# Reprogrammable Redundancy for SRAM-Based Cache $V_{\text{min}}$ Reduction in a 28-nm RISC-V Processor

Brian Zimmer, *Member, IEEE*, Pi-Feng Chiu, *Student Member, IEEE*, Borivoje Nikolić, *Fellow, IEEE*, and Krste Asanović, *Fellow, IEEE*

***Abstract*— Reducing the operating voltage of digital systems improves energy efficiency, and the minimum operating voltage of a system ($V_{\text{min}}$) is commonly limited by SRAM bitcells. Common techniques to lower SRAM $V_{\text{min}}$ focus on using circuit-level periphery-assist techniques to prevent bitcell failures at low voltage. Alternatively, this paper proposes architecture-level techniques to allow caches to tolerate significant numbers of failing bitcells at low voltage while maintaining correct operation. The presented processor lowers SRAM-based cache $V_{\text{min}}$ using three architectural techniques–bit bypass, *dynamic column redundancy*, and line disable–that use low-overhead reprogrammable redundancy (RR) to increase the maximum tolerable bitcell failure rate and decrease the minimum operating voltage in processor caches. In a fabricated 28-nm RISC-V-based processor chip, these RR techniques add 2% area overhead to the cache and reduce the $V_{\text{min}}$ of the 1-MB L2 cache by 25%, resulting in a 49% power reduction.**

***Index Terms*— Bit bypass (BB), dynamic column redundancy (DCR), error-correcting codes (ECC), line disable (LD), redundancy, reprogrammable redundancy (RR), SRAM, $V_{\text{min}}$.**

## I. Introduction

IMPROVING energy efficiency is critical for a wide variety of digital systems, from power-constrained servers to battery-limited mobile devices. Reducing the supply voltage of digital designs is one of the most common ways to improve energy efficiency. However, the minimum supply voltage of a system is generally limited by SRAM, because the need for maximum capacity SRAM-based caches forces the transistors within each bitcell to extremely small dimensions, which increases the effects of process variations and causes bitcell failures. Therefore, further energy efficiency improvements in advanced process nodes require withstanding or preventing variation-induced SRAM bitcell failures in caches at low voltage.

Manuscript received February 13, 2017; revised May 9, 2017; accepted June 5, 2017. Date of publication July 18, 2017; date of current version September 21, 2017. This paper was approved by Guest Editor Jaeha Kim. This work was supported in part by DARPA PERFECT under Award HR0011-12-2-0016, in part by BWRC, in part by ASPIRE, in part by Intel ARO, and in part by TSMC. *(Corresponding author: Brian Zimmer.)*

B. Zimmer is with the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94720 USA, and also with NVIDIA, Santa Clara, CA 95050 USA (e-mail: bmzimmer@eecs.berkeley.edu).

P.-F. Chiu, B. Nikolić, and K. Asanović are with the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94720 USA.

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

Digital Object Identifier 10.1109/JSSC.2017.2715798

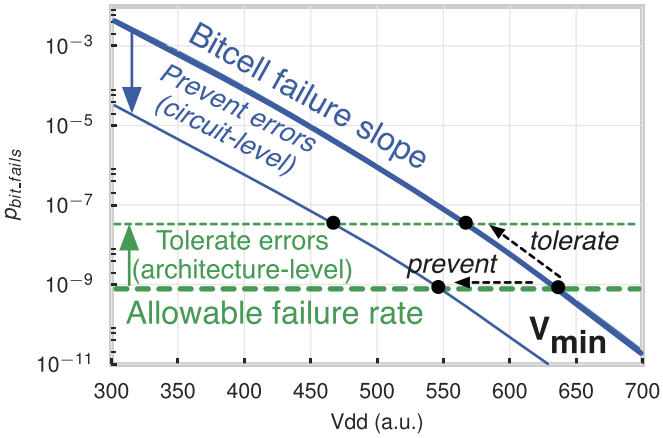



Fig. 1. Circuit-level techniques lower $V_{\text{min}}$ by improving bitcells, while architecture-level techniques lower $V_{\text{min}}$ by tolerating bitcell failures.

The transistor variation that causes SRAM failures can be overcome with peripheral circuit assist techniques that change wordline [1], bitline [2]–[4], cell supply [5], [6], or combination of voltages [7] on a cycle-by-cycle basis to strengthen or weaken particular devices during each operation. These techniques have been shown to significantly reduce $V_{\text{min}}$, but have a few major disadvantages. First, assist techniques require redevelopment for each new technology variant, as relative transistor strengths within the bitcell change. Second, the design cycle time is increased due to the need to qualify these techniques early in the design process. Third, these techniques cannot be targeted at only the weakened bitcells and must be enabled for all cells, which incur area and power overhead for the entire SRAM array.

Alternatively, architecture-level techniques increase the allowable failure rate by tolerating failing bitcells, as shown in Fig. 1, and can either supplant or supplement existing assist schemes. Parametric variations cause a wide spread of minimum operating voltages for bitcells in a chip. The increase in probability of bitcell failure ($p_{\text{bit\_fails}}$) due to a decrease in voltage, referred to here as the *failure slope*, is dictated by the process technology, SRAM bitcell, and SRAM periphery architecture. The 28-nm SRAM used in this paper has a measured failure slope of around 50 mV per decade—a 50-mV reduction in $V_{\text{DD}}$ increases the number of failures by ten times. Resiliency techniques are evaluated based on their ability to

increase the maximum allowable $p_{\text{bit\_fails}}$, which is translated into voltage reduction based on the failure slope. A gradual failure slope improves the effectiveness of architecture-level techniques, as the same $p_{\text{bit\_fails}}$ difference translates to a larger voltage difference.

To have adequate yield for large SRAM-based caches, the bitcell error rate must be extremely low—about $1\times10^{-10}$ for a 1-MB cache. A resiliency-modeling framework described in Section II evaluates the effectiveness of a variety of techniques by translating $p_{\text{bit\_fails}}$ to cache yield. Lightweight resiliency techniques, such as static redundancy [8], [9], achieve $V_{\text{min}}$ reduction at low cost by tolerating failures in a very small number of cells. However, static column and row redundancy can only cope with a limited number of failures as the overhead of the circuitry required scales poorly with increased protection. More aggressive techniques, such as line disable (LD) [10] or error-correcting codes (ECC) [11]–[13], can tolerate more failing cells and therefore a higher $p_{\text{bit\_fails}}$ than static redundancy, but still have limited effectiveness. The maximum $V_{\text{min}}$ reduction allowed by LD is limited by diminished cache capacity at high failure rates (as an entire line needs to be disabled to repair a single bit), and ECC effectiveness is limited by uncorrectable double-bit errors.

At high voltages, orders of magnitude changes in the failure rate translate to only small changes in the absolute number of failing bitcells, but at low voltages, the number of failing cells increases dramatically and trading off increased faults for decreased $V_{\text{DD}}$ becomes much less attractive. The limits of fault avoidance for $V_{\text{min}}$ reduction have been explored by aggressive architecture-level techniques. After identifying failing bits with an SRAM built-in-self-test (BIST), failing bits can be avoided by merging multiple words into a single operational word [14] or using offline graph algorithms to optimally match operational words [15], [16], but these approaches increase dynamic energy by accessing more bits than necessary. Another family of techniques uses multi-bit ECC to dynamically detect and repair errors at the cost of increased delay due to encoding, decoding, and correction [17]–[19], but are limited to use in last-level caches where the ECC latency can be tolerated. Words within a line can be disabled to trade capacity for $V_{\text{min}}$ [20], or a combination of LD to protect against multi-bit errors and ECC codes to protect against single-bit errors [21] can be used. The lack of known silicon implementations of these ideas reflects the high overhead costs and implementation complexity required to tolerate such high failure rates. This paper aims for a middle ground where minimal area overhead is emphasized over the lowest absolute $V_{\text{min}}$ possible.

This paper proposes a new set of architecture-level redundancy techniques—described in Section III—that avoid a large number of failing bitcells with low area overhead. The data portions of arrays are mainly protected by *dynamic column redundancy (DCR)*, which targets a sweet spot bitcell failure rate of $1\times10^{-4}$ and achieves a lower $V_{\text{min}}$ than other proposed techniques (such as static redundancy ECC and LD) with low area, delay, and energy overhead. $V_{\text{min}}$ is further reduced by supplementing DCR with LD to tolerate multi-bit failures,

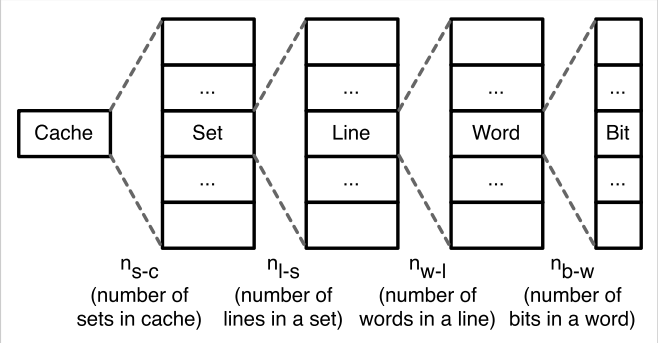



Fig. 2. Description of the notation used in error model framework that translates bitcell failure rate to system yield.

and another reprogrammable redundancy (RR) technique, *bit bypass* (BB), to protect against failures in the tag arrays without requiring SRAM assist techniques for the tag macros. The proposed techniques have a low enough overhead to be used in both the L1 and L2 caches, in comparison to many prior techniques that target L2 caches only. These techniques are verified through implementation in a 28-nm processor, and measurement results in Section IV show that the proposed RR techniques enable a 25% $V_{\text{min}}$ reduction with 2% area overhead.

The effectiveness of the proposed scheme is compared to existing schemes in Section V. The generic error model is further utilized to weigh the advantages and disadvantages of each technique, and provide intuitive explanations of the factors that determine the effectiveness of each architecture-level resiliency scheme.

## II. Generic Error-Modeling Framework

Investigating the relationship among voltage, bitcell failure rate, and cache yield is invaluable in developing effective $V_{\text{min}}$ reduction circuitry. This section proposes a generic framework that calculates cache yield as a function of bitcell error rate (or equivalently, voltage) to compare the effectiveness of the proposed scheme with that of prior schemes under a common set of assumptions. This generic framework uses a hierarchy of binomial distributions defined in Table I to translate bitcell failure probability to system failure probability for the cache structure described in Fig. 2. The distribution of failures in each level of the hierarchy can be represented by a binomial sample of the level below. For example, the number of failing lines in a set is a binomial sample of $n$ total lines in a set with a given probability $p$ of line failure. The probability that a given level in the hierarchy fails can be determined by evaluating the cumulative density function (CDF) of that level. For example, the probability that a set does not fail is the CDF evaluated at 0 (assuming no lines can fail in a set).

While using a binomial distribution to translate failure probabilities is a common practice, this new error model introduces a hierarchical combination of these distributions in a configuration that enables the evaluation of many different schemes by simply changing a few parameters. For example, single error correction and double error detection (SEC–DED)

TABLE I
GENERIC ERROR-MODELING FRAMEWORK

| Equations | Variables |
|---|---|
| $p_{bit\_fails} =$ Defined for given $V_{DD}$ | $a_{b-w} =$ Allowable number of bit failures in word |
| $X_{word} \sim Binomial(n_{b-w}, p_{bit\_fails})$ | $a_{w-l} =$ Allowable number of word failures in line |
| $p_{word\_fails} = 1 - \mathbb{P}(X_{word} \leq a_{b-w})$ | $a_{l-s} =$ Allowable number of line failures in set |
| $X_{line} \sim Binomial(n_{w-l}, p_{word\_fails})$ | $a_{s-c} =$ Allowable number of set failures in cache |
| $p_{line\_fails} = 1 - \mathbb{P}(X_{line} \leq a_{w-l})$ | $n_{b-w} =$ Number of bits in word |
| $X_{set} \sim Binomial(n_{l-s}, p_{line\_fails})$ | $n_{w-l} =$ Number of words in line |
| $p_{set\_fails} = 1 - \mathbb{P}(X_{set} \leq a_{l-s})$ | $n_{l-s} =$ Number of lines in set |
| $X_{cache} \sim Binomial(n_{s-c}, p_{set\_fails})$ | $n_{s-c} =$ Number of sets in cache |
| $p_{cache\_fails} = 1 - \mathbb{P}(X_{cache} \leq a_{s-c})$ | |

TABLE II
GENERIC ERROR MODEL REPORTS OF THE AVERAGE NUMBER OF FAILING BITS PER STRUCTURE IN THE 1-MB L2 CACHE

| Vdd | $p_{bit\_fails}$ | Per word (65,536 total) | | | | | | Per line (16,384 total) | | | | | | Per set (2,048 total) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | | 1 | | $\geq 2$ | | 0 | | 1 | | $\geq 2$ | | 0 | | 1 | | $\geq 2$ | |
| 325 | $1.8 \times 10^{-3}$ | 50999 | 77.8% | 12802 | 19.5% | 1735 | 2.6% | 6008 | 36.7% | 6033 | 36.8% | 4343 | 26.5% | 1 | 0.0% | 5 | 0.2% | 2042 | 99.7% |
| 350 | $6.9 \times 10^{-4}$ | 59601 | 90.9% | 5660 | 8.6% | 275 | 0.4% | 11208 | 68.4% | 4257 | 26.0% | 919 | 5.6% | 98 | 4.8% | 298 | 14.6% | 1651 | 80.6% |
| 375 | $2.4 \times 10^{-4}$ | 63375 | 96.7% | 2125 | 3.2% | 36 | 0.1% | 14327 | 87.4% | 1922 | 11.7% | 135 | 0.8% | 700 | 34.2% | 752 | 36.7% | 596 | 29.1% |
| 400 | $8.6 \times 10^{-5}$ | 64764 | 98.8% | 767 | 1.2% | 5 | 0.0% | 15626 | 95.4% | 741 | 4.5% | 18 | 0.1% | 1402 | 68.5% | 531 | 25.9% | 115 | 5.6% |
| 425 | $3.0 \times 10^{-5}$ | 65262 | 99.6% | 273 | 0.4% | 1 | 0.0% | 16112 | 98.3% | 270 | 1.6% | 2 | 0.0% | 1791 | 87.5% | 240 | 11.7% | 17 | 0.8% |
| 450 | $9.3 \times 10^{-6}$ | 65452 | 99.9% | 84 | 0.1% | 0 | 0.0% | 16300 | 99.5% | 84 | 0.5% | 0 | 0.0% | 1965 | 95.9% | 81 | 4.0% | 2 | 0.1% |
| 475 | $2.9 \times 10^{-6}$ | 65510 | 100.0% | 26 | 0.0% | 0 | 0.0% | 16358 | 99.8% | 26 | 0.2% | 0 | 0.0% | 2022 | 98.7% | 26 | 1.3% | 0 | 0.0% |
| 500 | $8.8 \times 10^{-7}$ | 65528 | 100.0% | 8 | 0.0% | 0 | 0.0% | 16376 | 100.0% | 8 | 0.0% | 0 | 0.0% | 2040 | 99.6% | 8 | 0.4% | 0 | 0.0% |
| 525 | $2.7 \times 10^{-7}$ | 65534 | 100.0% | 2 | 0.0% | 0 | 0.0% | 16382 | 100.0% | 2 | 0.0% | 0 | 0.0% | 2046 | 99.9% | 2 | 0.1% | 0 | 0.0% |
| 550 | $7.8 \times 10^{-8}$ | 65535 | 100.0% | 1 | 0.0% | 0 | 0.0% | 16383 | 100.0% | 1 | 0.0% | 0 | 0.0% | 2047 | 100.0% | 1 | 0.0% | 0 | 0.0% |
| 575 | $2.1 \times 10^{-8}$ | 65536 | 100.0% | 0 | 0.0% | 0 | 0.0% | 16384 | 100.0% | 0 | 0.0% | 0 | 0.0% | 2048 | 100.0% | 0 | 0.0% | 0 | 0.0% |

is described by the parameters $a_{b-w} = 1, a_{w-l} = 0, a_{l-s} = 0$, and $a_{s-c} = 0$, which represent the idea that one bit in every word can fail, no words can fail in a line, no lines can fail in a set, and no sets can fail in a cache.

Table II uses the generic error model to report the average number of failing bits in each cache structure for a L2 cache with $n_{b-w} = 138, n_{w-l} = 4, n_{l-s} = 8$, and $n_{s-c} = 2048$ (a 1-MB 8-way set associative cache with 64-B cache lines and 128-bit words with 10 extra bits for ECC and redundancy). The probability of bitcell failure ($p_{\text{bit\_fails}}$) is related to $V_{\text{DD}}$ using the measured SRAM failure rate from the prototype, described in Section IV. Then, the failing bitcells are examined in three ways: the per-word columns report the number of 138-bit words with 0, 1, or $\geq 2$ failing bits, the per-line columns report the number of $(138 \times 4)$-bit lines with 0, 1, or $\geq 2$ failing bits, and the per-set columns report the number of $(138 \times 4 \times 8)$-bit sets with 0, 1, or $\geq 2$ failing bits. At high voltages, only one or two bitcells will fail on average in each cache. However, as $V_{\text{DD}}$ decreases, $p_{\text{bit\_fails}}$ increases by about 10 times for every 50-mV reduction in $V_{\text{DD}}$, and the distribution of errors within each structure shifts from almost all entries having zero failing bits, to increasing chances of one or two bit failures. For example, at 450 mV, 96% of sets have zero failing bits, but at 350 mV, only 5% of sets have zero failing bits, and over 80% of sets have two or more failing bits.

### A. Determining $V_{min}$

$V_{\text{min}}$ is determined by finding which $p_{\text{bit\_fails}}$ corresponds to a target probability of cache failure ($p_{\text{cache\_fails}}$). Two different $p_{\text{cache\_fails}}$ are especially relevant, 0.5 and $1 \times 10^{-3}$, where the former reflects the voltage at which an average cache fails, and the latter reflects the voltage at which 99.9% of all caches work. Both assumptions account for the probability that a collection of bitcells with the same probability of failure at the same voltage create a condition that causes chip failure, but do not account for global process skew, which will cause the same $p_{\text{bit\_fails}}$ to occur at different voltages. If each die can be operated with a different $V_{\text{min}}$ value, then the $p_{\text{cache\_fails}}$ of 0.5 metric is more relevant as it represents the average $V_{\text{min}}$ reduction of all die, while a single $V_{\text{min}}$ target would require a $p_{\text{cache\_fails}}$ of $1 \times 10^{-3}$ to avoid yield fallout. This distinction will be revisited in Section V.

### B. Monte Carlo Verification

The generic error model has been verified through numerical Monte Carlo simulation, where 5000 fault maps are generated

at each $p_{\text{bit\_fails}}$ (voltage) by sampling from the binomial distribution for each bit in the 1-MB L2 cache. The bits are rearranged into words, lines, and sets based on the assumed cache organization. Different resiliency techniques can be implemented within this model, and each fault map returns a pass or fail value for each scheme. For example, a SEC-DED-protected cache will fail if any of the words contain more than one failing bit. Taking the mean of the pass value for many fault maps indicates the probability that the cache fails ($p_{\text{cache\_fails}}$). Although the Monte Carlo model is useful for validating the generic error model, it is slow to run and requires a huge number of samples to provide accurate information about the tail of the distribution.

### C. Optimal Bitcell Error Rate

Table II provides an intuitive basis to discussing the optimal $V_{\text{min}}$. Lower $V_{\text{min}}$ is not always better, as low voltages have a dramatic number of failures, and the energy overhead of techniques required to repair or avoid these failures will outweigh the energy reduction by a slightly lower $V_{\text{min}}$. In addition to dramatically increasing failure rates at low voltages, energy reduction returns diminish because exponential transistor delay increases near the threshold voltage cause a larger amount of leakage current to be integrated per clock cycle. As a result, the leakage portion of total energy consumption per operation dominates dynamic energy, and further reductions in voltage begin to actually increase energy per operation. This paper targets a $V_{\text{min}}$ with a failure rate of about $1 \times 10^{-4}$, which requires repairing less than 1000 bits in a 1-MB cache. A further 50-mV reduction beyond this point would require repairing closer to 10 000 bits.

## III. Reprogrammable Redundancy Implementation

Fig. 3 shows the system diagram and floorplan of the implemented processor with RR. The processor is based on a 64-bit RISC-V [22] single-issue in-order 6-stage pipeline [23]. To support dynamic voltage and frequency scaling, the processor is split into three independent voltage and frequency domains: the processor pipeline with L1 cache, the L2 cache, and the uncore I/O domain. The L1 consists of an 2-way 8-KB instruction and a 4-way 16-KB data cache with 8T-based macros. The L2 consists of an 8-way 4-bank 1-MB cache with high-density 6T-based macros. Both the L1 and L2 caches have 512-bit lines. The three architecture-level RR techniques protect all SRAM bitcells on the chip: BB protects tag arrays and DCR and LD protect the data portions of both the L1 and L2 caches. An at-speed SRAM BIST quickly identifies fault locations. Asynchronous first-input first-output (FIFO) queues and level shifters allow communication between the voltage and frequency islands. The host-target interface allows on-chip to off-chip communication for memory accesses (MEMIO) or configuration through the system control registers (SCR). Every SRAM includes SEC-DED protection. While testing RR, the correction capability is disabled and errors are simply logged to ensure identification of all SRAM errors. SEC-DED correction can be enabled to protect against soft errors or intermittent errors by reprogramming the redundant entries.

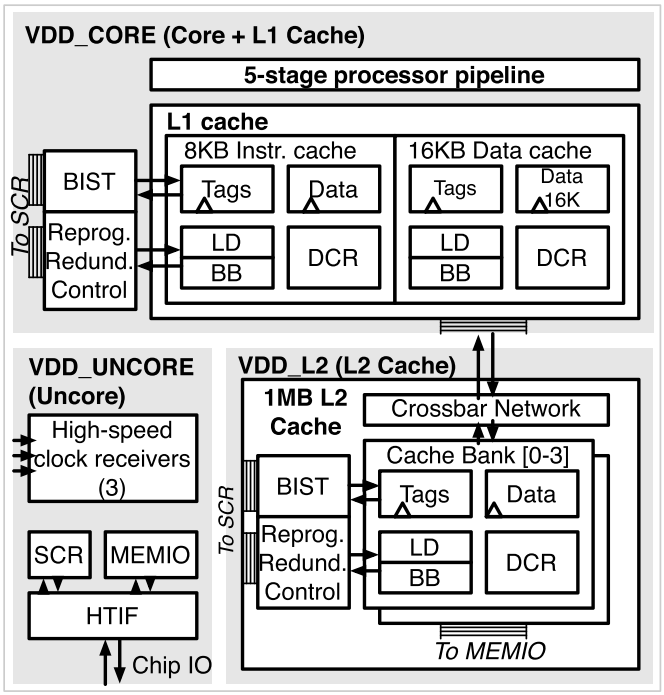



Fig. 3. Block diagram showing the organization of the 28-nm processor into three voltage domains containing the core with L1 cache, L2 cache, and uncore.

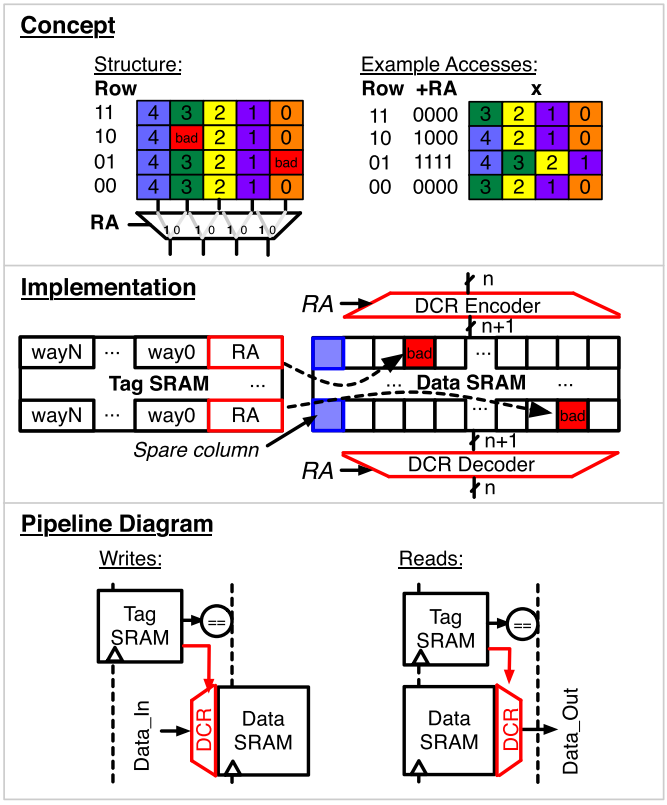



Fig. 4. DCR exploits cache architectures to repair a single bit per row in cache data arrays.

### A. Dynamic Column Redundancy (DCR)

Fig. 4 describes the dynamic column-redundancy technique. Column-redundancy schemes handle hard faults by adding

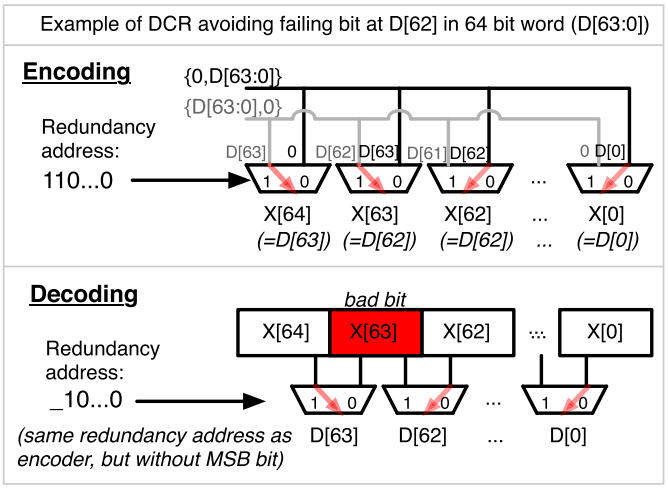



Fig. 5. DCR encoder and decoder add a single two-to-one multiplexer delay to the critical path.

a spare column to an SRAM array, with a two-way multiplexer at the bottom of each column to shift columns over to map out a failing column. Traditional static column-redundancy schemes are configured with fuses and correct one bit per array. The proposed DCR scheme associates a redundancy address (RA) with each row of the SRAM to dynamically select a different multiplexer shift position on each access. The RA is stored inside the cache tag array, and is shared between all lines in a set. In this implementation, shifting occurs outside the array to repair one bit per logical address in a set regardless of physical interleaving. The timing overhead is small, because the RA access occurs in parallel with the tag access and the shifting operations add a single multiplexer delay to the data setup and access time, as shown in Fig. 5. DCR offers similar resiliency to ECC, but at a much lower overhead. Unlike ECC, which generally requires codeword granularity to reflect access word size (to avoid read-modify-write operations), DCR granularity can be adjusted independent of access size. The RA width is log-base-two of the word size, which is assumed to be 128 bits for both the L1 and L2 caches. The proposed prototype repairs one bit per all 8 lines in a L2 cache set (4096 bits), requiring only a single 7-bit RA per 4096 protected bits. This technique is even more attractive for L1 caches, where ECC would require 8 checkbits for every 64-bit word, while DCR can use a 7-bit RA to protect 2048 bits. DCR enables a larger design space of resiliency versus area overhead tradeoffs. ECC is generally already required for soft error tolerance, and DCR can be easily supplemented by a SEC-DED code to protect against intermittent errors. In comparison, moving to double-bit ECC to protect against soft errors on a design that already uses single-bit ECC for voltage reduction is very expensive [13].

### *B. Bit Bypass*

Bitcell faults in tag arrays are protected using the BB scheme shown in Fig. 6. A redundant set repairs a single failing bitcell using flip-flops to store a replacement redundant bit along with the row and column address of the failing cell.

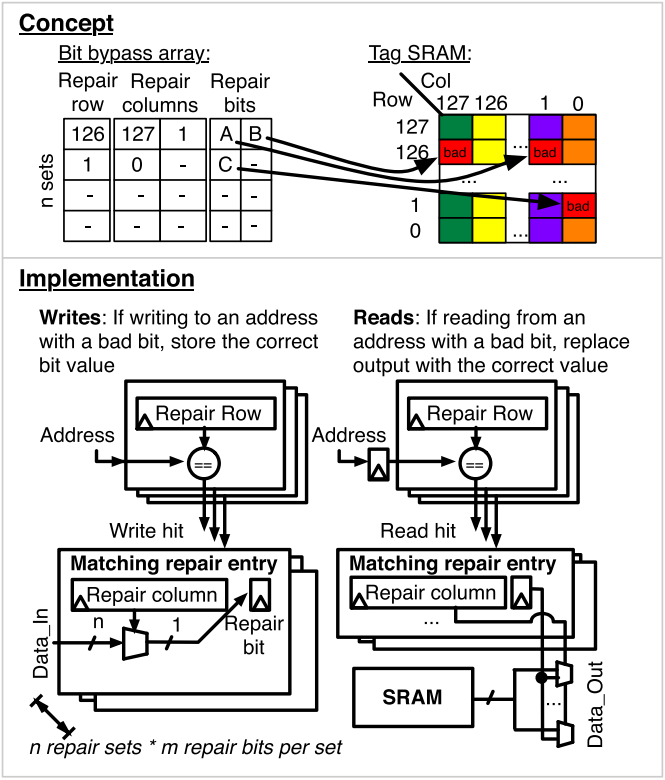



Fig. 6. Bit bypass protects against bit failures in any standalone SRAM macro by storing redundant copies of failing bits in flip-flops. The L1 BB array can correct up to 2 bits in 7 failing rows, while the L2 BB array can correct up to 2 bits in 22 failing rows. Note: only a single repair bit is shown for clarity, but there are actually two repair bits per repair entry.

Writes and reads to SRAM addresses with known faults use information from the redundant sets to bypass operations to failing bitcells. The targeted allowable $p_{\text{bit\_fails}}$ of a macro determines the total number of redundant sets required, and this technique can be added to protect any standalone SRAM macro in a digital design. BB adds a timing overhead of two multiplexers to the read delay. BB also requires a large write address setup time, but this overhead is not on the critical path.

In-array row redundancy could be more area efficient than BB. However, BB is proposed and used in this prototype as it requires no modifications to the SRAM compilers, which has significant practical advantages.

### *C. Line Disable (LD)*

$V_{\text{min}}$ is limited by a few multi-bit faults, instead of an overwhelming number of single-bit faults. While BB can repair multi-bit failures in the tag arrays, DCR can only repair single-bit failures in the data arrays. At the $V_{\text{DD}}$ where the first double-bit failure appears in a cache set, DCR is correcting less than 5% of the sets. LD [10] is used in the proposed scheme to prevent accesses to lines with multi-bit failures, which allows DCR correction to be better utilized. A maximum of 1% of lines are disabled to ensure that cache performance degradation due to diminished capacity is negligible. Disable bits are stored with each tag, and the cache-replacement algorithm avoids refills to disabled ways. Also, BB overhead could also be reduced using LD to disable ways with failing tag bits.

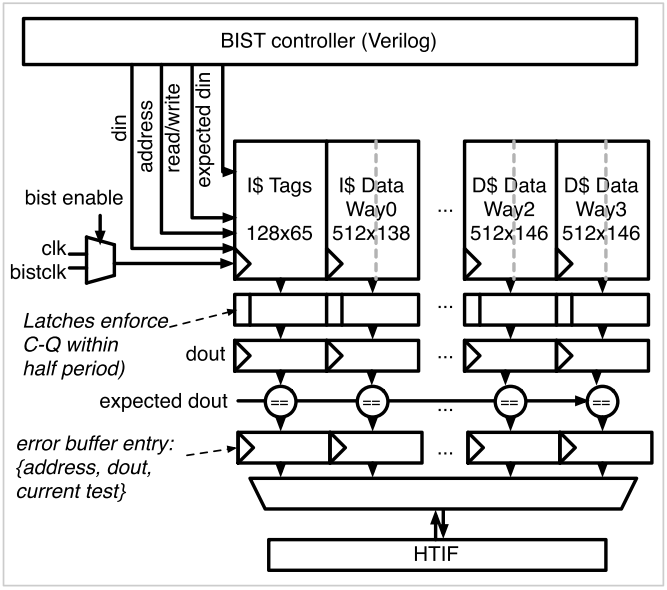



Fig. 7. SRAM BIST tests all SRAMs in parallel to identify fault locations.

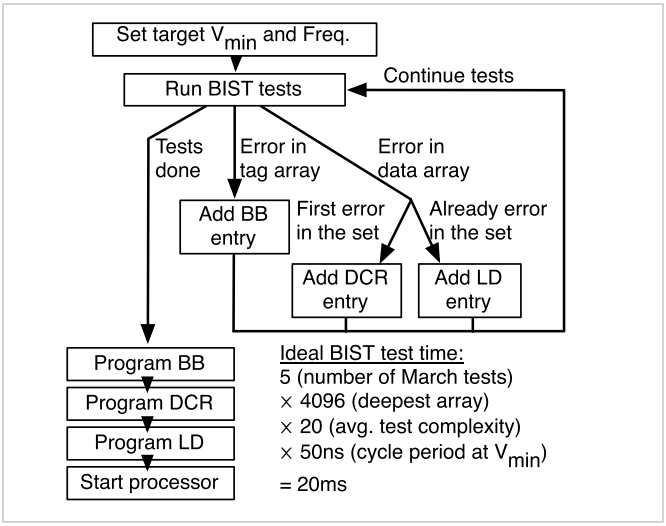



Fig. 8. Flowchart describing the RR programming algorithm.

### D. Runtime Configuration

The BIST detects SRAM error locations before operation, and uses the error information to program BB, DCR, and LD. Fig. 7 shows the BIST implementation that tests every SRAM array in parallel for errors, and sends the results off-chip. To ensure redundancy is correctly programmed during every power-up, testing results either need to be stored in non-volatile memory or retested and reprogrammed on every power-up. The flowchart in Fig. 8 summarizes the RR programming algorithm. In this testchip, the BIST is ran in a wide range of voltages and frequencies to identify $V_{\min}$ and the corresponding maximum frequency, while in a practical processor, existing binning techniques can be used to identify a smaller number of temperature, voltage, and frequency points. Other work has suggested that between two and thirteen tests are necessary to cover all faults [24]. Latency is estimated by assuming that five March tests with an average complexity of $20 \cdot n$ are run on every SRAM in parallel at a clock frequency of 50 ns (an approximate operation frequency at $V_{\min}$), and that the deepest array on the chip is 4096 entries.

To remove the need for non-volatile memory, BIST and RR programming can be run on every chip boot. As each memory is tested in parallel, the testing time is minimal. While the fabricated prototype uses an off-chip programming loop for flexibility, an on-chip implementation could test the SRAM in around 20 ms for each voltage and frequency point. The area of this hypothetical on-chip controller is not included in the area overhead calculation because the area should be very small. The controller only requires a small on-chip state machine that controls the read and write ports to the tag array based on failing bit locations. Because the DCR and LD information are stored in tag arrays, the bitcells in the tag array must be fixed first using BB before fixing bitcells in the data array with the LD and DCR entries. Power-on testing will still need to add margin for $V_{\min}$ changes due to temperature. However, it is assumed that these margins will also be necessary at nominal $V_{\min}$, so the overall voltage reduction will be almost identical. In addition, supplemental ECC could be leveraged to remove this margin with extremely minor impact on resiliency.

### E. Area Overhead

Table III reports the area overhead of the proposed techniques. For this implementation of DCR, the data arrays are nominally 137 bits wide (128 bits plus 9 ECC check bits), and require 1 extra column for DCR plus small shifting logic. The nominal tag array width is 216 bits, and LD adds 8 bits while the RA for DCR adds an additional 13 bits (8 bits plus 5 ECC check bits). The area overhead for DCR would be even lower without assuming SEC-DED protection. BB flip-flops and logic, implemented as standard cells, add about 15% to the tag macro area. The area overhead of the control logic to program both BB, DCR, and LD entries is tiny, as it only requires a 32-bit command from off-chip, which reuses an existing bus to access the SCRs. However, the relative area of the tag portion of the L2 cache is only 6% of the total area, so the entire RR scheme adds only 2% area overhead to the L2 cache. Using BB to protect the tag areas is the most expensive portion of the proposed technique, and assuming that tags are protected with an alternative mechanism would allow the overhead to be reduced to 1.2%. The area overhead of the BIST controller is not included as it is assumed that an on-chip BIST is already required for production testing. The area overhead of the unoptimized BIST in this implementation was 0.7% for the L2 cache.

### F. SRAM Coverage

The proposed techniques are intended to protect cache structures. The size of the cache will have an impact on effectiveness and area overhead. To help explore these factors, the proposed techniques are implemented on two very different cache structures: an L1 cache and an L2 cache. L1 caches are

TABLE III

AREA OVERHEAD OF THE PROPOSED RR SCHEMES. BASELINE DATA ARRAY SRAM IS 84% (90%) OF THE L1 (L2) CACHE AREA, AND TAG ARRAY SRAM IS 7% (6%) OF THE L1 (L2) CACHE AREA

| Technique | L1 Area Overhead | | | L2 Area Overhead | | |
|---|---|---|---|---|---|---|
| | Tag Array | Data Array | Total | Tag Array | Data Array | Total |
| BB | 20% | 0% | 1.4% | 15% | 0% | 0.9% |
| DCR | 12% | 1.4% | 2.0% | 6.3% | 0.70% | 1.0% |
| LD | 3.7% | 0% | 0.26% | 3.8% | 0% | 0.23% |
| DCR+BB | 32% | 1.4% | 3.4% | 21% | 0.70% | 1.9% |
| LD+BB | 24% | 0% | 1.7% | 19% | 0% | 1.1% |
| DCR+LD+BB | 36% | 1.4% | 3.7% | 25% | 0.7% | 2.1% |

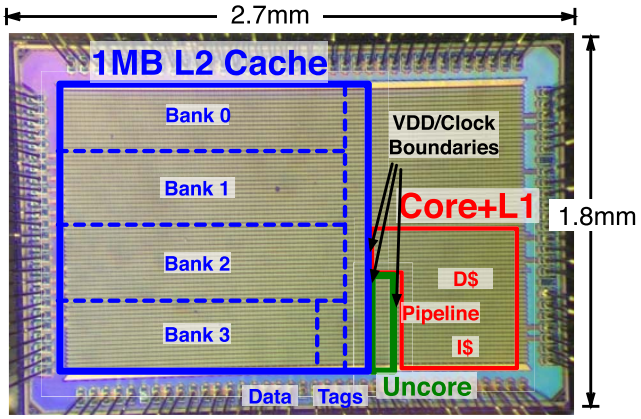



Fig. 9. Die micrograph with labeled processor core, L1 cache, and L2 cache.

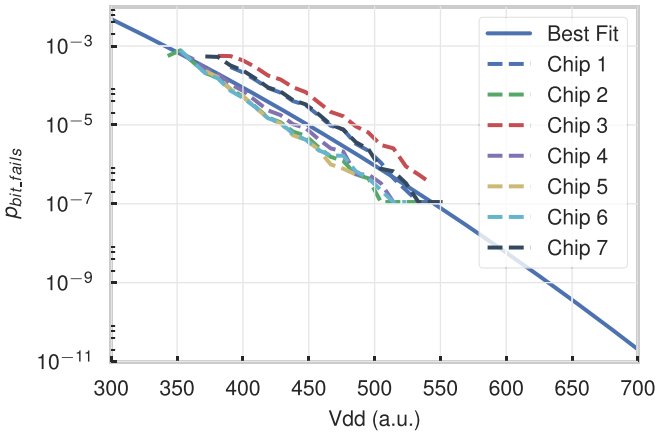



Fig. 10. Measured SRAM bitcell failure rate versus voltage of seven chips for the L2 cache (high-density 6T bitcells) with extrapolated best fit line.

TABLE IV

CHIP SUMMARY

| | |
|---|---|
| Technology | TSMC 28nm HPM |
| Die Size | 1.8 mm by 2.7 mm |
| Processor ISA | RISC-V RV64IMAFD/Sv39 |
| L1 Inst. Cache | 16KB, 2-way |
| L1 Data Cache | 32KB, 4-way |
| L2 Cache | 1MB, 8-way |

small and latency critical, while L2 caches are large and less latency sensitive, and provide points at the two extremes of the cache design space. The relative area overhead of RR will decrease for larger cache structures. The BB technique can protect all non-cache SRAMs. Even though BB has the highest overhead among the proposed techniques, the assumption is that non-cache SRAMs are a small portion of overall chip area overhead so the cost is reasonable.

The additional bad bit locations stored in the tag array for DCR and LD are also vulnerable to errors, but BB protects these bits along with the rest of the tag array based on fault information stored in flip-flops, which will not incur errors at low voltage.

## IV. MEASUREMENT RESULTS

The single-core RISC-V Rocket processor is fabricated in a TSMC 28-nm HPM process with a 1.8- by 2.7-mm die area, as shown in Fig. 9, with a chip summary provided in Table IV [25]. The test setup is similar to [26], where the die is wire-bonded on a PCB. The test board provides programmable voltage sources and connects the chip's digital IO to a Zedboard with a field programmable gate array containing a network-accessible ARM core. The BIST and RR algorithms are written in C and compiled to the testing ARM core, which communicates data and commands with the test chip. The measured bitcell failure rate of the L2 SRAM bitcells from a suite of March BIST tests is shown in Fig. 10. Measurements show that although the intrinsic absolute $V_{\min}$ of each die differs, the slope of the failure curve is consistent. The best-fit line represents the voltage versus bitcell failure characteristics for an average chip. The SRAM macros used in this paper are self-timed, and therefore, their failure rate is independent of operating frequency; faults are caused by all three modes of failure: read stability, readability, and writability. All following references to voltages are normalized, but this will not affect percent reduction results and these results hold for other processes with different absolute $V_{\min}$ values. The main technology feature that will affect these results is the slope of the bitcell failure rate curve. If the curve is steeper than shown in Fig. 10 (larger increase in failure rate for given voltage reduction), then the $V_{\min}$ reduction achieved by these techniques will be reduced, because the same difference in $p_{\text{bit\_fails}}$ corresponds to a smaller voltage difference. Likewise, if the curve is shallower than shown, then the $V_{\min}$ reduction achieved by these techniques will be improved. The results shown in this paper correspond to the measured failure slope

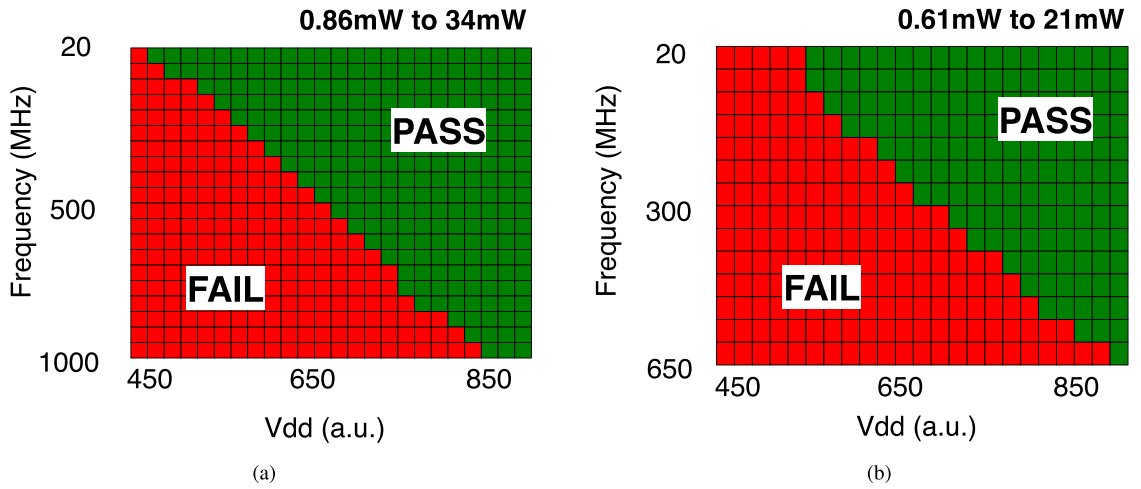



Fig. 11. Baseline L1 and L2 cache shmoo plots for benchmarks running on the processor without any of the RR techniques enabled. The L1 cache and processor core share the same voltage and frequency domain. (a) L1 cache and processor core. (b) L2 cache.

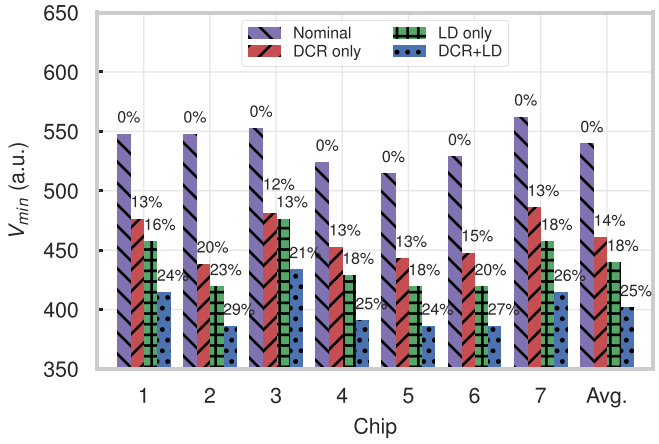



Fig. 12. Enabling the proposed RR reduces the minimum operating voltage of the L2 cache by 25%.

of this particular technology, and the generic error model of Section II can be used to re-evaluate these techniques with different assumptions about SRAM failure characteristics.

A variety of benchmarks have been ran on the processor at different voltages and frequencies with RR disabled. Fig. 11 shows the baseline shmoo for the L1 and L2 caches of seven measured chips with the proposed resiliency features disabled.

To verify the $V_{\min}$ reduction enabled by RR and test the actual implementation, RR is programmed with the fault information at $V_{\min}$, and the benchmarks are rerun at each voltage and frequency point. Fig. 12 shows the measured $V_{\min}$ of the L2 cache for three options for seven evaluated chips: only LD enabled, only DCR enabled, and a combination of DCR+LD. In all the three cases, BB is used to protect the tags, and less than 1% of cache lines are disabled. The frequency of operation is the same for all points, and is set to be the maximum frequency at $V_{\min}$ for the worst chip. Because the SRAM macros are self-timed, the frequency of operation does not affect the SRAM failure rate, so low frequency was used to ensure that critical paths did not affect the $V_{\min}$ reduction results. Enabling the RR techniques (BB and DCR+LD) decrease $V_{\min}$ by an average of 25% in the L2 cache, which resulted in a measured power reduction of 49%. BB is always enabled in this evaluation as it is required to protect the tag arrays and RAs from errors. The 8T-based cells in the L1 already have excellent low-voltage performance for the tested chips, so RR decreases $V_{\min}$ in the processor core as well, but only at voltages with extremely low frequencies that lie well below the energy-optimal point.

The correction capability of ECC is not used in Fig. 12 to decrease $V_{\min}$, and ECC is used only to confirm that all SRAM errors are identified during testing.

## V. Comparison to Existing Techniques

The effectiveness of the proposed technique is compared against previously proposed techniques using the generic error-modeling framework described in Section II.

Tables V and VI show the model parameters that are used to evaluate the effectiveness of the proposed scheme versus prior techniques. The exact L1 and L2 organization of the prototype was used as a representative sample of typical L1 and L2 caches. These parameters describe the data array portions of the cache. The number of BB entries was determined using the same generic error model applied to the tag arrays to determine how many errors would need to be tolerated in the tags to achieve the same $V_{\min}$ as DCR+LD. The static redundancy reference reports 8 column repairs and 16 row repairs per 2.5 MB [8], [9], which translates to 4 column and 8 repair rows per 1 MB for this paper, and can be simplified to simply checking for less than 12 failing sets in the cache. double error correction and triple error detection (DEC-TED) [11], [12] can correct a single failing bit per word (and the other correcting bit is reserved for soft errors). LD $V_{\min}$ is determined by allowing 1% of the lines to have errors (163/16 384 for the L2), and the capacity requirement of 99% is the limiting factor, not the probability that all lines in a set need to be disabled. DCR can correct 1 bit per set, and $n_{b-w}$ is modified to reflect this granularity of repair. DCR+LD enables a single failing bit per set, plus up to 1% failing lines. To aid analysis, the simplifying assumption

TABLE V

INPUTS TO EVALUATE THE MINIMUM OPERATING VOLTAGE ($V_{min}$) OF THE L1 CACHE USING THE PROPOSED GENERIC MODEL IN TABLE I FOR DIFFERENT ARCHITECTURE-LEVEL RESILIENCY TECHNIQUES

| Technique | Inputs to Table I | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $a_{b-w}$ | $a_{w-l}$ | $a_{l-s}$ | $a_{s-c}$ | $n_{b-w}$ | $n_{w-l}$ | $n_{l-s}$ | $n_{s-c}$ |
| Nominal | 0 | 0 | 0 | 0 | 73 | 8 | 4 | 128 |
| Static Redundancy [8], [9] | 0 | 0 | 0 | 12 | 73 | 8 | 4 | 128 |
| DEC-TED [11], [12], [13] | 1 | 0 | 0 | 0 | 73 | 8 | 4 | 128 |
| Line Disable [10] | 0 | 0 | 0 | 5 | 73 | 8 | 1 | 512 |
| **DCR+BB** | 1 | 0 | 0 | 0 | 2336 | 1 | 1 | 128 |
| **LD+BB** | 0 | 0 | 0 | 5 | 73 | 8 | 1 | 512 |
| **DCR+LD+BB** | 1 | 0 | 0 | 5 | 2336 | 1 | 1 | 128 |

TABLE VI

INPUTS TO EVALUATE THE MINIMUM OPERATING VOLTAGE ($V_{min}$) OF THE L2 CACHE USING THE PROPOSED GENERIC MODEL IN TABLE I FOR DIFFERENT ARCHITECTURE-LEVEL RESILIENCY TECHNIQUES

| Technique | Inputs to Table I | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $a_{b-w}$ | $a_{w-l}$ | $a_{l-s}$ | $a_{s-c}$ | $n_{b-w}$ | $n_{w-l}$ | $n_{l-s}$ | $n_{s-c}$ |
| Nominal | 0 | 0 | 0 | 0 | 138 | 4 | 8 | 2048 |
| Static Redundancy [8], [9] | 0 | 0 | 0 | 12 | 138 | 4 | 8 | 2048 |
| DEC-TED [11], [12], [13] | 1 | 0 | 0 | 0 | 138 | 4 | 8 | 2048 |
| Line Disable [10] | 0 | 0 | 0 | 163 | 138 | 4 | 1 | 16,384 |
| **DCR+BB** | 1 | 0 | 0 | 0 | 4416 | 1 | 1 | 2048 |
| **LD+BB** | 0 | 0 | 0 | 163 | 138 | 4 | 1 | 16,384 |
| **DCR+LD+BB** | 1 | 0 | 0 | 163 | 4416 | 1 | 1 | 2048 |

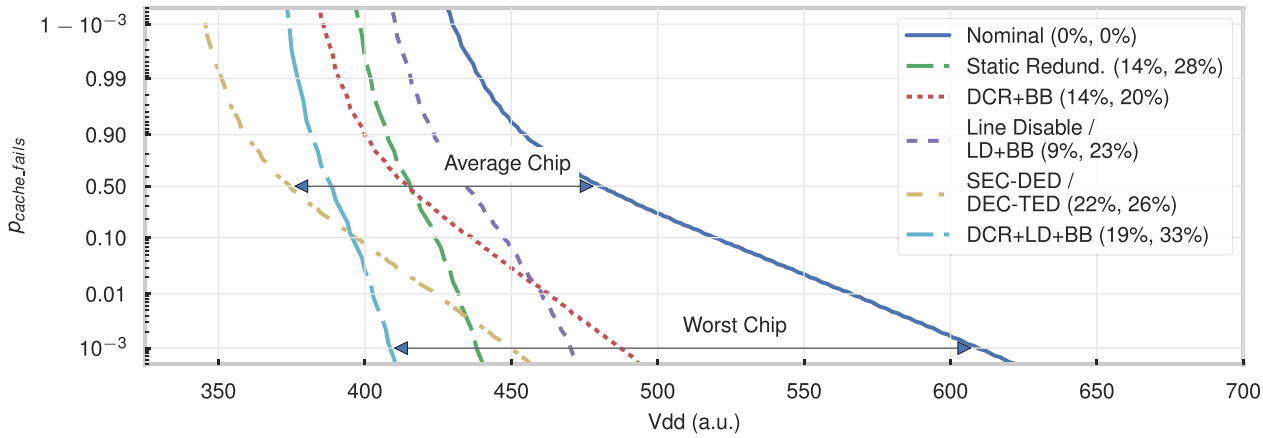
is made that every set failure is caused by a single line failure, which means that 163 failing sets out of 2048 total sets (for the L2) yield 99% capacity when disabling occurs on a line granularity. All simplifications were verified with Monte Carlo.

Fig. 13 compares the probability that a cache fails versus voltage for both the L1 and L2 caches, and reports the $V_{\text{min}}$ reduction percentage for both an average (50-th percentile) and worst case (99.9th percentile) chip as percentage reduction in the legend. The measured failure rate of the L2 data arrays from Fig. 10 was used to translate $p_{\text{bit\_fails}}$ to $V_{\text{DD}}$ for both the L1 and L2 caches for ease of comparison between different cache sizes. For the L2 cache, DCR+LD+BB achieves the largest $V_{\text{min}}$ reduction for both the average and worst case chip. For the L1, DEC-TED achieves slightly lower $V_{\text{min}}$ than DCR+LD+BB for the average case but higher $V_{\text{min}}$ for the worst case because the small number of entries makes double bit errors rare. Single-bit correction techniques show similar $V_{\text{min}}$ reduction for worst chips (27%) versus average chips (23%), because $V_{\text{min}}$ is always set by a rare failure condition—a single bit error in the entire cache for the nominal case or a double bit error in any word for the DEC-TED case. However, disable-based techniques such as LD show larger $V_{\text{min}}$ reduction for worst chips (34%) than average chips (21%), because while the nominal case is limited by rare conditions, $V_{\text{min}}$ of DCR+LD is limited by the maximum number of lines disabled, which essentially equals the average. The average chip case assumes unique per-chip $V_{\text{min}}$ binning, while the worst chip case assumes all chips are binned with the same exact $V_{\text{min}}$ (where low-voltage failures cannot cause greater than 0.1% yield fallout). For the following results, we use the more conservative metrics of average chip $V_{\text{min}}$ reduction, but the proposed techniques perform much better when evaluated using single-voltage binning assumptions. The predicted average $V_{\text{min}}$ reduction in Fig. 10 is the analytical prediction of the actual average measured $V_{\text{min}}$ reduction in Fig. 12.

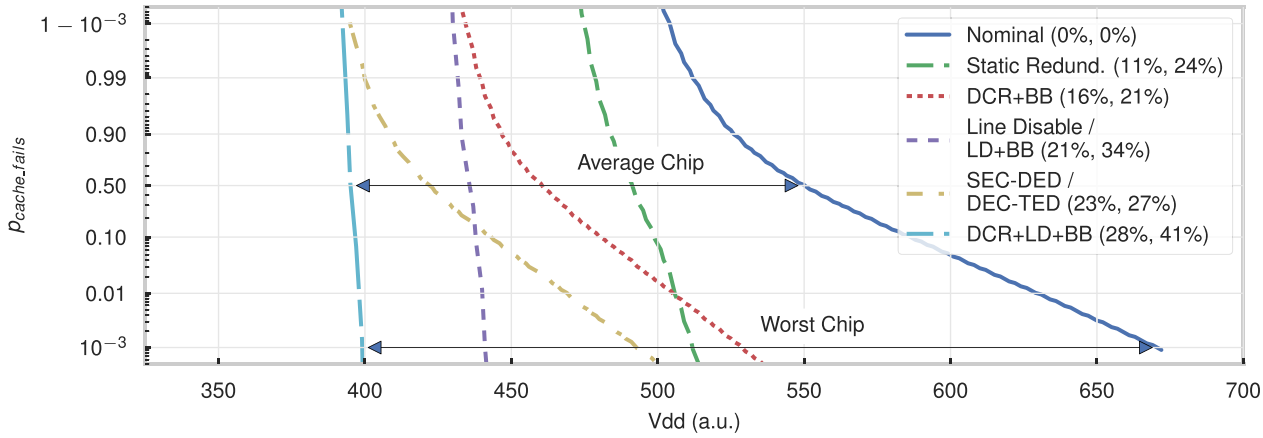
Table VII compares the resiliency, timing, and area overhead of the proposed technique compared to prior techniques. The cited techniques do not disclose their resiliency or overhead results, so the comparison points are best estimates.

Static redundancy is an efficient means to fix failures at low failure rates, but the area overhead scales poorly at higher failure rates. Power-on testing can be avoided with fuses, but large fuse area limits the number of possible repairs. Row redundancy needs to be integrated with the circuit-level SRAM design to gate broken wordlines.

DEC-TED is very expensive in terms of area overhead, especially for L1 caches with smaller word sizes. If soft-error protection is sacrificed, a lower overhead SEC-DED code could be used in place of a DEC-TED code, and would achieve the same $V_{\text{min}}$ reduction as reported for DEC-TED with less area overhead. However, both SEC-DED and DEC-TED require at least one cycle of latency to be inserted into the pipeline for decoding, which is problematic in an L1 cache. The advantage of ECC codes is that they do not require power-on testing.




(a)




(b)

Fig. 13. Comparison of different $V_{min}$ reduction techniques using the generic error model using parameters from Tables V and VI. The first number next to each technique in the legend reports the percentage $V_{min}$ reduction from nominal for an average chip, while the second number reports the percentage $V_{min}$ reduction for the 99.9th percentile (worst) chip. (a) 32-KB L1 data cache. (b) 1-MB L2 cache.

TABLE VII

COMPARISON OF L1 AND L2 CACHES AREA OVERHEAD AND ANALYTICAL $V_{min}$ REDUCTION FOR DIFFERENT TECHNIQUES. BASELINE DATA ARRAY SRAM IS 84% (90%) OF THE L1 (L2) CACHE AREA, AND TAG ARRAY SRAM IS 7% (6%) OF THE L1 (L2) CACHE AREA

| Technique | Protects Tag Arrays | Soft Error Protection | Requires SRAM Changes | Requires Power-on Testing | L1 Data Cache | | L2 Cache | |
|---|---|---|---|---|---|---|---|---|
| | | | | | $V_{min}$ (% Red.) | Area Overhead | $V_{min}$ (% Red.) | Area Overhead |
| Nominal | No | No | No | No | 480 (0%) | - | 550 (0%) | - |
| Static Redund. [8], [9]* | Yes | Compatible | Yes | Maybe | 415 (14%) | 1.0% | 491 (11%) | 1.0% |
| SEC-DED † | Yes | No | No | No | 374 (22%) | 11.4% | 423 (23%) | 6.7% |
| DEC-TED ††[11], [12], [13] | Yes | Yes | No | No | 374 (22%) | 20% | 423 (23%) | 12% |
| Line Disable [10] ‡ | No | Compatible | No | Yes | 435 (9%) | 0.26% | 435 (21%) | 0.23% |
| **DCR+BB** | Yes | Compatible | No | Yes | 415 (14%) | 3.4% | 461 (16%) | 1.9% |
| **LD+BB**‡ | Yes | Compatible | No | Yes | 435 (9%) | 1.7% | 423 (21%) | 1.1% |
| **DCR+LD+BB**‡ | Yes | Compatible | No | Yes | 389 (19%) | 3.7% | 395 (28%) | 2.1% |

* 12 repairs per MB, no power-on test if fused during manufacturing test
† 1 bit repair + 0 bits reserved for soft errors per 64 (L1) or 128 (L2) bits
†† 1 bit repair + 1 bit reserved for soft errors per 64 (L1) or 128 (L2) bits
‡ Up to 1% disabled

LD is extremely area efficient as only one bit is required to indicate a disabled line, but $V_{min}$ reduction is limited at higher failure rates due to a rapid decrease in capacity. LD is significantly more effective in L2 caches than L1 caches as ratio of line size to total cache size is smaller. LD alone requires a separate technique to protect the tag array,

where the disable bits are stored, which will add area overhead, and the total overhead of combining LD with BB to protect the tags is listed. Bit bypass looks expensive at around 15% area overhead, but is still significantly cheaper than transitioning from a 6T to an 8T bitcell, which would require a 100% area overhead.

For all schemes that are compatible with (but do not include) ECC, a SEC-DED code can be added for soft-error protection at the cost of 7% for the L2 (and 12.5% for the L1), but this decision can be decoupled from $V_{\min}$ reduction method.

If further $V_{\min}$ reduction is required, DCR+LD+BB can be programmed to allow up to 10% of cache lines to be disabled (instead of only 1% of cache lines), which will decrease $V_{\min}$ by an additional 8% for both the L1 and L2 caches, but could affect energy per operation results due to increased cache misses.

In comparison to circuit techniques that report 130–200 mV of $V_{\min}$ reduction for 5–7% area overhead [7], [4], the proposed techniques can achieve comparable effectiveness with less area and power overhead. However, direct comparison is difficult as modern bitcells are not designed to be used without SRAM assist, so the reported $V_{\min}$ reduction is optimistic. RR protects against all failure mechanisms, while circuit assist techniques must carefully tune assists to tradeoff between different failure mechanisms. In addition, RR effectiveness is easier to predict at design time than assist technique effectiveness as long as the failure slope can be bounded.

Comparison to conceptual architecture-level techniques without silicon implementations summarized in Section I has not been included, as area comparisons are unobtainable. These schemes can be analyzed with the generic error model to show that many achieve similar $V_{\min}$ as the proposed technique. All either require serial tag and data access or have long-latency ECC decoding overheads and, therefore, could work in L2 caches but not L1 caches.

The main disadvantage of the proposed techniques is that they require power-on BIST or non-volatile storage of bitcell fault locations. The RAs and disabled lines need to either be calculated on power-up by running BIST on every power up, or the results of the BIST need to be stored in a non-volatile memory off-chip.

## VI. Conclusion

The three proposed RR techniques, DCR, BB, and LD, reduce power in the L2 cache by 49% through improved supply voltage scaling with less than 2% area overhead and minimal timing overhead, and can be combined with existing assist techniques to enable further $V_{\min}$ reduction, enriching the tradeoff space between low voltage reliability and overhead. RR is particularly attractive because it can be used in conjunction with simple SEC-DED codes to protect against soft errors, and online reconfiguration based on ECC results or power-on tests can reduce voltage margin required for intermittent and aging-related faults.

## Acknowledgment

The authors would like to thank Y. Lee, A. Waterman, H. Cook, S. Twigg, B. Richards, J. Dunn, S. Liao, and J. Chang for their contributions.

## References

[1] M. E. Sinangil, H. Mair, and A. P. Chandrakasan, "A 28 nm high-density 6T SRAM with optimized peripheral-assist circuits for operation down to 0.6 V," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2011, pp. 260–262.

[2] M. Yabuuchi, K. Nii, Y. Tsukamoto, S. Ohbayashi, Y. Nakase, and H. Shinohara, "A 45 nm 0.6 V cross-point 8T SRAM with negative biased read/write assist," in *IEEE Symp. VLSI Circuits Dig.*, Jun. 2009, pp. 158–159.

[3] H. Pilo *et al.*, "A 64 Mb SRAM in 32 nm high-$k$ metal-gate SOI technology with 0.7 V operation enabled by stability, write-ability and read-ability enhancements," *IEEE J. Solid-State Circuits*, vol. 47, no. 1, pp. 97–106, Jan. 2012.

[4] T. Song *et al.*, "A 10 nm FinFET 128 Mb SRAM with assist adjustment system for power, performance, and area optimization," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Jan. 2016, pp. 306–307.

[5] E. Karl *et al.*, "A 4.6 GHz 162 Mb SRAM design in 22 nm tri-gate CMOS technology with integrated active $V_{MIN}$-enhancing assist circuitry," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers (ISSCC)*, Feb. 2012, pp. 230–232.

[6] E. Karl *et al.*, "A 0.6 V 1.5 GHz 84 Mb SRAM design in 14 nm FinFET CMOS technology," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2015, pp. 1–3.

[7] J. Chang *et al.*, "A 20 nm 112 Mb SRAM in high-$\kappa$ metal-gate with assist circuitry for low-leakage and low-$V_{MIN}$ applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2013, pp. 316–317.

[8] M. Huang, M. Mehalel, R. Arvapalli, and S. He, "An energy efficient 32-nm 20-MB shared on-die L3 cache for Intel Xeon processor E5 family," *IEEE J. Solid-State Circuits*, vol. 48, no. 8, pp. 1954–1962, Aug. 2013.

[9] W. Chen *et al.*, "A 22 nm 2.5 MB slice on-die L3 cache for the next generation Xeon processor," in *Proc. Symp. VLSI Technol.*, Jun. 2013, pp. C132–C133.

[10] J. Chang *et al.*, "The 65-nm 16-MB shared on-die L3 cache for the dual-core Intel Xeon processor 7100 series," *IEEE J. Solid-State Circuits*, vol. 42, no. 4, pp. 846–852, Apr. 2007.

[11] S. Sawant *et al.*, "A 32 nm Westmere-EX Xeon enterprise processor," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2011, pp. 74–75.

[12] G. Shamanna, R. Gaurav, Y. K. Raghavendra, P. Marfatia, and B. Kshatri, "Using ECC and redundancy to minimize VMIN induced yield loss in 6T SRAM arrays," in *Proc. IEEE Int. Conf. IC Design Technol. (ICICDT)*, May/Jun. 2012, pp. 1–4.

[13] R. Naseer and J. Draper, "DEC ECC design to improve memory reliability in sub-100 nm technologies," in *Proc. IEEE Int. Conf. Electron., Circuits Syst. (ICECS)*, Aug./Sep. 2008, pp. 586–589.

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

[15] T. Mahmood, S. Kim, and S. Hong, "Macho: A failure model-oriented adaptive cache architecture to enable near-threshold voltage scaling," in *Proc. IEEE 19th Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2013, pp. 532–541.

[16] A. Ansari, S. Feng, S. Gupta, and S. Mahlke, "Archipelago: A polymorphic cache design for enabling robust near-threshold operation," in *Proc. IEEE 17th Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2011, pp. 539–550.

[17] T. N. Miller, R. Thomas, J. Dinan, B. Adcock, and R. Teodorescu, "Parichute: Generalized turbocode-based error correction for near-threshold caches," in *Proc. 43rd Annu. IEEE/ACM Int. Symp. Microarchitecture*, Dec. 2010, pp. 351–362.

[18] Z. Chishti, A. R. Alameldeen, C. Wilkerson, W. Wu, and S.-L. Lu, "Improving cache lifetime reliability at ultra-low voltages," in *Proc. 42nd Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, vol. 42. Dec. 2009, pp. 89–99.

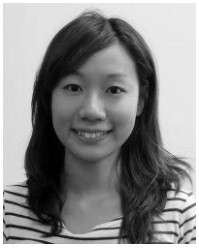
[19] A. R. Alameldeen, I. Wagner, Z. Chishti, W. Wu, C. Wilkerson, and S.-L. Lu, "Energy-efficient cache design using variable-strength error-correcting codes," in *Proc. 38th Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2011, pp. 461–472.

[20] J. Abella, J. Carretero, P. Chaparro, X. Vera, and A. González, "Low Vccmin fault-tolerant cache with highly predictable performance," in *Proc. 42nd Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, vol. 42. Dec. 2009, pp. 111–121.

[21] M. Zhang, V. M. Stojanovic, and P. Ampadu, "Reliable ultra-low-voltage cache design for many-core systems," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 59, no. 12, pp. 858–862, Dec. 2012.
[22] A. Waterman, Y. Lee, D. A. Patterson, and K. Asanović. (May 2014). "The RISC-V instruction set manual, volume I: User-level ISA, version 2.0," Dept. EECS, Univ. California, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2014-54. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-54.html
[23] Y. Lee *et al.*, "A 45 nm 1.3 GHz 16.7 double-precision GFLOPS/W RISC-V processor with vector accelerators," in *Proc. Eur. Solid State Circuits Conf. (ESSCIRC)*, Sep. 2014, pp. 199–202.
[24] M. Linder, A. Eder, U. Schlichtmann, and K. Oberlander, "An analysis of industrial SRAM test results—A comprehensive study on effectiveness and classification of march test algorithms," *IEEE Des. Test*, vol. 31, no. 3, pp. 42–53, Jun. 2014
[25] B. Zimmer, P.-F. Chiu, B. Nikolić, and K. Asanović, "Reprogrammable redundancy for cache $V_{min}$ reduction in a 28 nm RISC-V processor," in *Proc. IEEE Asian Solid-State Circuits Conf.*, Nov. 2016, pp. 121–124.
[26] B. Zimmer *et al.*, "A RISC-V vector processor with simultaneous-switching switched-capacitor DC-DC converters in 28 nm FDSOI," *IEEE J. Solid-State Circuits*, vol. 51, no. 4, pp. 930–942, Apr. 2016.

**Brian Zimmer** (S'09–M'15) received the B.S. degree in electrical engineering from the University of California, Davis, CA, USA, in 2010, and the M.S. and Ph.D. degrees in electrical engineering and computer sciences from the University of California, Berkeley, CA, USA, in 2012 and 2015, respectively.

He is currently with the Circuits Research Group, NVIDIA Corporation, Santa Clara, CA, USA. His current research interests include soft error resilience and energy-efficient digital design, with an emphasis on low-voltage SRAM design and variation tolerance.



**Pi-Feng Chiu** (S'10) received the B.S. and M.S. degrees in electronic engineering from National Tsing Hua University, Hsinchu, Taiwan, in 2009 and 2010, respectively. She is currently pursuing the Ph.D. degree in electrical engineering with the University of California, Berkeley, CA, USA.

She was with the Industrial Technology Research Institute, Hsinchu, from 2010 to 2012. She joined the Berkeley Wireless Research Center, Berkeley, in 2012. She was an Intern with Samsung Electronics in 2014 and with Western Digital, San Jose, CA, USA, in 2016. Her current research interests include resilient memory circuit design and emerging nonvolatile memories.

**Borivoje Nikolić** (S'93–M'99–SM'05–F'17) received the Dipl.Ing. and M.Sc. degrees in electrical engineering from the University of Belgrade, Belgrade, Serbia, in 1992 and 1994, respectively, and the Ph.D. degree from the University of California at Davis, Davis, CA, USA, in 1999.

He joined the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA, USA, in 1999, where he is currently a National Semiconductor Distinguished Professor of Engineering. He has co-authored the book *Digital Integrated Circuits: A Design Perspective* (2nd ed., Prentice-Hall, 2003). His current research interests include digital, analog, and RF integrated circuit design and VLSI implementation of communications and signal processing systems.

Dr. Nikolić was a recipient of the NSF CAREER Award in 2003, the College of Engineering Best Doctoral Dissertation Prize, and the Anil K. Jain Prize for the Best Doctoral Dissertation in Electrical and Computer Engineering at University of California at Davis in 1999, as well as the City of Belgrade Award for the Best Diploma Thesis in 1992. Working with his students and colleagues, he has received the Best Paper Awards at the IEEE International Solid-State Circuits Conference, the Symposium on VLSI Circuits, the IEEE International SOI Conference, the European Solid-State Device Research Conference, the European Solid-State Circuits Conference, the S3S Conference, and the ACM/IEEE International Symposium of Low-Power Electronics. He was a Distinguished Lecturer of the IEEE Solid-State Circuits Society from 2014 to 2015.

**Krste Asanović** (S'90–M'98–SM'12–F'14) received the B.A. degree in electrical and information sciences from the University of Cambridge, Cambridge, U.K., in 1987, and the Ph.D. degree in computer science from the University of California at Berkeley, Berkeley, CA, USA, in 1998.

He was an Assistant and later an Associate Professor of Electrical Engineering and Computer Science with the Massachusetts Institute of Technology, Cambridge, MA, USA, from 1998 to 2007. He is currently a Professor of Electrical Engineering and Computer Sciences with the University of California at Berkeley. His current research interests include computer architecture, very-large-scale integration design, and parallel programming and run-time systems.

Prof. Asanović is an ACM Distinguished Scientist.