Efficient Cache Architectures for Reliable Hybrid Voltage Operation Using EDC Codes

Bojan Maric[1,2] Jaume Abella[2] Mateo Valero[1,2]
[1]Barcelona Supercomputing Center (BSC-CNS) [2]Universitat Politecnica de Catalunya (UPC)
{bojan.maric, jaume.abella, mateo.valero}@bsc.es

***Abstract*—Semiconductor technology evolution enables the design of sensor-based battery-powered ultra-low-cost chips (e.g., below 1 €) required for new market segments such as body, urban life and environment monitoring. Caches have been shown to be the highest energy and area consumer in those chips.**

This paper proposes a novel, hybrid-operation (high Vcc, ultra-low Vcc), single-Vcc domain cache architecture based on replacing energy-hungry bitcells (e.g., 10T) by more energy-efficient and smaller cells (e.g., 8T) enhanced with Error Detection and Correction (EDC) features for high reliability and performance predictability. Our architecture is proven to largely outperform existing solutions in terms of energy and area.

***Index Terms*—Caches, Low Energy, Reliability, Real-Time**

I. Introduction

Higher semiconductor technology integration due to geometry scaling opens the door to new market segments. In particular, technology evolution enables adding some degree of intelligence to any control or measuring engine such as biomedical sensor applications to monitor the body, environment sensor applications to monitor wind, temperature, tsunamis, etc., by means of battery-powered ultra-low-cost (e.g., below 1 €) computing devices. The main requirements for this new market segment are: (i) ultra-low energy consumption in order to extend battery lifetime, (ii) very simple system design for increased yield and reduced cost and (iii) strong functional and timing guarantees required for the worst-case execution time (WCET) estimation, as needed for running critical applications on top. Typically, those computing systems have two operation modes and different optimal supply voltages (Vcc): (i) high-performance and low-power operation mode under high or moderate voltage (HP mode for short) during relatively short periods of time to react to some infrequent particular events (e.g., 0.01% - 1% of the time [19]) and (ii) low performance, ultra-low energy and reliable operation mode under near-/sub-threshold (NST) voltage (ULE mode for short) during most of the time until infrequent events arise (e.g., 99% - 99.99% of the time [19]).

Cache memories are used in those systems to reduce the number of slow and energy-hungry memory accesses, thus increasing the efficiency of the system. However, caches become the main energy consumer on the chip. Cheap solutions based on a single-Vcc domain have been demonstrated recently [14], [15]. Those caches use large memory cells to achieve high levels of reliability even at ULE mode, as needed by critical applications run on top. Decreasing the size of the memory cells for higher energy efficiency at the expense of higher failure rates is unacceptable in this environment. Faulty entries should be then disabled and strong performance guarantees required by critical applications would not be achievable [20].

This paper proposes a novel single-Vcc domain cache architecture whose main characteristics are: (i) low energy consumption, (ii) simple design and (iii) high reliability levels, outperforming existing solutions [14]. In particular, our cache design relies on replacing energy-hungry bitcells (e.g., 10T) by more energy-efficient and smaller cells (e.g., 8T) enhanced with error detection and correction (EDC) features. We illustrate our cache architecture with two scenarios, depending on the reliability level of the baseline (no coding or single error correction double error detection (SECDED)), where 10T cells are replaced by smaller 8T cells (a) by keeping no coding at HP mode and by adding SECDED at ULE mode, whenever no coding is in place or (b) by keeping SECDED at HP mode and by replacing SECDED by double error correction triple error detection (DECTED) at ULE mode, whenever SECDED is in place. Our cache architecture achieves significant energy savings (up to 14% and up to 42% on average at HP and ULE mode respectively) and small average performance degradation (up to 3%) with respect to existing solutions [14] while keeping the same guaranteed performance and reliability levels.

II. Related Work

There is an abundance of literature on low-power techniques for caches. Double-ended 6T (6 transistors) SRAM cells have been widely deployed for high voltage operation. Numerous SRAM cell designs such as 8T [16], Schmitt-Trigger 10T (10T) [12], etc. target different voltage and robustness scenarios. However, those SRAM cells introduce significant area and energy overheads w.r.t. 6T cells at high voltage, which is unaffordable in embedded cache design if used extensively.

Some authors present techniques to save energy by reconfiguring cache characteristics such as cache size and associativity [3] or lowering cache Vcc [9] (or even gating it [18]) for some cache sections or the whole cache. Other authors propose splitting the cache into different modules [11]. Zhou et al. [23] propose downsizing 6T cells of large on-chip caches combined with EDC techniques and extra cells to guarantee a target yield. In general, those techniques are unsuitable for our market since they fail to operate reliably at ULE mode.

Techniques based on having multiple Vcc domains are unaffordable for our target ultra-low-cost (e.g., below 1 €) market [8]. Likewise, techniques based on disabling faulty cache entries [21], [1], [7] *fail to provide strong timing guarantees required for the worst-case execution time (WCET) estimation, as needed for critical applications in our target market* [20]. A failure to perform an operation correctly and within a given time may have catastrophic consequences in these environments.

Maric et al. [14] propose hybrid-operation, single-Vcc domain cache architectures, suitable for our target market. Nevertheless, authors naively achieve robustness at ultra-low Vcc by simply

978-3-9815370-0-0/DATE13/©2013 EDAA

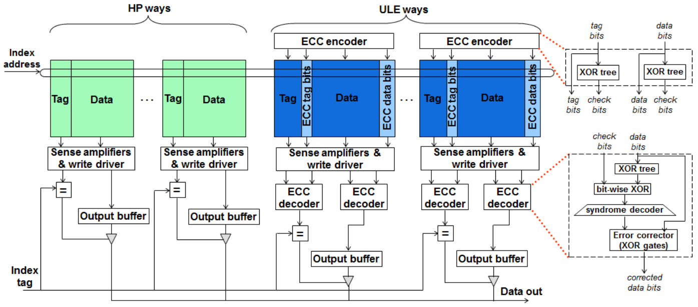

Fig. 1. Proposed cache architecture for scenario A.

increasing bitcells size, which translates into large area and energy overheads. Our approach builds upon the solution by Maric et al. reducing energy and area overheads while keeping robustness, simplicity and performance predictability.

III. Proposed Hybrid Cache Architecture

In this section, we first describe the cache architecture that we use as the baseline. Next, we present our proposal as well as the design methodology for the proposed architecture.

A. Baseline Architecture

Based on the fact that most L1 caches in existing chips are set-associative, we have chosen such organization as the target of our study, although significant parts of our study can be easily reused for direct-mapped and fully-associative caches.

We use a hybrid-operation, single-Vcc domain cache design particularly suited for our target market [14] as a starting point. The cache is designed in such a way that some of the cache ways are optimized to satisfy high performance requirements during high Vcc operation (HP ways) whereas the rest of the ways provide ultra-low energy consumption and reliability during NST Vcc operation (ULE ways). In particular, we use a 6T+10T hybrid cache as the baseline [14]. In this design, HP ways are implemented with differential 6T bitcells whereas the ULE ways consist of 10T bitcells, *although our proposal is not limited to this design*. During ULE mode, data processing is expected to be minimal and workloads are much smaller than during HP mode [19]. Workload discrepancy across HP and ULE mode justifies reducing the hardware resources at ULE mode. Since HP ways would experience many faults at NST Vcc and thus would not provide reliable operation, they are turned off at ULE mode. However, all cache ways are enabled at HP mode to fit larger workloads and provide high performance. ULE ways are reused at HP mode, in spite of their inefficiency at high Vcc, because they reduce the number of slow and energy-hungry memory accesses [15].

The main drawback of this design is using large 10T cells to guarantee robust fault-free NST operation. Simply decreasing the size of these large memory cells or replacing them by cheaper cells (e.g., 8T) for higher energy efficiency would increase failure rates. Faulty entries should be then disabled and strong performance guarantees required by critical applications would not be achievable [20].

In order to overcome the inefficiency of the large memory cells (e.g., 10T) in terms of area and energy, we propose a new, simple, energy-efficient cache design without jeopardizing reliability levels to still provide predictable performance.

B. Our Proposal

We illustrate our proposed cache architecture with two scenarios depending on the reliability level of the baseline cache. In the first scenario, we consider a 6T+10T baseline cache where **no coding** is in place. In the second scenario, the baseline cache has higher reliability and all ways are **SECDED** protected to deal with *soft errors* (6T+SECDED+10T+SECDED). Our cache design relies on replacing energy-hungry bitcells (e.g., 10T) in **ULE ways** by more energy-efficient and smaller cells (e.g., 8T) enhanced with error detection and correction features to keep the same reliability levels, which are particularly critical at ULE mode. Figure 1 depicts our proposed cache architecture for the first scenario.

Reliability of ULE ways at HP mode in both scenarios is not an issue, because both 8T and 10T cells are more reliable (by some orders of magnitude) than 6T ones at high voltage, thus the *same* coding (none or SECDED) as that used for the baseline cache suffices. However, at ULE mode, *stronger* codes (SECDED, if none in baseline or DECTED, if SECDED in baseline) must be used, because smaller 8T cells are less reliable than 10T at NST Vcc. Therefore, we have:

Scenario A. The baseline is a 6T+10T cache and no coding is in place. 10T cells are replaced by smaller and less reliable 8T cells by adding SECDED whenever no coding is in place (6T+10T vs. 6T+8T+SECDED). SECDED is only required to deal with *hard faults* in 8T cells at ULE mode. At HP mode, SECDED is simply turned off (6T+10T vs. 6T+8T).

Scenario B. The baseline has higher reliability than that of scenario A since all cache ways are SECDED protected to deal with *soft errors* (6T+SECDED+10T+SECDED). 10T cells are replaced by smaller and less reliable 8T cells by replacing SECDED (only for ULE ways) by DECTED whenever SECDED is in place to deal with *soft errors* (6T+SECDED+10T+SECDED vs. 6T+SECDED+8T+DECTED). DECTED is only required to deal with *hard faults* in 8T cells at ULE mode. At HP mode, DECTED is simply turned off since SECDED protection of 8T cells is sufficient to deal with *soft errors* at high Vcc (6T+SECDED+10T+SECDED vs. 6T+SECDED+8T+SECDED).

Using EDC introduces delay, energy and area overheads. As described later in Section IV, we consider those overheads in our calculations. Turning off HP ways at ULE mode is done by using the gated-Vdd technique [18]. The processor itself is responsible for gating or ungating the corresponding cache ways (or corresponding EDC block) on a Vcc change. Overheads are negligible, as explained in [18].

In the rest of the paper, we use differential 6T for HP ways, 8T for ULE ways, Hsiao SECDED and DECTED codes [5] and 32nm technology node. However, our architecture is not limited to any particular Vcc level, SRAM cell type, technology node, type of protection or reliability level as long as performance predictability is achievable. The type of protection used depends on the given baseline cache and its reliability level. *Since we maintain the same level of robustness as in the baseline cache, performance predictability features remain the same.*

C. Design Methodology

HP ways are designed with differential 6T bitcells. Depending on the cache size and target cache yield, the *hard* faulty bit rate (P_f) is obtained using elementary probability calculations. For example, to have a 99% yield for an 8KB cache, faulty bit rate P_f must be 1.22x10^{-6}. Then, using the analysis based on importance sampling proposed by Chen et al. [6] and calculated P_f, 6T bitcells size is determined.

```
10T bitcells sizing:
1. For chosen NST Vcc and reduced frequency at ULE mode, size 10T bitcell
   to match the same hard bit failure rate ($P_f$) as 6T bitcells at HP mode,
   using the analysis based on importance sampling proposed by Chen et al. [6]
2. Calculate cache yield ($Y_{10T}$) based on the given cache size and $P_f$
Replacing 10T bitcells with 8T bitcells and EDC:
1. Set minimal tranzistor sizes possible for targeted technology node
2. Calculate 8T bitcell's hard bit failure probability $P_{f8T}$ using Chen's
   analysis [6]
3. Calculate failure probability ($P_{total}$) of EDC-protected cache
4. Calculate cache yield ($Y$)
5. If ($Y < Y_{10T}$)
5a.        Increase transistor sizes by minimal amount possible for targeted
           technology
5b.        Go to step 2
6. Else
6a.        Optimal cell size is obtained
```

Fig. 2. Design methodology for scenario A.

The design methodology for the ULE ways for **scenario A** is shown in Figure 2. Remember that only ULE ways are active at ULE mode. For the chosen NST Vcc and reduced operating frequency at ULE mode, we first size 10T cells to match the same *hard* faulty bit rate as 6T cells at HP mode (P_f) using Chen's analysis [6]. Then, depending on the cache size and given P_f, cache yield (Y_{10T}) can be easily calculated. Note that in **scenario B** the 10T cells are SECDED protected to deal with soft errors, and cache yield in that case ($Y_{10T+SECDED}$) can be calculated analogously to scenario A.

Next, we determine the size of 8T bitcells protected with EDC in order to replace 10T bitcells in ULE ways as shown in Figure 2. We first set minimal transistors sizes for 8T bitcells and then calculate the hard bit failure probability (P_{f8T}) for the chosen NST Vcc by using Chen's analysis [6]. Then, we define data and tag words to have 32 and 26 bits respectively, and protect them at such granularity. The probability of having *fault-free* tag/data words and the cache yield (Y) are:

$$P(tag/data) = \sum_{i=0}^{1} (1 - P_{f8T})^{n+k-i} P_{f8T}^{i} \binom{n+k}{i} \quad (1)$$

$$Y = P(data)^{DW} P(tag)^{TW}, \quad (2)$$

where DW and TW are the total number of data and tag words in cache respectively, n is number of bits of tag or data words, k is number of added check bits (i.e. 7 bits for SECDED, 13 bits for DECTED) to each tag/data word and i is number of hard faults in a tag or data word. Note that in case of no coding (scenario A), SECDED suffices to correct a hard faulty bit in a word (8T+SECDED), whereas in scenario B, DECTED can correct both a soft error and a hard faulty bit in the same word (8T+ DECTED). If the yield obtained (Y) is lower than required (e.g., Y_{10T} for scenario A or $Y_{10T+SECDED}$ for scenario B), transistors sizes must be increased by the smallest amount possible for the target technology node and yield must be calculated again. Once yield is high enough, we have an optimal SRAM bitcell size.

IV. Evaluation

This section presents the evaluation methodology and performance/energy results to verify the efficiency of the proposed cache architecture.

A. Methodology

We have chosen a very simple processor architecture with one core and in-order execution, resembling a recently fabricated Intel processor for hybrid Vcc operation although not suited for the ultra-low-cost market [10]. Both on-chip L1 data (DL1) and instruction (IL1) caches implement the proposed design. 8KB 8-way caches are used, where 7 ways are implemented with 6T cells and 1 way with 10T cells (7+1 for short). We have considered other designs (e.g., 6+2), but they did not provide further insights. The relative memory latency is low (in the order of 20 cycles) given the simplicity required in those systems, its small size (typically few MBs) and its high integration with the processor itself. Given that all comparisons involve caches with the same characteristics in terms of cache size and associativity, other memory latencies do not change the trends reported later thus, we did not include memory energy in our results.

1) Benchmarks: To the best of our knowledge, a set of benchmarks specific for the domain that we target does not exist. We have chosen MediaBench [13], because they fit very well the expected needs of the ultra-low-cost segment: an abundant data processing during HP mode and relatively small workloads at ULE mode. We classify benchmarks into two categories, depending on the cache requirements: (i) *SmallBench* - workloads fit into very small cache sizes (e.g., 1KB) due to small data volume (adpcm_c, adpcm_d, epic_c and epic_d) and (ii) *BigBench* - larger cache space is required to fit the workload due to large data volume (g721_c, g721_d, gsm_c, gsm_d, mpeg2_c and mpeg2_d). *SmallBench* benchmarks are used during ULE operation whereas *BigBench* ones are used during HP operation.

2) Operating Modes: Our system has two distinct operating modes: HP and ULE. We have set Vcc to 1V and 350mV for HP and ULE mode respectively. Operating frequencies are set to 1GHz for HP mode, and 5MHz for ULE mode, which is in line with the Intel processor for hybrid Vcc operations [10].

3) System Modeling: The technology node considered is 32nm. L1 cache memories have been modeled using CACTI 6.5, which is a flexible and accurate cache delay, energy, power and area simulator [17]. To support two different operating modes, we have extended CACTI tool in order to implement accurate energy models for 8T and 10T SRAM cells when operating at high and NST Vcc by adapting capacitances, resistances and geometry. All SRAM cells have been sized as described in Section III. Several hybrid cache microarchitectures have been implemented using heterogeneous SRAM cell types at a coarse granularity as explained in Section III. Moreover, we have extended tag and data words (26 and 32 bits respectively in our case) with check bits (7 bits for SECDED, 13 bits for DECTED) and taken into account energy and area overheads introduced due to those check bits.

In order to understand the impact of different cache designs on the whole chip, we have incorporated our custom-modified CACTI tool into the MPSim [2] full-chip simulator. We have extended MPSim with power models analogous to those of Wattch [4], but using our enhanced CACTI version to model all SRAM array-like structures (Caches, TLB, etc.). All SRAM arrays except L1 caches have been implemented using 10T cells so they operate properly at any voltage level considered.

In our simulations, we account an additional latency of one clock cycle for SECDED/DECTED encoding and decoding as well as the energy consumed by the extra EDC circuits at ULE mode. Energy consumption of EDC encoders and decoders is obtained by performing HSPICE simulations. For that purpose, we used the 32nm Predictive Technology Model transistor model and 10% variation in threshold voltage (V_t) [22].

B. Results and Discussion

In this subsection, we present energy per instruction (EPI) and area results at HP and ULE modes comparing the proposed

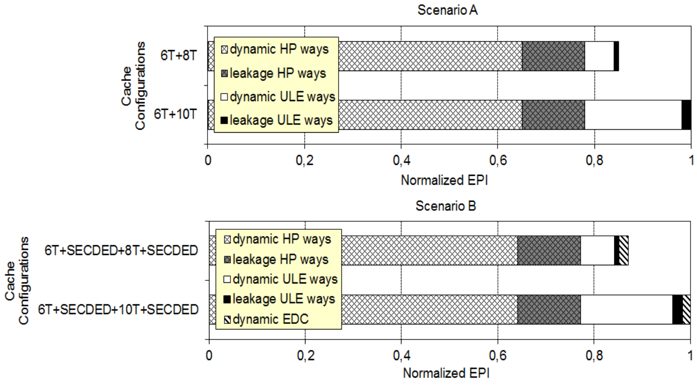

Fig. 3. Normalized average EPI breakdowns at HP mode for scenarios A and B.

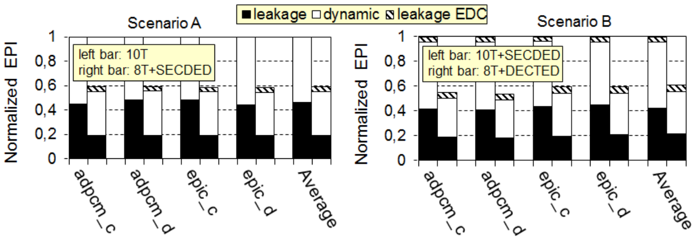

Fig. 4. Normalized EPI breakdowns at ULE mode for scenarios A and B.

cache architecture with the baseline designs for both scenarios described in Section III. Execution time and energy vary across scenarios. Thus, for the sake of clarity, results have been normalized with respect to the baseline configuration in both scenarios.

1) HP mode: Figure 3 shows the normalized average EPI for both scenarios at HP mode. All benchmarks show minor differences to the average. The main reason is that the fraction of cache memory accesses (instruction and data) and execution behavior of the different benchmarks is quite similar given that their workloads fit pretty well in cache, which will be the case in real systems. Since caches are the main energy contributor in these extremely simple processors, *cache behavior dominates full processor behavior*.

Our architecture shows energy savings of 14% and 12% on average for scenario A and scenario B respectively. This is due to the smaller transistor sizes for 8T cells with respect to the 10T cells and thus, reduced dynamic energy (which is the dominant energy factor at high voltage). *Our architecture does not experience any performance degradation (no latency overhead) since 8T cells are as reliable as 6T at high voltage, so they use exactly the same coding as in baseline.*

2) ULE mode: HP ways are turned off at this mode, so only ULE ways keep operating. Leakage increases at ULE mode whereas dynamic energy is still a significant energy factor. Figure 4 shows the normalized EPI breakdowns across all benchmarks for scenario A and B at ULE mode. Caches remain to be the main energy contributor and access frequency is not drastically different across benchmarks, so effects on different sources of energy on each benchmark are relatively similar, because dynamic and leakage cache energy is impacted in a very similar way. Thus, all benchmarks observe similar trends.

When EDC codes are used, smaller transistors are needed for 8T cells and thus, relative dynamic and leakage energy consumption is lower than for 10T cells. Smaller transistors keep capacitances lower and reduce dynamic energy, which scales *linearly* with capacitance, whereas delay and thus, leakage scales *exponentially*. Hence, the relative leakage energy savings are larger than those for dynamic energy. Taken all together, the normalized average EPI reductions are 42% and 39% for scenario A and B respectively. *Performance variation due to the extra cycle for EDC encoding/decoding is negligible (around 3% increase in execution time in all cases).*

V. Conclusions

We propose a new, efficient and simple, single-Vcc domain cache architecture for hybrid Vcc operations in ultra-low-cost (e.g., below 1 €) battery-powered systems. The cache design relies on replacing energy-hungry bitcells (e.g., 10T) by more energy-efficient and smaller cells (e.g., 8T) enhanced with error detection and correction features to improve energy and area efficiency without jeopardizing reliability levels to still provide predictable performance, as needed for critical applications. Our cache architecture achieves significant savings in energy (up to 14% and up to 42% on average at HP and ULE mode respectively) and negligible average performance degradation (up to 3%) with respect to existing solutions while keeping the same guaranteed performance and reliability levels.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Science and Innovation under grant TIN2012-34557, HiPEAC and the UPC under grant FPI-UPC.

References

[1] J. Abella et al. Low vccmin fault-tolerant cache with highly predictable performance. In *MICRO*, 2009.
[2] C. Acosta et al. The MPsim Simulation Tool. Technical Report UPC-DAC-RR-CAP-2009-15, in UPC, 2009.
[3] D. H. Albonesi. Selective cache ways: On-demand cache resource allocation. In *MICRO*, 1999.
[4] D. M. Brooks et al. Wattch: A framework for architectural-level power analysis and optimizations. In *ISCA*, 2000.
[5] C. L. Chen and M. Y. Hsiao. Error-correcting codes for semiconductor memory applications: A atate-of-the-art review. *IBM Journal of Research and Development*, 28(2), 1984.
[6] G.K. Chen et al. Yield-driven near-threshold SRAM design. In *ICCAD*, 2007.
[7] Y. G. Choi et al. Matching cache access behavior and bit error rate pattern for high performance low vcc l1 cache. In *DAC*, 2011.
[8] R.G. Dreslinski et al. Reconfigurable energy efficient near threshold cache architectures. In *MICRO*, 2008.
[9] K. Flautner et al. Drowsy caches: Simple techniques for reducing leakage power. In *ISCA*, 2002.
[10] S. Jain et al. A 280mv-to-1.2v wide-operating-range ia-32 processor in 32nm cmos. In *ISSCC, dig. Tech. Papers*, 2012.
[11] J. Kin et al. The filter cache: An energy efficient memory structure. In *MICRO*, 1997.
[12] J.P. Kulkarni, K. Kim, and K. Roy. A 160 mV, fully differential, robust schmitt trigger based sub-threshold SRAM. In *ISLPED*, 2007.
[13] C. Lee et al. Mediabench: A tool for evaluating and synthesizing multimedia and communication systems. In *MICRO*, 1997.
[14] B. Maric et al. Hybrid high-performance low-power and ultra-low energy reliable caches. In *ACM CF*, 2011.
[15] B. Maric et al. Adam: An efficient data management mechanism for hybrid high and ultra-low voltage operation caches. In *GLSVLSI*, 2012.
[16] Y. Morita et al. An area-conscious low-voltage-oriented 8t-sram design under dvs environment. In *IEEE Symposium on VLSI Circuits*, 2007.
[17] N. Muralimanohar, R. Balasubramonian, and N.P. Jouppi. CACTI 6.0: A tool to understand large caches. *HP Tech Report HPL-2009-85*, 2009.
[18] M. Powell et al. Gated-vdd: A circuit technique to reduce leakage in deep-submicron cache memories. In *ISLPED*, 2000.
[19] R. Szewczyk et al. Lessons from a sensor network expedition. In *European Workshop on Sensor Networks*, 2004.
[20] R. Wilhelm et al. The worst-case execution time problem: overview of methods and survey of tools. *ACM Trans. on Embedded Computing Systems*, 7(3):1–53, 2008.
[21] C. Wilkerson et al. Trading off cache capacity for reliability to enable low voltage operation. In *ISCA*, 2008.
[22] W. Zhao and Y. Cao. New generation of predictive technology model for sub-45nm design exploration. In *ISQED*, 2006.
[23] S.-T. Zhou et al. Minimizing total area of low-voltage sram arrays through joint optimization of cell size, redundancy, and ecc. In *ICCD*, 2010.