Suitable cache organizations for a novel biomedical implant processor

Christos Strydis
Computer Engineering Lab, Delft University of Technology, P.O. Box 5031, 2600 GA Delft, The Netherlands
E-mail:christos@ce.et.tudelft.nl

***Abstract*— This paper evaluates various instruction- and data-cache organizations in terms of performance, power, energy and area on a suitably selected biomedical benchmark suite. The benchmark suite consists of compression, encryption and data-integrity algorithms as well as real implant applications, all executed on biomedical input datasets. Results are used to drive the (micro)architectural design of a novel microprocessor targeting microelectronic implants. Our profiling study has revealed a L1 instruction-cache of 8 *KB* size (when relaxed area constraints are imposed) and a L1 data-cache of 4 *KB* size, both structured as 2-way associative caches, as optimal organizations for the envisioned implant processor.**

I. Introduction

Biomedical microelectronic implants have been around for more than 50 years. Perhaps the most well-known instance of such devices is the implantable pacemaker which, apart from saving lives, has acted as a catalyst on the general public closed-mindedness against biomedical implants. Indicative of the penetration and impact pacemakers have achieved is the fact that, in the U.S. alone, a total number of 180,000 implantable pacemakers have been registered for the year 2005 (source: American Heart Association [1]). Overall, biomedical implants are now being designed for an expanding range of applications.

In a world where clinical healthcare costs are increasing and population is aging, implant applications are expected to multiply in the years to come. A future where people are moving around performing their everyday tasks while tiny implants are monitoring or assisting their body in various ways is not so distant. Implants are expected to monitor and log biological data in-vivo and, depending on the application, to act on those readouts by regulating some physiological quantity in the body e.g. to release insulin to the blood stream when high blood-glucose levels are detected.

With a market finally mature enough to embrace implants and the technological innovations of late to support them, implant designers are slowly changing their approach. Already established product cases such as the family of pacemakers introduced by Medtronic [2], where previous design expertise is (re)used to enhance the next device version, are currently the exception. It has come to our attention that implant design has been largely custom-based; that is, implants have been developed as ASIC circuits tightly fitting the application requirements at hand.

However, this is nowadays changing with implants moving from custom-designed, application-specific (e.g. FSM-based) systems [3], [4], [5] to more generic and software-based

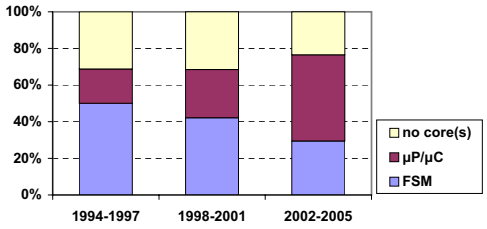

Fig. 1. Relative distribution of implant-core architecture types over the last 12 years (Source: [9]).

($\mu P/\mu C$-based) ones [6], [7], [8]. This trend has been well-studied [9] and is depicted in Fig.1. What the figure tells us is that implant-processor design is becoming more streamlined and structured than it used to be and that in the near future implant functionality will be based on executed software (written in some high-level, established language like C) rather than pure, hardwired circuitry.

The observed rapid expansion of implant applications in the years to come calls for a formal, standardized way of designing future implant architectures. Our long-term work focuses on designing a novel, minimalistic, low-power processor suitable for a large subset of biomedical applications as the ones mentioned above. We are currently at the process of defining the architecture of such a digital processor. In this paper, we profile various instruction- and data-cache organization alternatives for this processor against metrics of performance, power, energy and area. We, then, select the ones with the best characteristics for the targeted application domain. We, thus, offer insights on the design and implementation of the cache subsystem of the targeted processor. Concisely, the contributions of this work are:

- With respect to a given collection of typical and representative biomedical workloads, to specify optimal I- and D-cache geometries under performance, power, energy and area constraints;
- To offer original, quantitative data on the behavior and specifications of I- and D-caches for current and future implant processors; and
- To propose a sound methodology and toolset for selecting optimal I- and D-cache geometries for different biomedical (or other) workloads.

The rest of the paper is organized as follows: section II gives an overview of related works in the field. Section III provides the details of our selected input datasets, application benchmarks as well as the profiling testbed used. Section IV contains, in detail, the findings of this work. Overall conclusions and future work are drawn in section V.

978-1-4244-2658-4/08/$25.00 ©2008 IEEE

II. Related work

So far, extensive work has been put in identifying and profiling common applications to be executed on the targeted implant architecture. Algorithms for lossless data compression [10], symmetric-key encryption [11] and data integrity as well as representative real-world applications have been evaluated and suitable candidates have been isolated. Moreover, a carefully selected benchmark suite for microelectronic implants has been proposed [12], based on the profiled applications, to guide and assist future implant design. This benchmark suite has been shown to offer diverse program behaviors and, thus, be able to capture corners of our design space. We build on our previous work by using it in our following exploratory study on suitable cache organizations.

Besides, a significant body of prior work has been published on cache behavior with respect to traditional metrics (e.g. cache misses) as well as recent ones (e.g. power or energy). Fornaciari et al. [13] have proposed a design framework for fast exploration of energy and performance constraints (ED metric) at the system level. Their framework, among others, supports the investigation of I- and D-cache configurations of different cache sizes, block sizes and associativity. Its applicability is limited by the fact that a complete specification of the processor core is needed, which is not available in our case, yet.

Hicks et al. [14] present an exhaustive analysis of power consumption in caches when varying all cache configuration parameters. Unfortunately, their working dataset has been a subset of SPECint92 benchmarks which does not apply in our case biomedical implants. Kamble and Ghose [15], on the other hand, take a different approach and propose analytical energy models for caches but their work is not applicable in our case for the same reason as that of Hicks et al. Givargis et al. [16] have evaluated the power consumption of various cache and bus architectures with parameterizable characteristics. Su and Despain [17] have performed a case study on power-performance trade-offs for various conventional and new cache designs targeted for low power. Shiue and Chakrabarti [18] investigate suitable cache configurations for low-power embedded systems. They correct and improve on the Kamble-Ghose and Hicks analytical models and propose algorithms for finding optimal configurations.

A problem with the above works is that caches are studied in isolation from the rest of the system and, thus, no overall performance behavior is attached to the various power figures, while information about the interplay between different cache configurations and other components of a processor core cannot be acquired. The work presented here is original in that it targets a different class of low-power devices with particular idiosyncrasies. To the best of our knowledge, no similar effort has been reported so far in explicitly studying cache structures for an implant processor. Further, we have considered aspects of performance and power but also energy and area in our study, to drive our selection process.

III. Experimental setup

In order to correctly set up our experiments as well as to select suitable cache geometries, to be discussed in the following section, we first elaborate on the particular idiosyncrasies of microelectronic implants. Such implants are highly resource-constrained devices. The (re)implantation frequency for battery replacement - a costly and risky undertaking - is directly related to the operational life of a device. In order to achieve long in-vivo operation times, we are aiming at a **tight power budget** (μW order of magnitude).

An **ultra-small form factor** is also required for such devices considering the space available for implantation inside the body. This means that available processor area is also limited. Besides there are further aspects benefitting from low transistor counts (but out of the scope of this paper) such as higher device yield, increased testability and higher coverage for fault-tolerant design.

There has been shown to exist [19] and we are targeting a significant category of biomedical applications displaying **moderate performance** requirements, e.g. a feedback loop periodically regulating the functionality of bioactuators based on readouts from biosensors. Even so, under tight power and area budgets, the implant still has to complete its real-time (repetitive) duties within specific time margins. To do so, it must maintain a minimal instruction rate under the worst-case scenario.

Typical biomedical readouts are often highly periodic signals (e.g. heart beat) or stable signals (e.g. blood temperature) which can, under specific circumstances, display gradual or abrupt changes in value (e.g. a sudden muscle contortion). We have collected and used various workloads, representative of such behaviors, capturing both stable as well as rapidly changing patterns. The original **workloads** have been provided from the BIOPAC (R) Student Lab PRO v3.7 Software. Paper-size limitations do not allow for an extensive description of the various workloads; for the work presented here, we have used a biological dataset containing 10 KB of ECG data and representative of all examined workloads. Reported literature [19] and an extensive study [9] on implants have revealed that typical memory sizes inside the implants range from 1 KB to 10 KB; thus, the use of $10-KB$ ECG data. More particularly, they have revealed instruction-memory sizes in the $10-KB$ locus and data-memory sizes in the $1-KB$ locus.

Eight **benchmark applications** have been used to evaluate different cache configurations. They comprise the ImpBench benchmark suite [12] and consist of *lossless data compression* algorithms, *symmetric-key encryption* algorithms and *data-integrity* algorithms as well as synthetic *real applications*). The benchmarks are reported in Table I for convenience; they represent common tasks in present and future implant applications and also exhibit varied characteristics.

Our cache evaluation study has been based on the XTREM [26] **simulator**, a modified version of SimpleScalar [27]. The XTREM simulator is a cycle-accurate, microarchitectural, power- and performance- functional simulator for the

Benchmark	name	size (KB)
Compression	miniLZO [20]	16.3
	Finish [21]	10.4
Encryption	MISTY1 [22]	18.8
	RC6 [22]	11.4
Data integrity	checksum [23]	9.4
	CRC32 [24]	9.3
Real applications	motion [25]	9.44
	DMU [7]	19.5

TABLE I

IMPBENCH BENCHMARKS.

feature	value
ISA	32-bit ARMv5TE-compatible
Pipeline depth	7/8-stage, super-pipelined
Datapath width	32-bit
RF size	16 registers
Issue policy/Instr.window	in-order/single-instruction
I/D-Cache, L1 (separ)	2B/block, **1-cc hit/170-cc miss latency**
BTB/TLB	2-entry direct-mapped/1-entry
Branch Predictor	2-bit Bimodal (32-entry ret. addr. stack)
Write/Fill Buffer (separ)	2-entry/2-entry
Mem. bus width	1 Byte (1 mem. port)
INT/FP ALUs	1/1
Clock freq.	2 MHz
Implem. tech.	0.18 μm @ 1.5 Volt

TABLE II

XTREM (MODIFIED) ARCHITECTURE DETAILS.

Intel XScale core [28]. It models the effective switching node capacitance of various functional units inside the core, following a similar modeling methodology to the one found in Wattch [29]. XTREM has been selected for its straightforward functionality but mostly for its high performance- and power-modeling precision. It exhibits an average performance error of 6.5% and an average power error of 4%.

Many of the XScale architectural features have been integrated into XTREM. Thumb instructions and special memory-page attributes are not supported but they do not affect simulation results since they are not used by our benchmarked applications. XTREM allows monitoring of 14 different functional units of the Intel XScale core: Instruction Decoder (DEC), Branch-Target Buffer (BTB), Fill Buffer (FB), Write Buffer (WB), Pend Buffer (PB), Register File (REG), Instruction Cache (I$), Data Cache (D$), Arithmetic-Logic Unit (ALU), Shift Unit (SHF), Multiplier Accumulator (MAC), Internal Memory Bus (MEM), Memory Manager (MM) and Clock (CLK). However, to better match our application field and, also, to isolate cache behavior as much as possible, many of XTREM's architectural parameters have been cut down or disabled to better reflect the highly constrained implantable processors. The modified XTREM characteristics are summarized in Table II. Performance/power figures have been checked and scale properly with the changes.

IV. PROFILING ANALYSIS

The XScale core (and thus XTREM) assumes a Harvard architecture, with separate L1 I-cache and D-cache and no L2 caches so as to relax the bandwidth requirements on the memory bus. Most implantable systems we have studied so far feature separate caches (or memories, in general), and thus we have limited our study to split caches as well.

To perform a thorough but realistic investigation of cache sizes for biomedical implants, we have evaluated sizes in

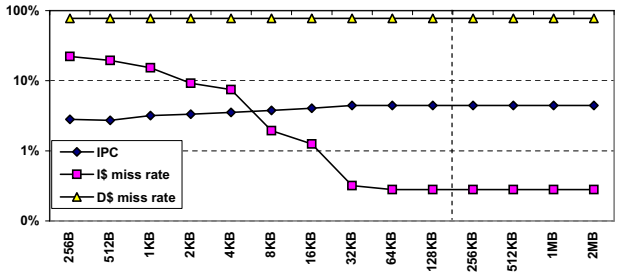

Fig. 2. Averaged, average IPC and I/D-cache miss rates for various direct-mapped, I-cache sizes.

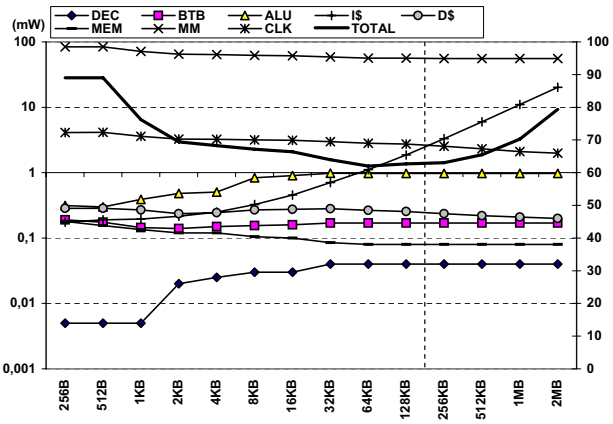

Fig. 3. Averaged, total (right axis) and per-component (left axis) average power consumption (in mW) for various direct-mapped, I-cache sizes.

the range: [32B, 16KB], in accordance with our prior study of existing implantable devices. However, as seen in Table II, XTREM simulates a 32-bit wide architecture which is unrealistic for the ultra-low-power processor that we are targeting. By conservatively assuming an average size of 8-bits for our implant-processor ISA, we had to scale up by a factor of 4x, to move from 8-bit to 32-bit quantities (which are supported by our simulation testbed). Further, since the minimal block size supported by XTREM is 2, the scaling factor becomes 8x. In effect, the properly scaled, final cache-size range becomes: [256B, 128KB]. We are well aware that the final, scaled-down findings might be suboptimal when mitigated to our actual implant processor however they will give us useful hints and a good starting point for further architectural design-space exploration. All 8 benchmarks have been profiled against each cache size and average values are reported. Unless otherwise stated, all average values in fact are median values since we cannot guarantee normal distribution of the data in the general case.

A. Cache sizes

The first step in our methodology involves finding the pair of optimal L1 I- and D-cache sizes under constraints of performance, power, energy and area. First, we have kept D-cache size constant at 32 *KB* and we have gradually increased I-cache size from 256 *B* to 128 *KB*, each step featuring double the size of the previous one. Both caches have been configured as direct-mapped structures for this step. Figure 2 illustrates the variation of IPC and I/D-cache miss ratios as a function of I-cache size. The figure actually plots also

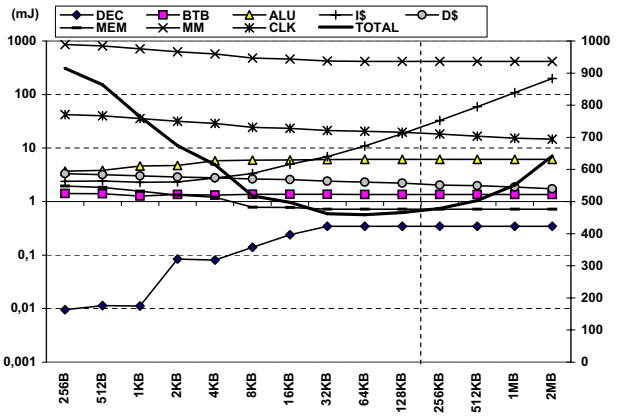

Fig. 4. Averaged, total (right axis) and per-component (left axis) energy budget (in mJ) for various direct-mapped, I-cache sizes.

larger cache sizes to give a better overview of the observed trends, but such excess sizes are not considered as viable for our application domain.

Expectedly, the D-cache miss rate is not affected by the I-cache size, while the I-cache miss rate drops rapidly and practically assumes a constant miss rate at 32 *KB* and onwards. This affects the IPC which assumes a constant value at the same point. This comes as little surprise since each benchmark in our collection (see Table I) essentially fits in the I-cache for sizes of 32 *KB* or more. However, it is interesting to observe that the IPC value does not, in an overall, change drastically with improving miss rates (viz. from 0.027% it saturates at 0.044%).

The next metric we examine is average power consumption. In Fig. 3, total and per-component power figures are plotted for the investigated I-cache sizes. XTREM components with zero power consumption have been omitted from the plot. We can readily see that, while the I-cache power increases exponentially with size, it is one to two orders of magnitude smaller than that of the main power culprit; the MM unit. The decoder and ALU components present the most notable increase in their power profile with increasing I-cache size, in response to the increased IPC, while the clock, D-cache and memory bus in fact display dropping power trends. Overall, average power consumption in the processor reaches a minimum for an I-cache size of 64 *KB*.

Apart from average power consumption, for embedded systems with a very constrained energy budget such as implants are, it is also important to examine the overall energy spent by the processor for executing all assigned tasks. Energy has been shown to depend heavily on execution time and, thus, energy plots are not necessarily identical to power plots. In Fig. 4 overall energy budgets for different I-cache sizes are plotted. Energy profiles in this case are similar to the power profiles with the minimum again observed for the $64-KB$ case. However, as can be observed from the "TOTAL" line plots, contrary to power, energy budget drops more steeply in the range from 256 *B* to 64 *KB* which makes moving to I-cache sizes smaller than 64 *KB* more attractive.

In a fashion identical to I-cache sizes, we further investigate D-cache sizes. In Fig. 5, the average IPC and I/D-cache

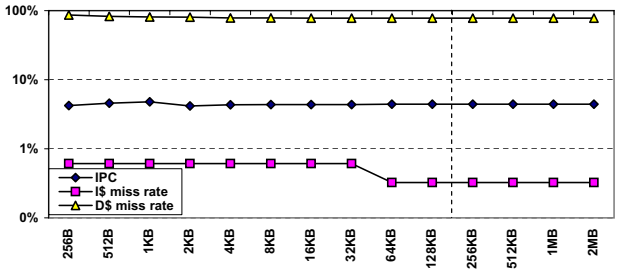

Fig. 5. Averaged, average IPC and I/D-cache miss rates for various direct-mapped, D-cache sizes.

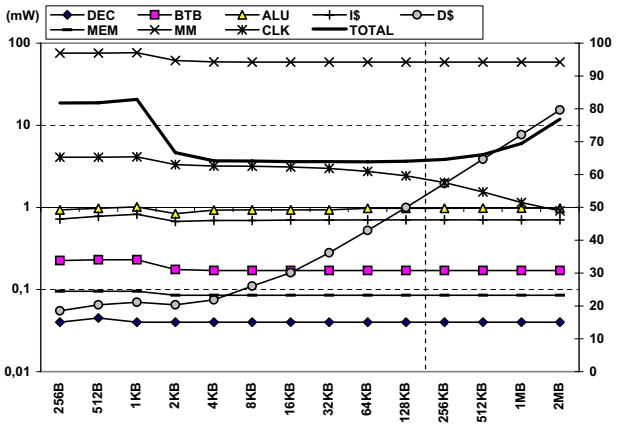

Fig. 6. Averaged, total (right axis) and per-component (left axis) average power consumption (in mW) for various direct-mapped, D-cache sizes.

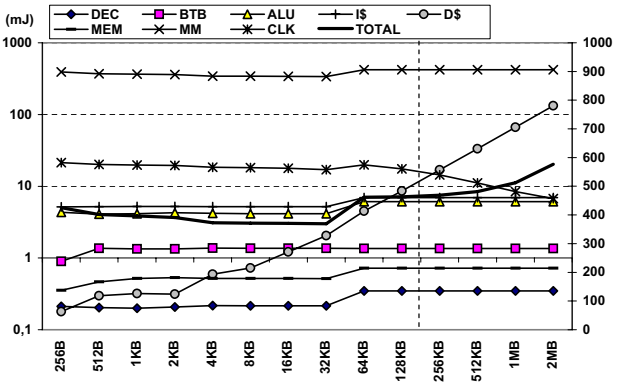

Fig. 7. Averaged, total (right axis) and per-component (left axis) energy budget (in mJ) for various direct-mapped, D-cache sizes.

miss rates for a constant I-cache size of 32 *KB* and variable D-cache sizes are plotted. Contrary to I-cache behavior, we can readily observe that increasing the D-cache size has minimal impact on its miss rate. To be precise, D-cache miss rates drop from an initial maximum of 0.863% to a final minimum of 0.776% (first observed at 512 *KB*) for our selected benchmark suite. I-cache miss rates by comparison present a *proportionally* larger drop in the locus of 64 *KB*, but in absolute terms remain roughly unaffected by the D-cache size. In effect, the IPC presents a positive spike at 1 *KB* but then stabilizes to a constant value for a $64-KB$ size and onwards.

As far as power consumption is concerned, results are plotted in Fig. 6. With the exception of the clock network and of course the D-cache itself, D-cache size increases do not affect other processor subsystems as radically as the I-cache.

TOLERANCE LEVELS		
metric	**I$-size var.**	**D$-size var.**
IPC	**1.0000**	**0.9650**
power	**1.0000**	**0.9700**
energy	**1.0000**	**0.9700**

TABLE III

TOLERANCE LEVELS FOR IPC, POWER AND ENERGY IN CACHE-SIZE OBJECTIVE FUNCTION.

The IPC spike observed in the previous figure, manifests here also as a power spike in the locus of 1 *KB*. Minimum power is located again in the $64-KB$ locus but, opposite to the I-cache case, overall power consumption drops steeply to this value immediately after D-cache sizes of 2 to 4 *KB*. A last observation is that, in an overall, I-cache size variation has a stronger impact on power than D-cache size variation.

Figure 7, last, illustrates energy results for various D-cache sizes. As was also the case with power, D-cache size variation has a smaller impact on energy than I-cache variation. However, the energy and power profiles in the D-cache case are less consistent between them. Minimum energy in this case is clearly observed in the $32-KB$ locus, followed by a steep ramp-up for larger sizes. This gives us a clear indication that, energy-wise, we should focus on D-cache sizes of 32 *KB* or less.

For selecting the best sizes for the I-cache and D-cache structures, we have based our evaluation on performance, power consumption and energy expenditure. As a performance metric, we have chosen the IPC instead of the miss rates since we do not wish to study the caches in an isolated environment but, rather, we wish to capture overall system performance as a function of cache size. That is why we have used a processor (rather than cache) simulator as our testbed. For the very same reason we have also used total average power consumption and total energy budget as our second and third metric, respectively. To find optimal solutions, we have used the following formula as our objective function for minimization:

$$F(x) = IPC_{PD}(x) + P_{PD}(x) + E_{PD}(x), \quad (1)$$

where x represents a single cache-size node. Each term $VAR_{PD}(x)$ represents the *percentage difference* between the VAR value at node x and the best VAR value across all cache-size nodes (maximum value for IPC, minimum value for power and energy). This percentage difference is given by the formula:

$$VAR_{PD}(x) = \frac{|VAR(x) - VAR_{OPT}|}{(1/2) * (VAR(x) + VAR_{OPT})} * 100, \quad (2)$$

where $VAR_{OPT} = max(VAR(x))$ or $min(VAR(x))$, with x in the range [256B,128KB]. We have chosen to use percentage differences in our objective function (1) so as to normalize all involved variables by calculating their "relative" deviation from the per-case optimal value.

We initially sought a cache size that optimizes all three imposed metrics. For the case of the I-cache, the size of 64 *KB* (or 8 *KB* for our targeted implant processor) gave the best results across IPC, power and energy. This is reflected in Table III which shows tolerance levels of 1.000 for all metrics; in essence, no compromises had to be made in the decision.

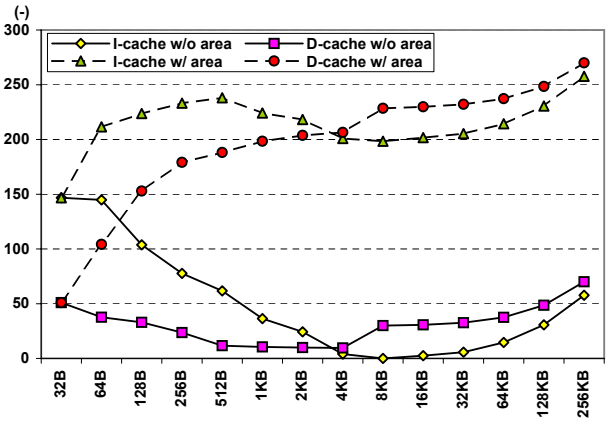

Fig. 8. Results for various I- and D-cache sizes of objective function (1).

Some commenting on this result is needed here. It is obvious that we have avoided including an area metric in the objective function above. The reason for this is the following: Area doubles with each increasing node and this represents a large percentage difference resulting in the "optimal" value for the area to be the very first node (smallest size). Further, due to this doubling of values, the area metric becomes dominant compared to the other three metrics which are changing slowly by comparison. In effect, the objective function would be strongly dominated by the area metric, returning the smallest cache size as the optimal one. At this point, we do not have a specific upper bound for the overall (and thus cache) size of our targeted processor nor can we make any educated guess on the weight of the area metric in the above optimization function. Thus, we chose to omit the area metric from this part and conclude that, under no area constraints, for the given representative benchmark collection, the optimal I-cache size is 8 *KB* for our processor.

However, for the D-cache case optimal results are not directly related to program size and, what is more, are more dispersed, as has been also observed in the previous discussion on power and energy profiles. In this case, we had to lower the objective-function tolerance levels up to the point that we found a valid D-cache configuration. As can be seen from Table III, a slight bias has been given to power and energy over IPC for two reasons: i) the IPC displayed insignificant variations with increasing D-cache sizes, and ii) in our targeted processor we consciously want to emphasize more on achieving low power and energy and less on performance. With the tolerance levels lowered as little as possible, the D-cache size giving the best results across all three metrics was 32 *KB* (or 4 *KB* for our implant processor). For the same reasons as for the I-cache case, the area metric has been omitted here, too. Cumulative results for objective function (1) for various I- and D-cache sizes and direct-mapping of cache data are given in Fig. 8 where the effect the area metric would have - should it be included - is also shown.

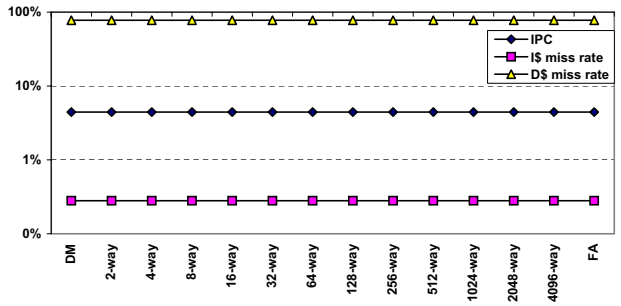

Fig. 9. Averaged, average IPC and I/D-cache miss rates for various I-cache associativity degrees.

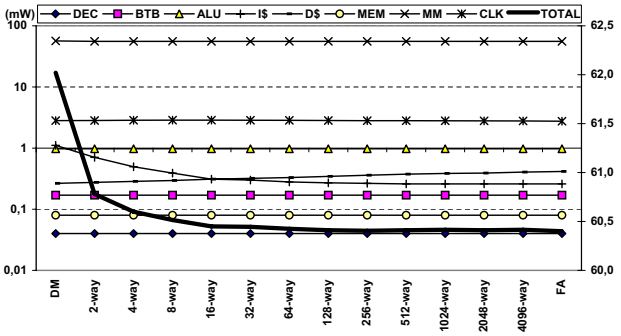

Fig. 10. Averaged, total (right axis) and per-component (left axis) average power consumption (in mW) for various I-cache associativity degrees.

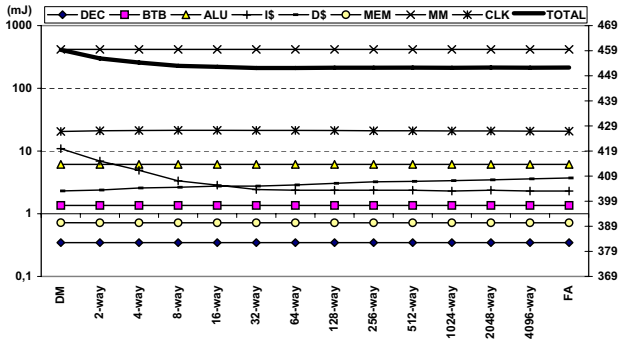

Fig. 11. Averaged, total (right axis) and per-component (left axis) energy budget (in mJ) for various direct-mapped, I-cache associativity degrees.

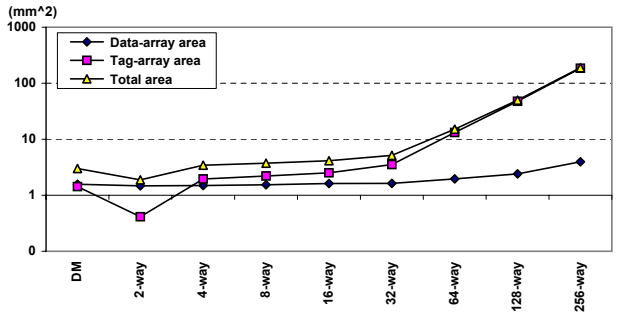

Fig. 12. Data-array, tag-array and total area (in mm^2) for various I-cache associativity degrees.

B. Cache associativity

Having selected optimal I- and D-cache sizes, we fix our simulator I/D-caches to 64 *KB* and 32 *KB* respectively and move to the second step of our study, which is the evaluation of different degrees of associativity for both structures. Starting with the I-cache, in Fig. 9 IPC and miss-rate results are plotted for various associativity configurations, ranging from direct-mapped (DM) to fully associative (FA). It can be easily observed that increasing the I-cache ways has no effect on the processor performance. Going back to Fig. 2, we can recall that with an I-cache of 64 *KB* (and onwards) the miss rate was essentially eliminated. In effect, the IPC figure here points towards a direct-mapped or few-way organization for the I-cache.

In Fig. 10, power figures are given for various I-cache ways. As expected, changing the cache associativity hardly affects the power behavior of the processor subsystems except, of course, for the I-cache itself. It is interesting to see that although required hardware area increases with the ways, overall I-cache power consumption drops. We attribute this to the way the cache is constructed (e.g. cache-line buffering etc.). In a processor employing aggressive low-power techniques such as XScale (and, thus, XTREM) is, increasing the number of ways implies reducing the number of active sets per cache access and, thus, the cache overall power consumption. On the other hand, from the same figure we can also observe a slight increase in the D-cache power when more ways in the I-cache are implemented. Given that the IPC is not notably impacted, we have so far been unable to find the reason for that phenomenon. In any case, the combined result of the above two cache trends (plus an initial drop in the MM unit) is a net decrease of the overall, average power consumption in the processor which, for the considered ultra-low-power implants we envision, is non-negligible. In effect, with 64 ways or more the I-cache power consumption settles to its overall minimum.

The energy budgets for different I-cache ways are illustrated in Fig. 11. In a similar manner to power, albeit slower, overall energy costs drop with more cache ways due to the previously discussed I-cache and D-cache behaviors. At the 32- to 64-way nodes, the processor achieves the lowest energy expenditure throughout.

In this part of our analysis, it also makes sense to consider the area cost of the I-cache when moving to more associativity ways. Since moving to a higher associativity degree (while keeping the overall cache size constant) results in a slower area increase compared to doubling the cache size (with a given associativity degree), our objective function shall show weak biasing towards the area metric. We have, therefore, properly configured and run CACTI v6.0 to collect area-utilization figures for various cache geometries. Findings for up to a realistic number of ways are illustrated in Fig. 12. We can easily observe that the global area minimum lies at an associativity degree of 2. The 4-way or 8-way configurations are also attractive alternatives with similar area costs.

We now move to investigating the optimal D-cache associativity. Figure 13 reveals that changing the associativity of the D-cache has the same marginal effect to the IPC as for I-cache. Miss rates are equally unaffected, the reason being that higher associativity does not seem to offer any *additional* speedup to the execution of the benchmarks.

As far as power consumption is concerned, Fig. 14 has been plotted. As expected, I-cache power does not change

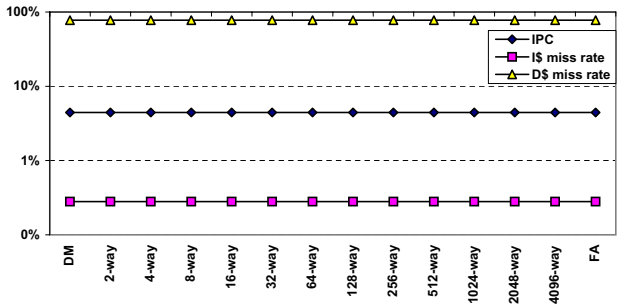

Fig. 13. Averaged, average IPC and I/D-cache miss rates for various D-cache associativity degrees.

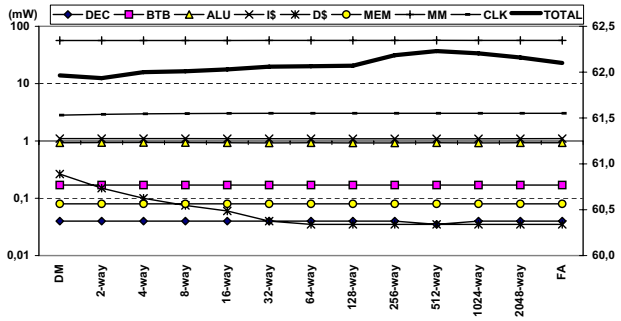

Fig. 14. Averaged, total (right axis) and per-component (left axis) average power consumption (in mW) for various D-cache associativity degrees.

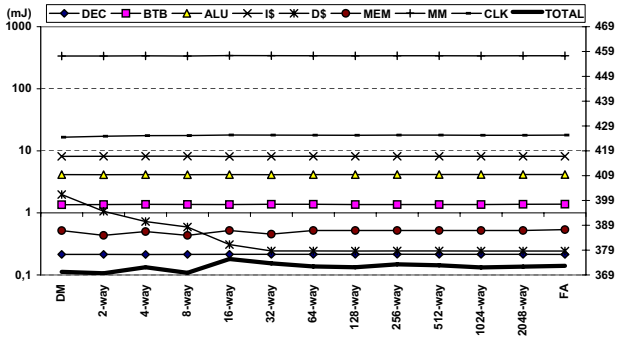

Fig. 15. Averaged, total (right axis) and per-component (left axis) energy budget (in mJ) for various D-cache associativity degrees.

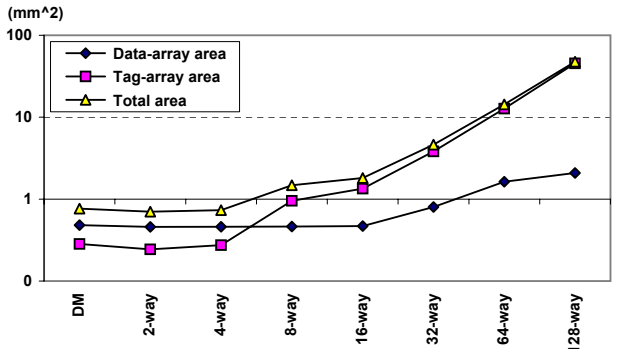

Fig. 16. Data-array, tag-array and total area (in mm^2) for various D-cache associativity degrees.

significantly with D-cache associativity, while the power consumption of the D-cache gradually drops. Overall power presents a slow climbing trend mainly due to the contributions of the clock network and the MM unit. This implies that, in the general case, less ways for the D-cache should be sought in terms of power, but the correlation is weak.

TOLERANCE LEVELS		
metric	**I$-size var.**	**D$-size var.**
IPC	**0.9985**	**0.9985**
power	**1.0000**	**1.0000**
energy	**1.0000**	**1.0000**
area	**0.9865**	**0.9900**

TABLE IV

TOLERANCE LEVELS FOR IPC, POWER, ENERGY AND AREA IN CACHE-WAY OBJECTIVE FUNCTION.

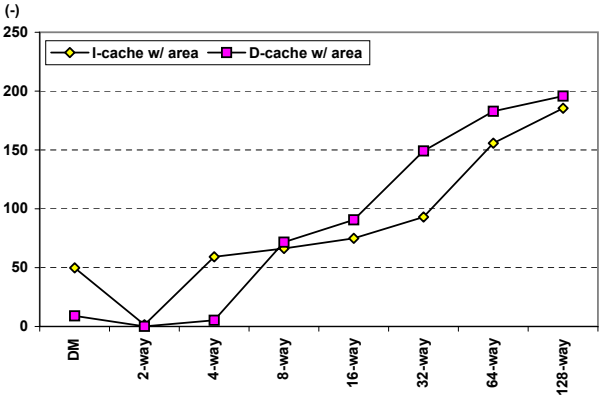

Fig. 17. Results for various I- and D-cache associativity degrees of objective function (3).

D-cache associativity versus energy cost has been plotted in Fig. 15. Observations are identical to the ones previously made for power, however in this case we witness a less uniform profile in the memory bus, the MM unit and other components, resulting in a high-energy spike at the 16-way node. This prepossesses us in favor of a D-cache design with less than 16 ways.

Last, the D-cache area cost with increasing associativity has been plotted in Fig. 16. As was the case for the I-cache, again the globally minimal area is found for 2-way associativity with the direct-mapped and 4-way alternatives being also viable choices.

For identifying the best I-/D-cache associativity degrees, we have used a cache-associativity objective function similar to (1) and percentage differences given by (2). The new objective function (3) varies only in the fact that the area percentage difference has been incorporated in the summation:

$$F(x) = IPC_{PD}(x) + P_{PD}(x) + E_{PD}(x) + A_{PD}(x), \quad (3)$$

We have, once more, favored power and energy slightly more than performance and, in this case, area. In so doing, we have acquired the best associativity degree for both the I-cache and for the D-cache to be 2-way. For convenience, cumulative results for objective function (3) for various I- and D-cache associativity degrees and fixed sizes are given in Fig. 17.

V. CONCLUSIONS

In this paper we have provided a detailed investigation of various instruction- and data-cache configurations, tested on a specially modified, low-power, cycle-accurate processor simulator. We have fed the machine with benchmarks suitable for profiling biomedical-implant applications and

have focused on performance, power, energy and area results given by the various cache configurations. We have, then, run sequential optimization (minimization) functions on the specified design space and have identified best instruction- and data-cache candidates for our end goal which is the design of an implant processor. Concisely, a 2-way L1 instruction-cache of 8 *KB* size and a 2-way associative L1 data-cache of 4 *KB* size have been selected. We are fully aware of the fact that we have scaled our simulation parameters and the produced results to reflect the targeted biomedical-implant processor. This does not necessarily mean that these results will represent optimal selections in our final processor design. However, the paper offers a sound methodology and, at the same time, defines a suitable starting point for the design-space exploration work for our envisioned processor. Besides, the methodology we have used, supported by a cycle-accurate power/performance simulator and our developed toolflow, can be used to find optimal cache configurations for different application scenarios. Updating objective functions with more variables (i.e. design parameters) is straight-forward and adjusting their contribution (weight) and tolerance levels to the optimization problem can be modified just as easily. Last, this work to our best knowledge is the first attempt to study cache behavior and geometries for the application field of biomedical implants. Based on our previous profiling study and the current work, our future work entails the full (micro)architectural specification and prototyping of our targeted biomedical-implant processor.

VI. Acknowledgements

This work has been partially supported by the ICT Delft Research Centre (DRC-ICT) of the Delft University of Technology. Many thanks are due to Kyriakos Stavrou for his help with CACTI.

References

[1] H. Ector and P. Vardas, "Heart disease and stroke statistics - 2008 Update (At-a-Glance Version)," *American Heart Association*, 2008.

[2] "Medtronic - Cardiology product list," http://www.medtronic.com/physician/cardiology.html.

[3] M. Ghovanloo and K. Najafi, "A modular 32-site wireless neural stimulation microsystem," in *IEEE Journal of Solid-State Circuits*, vol. 39, December 2004, pp. 2457–2466.

[4] P. Mohseni and K. Najafi, "Wireless multichannel biopotential recording using an integrated FM telemetry circuit," in *26th Annual International Conference of the IEEE in Engineering in Medicine and Biology Society (EMBS)*, San Francisco, CA, USA, 1-5 September 2004, pp. 4083–4086.

[5] H. Park, H. Nam, B. Song, and J. Cho, "Design of miniaturized telemetry module for bi-directional wireless endoscopy," *IEICE Transactions on Fundamentals on Electronics, Communications and Computer Sciences*, vol. 6, pp. 1487–1491, June 2003.

[6] C. Liang, J. Chen, C. Chung, C. Cheng, and C. Wang, "An implantable bi-directional wireless transmission system for transcutaneous biological signal recording," *Physiological Measurement*, vol. 26, pp. 83–97, February 2005.

[7] P. Cross, R. Kunnemeyer, C. Bunt, D. Carnegie, and M. Rathbone, "Control, communication and monitoring of intravaginal drug delivery in dairy cows," in *International Journal of Pharmaceuticals*, vol. 282, 10 September 2004, pp. 35–44.

[8] H. Lanmuller, E. Unger, M. Reichel, Z. Ashley, W. Mayr, and A. Tschakert, "Implantable stimulator for the conditioning of denervated muscles in rabbit," in *8th Vienna International Workshop on Functional Electrical Stimulation*, Vienna, Austria, 10-13 September 2004.

[9] C. Strydis *et al.*, "Implantable microelectronic devices: A comprehensive review," Computer Engineering, TU Delft," CE-TR-2006-01, Dec. 2006.

[10] C. Strydis and G. Gaydadjiev, "Lossless data compression in ultra-low-power embedded systems: An analysis," in *Submitted to International Conference on Hardware-Software Codesign and System Synthesis (CODES'08)*, Atlanda, Georgia, 19-24 October 2008.

[11] C. Strydis, D. Zhu, and G. Gaydadjiev, "Profiling of symmetric encryption algorithms for a novel biomedical-implant architecture," in *ACM International Conference on Computing Frontiers (CF'08)*, Ischia, Italy, 5-7 May 2008, pp. 231–240.

[12] C. Strydis, C. Kachris, and G. Gaydadjiev, "ImpBench - A novel benchmark suite for biomedical, microelectronic implants," in *To appear in International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS'08)*, Samos, Greece, 21-24 July 2008.

[13] W. Fornaciari, D. Sciuto, C. Silvano, and V. Zaccaria, "A design framework to efficiently explore energy-delay tradeoffs," in *CODES '01: Proceedings of the ninth international symposium on Hardware/software codesign*, New York, NY, USA, 2001, pp. 260–265.

[14] P. Hicks, M. Walnock, and R. Owens, "Analysis of power consumption in memory hierarchies," *Low Power Electronics and Design, 1997. Proceedings., 1997 International Symposium on*, pp. 239–242, Aug 1997.

[15] M. Kamble and K. Ghose, "Analytical energy dissipation models for low power caches," *Low Power Electronics and Design, 1997. Proceedings., 1997 International Symposium on*, pp. 143–148, Aug 1997.

[16] T. Givargis, F. Vahid, and J. Henkel, "Evaluating power consumption of parameterized cache and bus architectures in system-on-a-chip designs," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 9, no. 4, pp. 500–508, Aug 2001.

[17] C.-L. Su and A. M. Despain, "Cache design trade-offs for power and performance optimization: a case study," in *ISLPED '95: Proceedings of the 1995 international symposium on Low power design*, New York, NY, USA, 1995, pp. 63–68.

[18] W. Shiue and C. Chakrabarti, "Memory exploration for low power, embedded systems," in *DAC'99*, 1999, pp. 140–145.

[19] C. Strydis, G. Gaydadjiev, and S. Vassiliadis, "The case for standard processing elements in biomedical, microelectronic implants," *Submitted to IEEE Transactions on Computers - Special Issue: Chips and Architectures for Emerging Technologies and Applications*, February 2009.

[20] M. Oberhumer, ""LZO v2.0.2"," http://www.oberhumer.com/opensource/lzo/.

[21] N. de Vries, "Lossless Data-Compression Kit, LDS v1.1," http://www.nicodevries.com/nico/lds13.zip.

[22] Y. Law, J. Dourmen, and P. Hartel, "Survey and benchmark of block ciphers for wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 2, pp. 65–93, February 2006.

[23] R. Braden, D. Borman, and C. Partridge, "Computing the internet checksum," *SIGCOMM Comput. Commun. Rev.*, vol. 19, no. 2, pp. 86–94, 1989.

[24] "Cell Relay Retreat: CRC-32 Calculation, Test Cases and HEC Tutorial," http://cell.onecall.net/cell-relay/publications/software/.

[25] P. Wouters, M. D. Cooman, D. Lapadatu, and R. Puers, "A low power multi-sensor interface for injectable microprocessor-based animal monitoring system," in *Sensors and Actuators A: Physical*, vol. 41-42, 1994, pp. 198–206.

[26] G. Contreras *et al.*, "XTREM: A Power Simulator for the Intel XScale Core," in *LCTES'04*, 2004, pp. 115–125.

[27] T. Austin *et al.*, "SimpleScalar: an infrastructure for computer system modeling," *IEEE Computer*, vol. 35, no. 2, pp. 59–67, Feb. 2002.

[28] *Intel XScale Microarchitecture for the PXA255 Processor: User's Manual*, Intel Corp., March 2003.

[29] D. Brooks *et al.*, "Wattch: A Framework for Architectural-Level Power Analysis and Optimizations," in *ISCA'00*, 2000, pp. 83–94.