PowerPC604e system. The authors pay special attention to the power dissipation in the system-level address and data buses. In [5] the influence of compiler optimizations in the power consumption of a processor is analyzed in depth. In the work we present here we can estimate power consumption in the cache hierarchy for different cache configurations and many target processors. We can also study the effect of data correlations and the various compiler optimizations that can be applied to a given source code.

The structure of the paper is as follows. First, the main objectives of this work will be presented. After that, the tool that has been used to simulate and estimate energy dissipation in the memory hierarchy will be introduced. Next, the experiments performed on the case study will be described in depth. Finally some conclusions will be drawn.

II. OBJECTIVES

In this work we present a case study on power consumption in the cache hierarchy, where the influence of key parameters (switching activity, compiler optimizations, data correlations and cache configuration) has been analyzed.

The influence of compiler optimizations on power consumption has been studied by generating different compiled versions of the benchmark (an image processing tool). On the other hand, the influence of data correlation has been studied by using two data sets with high and low correlation. Also, the influence of the switching activity factor has been analyzed by explicitly calculating this or by supposing a homogeneous distribution on the values. Finally, the effect of the cache hierarchy on the power consumption has also been evaluated by studying different cache configurations and sizes.

The experiments to characterize the power consumption on the cache hierarchy have been carried out with the help of a simulation and power estimation environment designed by
our research group [8]. This environment extends previous approaches and tools by combining a functional simulator with a power estimator, improving the accuracy of the estimation with simulated results, and automatically generating the set of co-design tools to a large set of target architectures.

The goal of our work is to simplify the design space exploration task accomplished by the designer by providing power and performance information from the very early design phases. Also, the conclusions and information extracted from the presented case study can be used to promote future power-aware compiler optimizations and cache hierarchies.

III. OVERVIEW OF THE SIMULATION/ESTIMATION ENVIRONMENT

The power estimation tool is based on CGEN (“Cpu tool GENerator”) [9], a unified framework and toolkit for writing programs like assemblers, disassemblers and simulators. CGEN is centered around application independent descriptions of different CPUs (ARM, MIPS32, SPARC...), many of them present in current embedded systems. Our work is built over the CGEN core to generate embedded system design tools with explicit information of power dissipation in caches. As CGEN successfully does with cross-design tools, the cache power estimator can be automatically generated for the whole set of available target processors in order to free the designer from this annoying task.

A. Analytical Model

In the last few years, special emphasis has been put on power estimation and high-level optimization tools attending to the designers’ demands. Some of these works are based on analytical power models that can predict with high accuracy the power dissipation in certain processor modules (cache, system clock, data path, etc).

Our tool uses the analytical power model for caches developed by Kamble and Ghose [10]. This analytical model can be very accurate but can also exhibit big deviations due to the statistical distribution of the switching activity value of the bus, that is assumed homogeneous. This is due to the lack of exact simulation results (the exact value of the switching activity can only be known when an application runs on the target architecture and the transferred data and addresses are tracked). This limitation has been solved with the design of this simulation/estimation environment.

As was previously said, in order to get an accurate estimation of cache power dissipation, the exact memory access pattern must be known to feed the analytical model of power consumption [10]. This model provides mathematical expressions for the several energy dissipation sources in the cache architecture and takes as arguments parameters of different kinds: architectural (cache size, way configuration, word line size, etc), technological (line and device input/output capacitors, etc) and statistical (switching bus activity, cache accesses, etc). One of such parameters is the switching factor ($swf$), which deserves special attention. These expressions can be deduced from the general equation for the power dissipation $E = swf \cdot C \cdot V^2_{dd}$ taking into account the particular cache architecture and the statistics for the access pattern. While the technology and architectural parameters are known from the architecture design phase, the access pattern depends on the particular data that are in use.

The switching activity factor reflects the number of bit changes at the data or address bus divided by the bus width (bit change factor). The usual approach to deal with this unavailable term is supposing half of the bits change their value, what is far away from being true when strong data or address correlation appears in the application and a reduced number of bits switches. Therefore, over and under-estimation on cache energy consumption can be explained by this limitation in the analytical model. The approach presented here overcomes this past limitation by explicitly calculating the switching activity term thanks to the information provided by the functional simulator.

B. Design Flow

The whole description of the simulation/estimation environment used for our experiments is out of the scope of this paper [8]. However, next figures present the design-flow and the cross-tool generation flow proposed to deal with the automatic generation of tools to estimate power consumption on the cache hierarchy.

Figure 1 shows the design flow corresponding to a typical cross-design environment (host and target machine are not the same) with the particularity of functional and power simulation in parallel. The designer should only provide the source code
IV. EXPERIMENTAL SETUP

The previously described environment has been used to analyze a case study on power consumption in the cache hierarchy when modifying several parameters as they are the compiler optimizations or the data correlations. The cross-design tools have been generated for a MIPS32 target processor with a single level cache hierarchy (8k for the data cache and 16k for the instruction cache) with 16 and 32 data and instruction block size, respectively.

The benchmark selected for performing the analysis has been susan, an image recognition package coming from the MiBench suite [4]. In concrete, the benchmark has been run up to completion when performing an smoothing task and an edge detection task over two different input images.

The code for susan has been compiled with the cross-version of the GCC compiler with different levels of optimizations. In this way, susan_O0 has been generated with the -O0 compiler option which means without optimizations. In the same way, susan_O1, susan_O2 and susan_O3 have been generated with the -O1, -O2 and -O3 options, which perform from the most basic transformations to the whole set of compiler optimizations (dead code elimination, loop splitting, inlining, etc), respectively.

Finally, to explore the influence of data correlations, two input images have been used for the experiments. The first one is a gray scale figure with high data correlation, while the second one is a chess board figure with a minimum of data correlation.

V. INFLUENCE OF THE SWITCHING FACTOR

As was previously said, the switching activity can drastically impact in the final estimated energy, but its value can only be known when simulation results are collected. Previous approaches overcame the lack of simulation results by supposing a homogeneous distribution in the bus activity (switching factor = 0.5). However, this simple approximation tends to over-estimate the power consumption when input data present high correlation or (what could be even worse in the design of power-aware systems) it could under-estimate the power consumption in the system when input data are strongly uncorrelated. Our tool explicitly calculates the value of this factor during the functional simulation by attending to the data transferred by the bus.

The gray scale figure has been used as input data to the susan_O3 benchmark when performing an smoothing task and power consumption results for the cache hierarchy (energy per read access, energy per write access, energy in the data bus, energy in the address bus, total energy, etc.) have been collected. This power estimation has been performed with two different approaches, the first one assuming a homogeneous distribution of the switching activity, and the other one letting the tool to calculate the exact value of the switching factor.

Figure 3a shows the results for the energy consumption per write access when an homogeneous distribution in the switching activity is supposed. On the other hand, Figure 3b shows the results for the energy consumption per write access in the cache hierarchy when the switching activity is explicitly calculated. As can be noticed, the first approximation considerably over-estimates the energy consumption per access since it does not take into account the correlation in the input data, which conducts most of the time to a switching factor lower than 0.5. Also, it is shown in Figure 3b how the data correlation presents much more impact on the second part of the benchmark (manipulation of the image) than on the first part (load of the image into the cache). However, the assumption of homogeneous distribution on the data correlation is not able to reflect this effect.

The over-estimation on the energy consumption also affects
to the total energy consumption. While the homogeneous assumption estimates a total energy consumption of 199.63 mJ for the whole running, the exact calculation of the switching activity reduces this value to 36.266 mJ (81.83% reduction).

VI. INFLUENCE OF DATA CORRELATION

Once the influence of the switching activity has been proved, the rest of experiments use the exact calculation of this factor. In this set of experiments, the benchmark susan_O3, performing the edge detection task, has been run with the gray scale figure and the chess board figure as input data.

Figures 4a and 4b show the total energy consumption when the input data is the gray scale figure or the chess board figure, respectively, where the X-axis represents the time evolution of the cache accesses (every 1000 accesses). As can be noticed, for a same size input², the low data correlation of the chess board figure impacts on power consumption by powering this from 4.014 mJ to 7.1615 mJ (78.41% increase) for the whole running. It is also shown how the edge detection task for the chess board image is more computationally intensive than the same for the gray scale image (where less edges can be found), and that is reflected in the increase of the number of cache accesses.

Most of the energies involved in the total energy consumption (i.e. energy per read access, energy per write access, energy consumed in the data bus, energy consumed in the address bus, etc.) are affected by the switching factor value (and, consequently, by the data correlation). However, energy consumed in the data bus is the energy factor which most aggressively changes with strongly correlated input data. This is easy to understand since the data bus presents a strong relation with the switching factor (reducing the switching factor decreases the number of bit commutations at the data bus), and quickly reduces its energy dissipation when the switching activity becomes smaller. In this example, total energy consumed in the output data bus during the whole running increases from 1.401 mJ to 2.320 mJ (65.59% increase) when the strongly uncorrelated input image is used, which represents a 29.19% of the variation in the total energy consumption.

VI. INFLUENCE OF COMPILER OPTIMIZATIONS

The purpose of this set of experiments is to analyze the effect of compiler optimizations on the energy consumption. In this case, the two input images and the four compiled versions of the benchmark performing the smoothing task have been used.

The compiler optimizations have a dramatic impact on the total energy consumption by reducing the number of cache accesses. However, these compiler transformations also increase the frequency of cache accesses; therefore, the power (as opposed to energy) per access is increased.

Figure 5a shows the total energy consumption when the susan_O0 benchmark (without compiler optimizations) is used with the gray scale image, while Figure 5b corresponds to the total energy consumed by susan_O3 with the same input image. As can be noticed, the whole running of the opti-
mized benchmark is less energy-consuming than the optimized benchmark. The total number of cache accesses are highly reduced with the compiler optimizations reducing in this way the total energy consumption from 12.892 mJ for Susan_00 to 4.014 mJ for Susan_O3 (68.86% reduction).

This effect is the same for other energy consumptions as the energy consumed per write access but, in this case, the energy savings are even more representative (2434.2 µJ is reduced to 34.942 µJ in the Susan_O3 benchmark, 98.56% reduction). Benchmarks where the cache accesses were dominated by the write accesses would also exhibit this behavior.

Different optimization levels (-O1, -O2) have also a big impact on total energy consumption, and this is only noticeable during the processing of the image. The benchmark we have selected (edge detection task of an image by the Susan package) has high processing complexity with respect to the load of the input image into the cache hierarchy. Other algorithms with complex processing of the input data would also present this strong dependence of the energy consumption with the optimization levels. Table I shows the values of total energy consumption for different accesses, input images and compiler optimizations; we will analyze some of the dependencies next.

As can be seen in the table, the total energy consumption has an slight dependence with the compiler optimization level, reaching the minimum consumption for a specific optimization level which is not the same for both input data. Power consumption on the data bus and write access are very dependent on compiler optimizations; therefore, benchmarks where cache accesses are limited by these two kind of accesses would benefit from aggressive compiler optimizations. However, as was previously shown, if the design goal is to reduce instantaneous power dissipation (not battery operated devices but thermally constrained), compiler transformations have a malicious effect on this, even though there is a decrease on the number of cache accesses and the execution time. Also, the results presented in the table encourage the design of new compiler optimizations; we will analyze some of the dependencies next.

Finally, figure 6 shows how energy savings due to compiler optimizations are more effective when high correlation is found in the input data (regarding the baseline power consumption), as the darker bars show for the gray scale image.

VIII. INFLUENCE OF THE CACHE HIERARCHY

The configuration of the cache in one or two levels, the size of the memory banks and the use of a fully-associative or a set-associative cache, has a strong relation with the performance (number of cache misses and memory accesses) as well as the power consumption.

To evaluate the effect of cache configuration in the power consumption of the system, four different memory hierarchies (all suitable for embedded systems) have been analyzed (table II). For these configurations, the power dissipation has been evaluated. Also, the performance of these hierarchies in terms of clock cycles has been measured. Figure 7a represents the absolute energy consumption in the cache configurations and benchmark compilations, while Figure 7b shows the execution time increase or reduction for the same cache hierarchies.

As can be noticed in these figures, the power consumption is very dependent on the cache hierarchy. In general terms, bigger memory size (both L1 and L2 levels) implies higher energy dissipation as usually happens in memory devices. However, looking for smaller memories to save power can negatively impact on performance increasing the execution time by cache access misses and fetches from the main memory (CONF 4 is the best configuration in terms of power consumption but it also presents the worst performance behavior). The designer has to trade off both metrics (power and execution time) by

---

**TABLE I**

**ENERGY VALUES (µJ)**

<table>
<thead>
<tr>
<th></th>
<th>Gray scale</th>
<th>Chess board</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-O0</td>
<td>-O1</td>
</tr>
<tr>
<td>Latches</td>
<td>180.68</td>
<td>38.067</td>
</tr>
<tr>
<td>Precharge</td>
<td>6.309.9</td>
<td>15.696.6</td>
</tr>
<tr>
<td>Input address bus</td>
<td>100.62</td>
<td>31.496</td>
</tr>
<tr>
<td>Output address bus</td>
<td>0.25945</td>
<td>0.077614</td>
</tr>
<tr>
<td>Output data bus</td>
<td>3777.83</td>
<td>4997.99</td>
</tr>
<tr>
<td>Write access</td>
<td>2434.2</td>
<td>599.69</td>
</tr>
<tr>
<td>Total</td>
<td>12892</td>
<td>7263.2</td>
</tr>
<tr>
<td></td>
<td>L1 data</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td></td>
<td>Size</td>
<td>Block Size</td>
</tr>
<tr>
<td>CONF 1</td>
<td>8k</td>
<td>16</td>
</tr>
<tr>
<td>CONF 2</td>
<td>8k</td>
<td>16</td>
</tr>
<tr>
<td>CONF 3</td>
<td>64k</td>
<td>16</td>
</tr>
<tr>
<td>CONF 4</td>
<td>256</td>
<td>16</td>
</tr>
</tbody>
</table>

TABLE II
CACHE Hierarchies

Switching activity, input data correlations and compiler optimizations have been identified as high-level factors with strong influence on power consumption. Interesting results have been obtained when tuning these factors, and power dependencies have also been found. The results and conclusions collected from this work enable the design of new compiler optimizations oriented to low-power issues, outline the utility of taking advantage of inherent data correlations, and validate the increased accuracy of our estimator which associates the functional simulation with the power estimation. Furthermore, the presented methodology and tool simplifies the design space exploration task by providing functional and power information from the very early design phases and speeds up the full design flow.

ACKNOWLEDGEMENTS

This work was supported by the Spanish Ministry of Science and Technology under contract TIC2000-0583-C02-02.

REFERENCES


Fig. 7. Modifications in the cache hierarchy: a) Energy reduction; b) Execution time attending to this information during the design specification phase.

IX. CONCLUSIONS

One of the most influencing factors in this power consumption is the energy dissipation in the cache hierarchy, which has become one of the hot-topics in recent low-power research works. Although there has been significant progress in low-power circuit design and low-power CAD and some work in low-power microarchitectures, there has been little work to date at analyzing which high-level transformations and data characteristics influence this power dissipation.

In this paper, a detailed case study on cache power consumption has been presented. The design and implementation of a simulation/power estimation environment, fully retargetable to modern processor architectures, has allowed us to perform such in depth characterization. Previous works were not able to focus on real architectures or perform this detailed study.