# Block Disabling Characterization and Improvements in CMPs Operating at Ultra-low Voltages

Alexandra Ferrerón*, Darío Suárez-Gracia†, Jesús Alastruey-Benedé*, Teresa Monreal‡, and Víctor Viñals*
*Universidad de Zaragoza, Spain, Email: {ferreron, jalastru, victor}@unizar.es
†Qualcomm Research Silicon Valley, USA, Email: dgracia@qti.qualcomm.com
‡Universidad Politécnica de Cataluña, Spain, Email: teresa@ac.upc.edu

***Abstract*—Power density has become the limiting factor in technology scaling as power budget restricts the amount of hardware that can be active at the same time. Reducing supply voltage to ultra-low voltage ranges close to the threshold region has the promise of great energy savings. However, the potential savings of voltage scaling are limited by the correct operation of SRAM cells, which is not guaranteed below $V_{dd_{min}}$, the minimum voltage in which cache structures operate reliably.**

**Understanding the effects of operating below $V_{dd_{min}}$ requires complex modeling, so we introduce an updated probability failure model of SRAM cells at 22nm and explore the reliability impact of lowering the chip voltage supply below $V_{dd_{min}}$ in shared-memory coherent chip-multiprocessors (CMP) running a variety of parallel workloads. A microarchitectural technique to cope with cache reliability at ultra-low voltages is block disabling; however, in many cases, the savings in on-chip caches do not compensate for the consumption in the rest of the system, as the consumption increase of the off-chip memory may offset the on-chip gain.**

**We make the case that existing coherence mechanisms can provide the substrate to improve energy savings with block disabling and propose two low-complexity techniques. Taking the best of both techniques we can scale voltage below $V_{dd_{min}}$ and reduce system energy up to 39%, and system energy-delay up to 10%. Besides, by lowering the CMP consumption in a power-constrained scenario, we could activate offline cores, reaching a potential speedup between 3.7 and 4.4.**

## I. Introduction

Power density has become the limiting factor in technology scaling. Technology improvements allow to increase the number of transistors and integration density following Moore's Law, but the power budget caps the amount of hardware that can be active at the same time, leading to dark silicon [1].

Voltage scaling is one of the most effective mechanisms to reduce microprocessor power consumption, as dynamic power scales quadratically with voltage. Modern systems include dynamic voltage and frequency scaling (DVFS) capabilities, and run at different predefined energy/performance points to trade-off performance for power consumption. Scaling voltage beyond to approach the threshold voltage would allow further reductions into the power consumption of the chip [2].

However, the increased severity of manufacturing-induced parameter variations at lower voltages limits voltage scaling to a minimum voltage, $V_{dd_{min}}$, below which a processor cannot operate reliably. Logic is more voltage scaling friendly than memory, as memory is sized more aggressively to satisfy high-density requirements. Thus, parameter variations particularly impact memory cells, and failures in memory structures typically determine the $V_{dd_{min}}$ of the whole processor, or implies using different voltage domains for logic and memory. For example, Intel uses 0.7V for logic and 1.05V for memory in one of its latest processors [3].

Overcoming $V_{dd_{min}}$ would allow to operate at lower voltages, improving power consumption and battery life. In the literature different solutions have been proposed to deal with faulty memory cells, such as spare and robust cells, frequency/voltage binning, error correcting codes, or remapping mechanisms to couple faulty entries in order to recreate fault-free cache blocks [4], [5], [6]. In general, these methods have a noticeable area overhead or sacrifice an important fraction of the cache capacity and associativity.

First-level caches in chip multiprocessors (CMPs) are usually private, occupy little area, and their access time often determines the processor cycle time. The use of complex remapping mechanisms may imply penalizing the access time, already critical, and the loss of cache capacity might degrade performance, as the miss rate increases. Commercial processors, such as the Intel Nehalem family, use robust cells (8T SRAM cells) in the first-level caches to improve resilience [7]. On the other hand, last-level caches (LLCs) are usually shared and have bigger size and associativity, occupying a great percentage of the chip area. Reducing LLC capacity and associativity potentially increases the off-chip memory accesses, translating into extra energy consumption that might spoil the energy savings coming from voltage reduction.

Our goal is twofold: to explore the implications of lowering $V_{dd_{min}}$ in shared-memory coherent CMPs and to provide system-conscious block disabling mechanisms enabling ultra-low voltage operation without penalizing LLC associativity.

This work makes the following contributions. First, we present an up-to-date SRAM cell failure probability model ($P_{fail}$) based on previous models, studies, and future technology predictions, taking into account emerging circuit implementations and topologies. Second, for the best of our knowledge, we provide the first evaluation of block disabling techniques in a shared-memory coherent CMP running parallel workloads with a complete and detailed memory model, including different $P_{fail}$ points and main memory. Off-chip memory energy consumption at lower chip voltages plays an important role, and not considering its contribution might lead to suboptimal design points. Finally, we introduce two low-complexity techniques that allow blocks disabled in the LLC to be present in the L1 caches. The first relies on the existing cache coherence

This work was supported in part by grants TIN2010-21291-C02-01, TIN2012-34557, TIN2013-46957-C2-1-P (Spanish Gov. and European ERDF), gaZ: T48 research group (Aragón Gov. and European ESF), and HiPEAC-3 NoE (European FET FP7/ICT 287759).

mechanism, keeping operational the tags of the disabled blocks, and so also the tracking capabilities of the coherence directory. The second technique builds upon, adding cache-to-cache service of clean copies as an alternative to main memory supply.

The rest of this paper is organized as follows: Section II presents the background and motivation. Section III comments on block disabling techniques and their impact on large cache structures. Section IV explores the implications of these techniques from the system perspective. In Section V we introduce tracking of disabled blocks as a means to lower energy consumption. Section VI describes the methodology and presents our evaluation. Section VII comments on the related work, and Section VIII concludes.

## II. Background and Motivation

Historically, scaling the supply voltage ($V_{dd}$) of MOS transistors reduced power consumption. Simultaneous frequency increments required lowering the threshold voltage ($V_{th}$). But voltage scaling has stopped because leakage current is exponentially related to $V_{th}$, and the dynamic power savings from lowering $V_{dd}$ went along with increments in static power due to leakage. Thus, $V_{dd}$ and $V_{th}$ are no longer scaling parameters, rather they are set by the results of a power optimization.

CMOS circuits operate at very low voltages, and remain functional even below the threshold voltage $V_{th}$, but concerns about failures and performance loss limit the scaling of voltage supply in commercial systems. When operating below $V_{th}$, circuit delay and leakage current increase exponentially with $V_{dd}$, discouraging $V_{dd}$ from being scaled below $V_{th}$.

Scaling supply voltage (and consequently frequency) to the near-threshold region would allow a cubic reduction in power, as dynamic power scales quadratically with voltage, and linearly with frequency. However, at lower voltages, manufacturing-induced parameter variations limit the functionality of SRAM structures. Intra-die variation has a systematic and a random component. The systematic component is typically caused by lithographic irregularities, while the random component is caused by varying dopant concentrations. Intra-die random dopant fluctuations (RDFs) play a primary role in cell failure by arbitrary impacting the number of locations of dopant atoms in transistors, resulting in different voltage thresholds for matched SRAM devices. These defective cells, randomly distributed throughout large memory structures, may prevent caches from operating below the minimum voltage $V_{dd_{min}}$. Excessive parametric variability can cause circuit behaviour consistent with a permanent or hard fault. Unfortunately, the severity of these variations gets worse with each new technology generation, limiting the potential of technology scaling [8].

Since large SRAM structures are more affected by parameter variations, determining the $V_{dd_{min}}$ of the system, it seems clear that the way towards lowering the system voltage requires to improve the reliability of SRAM.

### A. Model for Probability Failure

Next we introduce a probability failure model, $P_{fail}$, intended to capture the failure behaviour at 22nm. $V_{dd_{min}}$ is closely related to the yield of the fabrication process, but there are not much data available in the open literature for industrial production runs. For example, current implementations of SRAM at 22nm with Tri-gate CMOS technology reach $V_{dd_{min}}$ ranges between 0.7V and 0.8V, using $V_{dd_{min}}$–enhancing assist circuitry [9].

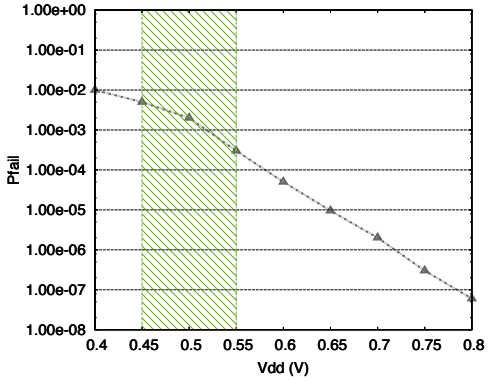


Fig. 1. Probability of failure ($P_{fail}$) estimated at 22nm.

We model the probability of fail of an SRAM cell ($P_{fail}$) based on the model of Wilkerson *et al.* for 65nm [5]. We incorporate the study from Zhou *et al.* at 32nm [10], and the predictions of Nassif *et al.* to scale down to 22nm [8]. Zhou *et al.* study six different 6T SRAM cells with different trade-offs in area and reliability at 32nm. Nassif *et al.* predicted in their resilience map, starting at 65nm, a worsening in the probability of fail of SRAM cells. However, they hold constant circuit implementations and topologies without considering new advances such as FDSOI, FinFET, Tri-gate, and gate-all-around devices with light channel doping, which have been proposed to reduce $V_{th}$ variability and thereby enable CMOS scaling. Taking all into account, we extrapolate previous models based on simulated/measured results reported by different authors, assuming a similar $V_{dd}/P_{fail}$ slope [5], [8], [10], [11], and adjust the values considering that new technological nodes incorporate modern circuit implementations and topologies. To do this, we incorporate to the model the available information from Intel at 22nm [9]. We assume 0.8V $V_{dd_{min}}$ as our baseline.

Figure 1 shows the variation of $P_{fail}$ with $V_{dd}$, according to our model for 22nm. Our objective is to reach voltages at the near-threshold region (the shadowed region on Fig. 1).

Note that, as pointed out by Zhou *et al.* [10], $P_{fail}$ may vary significantly depending on many parameters, which further complicates the search for a proper formalization of the model. At this point, our model should be seen as a collection of different design points and trade-offs between available capacity and power consumption. Our objective is to study the behaviour of a CMP under an ample $P_{fail}$ design space, rather than obtaining the maximum $V_{dd}$ reduction, which highly depends on the particular technology and fabrication process.

As in previous work, our model assumes that faulty cells occur randomly and are uncorrelated, which, as we mentioned above, corresponds to faults due to random dopant fluctuations. Hard faults can be detected with post-manufacturing and boot time tests, by tracking errors reported by error-correcting codes (ECC) during operation, built-in self-tests, or other similar methods [12].

## III. Impact of Variability on Large Caches at Ultra-low Voltages

Excessive parameter variations in SRAM cells limit the voltage scaling of memory structures to a minimum voltage below which SRAM cells may not operate reliably, and show a behaviour consistent with a hard fault.

A simple approach to mitigate hard faults is disabling faulty entries (cache entries with faulty bits, which cannot store a complete cache block). This kind of technique is called *graceful degradation technique* [13], and it is already implemented in modern processors to protect against hard faults [12].

Block disabling techniques have been studied for operation at lower voltages because of their easy implementation and low overhead [14]. From the implementation perspective, only one bit per entry suffices. This bit needs to be checked on every cache look-up. Their main disadvantage is that the amount of capacity dramatically decreases when the probability of failure grows. The probability of having a faulty entry, $P_{bf}$, where each entry has $k$ bits and each bit has a failure probability of $P_{fail}$, can be expressed as:

$$P_{bf} = 1 - (1 - P_{fail})^k \quad (1)$$

When supply voltage lowers to the ultra-low operation region or near-threshold ($V_{dd}$ between 0.45V and 0.55V), the amount of total faulty cells in the cache is not very high (less than 1%), but due to their random distribution, disabling the complete cache entry considerably reduces the available capacity (Table I).

TABLE I. Capacity available at low-voltages for 16-way, 1MB cache bank with block disabling (block size is 64 bytes).

| Vdd | Available capacity (KB) |
|---|---|
| 0.55V | 887 KB |
| 0.50V | 408 KB |
| 0.45V | 138 KB |

Block disabling results in caches with variable associativity per set, determined by the number and distribution of faults in the cache. Tracking faulty cells at finer granularity, usually combined with a remapping mechanism, offers a great increase in the effective cache capacity, and some recent proposals exploit this observation with complex mechanisms [4], [5], [15]. In this paper we focus on the simpler block disabling for shared LLCs, and we leverage cache coherence to track and manage disabled blocks in private caches.

## IV. Block Disabling for Shared Last Level Caches in Chip Multiprocessors

In this section we describe our baseline system and explore the implications of scaling the supply voltage to the near-threshold region in shared-memory coherent CMPs.

### A. Overview of the System

Our baseline system consists on a tiled CMP, with an inclusive two-level cache hierarchy, where the second level (L2) is shared and distributed among the processor cores. Tiles are interconnected by means of a mesh. Each node has a processor core with a private first level cache (L1) split in instructions and data, and a bank of the shared L2, both connected to the router (Fig. 2). Some tiles also include a memory controller. Table II shows the parameters of the baseline processor, memory hierarchy, and interconnection network.

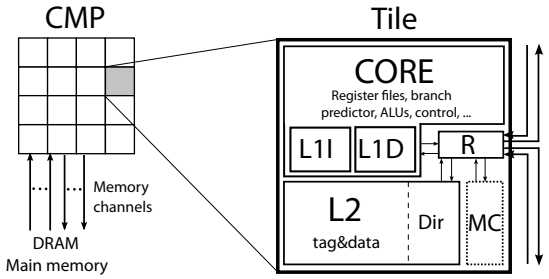


Fig. 2. Tiled CMP with 16 processors and four memory controllers.

At lower voltages near to the threshold voltage, frequency degrades about 5 to 10x [2], [16]. Our baseline processor runs at 2GHz, and we assume a linear frequency degradation to 400MHz at 0.4V. Note that the DRAM module voltage is not scaled as the rest of the system. Thus, the relative speed of main memory with respect to the chip gets faster as the voltage decreases. This model is consistent with prior work.

Our baseline coherence protocol relies on a full-map directory with MESI states. We follow a similar approach as AMD's HT assist protocol, and use explicit eviction notifications of both dirty and clean exclusively owned blocks [17]. Replacements of shared blocks are silent and, therefore, the baseline protocol does not track the exact number of sharers of a given block, and requests are served by the L2 cache (last-level cache, LLC).

L1 caches are built with robust SRAM cells and run at lower voltages without suffering from parameter variation, while L2 banks are built with conventional 6T SRAM cells and, therefore, they are sensitive to failures [7].

### B. Impact of Block Disabling from the System Perspective

From the system perspective, the interaction between the block disabling technique and the organization of the CMP might play an important role. Modern chips such as the Intel Core i7 implement inclusive cache hierarchies, as they ease the coherence management. Inclusive cache hierarchies have been pointed out as key to scale cache coherence with the number of cores in shared memory multicore chips [18]. The coherence is maintained embedding the state information of the block in the shared caches (i.e, each block of the shared cache is augmented with sharing state and a bit vector to represent the current sharers). The inclusion property enforces that every block cached in the private level is also present in the shared level, resorting to invalidate private cache blocks explicitly when required (inclusion victims) [19].

In a block disabling context, this problem is exacerbated and performance might degrade when blocks actively used in private levels are invalidated due to the reduction of capacity and associativity of shared caches. Thus, the amount of new inclusion victims depends on the associativity reduction in the cache. The number of faulty ways per set can be expressed as a probability, where $P_{wi}$ is the probability of having $i$ faulty ways (out of $n$, where $n$ is the associativity of the cache), and $P_{bf}$ is the probability of block failure from Eq. 1:

TABLE II. MAIN CHARACTERISTICS OF THE CMP SYSTEM.

| | |
|---|---|
| Cores | 16, Ultrasparc III Plus, in-order, 1 instr/cycle, single-threaded, 2GHz at $V_{dd}$ 0.8V |
| Coherence protocol | MESI, directory-based full-map distributed among L2 cache banks |
| Consistency model | Sequential |
| L1 cache | 64KB data and inst. caches, 4-way, 2-cycle hit access time, 64B line size<br>Private, pseudo-LRU replacement policy |
| L2 cache | Distributed, 1 bank/tile, 1MB/bank, 16-way, 8-cycle hit access time (4-cycle tag access), 64B line size<br>Shared, inclusive, interleaved by line address, pseudo-LRU replacement policy |
| Memory | 4 memory controllers, distributed on the edges of the chip<br>4 channels, Double Data Rate (DDR3 1333 MHz), 4Gb/channel, 8 banks, 8KB page size, open page policy |
| NoC | Mesh, 2 Virt. Networks (VN): requests and replies, 2 Virt. Chan. (VC) per VN; 16-byte flit size, 1-cycle latency hop |
| Routers | 2-stage pipeline: Routing and Input Buffer + VC Allocation, Switch Allocation + Switch Traversal<br>Round-robin 2-phase VC/Switch Allocation.; 5-flit buffers per VC |

$$P_{wi} = \binom{n}{i} P_{bf}^{i}(1 - P_{bf})^{n-i} \quad (2)$$

Fig. 3 illustrates the associativity loss for four different values of $V_{dd}$. As it can be seen, at ultra-low voltages this degradation is unsustainable.

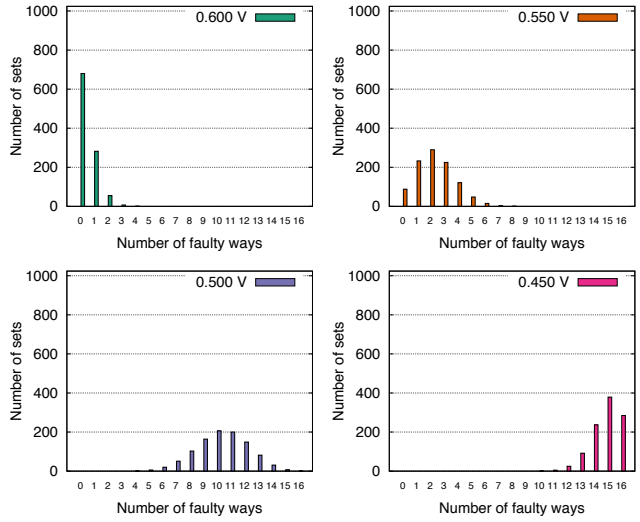


Fig. 3. Associativity loss of a 16-way set associative 1MB cache bank with 64B block size at different voltages (0.45V-0.6V).

## V. EXPLOITING COHERENCE TO ENABLE EFFICIENT ULTRA-LOW VOLTAGE OPERATION

Here, our explicit goal is to leverage a conventional two-level cache-inclusive CMP organization and assess low complexity modifications to the existing coherence protocol, leaving unchanged the cache structure, timing, and replacement.

The **block disabling (BD)** scheme simply assumes one extra resilient bit[1] per block to identify faulty cache blocks in the data array (one or more faulty bits). Faulty data blocks are excluded from tag search and replacement, involving a net reduction in associativity (Fig. 3).

This last evidence suggests that inclusive hierarchies are not optimal for systems which implement block disabling techniques in presence of a significant number of hard faults. However, from the coherence management perspective, only directory inclusion is required: lines present in the private levels only have to be tracked in the tag array, without the need of having a replica in the shared level data array [19].

This is the basis for the first technique that we call **block disabling with operational tags (BDOT)**. To keep directory inclusion, we turn on the tags of faulty blocks, just including them in the regular actions of search and replacement. Enabling the tags of the faulty blocks restores the associativity of the shared cache as seen by the first-level private caches. The tag of a faulty block, if valid, tracks a cache block that might be present in the private caches, but not in the shared cache.

To fully exploit this scheme, no failures can occur in the cells of the tag array. This can be accomplished, for example, by using robust cells, from 6T to 8T, or increasing the strength of the ECC. Tags occupy very little area in comparison to the data array (less than 10%), and increasing the cell size by 33% will end up increasing the total area by 3%. Since using sophisticated ECC could increase the access latency to the tag array, and using resilient tag cells involves little overhead, we will resort to the latter.

The coherence protocol also needs to be adapted to this new situation, where a faulty block only stores its tag and directory state. From the implementation perspective, still one resilient bit suffices to indicate whether the block is faulty (only tag stored) or not (both tag and data stored). Whenever a request to a faulty block arrives to the LLC cache bank, the request needs to be sent to the upper level (off-chip) to recover the block, and the same occurs with dirty blocks, which need to be written back to memory after being evicted in a private cache.

By protecting directory information, the increase in the number of inclusion victims disappears, as the associativity of the cache remains the same. But, still the capacity of the shared cache is compromised, with a potential increase in the off-chip traffic.

Our second proposed technique works on top of the previous one. It avoids the main memory read requests caused by faulty blocks, as long as one or more copies exist in the private caches. We call this technique **block disabling with operational tags and cache-to-cache copy (BDOT-C2C)**.

An interesting consequence of directory inclusion is that the directory information appears to be an efficient mechanism for tracking replicated blocks. Parallel workloads, which share data and instructions, could benefit from this replication. We can leverage directory inclusion to reduce the off-chip traffic by L1 caches cooperation. An L1 request to a shared block mapped to a faulty entry is forwarded to one of the sharers of

[1]Note that this bit needs to be resilient to ensure correctness.

the block (another L1). This L1 will serve the block through a cache-to-cache transfer. Cache-to-cache transfers are already implemented in the baseline coherence protocol for exclusively owned blocks. Thus, no additional hardware is needed. BDOT-C2C requires a slight modification of the directory protocol to trigger cache-to-cache communication for both faulty and evicted shared blocks, and to track the exact number of sharers of a given block. As we will show in Section VI, the increase in the network traffic is negligible.

## VI. Methodology and Evaluation

This section presents the methodology and evaluates the potential of the proposed techniques.

Regarding our experimental set-up, we use Simics [20] in combination with GEMS Multifacet's multiprocessor simulator toolset to model the on-chip memory hierarchy [21], an extended version of GARNET for the on-chip interconnection network [22], and DRAMSim2 for detailed DDR3 DRAM model [23]. To get timing, area, and energy consumption, we use McPAT framework, with 22nm technology [24].

We use a selection of ten shared-memory parallel applications from PARSEC with the *sim-medium* input. The parallel region of all benchmarks is simulated until completion [25].

In Section II-A we introduced our model of probability of fail ($P_{fail}$). We assume this probability is uniform across the chip, and leave for further studies the impact of the variation of the $P_{fail}$ levels, frequency, or leakage across the chip [16]. We create random fault maps and run Monte Carlo simulations to ensure results are within an error of 5%, and a confidence level of $(1-\alpha) = 0.95$[2]. We compute the faultiness of each memory cell randomly and independently of other cells, resulting in configurations with similar number of faults, but different locations.

Regarding the application of block disabling and its proposed enhancements, we will look into three potential markets, selecting for each one the relevant metric.

From the energy efficiency perspective, one objective seeks to obtain the minimum energy consumption of the system, for example for battery-constrained devices, such as mobile and wearable devices, which search for the maximum battery life. Accordingly, we compute the total energy consumption (joules) to execute the application's parallel region.

Another objective is to reach a trade-off between energy reduction and performance sustaining, for example, to execute background tasks in data centres, reducing the energy bill without noticeable performance degradation. In this context Energy-Delay product (ED) weights equally energy and performance. When DVFS is the primary power-performance trade-off controller, the power reduction is cubic in relation to the voltage reduction, while frequency degradation (assuming no increase in the work needed to execute the task) is linear to voltage scaling (frequency). Thus, voltage/frequency scaling would always improve ED. However, this is true if only dynamic power is taken into account. Current technologies suffer from increasingly higher leakage power, and this static energy consumption is proportional to the delay of the program, breaking even for ED. So, both from a data centre manager and a computer architect perspective, ED seems a reasonable metric to look into.

Finally, in a third scenario we can search for minimum CMP energy, to be able to wake up sleeping cores in power-constrained CMPs (dark or dim silicon scenarios) [1]. Here the goal is to decrease execution time or increase throughput in a high-performance environment. Indeed, if we know how to perform a task with a given set of cores with less energy (and more delay), the average power is going to decrease. If the saved power is used to turn on offline cores, the net effect is a performance increase.

### A. Energy Consumption

Fig. 4 shows the on-chip and off-chip energy consumption of the different techniques for voltage ranges between 0.45V and 0.7V, normalized to the baseline (0.8V). The technique with the best results in terms of reaching the minimum energy point is block disabling (7 out of 10 workloads). This minimum energy consumption point is achieved at 0.55V for 8 of the workloads; *canneal* and *vips* reach the minimum energy at 0.6V. If we keep decreasing the supply voltage, in general terms, block disabling (BD) increases the energy consumption of the system (respect to the baseline at 0.8V). BDOT and BDOT-C2C always offer an advantage respect to the baseline at 0.8V, except for the case of *blackscholes* and *canneal*. This is an important result, because it means that a simple technique such as block disabling in combination with directory inclusion always offers energy savings.

Digging into the results, we observe three differentiated patterns regarding energy consumption. The first pattern is observed in benchmarks such as *raytrace*, *swaptions*, and *vips*, where our techniques always reduce the energy consumption respect to BD at any voltage. These workloads are specially affected by the increase in inclusion victims, and keeping directory inclusion eliminates this problem.

A second pattern arises in benchmarks such as *blackscholes*, *bodytrack*, *canneal*, *fluidanimate*, *raytrace*, and *swaptions*. At ultra-low voltages and using BD these benchmarks have a huge increase in the off-chip memory energy consumption due to the effective on-chip cache capacity and associativity reduction, which translates into a great increase in the memory traffic. Our techniques reduce this consumption up to 85% and 86% at 0.45V, and 58% and 60% at 0.5V for BDOT and BDOT-C2C, respectively.

Lastly, and at certain voltage points, some workloads consume less energy with BD than with BDOT. At least two reasons explain this behaviour. First, some workloads such as *blackscholes*, *bodytrack*, *canneal*, and *x264*, show a considerable amount of reused faulty blocks in a short period of time (i.e., a block evicted from the private level is shortly after requested again). With BDOT, this kind of block can be allocated to a faulty entry. After being replaced from L1, its tag is kept, but its next reference will pay the penalty of an off-chip transaction. Besides, the recently fetched block will occupy the MRU position of the LLC stack, increasing the chances that newer reuses to that block will have to pay again the off-chip latency (and energy) penalty. Second, *dedup* and *ferret* exhibit

[2]We increase the number of samples as necessary to reach the target error within the given confidence level.

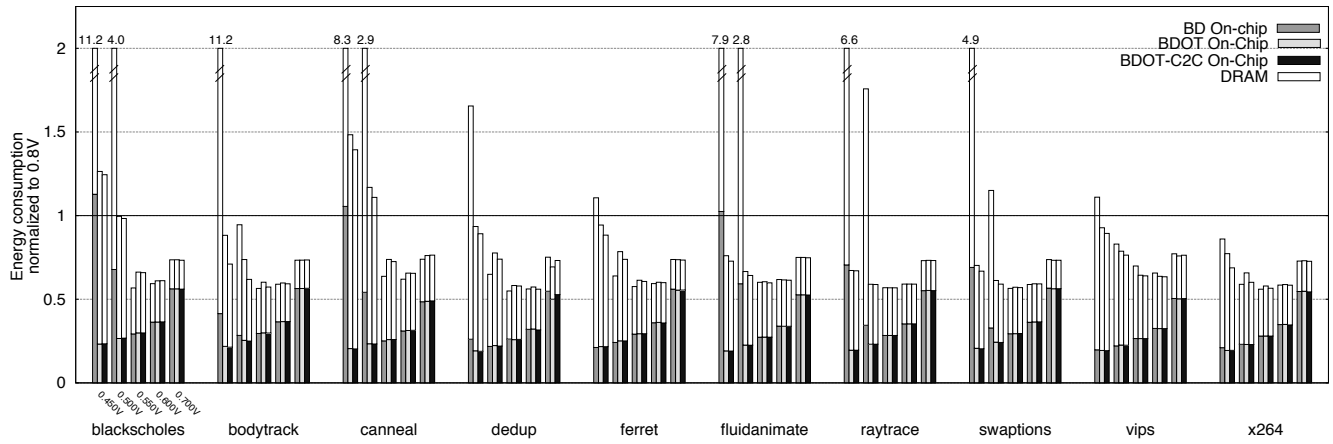


Fig. 4. Energy consumption: block disabling (*BD*), block disabling with operational tags (*BDOT*), and BDOT with cache-to-cache transfers (*BDOT-C2C*). Energy is divided in on-chip and off-chip (DRAM). Values are normalized to baseline (0.8V).

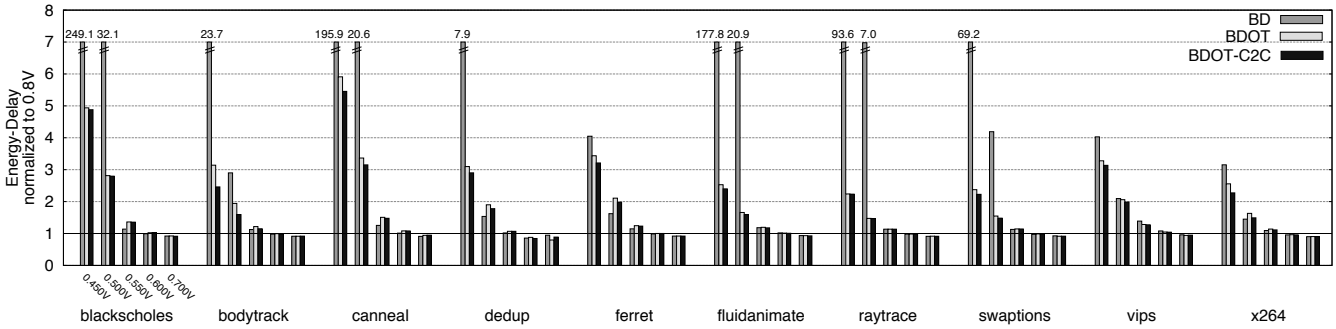


Fig. 5. Energy delay product: block disabling (*BD*), block disabling with operational tags (*BDOT*), and BDOT with cache-to-cache transfers (*BDOT-C2C*). Values are normalized to baseline (0.8V).

a similar behaviour. They are benchmarks that have a lot of stores, which do not benefit from BDOT in case the write-miss block is stored in a tag-only entry (a faulty entry).

For both cases (short reuses and write-misses), our last technique BDOT-C2C not suffices to overcome this problem. Namely, BDOT increases the energy consumption by 5% and BDOT-C2C by 3%, respect to BD at 0.55V. It seems clear that in order to fully exploit the potential of data-only disabling techniques the replacement policy needs to be considered and modified accordingly. We will cover this in future work.

Finally, *bodytrack* exhibits substantial amount of sharing, and that is why it benefits from the BDOT-C2C technique. In general, the considered applications do not have a high sharing degree, and providing cache-to-cache transfers of shared blocks translates in limited benefits (on average 3% energy save over BDOT at 0.5V and 0.45V). Likewise, the latency of the network-on-chip (NoC) remains unaffected, because there is no congestion in the NoC. BDOT-C2C increases the network traffic due to the cache-to-cache transfers and the L1 replacement messages to LLC, but with respect to BDOT, the increase is 4% and 3% at 0.5V and 0.45V, respectively. However, the impact in the NoC energy consumption is negligible.

### B. Energy-Delay

Fig. 5 shows the Energy-Delay product (ED). As expected, the minimum ED point shifts to higher voltages compared to the minimum energy consumption point. When a few loss in performance is acceptable, DVFS to lower than $V_{dd_{min}}$ voltages (0.7V-0.6V) still offers some advantage respect to the baseline. Regardless of the considered technique, at 0.7V ED improves on average 10%, but at 0.6V the improvement falls to less than 1%. Considering the energy reductions shown above at these voltages, both options are appealing to reach a good tradeoff between energy reduction and delay worsening. The three techniques perform very similar for this goal, so the coherence support here is really not needed. However, aggressive DVFS to the ultra-low voltage region is not a very favorable option. At very low voltages, both frequency scaling and miss ratio increase due to lower available capacity, delays execution, which in turn increases energy consumption, mainly in form of static energy. For instance, BD at 0.45V multiplies execution cycles by 3.5 on average (and some benchmarks such as blackscholes by 6.5). BDOT and BDOT-C2C do not show such an extreme behavior at 0.45V, because they take more profit of the scarce cache capacity, but anyway from the ED standpoint ultra-low voltage operation is not appealing.

### C. Transferring Power and Energy Reduction into System Performance

So far we have seen the effectiveness of BD in reducing energy and ED from a system standpoint. Now we focus on a power-constrained CMP and the capabilities of BD to improve system performance. To make it clear, consider the example of

Figure 6, showing three ways of doing a parallel computation.

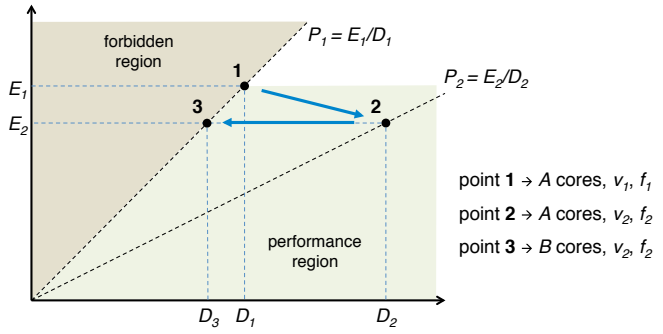


Fig. 6. Energy-Delay plot showing three operating points of a given parallel computation. Here the core count increase factor, $B/A$, equals $2 = P_1/P_2$, while the maximum speedup, $D_1/D_3$, is bound to the energy saving factor $E_1/E_2$.

The shaded region above the constant power line $P_1$ represents the CMP power limit; working above this line is thus forbidden in steady operation. The other shaded region below the lines $P_1$ and $E_1$ contains operating points with speedup potential. Point (1) represents a baseline operating mode in which $A$ cores perform a parallel task at voltage $v_1$ and frequency $f_1$. Point (2) represents a block disabling mode, in which the same number of cores performs the same task with lower power and energy, possibly at a lower voltage and frequency, $v_2$ and $f_2$, respectively. The key observation here is that we can use the saved power to wake up as many offline cores as needed to come back to the power limit. The point (3) represents such an action, where $B$ cores operating at $v_2$ and $f_2$ reach again the power limit $P_1$. We can compute a performance bound when going from point (1) to (3) assuming i) a perfectly parallelizable task, and ii) the existence of sufficient number of offline cores[3]. Under these assumptions it is quite straightforward to state that, first, the (maximum) attainable speedup $D_1/D_3$, equals the energy saving factor $E_1/E_2$, and second, the core count increase factor $B/A$, equals the power reduction factor $P_1/P_2$.

Table III summarizes the potential speedups reached by all the BD techniques. BDOT-C2C is the most promising technique, reaching speedups ranging between 3.7 and 4.4. Summarizing, block disabling enables working at ultra-low voltages, offloading a limited fraction of memory references to the off-chip DRAM. For instance, BDOT-C2C achieves the best CMP energy reduction, at the expenses of requiring somewhat large system energy (see Fig. 4). In a future power-constrained CMP, block disabling can be viewed as a control mechanism able to transfer energy inside-out, adding to other solutions proposed to mitigate the dark/dim silicon fact [1].

## VII. Related Work

One of the main barriers to ultra-low voltage operation is the increasing parameter variation, which translates into functional failure, especially affecting SRAM cells [2]. Circuit and architecture techniques have been proposed to tolerate faults in SRAM memories, especially when operating at low voltages, including spare rows/columns [26], robust cells [11], and error correction codes (ECC) [6]. Redundancy techniques have obvious limitations in the amount of faulty rows/columns they can handle, due to resource limitations, area scaling rate, and design complexity. For example, Intel includes 2 bits per cache entry to replace defective bits [12].

8T, 10T or ST SRAM cells have been proposed to allow operating at lower voltages at the expense of a significant increase in area (ST SRAM cells increase 100% area respect to a 6T cell) [11]. Mixed-cell memory designs offer a trade-off between area and resilience [27].

Regarding block disabling [9], as presented in this paper, it is worth referring to the work of Lee *et al.* [13], which examines performance degradation of disabling cache lines, sets, ways, ports, or the complete cache in a single processor environment. For the best of our knowledge, we provide the first evaluation of a shared-memory coherent CMP running parallel workloads in a block disabling context at ultra-low voltages with a complete memory model.

The granularity of the amount of capacity disabled might be finer, adding more complexity to cache accesses. Word disabling couples two consecutive cache entries to store a single fault-free block, halving both associativity and capacity [5]. Note that this mechanism could be extended with any of our techniques. By protecting the tag array (a modest increase in area), word disabling could result into a cache which has twice the number of tags than data entries, possibly decreasing the number of inclusion victims and potentially improving its efficiency.

Ladas *et al.* revisited block disabling schemes for low-power operation, and proposed to implement a victim cache to compensate for the associativity loss [14]. One weakness of our proposals is the penalty that blocks with a short reuse distance have to pay in case they are allocated in tag-only entries. A victim cache would capture this short reuse blocks, improving overall performance and energy.

More complex mechanisms couple faulty entries using a remapping mechanism, which adds a level of indirection to the cache access, making more complicated the direct adoption of our techniques [4], [15].

## VIII. Conclusions

Insufficient voltage reduction in ultra-deep technologies jeopardizes the benefits that scaling has been providing to the processor industry in the last decades. Within the processor, SRAM cells limit the minimum voltage because at lower voltages, manufacturing induced parameter variations limit the correct functionality of SRAM structures. Block disabling techniques disconnect cells that do not fulfill the stability requirements, in order to enable further voltage reductions, thus improving energy efficiency. In this work we introduce a model that relates voltage with probability of being a faulty bit cell ($P_{fail}$), intended to work at 22nm. Using such a model and a detailed experimental setup, we simulate block disabling taking into account the energy consumption of the whole CMP and the main memory. Simulations consider full execution of PARSEC applications running by scaling frequency and voltage into the ultra-low region. From this evaluation we conclude that block disabling may lose all their energy reduction benefits

[3]We do not take into account the potential benefits of the increased cache capacity coming from activating more LLC banks.

TABLE III. MAXIMUM SPEEDUP. THE FIRST EIGHT COLUMNS TO THE RIGHT OF BENCHMARKS NAMES ARE ENERGY-VOLTAGE PAIRS (JOULES-VOLTS) OF MINIMUM ENERGY (BASELINE, BD, BDOT, AND BDOT-C2C, RESPECTIVELY). THE RIGHTMOST THREE COLUMNS ARE THE ENERGY REDUCTION FACTORS, AND HENCE A BOUND ON THE MAXIMUM SPEEDUP. BOLDED NUMBERS HIGHLIGHT THE BEST CONFIGURATIONS.

| **Benchmkark** | Base E | Base $V_{dd}$ | BD E | BD $V_{dd}$ | BDOT E | BDOT $V_{dd}$ | C2C E | C2C $V_{dd}$ | Base/BD | Base/BDOT | Base/C2C |
|---|---|---|---|---|---|---|---|---|---|---|---|
| blackscholes | 0.62 | | 0.21 | 0.55 | **0.17** | | 0.17 | | 2.94 | **3.71** | 3.69 |
| bodytrack | 7.99 | | 2.62 | 0.50 | 2.02 | | **1.97** | | 3.05 | 3.96 | **4.05** |
| canneal | 1.05 | | 0.35 | 0.55 | 0.28 | | **0.28** | | 3.01 | 3.69 | **3.73** |
| dedup | 6.04 | | 1.58 | 0.50 | 1.39 | | **1.37** | | 3.81 | 4.34 | **4.41** |
| ferret | 16.59 | 0.80 | **4.11** | 0.45 | 4.22 | 0.45 | 4.20 | 0.45 | **4.04** | 3.94 | 3.95 |
| fluidanimate | 3.98 | | 1.34 | 0.55 | 0.94 | | **0.93** | | 2.96 | 4.23 | **4.26** |
| raytrace | 1.31 | | 0.43 | 0.55 | **0.30** | | **0.30** | | 3.02 | **4.38** | **4.38** |
| swaptions | 2.54 | | 0.86 | 0.55 | 0.61 | | **0.60** | | 2.94 | 4.18 | **4.23** |
| vips | 9.56 | | 2.44 | 0.45 | 2.39 | | **2.38** | | 3.91 | 3.99 | **4.01** |
| x264 | 3.09 | | 0.77 | 0.45 | **0.71** | | 0.71 | | 4.04 | **4.38** | 4.37 |

because of the extra main memory accesses, due to the reduction of cache capacity and associativity.

This work proposes two techniques to improve block disabling in CMPs leveraging current coherence mechanisms. First, associativity degradation causes unnecessary invalidations in inclusive hierarchies; however, inclusion is tracked in the directory and the tag array. So we propose to use robust cells in the directory tags to avoid unnecessary invalidations: we call it block disabling with operational tags (BDOT). Second, directories also track replicated blocks among caches, and therefore, request to faulty entries in the LLC cache can be forwarded to a L1 sharer of the block with a valid copy. We call this technique BDOT with cache-to-cache service (BDOT-C2C)

Block disabling (BD), without and with operational tags has been assessed through three different metrics, namely, system energy, system energy-delay product, and CMP energy. Regarding system energy, BD reaches the absolute minimum at 0.55V in most of workloads, while the other two techniques reduce energy consumption towards lower voltages. Regarding system energy-delay, the baseline configuration running at 0.8 is improved by 10% and 1% at 0.7V and 0.6V, respectively, equally well by all three techniques; no need to use operational tags is therefore concluded. Finally, searching for CMP energy reduction in a power-constrained CMP seeks to wake up offline cores. To this end, both BDOT and BDOT-C2C excel, showing the potential of speedups ranging from 3.7 to 4.4.

## REFERENCES


[1] M. Taylor, "A landscape of the new dark silicon design regime," *IEEE Micro*, vol. 33, no. 5, pp. 8–19, Sept 2013.

[2] R. Dreslinski *et al.*, "Near-threshold computing: Reclaiming moore's law through energy efficient integrated circuits," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 253–266, Feb 2010.

[3] R. Zahir *et al.*, "The medfield smartphone: Intel architecture in a handheld form factor," *IEEE Micro*, vol. 33, no. 6, pp. 38–46, Nov 2013.

[4] A. Ansari *et al.*, "Archipelago: A polymorphic cache design for enabling robust near-threshold operation," in *IEEE 17th Int. Symp. on High Performance Computer Architecture*, 2011, pp. 539–550.

[5] C. Wilkerson *et al.*, "Trading off cache capacity for reliability to enable low voltage operation," in *Proc. of the 35th Annual Int. Symp. on Computer Architecture*, 2008, pp. 203–214.

[6] Z. Chishti *et al.*, "Improving cache lifetime reliability at ultra-low voltages," in *Proc. of the 42nd Annual IEEE/ACM Int. Symp. on Microarchitecture*, 2009, pp. 89–99.

[7] R. Kumar *et al.*, "A family of 45nm IA processors," in *IEEE Int. Solid-State Circuits Conf. Digest of Technical Papers*, Feb 2009, pp. 58–59.

[8] S. R. Nassif *et al.*, "A resilience roadmap (invited paper)," in *Proceedings of the Conference on Design, Automation and Test in Europe*, 2010, pp. 1011–1016.

[9] S. Damaraju *et al.*, "A 22nm IA multi-CPU and GPU System-on-Chip," in *IEEE Int. Solid-State Circuits Conf. Digest of Technical Papers*, Feb 2012, pp. 56–57.

[10] S.-T. Zhou *et al.*, "Minimizing total area of low-voltage SRAM arrays through joint optimization of cell size, redundancy, and ECC," in *IEEE Int. Conf. on Computer Design*, Oct 2010, pp. 112–117.

[11] J. Kulkarni *et al.*, "A 160mV robust schmitt trigger based subthreshold SRAM," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 10, pp. 2303–2313, 2007.

[12] J. Chang *et al.*, "The 65-nm 16-MB Shared On-Die L3 Cache for the Dual-Core Intel Xeon Processor 7100 Series," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 4, pp. 846–852, April 2007.

[13] H. Lee *et al.*, "Performance of graceful degradation for cache faults," in *IEEE Computer Society Annual Symp. on VLSI*, March 2007, pp. 409–415.

[14] N. Ladas *et al.*, "Performance-effective operation below Vcc-min," in *IEEE Int. Symp. on Performance Analysis of Systems Software*, March 2010, pp. 223–234.

[15] A. BanaiyanMofrad *et al.*, "REMEDIATE: A scalable fault-tolerant architecture for low-power NUCA cache in tiled CMPs," in *Int. Green Computing Conf.*, 2013, pp. 1–10.

[16] U. Karpuzcu *et al.*, "Coping with parametric variation at near-threshold voltages," *IEEE Micro*, vol. 33, no. 4, pp. 6–14, July 2013.

[17] P. Conway *et al.*, "Cache hierarchy and memory subsystem of the AMD opteron processor," *IEEE Micro*, vol. 30, pp. 16–29, 2010.

[18] M. M. K. Martin *et al.*, "Why on-chip cache coherence is here to stay," *Communications of the ACM*, vol. 55, no. 7, pp. 78–89, Jul. 2012.

[19] J. L. Baer *et al.*, "On the inclusion properties for multi-level cache hierarchies," in *Proc. of the 15th Annual Int. Symp. on Computer Architecture*, 1988, pp. 73–80.

[20] P. Magnusson *et al.*, "Simics: A full system simulation platform," *Computer*, vol. 35, no. 2, pp. 50–58, Feb 2002.

[21] M. M. K. Martin *et al.*, "Multifacet's General Execution-driven Multiprocessor Simulator (GEMS) toolset," *SIGARCH Comput. Archit. News*, vol. 33, no. 4, pp. 92–99, Nov. 2005.

[22] N. Agarwal *et al.*, "GARNET: A detailed on-chip network model inside a full-system simulator," in *IEEE Int. Symp. on Performance Analysis of Systems and Software*, 2009, pp. 33–42.

[23] P. Rosenfeld *et al.*, "DRAMSim2: A cycle accurate memory system simulator," *Computer Architecture Letters*, vol. 10, no. 1, pp. 16–19, Jan 2011.

[24] S. Li *et al.*, "McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures," in *42nd Annual IEEE/ACM Int. Symp. on Microarchitecture*, 2009, pp. 469–480.

[25] C. Bienia *et al.*, "The PARSEC benchmark suite: characterization and architectural implications," in *Proc. of the 17th Int. Conf. on Parallel Architectures and Compilation Techniques*, 2008, pp. 72–81.

[26] S. Schuster, "Multiple word/bit line redundancy for semiconductor memories," *IEEE Journal of Solid-State Circuits*, vol. 13, no. 5, pp. 698–703, Oct 1978.

[27] S. M. Khan *et al.*, "Improving multi-core performance using mixed-cell cache architecture," in *IEEE 19th Int. Symp. on High Performance Computer Architecture*, 2013, pp. 119–130.