Reliable Ultra-Low-Voltage Cache Design for Many-Core Systems

Meilin Zhang, *Student Member, IEEE*, Vladimir M. Stojanovic, *Member, IEEE*, and Paul Ampadu, *Senior Member, IEEE*

***Abstract*—We reduce cache supply voltage below the normally acceptable V_{DDMIN}, in order to improve overall many–core system energy efficiency. Based on the observation that cache lines contain mostly one *hard* faulty cell at these ultra-low supply voltages, we exploit existing double-error correcting triple-error detecting codes, together with cache line disabling, to handle both *soft* and *hard* cache faults, thus enabling reliable ultra-low supply voltage cache operation. Compared to the next-best approach in the research literature, the proposed method reduces system energy consumption by up to 25% and energy-execution time product by nearly 10%, while introducing only 0.28% storage overhead and marginal instruction per cycle degradation, when the target yield loss rate is 1/1000.**

***Index Terms*—Cache, fault tolerance, low-power design, many-core, very-large-scale integration (VLSI).**

I. Introduction

ENERGY efficiency is a key concern in future many-core (1000+ processors) system design. Among all processor components, the cache consumes about 40% of the total area and 20% of power [1]. It is well known that supply voltage scaling can reduce system energy consumption at the cost of performance and reliability. Low supply voltages increase the impact of process and other variations on circuit functionality and performance. Furthermore, the current trend toward many-core processor and deep nanometer designs exacerbates reliability issues at low supply voltage. Eventually, the system will fail below a minimal supply voltage V_{DDMIN}. Typically, the cache arrays in the system limit V_{DDMIN} of the whole processor [2]. Consequentially, it is necessary to provide fault tolerance for the cache under low supply voltages to improve system energy efficiency while maintaining reliability.

To reduce the supply voltage beyond normally acceptable V_{DDMIN} and maintain appropriate yield and reliability, we exploit existing double-error correcting triple-error detecting (DECTED) codes, together with cache line disabling in an efficient way to handle both *persistent* (hard) and *non-persistent* (soft) errors, and provide a comprehensive evaluation of energy, performance, reliability, and overhead. Section II of this paper briefly describes related works on low-power cache design. In Section III, our proposed approach is described. Sections IV and V provide analysis and experimental results. Conclusions are presented in Section VI.

Manuscript received July 18, 2012; revised September 18, 2012; accepted October 27, 2012. Date of publication January 18, 2013; date of current version February 1, 2013. This work was supported in part by the U.S. National Science Foundation under Grants ECCS-0925993, ECCS-0903448, CAREER Award ECCS-0954999, and the Semiconductor Research Corporation Award SRC-2009-HJ-2000. This brief was recommended by Associate Editor M. Alioto.

M. Zhang is with the University of Rochester, Rochester, NY 14627, and currently is a visiting student at the Massachusetts Institute of Technology, Cambridge, MA 02139, USA (e-mail: mezhang@mit.edu).

V. M. Stojanovic is with the Massachusetts Institute of Technology, Cambridge, MA 02139, USA (e-mail: vlada@mit.edu).

P. Ampadu is with the ECE Department at the University of Rochester, Rochester, NY 14627, USA, currently on leave at the Massachusetts Institute of Technology, Cambridge, MA 02139, USA (e-mail: ampadu@mit.edu).

Color versions of one or more of the figures in this brief are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2012.2231013

II. Related Work

Some traditional approaches to reliable ultra-low-voltage cache introduce spare SRAM rows/columns [3], [4]. The spare rows/columns replace those containing faulty cells. While this approach is useful at low cell failure rates, it introduces excessive area and power overheads when cell-failure probability increases at low supply voltages [5].

Agarwal *et al.* propose a process-tolerant cache architecture using a cache line disabling technique, which detects and deletes cache lines with faulty cells [6]. In their scheme, the number of faulty cells and their locations are obtained and stored in a configurator, by performing a conventional Built-In Self-Test (BIST). Based on this information, the configurator will disable cache lines with faulty cells. Cache line disabling can be effective when the cell-failure probability is low. However, as the number of faulty cells increases at lower supply voltages, cache capacity shrinks rapidly.

Others use various error-correction codes (ECC) to enable fault tolerance for cache design. IBM Power 6 processors use single-error correcting double-error detecting (SECDED) extended Hamming codes to protect the L2 cache [7]. Kim *et al.* [8] use 2-D ECC to manage multi-bit clustered errors; unfortunately, this approach is not very effective in handling random faults (such as those associated with random dopant distribution in SRAM cells) at low supply voltages. Chishti *et al.* apply orthogonal Latin square codes (OLSC) to each cache segment and employ multi-bit segmented ECC (MS_ECC) to correct faulty cells in the low-power mode [9]. However, the OLSC code rate is low and between half and one-quarter of the cache is sacrificed to ECC check-bit storage. Alameldeen *et al.* employ variable-strength ECC (VS-ECC) [5]. They use low error correcting ECC in the common case and only use high error correcting ECC for those ways containing multiple faulty cells. Three different variable-strength ECC schemes are proposed and VS_ECC_Disabling is the most effective method among them. This scheme reserves one-bit error correction for soft errors. As for hard errors, it provides two-bit error correction for only 1/4 of 16 ways. No hard error protection is provided

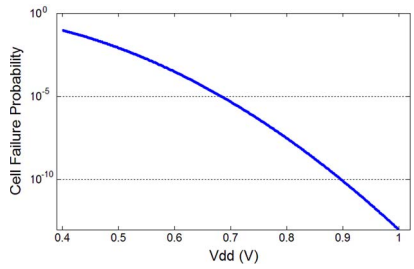

Fig. 1. SRAM cell failure probability VS. V_{DD}.

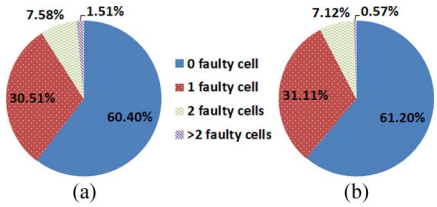

Fig. 2. Distribution of the number of persistent faulty cells in a cache line: (a) analysis and (b) simulation.

for the remaining 12 ways, making this approach ineffective for ultra-low supply voltage operation, where hard faults persist.

III. Our Proposed Approach: DECTED With Cache Line Disabling

In this section, we provide cell-failure models, on which our analysis and experiments are based. Then, we explore the distribution of *persistent* faulty cells in a single cache line to provide motivation for our approach. Our DECTED with cache line disabling method is then detailed.

A. SRAM Cell-Failure Models

Reducing V_{DD}, in the presence of process variations, can cause SRAM cell operation failure. Fig. 1 shows SRAM cell-failure probability (P_{FAIL}) as a function of supply voltage V_{DD}, extracted from [10], for a 45-nm process technology. Our current work uses 11-nm predictive technology but assumes a similar P_{FAIL}/V_{DD} ratio. Indeed, worsening variations will make P_{FAIL} higher and increase the need for improved fault-tolerance mechanisms [12].

B. Distribution of the Number of Persistently Faulty Cells in Single Cache Line

The majority of bit failures are *persistent*, and random dopant fluctuation (RDF) plays a primary role in causing them [9]. Cell failure resulting from RDF is believed to be random and independent. Therefore, the probability P_e of a cache line containing e persistently faulty cells can be related to the cell failure probability (P_{FAIL}) as

$$P_e = \binom{s}{e}(1 - P_{FAIL})^{s-e} P_{FAIL}^{e} \tag{1}$$

where s is the cache line size in bits.

Fig. 2(a) and (b) show the distribution of the number of **persistent** faulty cells in single cache line obtained by the analytical model as shown in (1) and simulation, respectively. We use a cache line size of 64 bytes, supply voltage of 0.34 V, and an 11-nm predictive technology node. As shown here, most

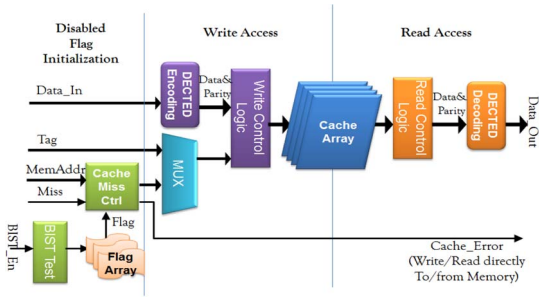

Fig. 3. Cache architecture of DECTED with cache line disabling.

cache lines contain only one *persistent* faulty cell, while only about 9% of cache lines contain multiple *persistent* faulty cells. Therefore, our idea is to provide 1-bit *hard* error correction for each cache line, and only disable cache lines containing *multiple* faulty cells. Because the probability of a cache line containing multiple faulty cells is low, we can maintain our cache capacity even at this low supply voltage. The other single-bit error correction of our DECTED code is reserved for *soft* errors encountered at runtime.

C. DECTED With Cache Line Disabling

Our key idea is to provide 1-bit error correction in each cache line for *hard* errors caused by low supply voltage, while disabling cache lines containing multiple faulty cells. In order to provide similar *soft* error correction capability as current commercial processors (e.g., IBM Power 6 and 7), we use DECTED codes for each cache line—1-bit error correction for *hard* errors and the other 1-bit error correction for *soft* errors. As in [6], the location and number of faulty cells (*hard error*) can be obtained statically using conventional BIST. Cache lines containing 0 or 1 error can be protected by 1-bit *hard* error correction capability. When there are multiple faulty cells, the cache lines will be disabled. Because the location of soft errors cannot be determined statically, one error-correction capability is reserved to provide similar *soft* error protection with SECDED and VS_ECC_Disabling.

The cache architecture of DECTED with cache line disabling is shown in Fig. 3. In the ultra-low-voltage mode, a memory test using BIST is first performed to determine the number of faulty cells in each cache line. Then, a *Disable* flag is set for cache lines with multiple faulty cells. On a cache miss, we only load data to healthy cache lines (those cache lines with 0 or 1 faulty cell) by checking the *Disable* flag. On a write access, we send the data and corresponding ECC parity bits into the healthy cache lines. On a read access, both the data and corresponding ECC check bits will be loaded from the cache array; the parity check bits will be used to correct faulty cell in the cache line.

IV. Cache Failure Probability Analysis

We compare, by analysis, cache failure probability of various cache fault-tolerance mechanisms. In order to provide a fair comparison, we assume that, as will be expected with modern cache designs, each method reserves 1-bit error correction for each cache line to address soft errors. We also assume that at least half of the cache lines are available in each cache set.

Our analysis is based on an eight-way cache, but can be easily adapted to other types of cache.

A. Cache Failure Probability Analysis

The cache failure probability P_{CFAIL} can be related to the cache set failure probability (P_{SFAIL}) as

$$P_{CFAIL} = 1 - (1 - P_{SFAIL})^{SET\_NUM} \quad (2)$$

where SET_NUM is the total number of sets in the cache. Now, we compare set failure probability for various cache fault-tolerance mechanisms.

SECDED can provide one-bit error correction for each cache line, MS_ECC provides four-bit error correction for every 64-bit data block, and cache line disabling (CLD) simply discards the cache line with faulty cells. Their respective cache set failure probabilities are

$$P_{SFAIL\_SECDED} = 1 - (1 - P_{e>1})^{s} \quad (3)$$

$$P_{SFAIL\_MS\_ECC} = 1 - \left(1 - \sum_{i=0}^{4} \binom{64}{i} P_{FAIL}^{i} (1 - P_{FAIL})^{64-i}\right)^{M_{64}} \quad (4)$$

$$P_{SFAIL\_CLD} = 1 - \sum_{i=w/2}^{w} \binom{w}{i} P_{e=0}^{i} (1 - P_{e=0})^{w-i} \quad (5)$$

where s is the set size in bits, w is the number of ways in each set, M_{64} is the number of 64-bit blocks in each set, $P_{e=0}$ is the probability of cache line with 0 faulty cell, and $P_{e>0}$ is the probability of cache line with > 0 faulty cells.

For hard errors, VS_ECC_Disabling provides two-error correction for 1/4 of all ways and no error correction for the remaining ways. Our analysis is based on an eight-way cache. Therefore, we assume two ways with two-error correction and no protection for the remaining ways. Then, the set failure probability using VS_ECC_Disabling (P_{SFAIL_VS}) is

$$P_{SFAIL\_VS} = 1 - \sum_{i=\frac{w}{2}-2}^{w} \times \sum_{j=\max\left(0,\frac{w}{2}-i\right)}^{w-i} \binom{w}{i+j} \binom{i+j}{j} P_{e=0}^{i} P_{1\le e\le 2}^{j} P_{e>2}^{w-i-j}. \quad (6)$$

The proposed approach provides one hard error correction capability and those cache lines containing multiple faulty cells will be discarded. Therefore, the cache failure probability using the proposed approach (P_{SFAIL_P}) is

$$P_{SFAIL\_P} = 1 - \sum_{i=w/2}^{w} \binom{w}{i} P_{e\le 1}^{i} (1 - P_{e>1})^{w-i}. \quad (7)$$

B. Cache Failure Probability and V_{DDMIN} Comparison

Fig. 4 plots cache failure probability versus V_{DD} using various fault-tolerance approaches. Here, our analysis is based on a 1024-tile system, where each tile contains an eight-way 256-KB cache, and cache line size is 64 bytes. As shown in the figure, the redundancy and parity introduced by various error control mechanisms can help improve yield. Among them, SECDED design provides the highest cache failure rate and the proposed approach obtains the lowest cache failure rate.

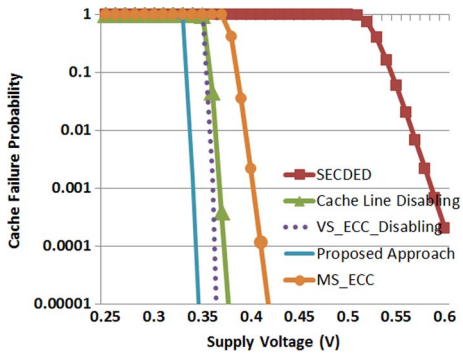

Fig. 4. Cache failure probability versus V_{DD}.

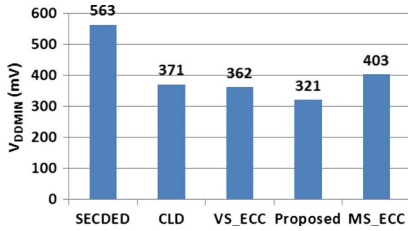

Fig. 5. V_{DDMIN} comparison.

Fig. 5 shows V_{DDMIN} comparison of cache using various fault-tolerance techniques. Here, we assume the yield loss rate target is 10^{-3}. As shown in the figure, the proposed approach reduces V_{DDMIN} by 43%, 14.5%, 11.4%, and 21.4%, respectively, comparing to SECDED, cache line disabling, VS_ECC_Disabling, and MS_ECC.

V. Experimental Results

In this section, we provide experimental results to evaluate and compare various cache fault-tolerance mechanisms in terms of energy consumption, energy-execution time product, and performance. Our experimental results show that the proposed approach achieves the best energy consumption and energy-execution time with negligible performance degradation.

A. Experimental Setup

Our evaluation is based on a 1024-tile many-core system in an 11-nm predictive tri-gate technology [14]. Each tile consists of a single-issue 64-bit core, four-way 32-KB private L1 instruction cache, four-way 32-KB private L1 data cache, and eight-way 256-KB private L2 cache (see Table I and Fig. 6). A mesh network-on-chip with worm-hole flow control is adopted. A Least Recently Used cache replacement policy is used for both L1 and L2 caches. We use a parallel many-core simulator, Graphite [16], which integrates CACTI [17], to evaluate performance and energy consumption.

B. Effective Cache Size and Cache Miss Rate Comparison

SECDED design provides a full-functional cache (256 MB). MS-ECC sacrifices half of the cache size; cache line disabling, VS_ECC_Disabling, and the proposed approach shrink cache size as supply voltage is reduced, and cell failure probability

TABLE I
1024-TILE SYSTEM PARAMETER

Parameters		Value
Tile Number		1024
Core Type		64 bits single-issue
L1-I/L1-D Cache, Private		
	Cache Block Size	64 bytes
	Cache Size	32 Kbytes
	Associativity	4
	Replacement Policy	LRU
L2 Cache, Private		
	Cache Block Size	64 bytes
	Cache Size	256 Kbytes
	Associativity	8
	Replacement Policy	LRU

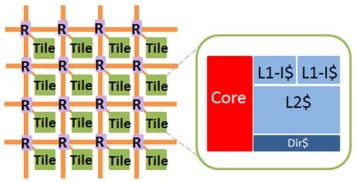

Fig. 6. 1024-Tile system architecture.

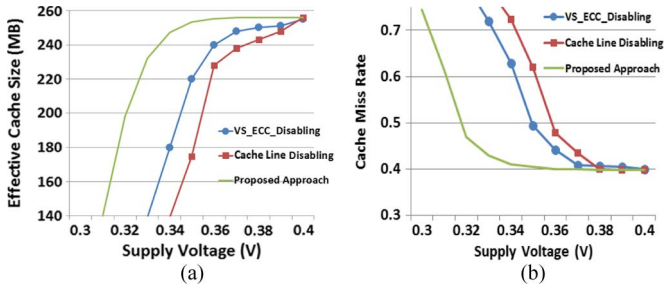

Fig. 7. (a) Effective cache size versus Vdd and (b) Cache miss rate versus Vdd (Barnes).

increases. Fig. 7(a) and (b) shows effective cache size versus supply voltage and cache miss rate versus supply voltage, respectively. As shown in the figure, our proposed approach achieves the highest effective cache size and the lowest cache miss rate among these three approaches at low supply voltages. As an example, when the supply voltage is 0.33 V, the proposed approach still provides 241-MB effective cache size, sacrificing no more than 10% cache size compared to the SECDED, while VS_ECC_Disabling and cache line disabling only provide 160 MB and 79.8 MB effective cache size, respectively. Meanwhile, our approach only increases the cache miss rate by 3% compared to the SECDED, while VS_ECC_Disabling and cache line disabling increase cache miss rate by 58% and 82%, respectively.

C. Energy Consumption and Energy-Execution Time Product

Figs. 8 and 9 show energy and energy-execution time product results of our 1024-tile many-core system using various fault-tolerance approaches. Leakage power because of disabled cache lines is also considered. As shown in the figure, our approach achieves the best energy consumption and energy-execution time product among all fault-tolerance methods. Specifically, our approach reduces energy consumption by 65%–76%, 26%–29%, 23%–25%, and 39%–45%, respectively, compared to SECDED, cache line disabling, VS_ECC_Disabling, and MS-ECC. Furthermore, our approach reduces energy-execution time product by 49%–53%, 7.8%–10.2%, 7%–9.1%, and 14.3%–22.4% compared to SECDED, cache line disabling, VS_ECC_Disabling, and MS_ECC.

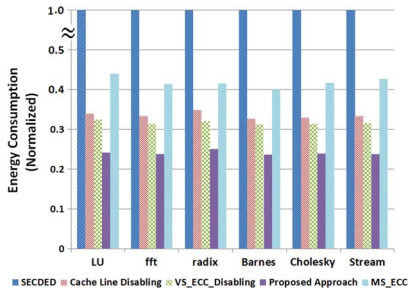

Fig. 8. Energy-consumption comparison.

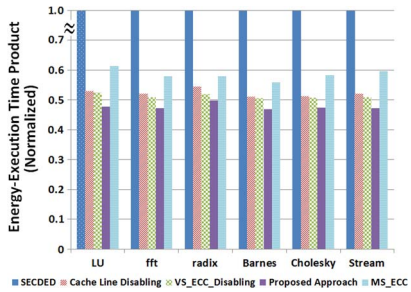

Fig. 9. Energy-execution time comparison.

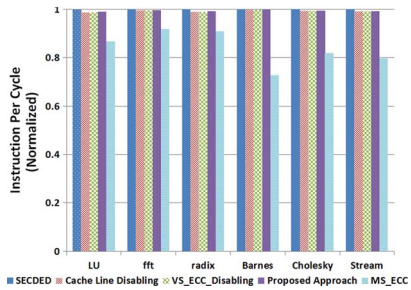

Fig. 10. Instructions per cycle (IPC) comparison.

D. IPC Performance Comparison

Various fault-tolerance mechanisms sacrifice effective cache size compared to the SECDED design to some extent. In addition, different error-correction codes can introduce extra access cycles because of critical time overhead introduced by encoding/decoding process. In our evaluation, we add one-cycle latency for SECDED, two-cycle latency for DECTED, and seven-cycle latency for 4EC5ED [5], [18]. To evaluate system performance of various mechanisms, Fig. 10 plots normalized instruction per cycle (IPC) using various fault-tolerance mechanisms. As shown in the figure, SECDED design provides the best IPC because all cache lines are used. Cache line disabling, VS_ECC_Disabling, and our proposed approach sacrifice marginal IPC degradation ($< 2\%$) because only a small portion of the cache lines is disabled in all these mechanisms. MS-ECC introduces the most severe IPC degradation because only half of the original cache is effectively available.

E. Supply Voltage Variation Tolerance Evaluation

Supply voltage variation, caused by IR or L*di/dt drop, is another source of cell failure. In this section, we evaluate supply voltage variation tolerance capability of cache line disabling,

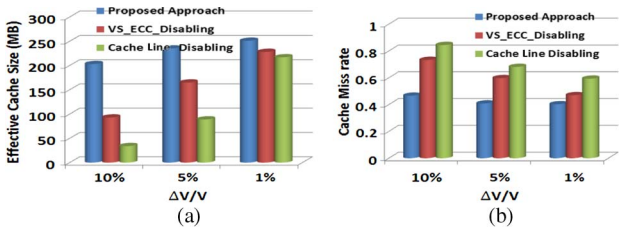

Fig. 11. Impact of supply voltage variation on (a) effective cache size and (b) cache miss rate.

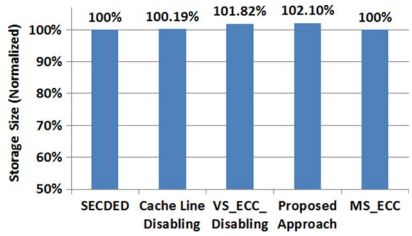

Fig. 12. Storage requirement comparison.

VS_ECC_Disabling, and the proposed approach. Fig. 11(a) and (b) show effective cache size and cache miss rate versus supply voltage variation $(\Delta V/V)$ for the Barnes benchmark at the supply voltage of 0.36 V (60% of nominal supply voltage). As shown in the figure, the proposed approach achieves higher effective cache size and lower cache miss rate for different supply voltage variation rate.

F. Storage Overhead Evaluation

To correct t-bit errors and detect $(t+1)$-bit errors, a BCH code requires t^*m+1 check bits, where m is the smallest possible number that meets the requirement $(k+t^*m+1) < 2^{\mathrm{m}}$.

Therefore, for a 64-byte (512-bits) cache line, SECDED ECC requires 11 check bits, DECTED requires 21 check bits, and 4EC5ED requires 41 check bits.

For SECDED and MS_ECC design, each set consists of eight cache lines, which has 523 bits. Thus, each set has 4184 bits. For cache line disabling, each cache line has 524 bits because it requires one extra status bit. Thus, each set has 4192 bits. For VS_ECC_Disabling, among one set there are two cache lines with 4EC5ED codes and six cache lines with SECDED codes. Moreover, each cache line has two status bits, and therefore there are 16 status bits in each set. Thus, each set has 4260 bits. For the proposed approach, each cache line has 534 bits (512 data bits +21 check bits + 1 status bit), and there are 4272 bits in each set. Therefore, compared to the SECDED design, MS_ECC has a storage size of 100%, cache line disabling has a storage size of 100.19%, VS_ECC_Disabling has a storage size of 101.82%, and the proposed approach has a storage size of 102.10% (as shown in Fig. 12). Compared to VS_ECC_Disabling, the proposed approach only introduces 0.28% storage overhead.

VI. Conclusion

Scaling cache supply voltage is an effective approach to improving overall system energy efficiency. However, the reduced supply voltage degrades performance and reliability. Our proposed approach exploiting DECTED with cache-line disabling maintains reliability and performance, while achieving the best energy consumption and energy-execution time product at the cost of trivial storage overhead. The experimental result shows that the proposed approach can reduce system energy consumption by up to 25% and energy-execution time by nearly 10% when compared to the next-best approach in the research literature, while only introducing 0.28% storage overhead and no IPC degradation. Our subsequent work will provide a fully dynamic fault-tolerance cache for dynamic frequency and voltage scaling processor design.

References

[1] U. Nawathe, M. Hassan, K. C. Yen, A. Kumar, A. Ramachandran, and D. Greenhill, "Implementation of an 8-Core, 64-Thread, power-efficient SPARC server on a chip," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, pp. 6–20, Jan. 2008.

[2] H. R. Ghasemi, S. C. Draper, and N. S. Kim, "Low-voltage on-chip cache architecture using heterogeneous cell sizes for high-performance processors," in *Proc. IEEE Int. Symp. High-Performance Comput. Architecture*, Feb. 2011, pp. 38–49.

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

[4] M. Horiguchi, "Redundancy techniques for high-density DRAMS," in *Proc. IEEE 2nd Annu. Int. Conf. Innov. Syst. Silicon*, Oct. 1997, pp. 22–29.

[5] A. Aladmeldeen, I. Wagner, Z. Chishti, W. Wu, C. Wilkerson, and S.-L. Lu, "Energy-efficient cache design using variable-strength error-correcting codes," in *Proc. 38th Int. Symp. Comput. Architecture*, 2011, pp. 461–472.

[6] A. Agarwal, B. C. Paul, H. Mahmoodi, A. Datta, and K. Roy, "A process-tolerant cache architecture for improved yield in nanoscale technologies," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 1, pp. 27–38, Jan. 2005.

[7] K. Reick, P. N. Sanda, S. Swaney, J. W. Kellington, M. J. Mack, M. S. Floyd, and D. Henderson, "Fault-tolerant design of the IBM Power6 microprocessor," *IEEE Micro*, vol. 28, no. 2, pp. 30–38, Mar./Apr. 2008.

[8] J. Kim, N. Hardavellas, K. Mai, B. Falsafi, and J. Hoe, "Multi-bit error tolerant caches using two-dimensional error coding," in *Proc. Int. Symp. Microarchitecture*, Dec. 2007, pp. 197–209.

[9] Z. Chishti, A. R. Alameldeen, C. Wilkerson, W. Wu, and S.-L. Lu, "Improving cache lifetime reliability at ultra-low voltages," in *Proc. Int. Symp. Microarchitecture*, Dec. 2009, pp. 89–99.

[10] Z. Guo, A. Carlson, L.-T. Pang, K. T. Duong, T.-J. K. Liu, and B. Nikolic, "Large-scale SRAM variability characterization in 45 nm CMOS," *IEEE J. Solid-State Circuits*, vol. 44, no. 11, pp. 3174–3192, Nov. 2009.

[11] S. T. Zhou, S. Katariya, H. Ghasemo, S. Draper, and N. S. Kim, "Minimizing total area of low-voltage SRAM arrays through joint optimization of cell size, redundancy, and ECC," in *Proc. IEEE ICCD*, 2010, pp. 112–117.

[12] C. Wilkerson, H. Gao, A. Alameldeen, Z. Chishti, M. Khellah, and S.-L. Lu, "Trading off cache capacity for reliability to enable low voltage operation," in *Proc. 35th ISCA*, 2008, pp. 203–214.

[13] J. Abella, J. Carretero, P. Chaparro, X. Vera, and A. Gonzáles, "Low VCCMIN fault-tolerant cache with highly predictable performance," in *Proc. MICRO*, Dec. 2009, pp. 111–121.

[14] G. Kurian, C. Sun, C.-H. O. Chen, J. E. Miller, J. Michel, L. Wei, D. A. Antoniadis, L.-S. Peh, L. Kimerling, V. Stojanovic, and A. Agarwal, "Cross-layer energy and performance evaluation of a nanophotonic many-core processor system using real application workloads," in *Proc. IEEE IPDPS*, 2012, pp. 1117–1130.

[15] S. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH-2 programs: Characterization and methodological considerations," in *Proc. 22nd Annu. Int. Symp. Comput. Architecture*, 1995, pp. 24–36.

[16] J. Miller, H. Kasture, G. Kurian, C. Gruenwald, N. Beckmann, C. Celio, J. Eastep, and A. Agarwal, "Graphite: A distributed parallel simulator for multicores," in *Proc. IEEE HPCA*, 2009, pp. 1–12.

[17] S. Thoziyoor, J. Ahn, M. Monchiero, J. Brockman, and N. Jouppi, "A comprehensive memory modeling tool and its application to the design and analysis of future memory hierarchies," in *Proc. ISCA*, 2008, pp. 51–62.

[18] R. Naseer and J. Draper, "Parallel double error correcting code design to mitigate multi-bit upsets in SRAMs," in *Proc. 34th Eur. Solid-State Circuits Conf.*, Edinburgh, U.K., 2008, pp. 222–225.