Reprogrammable Redundancy for Cache V_{min} Reduction in a 28nm RISC-V Processor

Brian Zimmer, Pi-Feng Chiu, Borivoje Nikolić, and Krste Asanović

Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley

Abstract

The presented processor lowers SRAM-based cache V_{min} by using three architectural techniques–bit bypass (BB), dynamic column redundancy (DCR), and line disable (LD)–that use low-overhead reprogrammable redundancy (RR) to avoid failing bitcells and therefore increase the maximum bitcell failure rate in processor caches. In the 28nm chip, the V_{min} of the 1MB L2 cache is reduced by 25%, resulting in a 49% power reduction with a 2% area overhead and minimal timing overhead.

Introduction

Lowering the minimum operating voltage of SRAM-based caches (V_{min}) improves the energy efficiency of digital systems. A wide variety of circuit-level assist techniques have been proposed to reduce V_{min} by reducing bitcell failure rate [1], but require redevelopment for each new technology and present high area and power overhead. Alternatively, architecture-level techniques increase the allowable failure rate by tolerating failing bitcells, as shown in Figure 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 bitcell error rate (BER) 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 28nm SRAM used in this work has a measured failure slope of around 50mV per decade—a 50mV reduction in V_{DD} increases the number of failures by ten times. Resiliency techniques are evaluated based on their ability to increase the maximum allowable BER, and BER is translated into voltage reduction based on the failure slope. A gradual failure slope improves the effectiveness of architecture-level techniques, as the same BER 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 1×10^{-10} for a 1MB cache. A resiliency modeling framework translates the probability of bitcell failure to cache yield for a variety of architecture-level resiliency techniques to evaluate the relative effectiveness. Lightweight resiliency techniques, such as static redundancy [2], achieve significant V_{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 [3] or error correcting codes (ECC) [4], can tolerate more failing cells and therefore a higher BER than static redundancy, but still have limited effectiveness. The maximum V_{min} reduction allowed by line disable 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 number of absolute failing bitcells, but at low voltages, the number of failing cells increases dramatically and trading off increased faults for decreased V_{DD} becomes much less attractive. The limits of fault avoidance for V_{min} reduction have been explored by aggressive techniques [5], [6], and the lack of known silicon

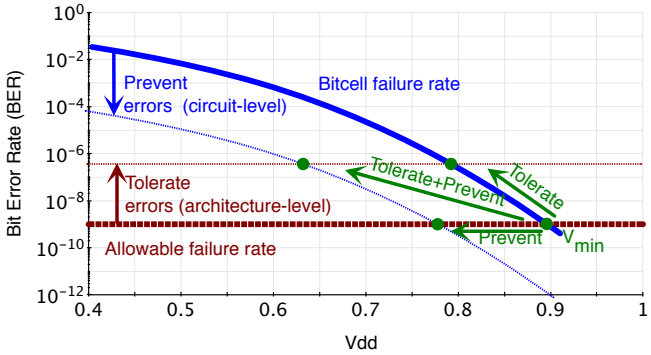

Fig. 1: Circuit-level techniques lower V_{min} by improving bitcells while architecture-level techniques lower V_{min} by tolerating bitcell failures.

implementations of these ideas reflect the high overhead costs and implementation complexity required to tolerate such high failure rates.

This work proposes a new architecture-level redundancy technique, called *dynamic column redundancy (DCR)* that targets a sweet spot failure rate of 1×10^{-4}, and achieves a lower V_{min} than other proposed techniques (such as static redundancy, ECC, and line disable) with low area, delay, and energy overhead. V_{min} is reduced further by supplementing DCR with line disable (LD) to tolerate multi-bit failures, and another reprogrammable redundancy technique, *bit bypass*, 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 cache, in comparison to many prior techniques that target L2 caches only. These techniques are verified through implementation in a 28nm RISC-V [7] processor where the proposed RR techniques enable a 25% V_{min} reduction with 2% area overhead.

Reprogrammable Redundancy Implementation

Figure 2 shows the system diagram of the implemented processor with reprogrammable redundancy. The processor is based on a 64-bit RISC-V single-issue in-order 6-stage pipeline [8]. To support DVFS, 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 8KB instruction and 16KB data cache with 8T-based macros. The 1MB, 4-bank, L2 cache uses high-density 6T-based macros. 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 cache. An at-speed SRAM built-in-self-test (BIST) quickly identifies fault locations. Asynchronous FIFOs and level shifters allow communication between the voltage and frequency islands. Every SRAM includes single error correction and double error detection (SECDED) protection. While testing RR, the correction capability is disabled and errors are simply logged to ensure that all SRAM faults are identified. SECDED correction can be enabled to protect against soft errors, or protect against intermittent errors by reprogramming the redundant entries.

978-1-5090-3700-1/16/$31.00 ©2016 IEEE

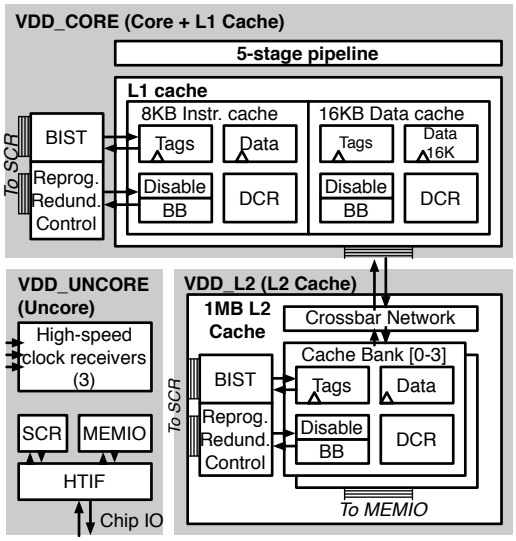

Fig. 2: Block diagram and die photo showing the organization of the 28nm processor into three voltage domains containing the core with L1 cache, L2 cache, and uncore.

Figure 3 describes the dynamic column redundancy technique. Column-redundancy schemes handle hard faults by adding a spare column (SC) 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. DCR offers similar resiliency to ECC, but at much lower cost. Unlike ECC, which generally requires codeword granularity to reflect access word size (to avoid read-modify-write operations), DCR granularity can be adjusted independently of access size. In this implementation, flexible protection granularity is leveraged to repair 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 trade-offs. Additionally, ECC is generally already required for soft error tolerance, and DCR can be easily supplemented by a SECDED code to protect against intermittent errors. In comparison, adding soft error protection to a design that already uses single-bit ECC for voltage reduction by adding double-bit ECC is very expensive [9], even though in some situations it may be acceptable to only use SECDED and simply leave some words unprotected.

Bitcell faults in tag arrays are protected using the bit bypass (BB) scheme shown in Figure 4. A redundant set repairs a single failing bitcell by using flip-flops to store a replacement redundant bit along with the row and column address of the failing cell. Writes and reads to SRAM addresses with known faults use information from the redundant sets to bypass operations to failing bitcells. The targeted allowable

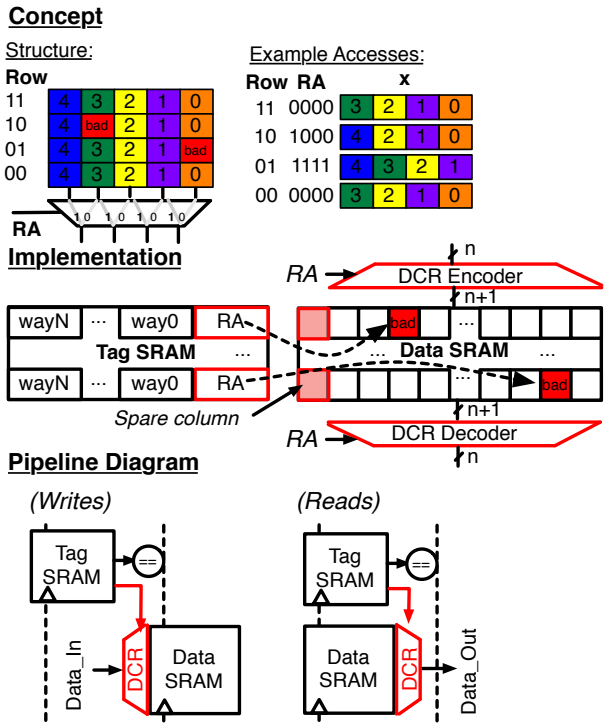

Fig. 3: Dynamic column redundancy exploits cache architectures to repair a single bit per row in cache data arrays.

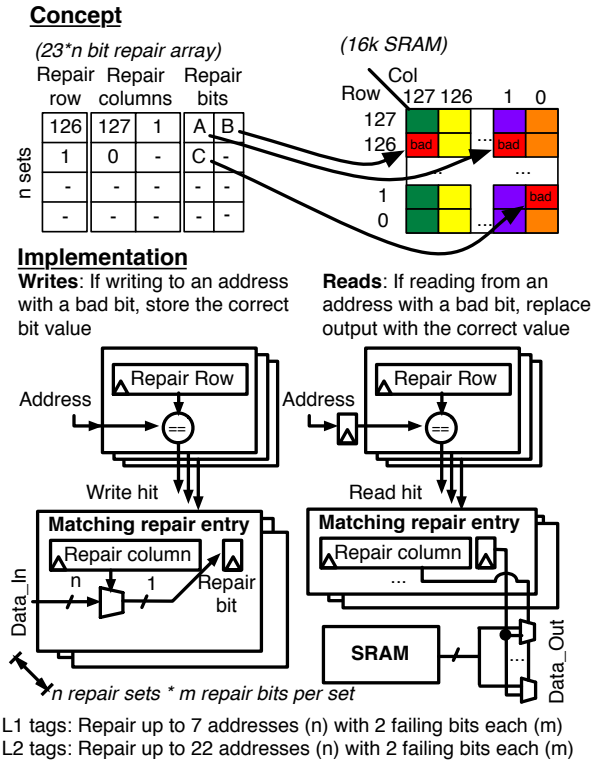

Fig. 4: Bit bypass protects against bit failures in any standalone SRAM macro by storing redundant copies of failing bits in flip-flops.

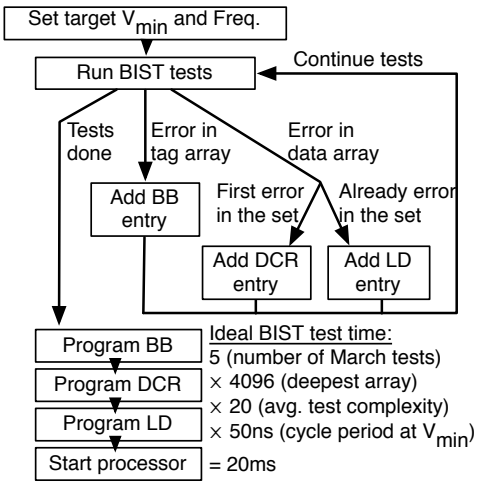

Fig. 5: Flow chart describing the RR programming algorithm.

bitcell failure rate 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.

The minimum operating voltage 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 V_{DD} where the first double-bit failure appears in a cache set, DCR is correcting less than 1% of the sets. Line disable [3] is added to prevent accesses to lines with multi-bit failures, which allows DCR correction to be better utilized. Disable bits are stored with each tag, and the way-replacement algorithm avoids refills to disabled ways. Also, BB overhead could be reduced by using LD to also disable ways with failing tag bits.

The BIST detects SRAM error locations before operation, and uses the error information to program BB, DCR, and LD. 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 flow chart in Figure 5 summarizes the RR programming algorithm. In this testchip, the BIST is run at 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 voltage and frequency points. 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.

Table I compares the resiliency, timing, and area overhead of the proposed technique compared to prior techniques for the L2 cache. The cited techniques do not disclose any resiliency or overhead results, so the comparison points are estimates. 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). BB flip-flops and logic, implemented as standard cells, add about 15% to the tag macro area. 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.

Technique	Maximum BER	Protects tags?	Area Overhead		Total Cache Overhead
			Tags	Data	
Proposed (BB +DCR+LD) §	9.8×10^{-5}	Yes	BB: 15% DCR: 6% LD: 4%	DCR: 0.7%	2.2%
Extreme Redundancy**	2.4×10^{-5}	Yes	SC: 6.4% Flops: 0.5%	SC: 6.4% Flops: 0.5%	6.6%
Line Disable § [3]	1.8×10^{-5}	No	4% #	-	0.2% #
ECC† [4]	1.3×10^{-6}	Yes	7% #	7%	6.7% #
Static Redundancy* [2]	4.4×10^{-7}	Yes	SC: 1% # Fuses: 0.12%	SC: 1% # Fuses: 0.12%	1.1% #
Nominal	1.1×10^{-10}	No	-	-	-

§ Up to 1% disabled **1 column repair/KB and 1 row repair/32KB
†1 repair/128 bit *10 repairs/MB
Estimate (not reported), fuse=100× bitcell, flop=10× bitcell
Note: Data portion is 90% of cache area, tag portion is 6% of cache area

TABLE I: Comparison of L2 cache area overhead and analytical estimates of allowable bitcell error rates for different techniques.

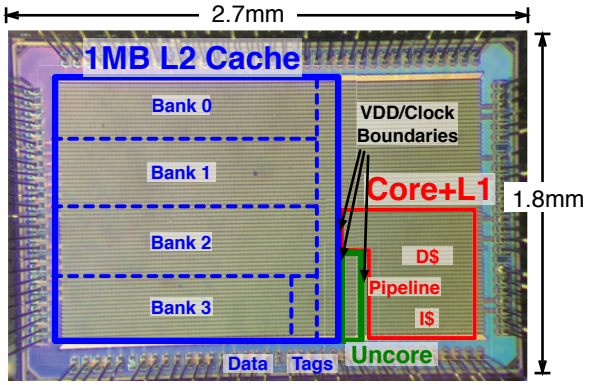

Fig. 6: Die micrograph with labeled processor core, L1 cache, and L2 cache.

Measurement Results

The single-core processor is fabricated in a TSMC 28nm HPM process with a 1.8 mm by 2.7 mm die area, as shown in Figure 6. The measured bitcell failure rate of the SRAM bitcells from a suite of March BIST tests is shown in Figure 7. The L1 cache is implemented with 8T-based bitcells, while the L2 cache is implemented with 6T-based bitcells. As expected, the 8T-based bitcells have a lower intrinsic V_{min} than the 6T cells, and the smaller absolute number of bits in the L1 provides less information about small bit error rates. The maximum allowable BER for the proposed technique and prior techniques from Table I is annotated on the figure.

A variety of benchmarks are run on the processor at different voltages and frequencies with RR disabled. Figure 8 shows the baseline shmoo for the L1 and L2 cache of 7 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. Figure 9 shows the measured V_{min} of the L2 cache for three options for seven evaluated chips: only line disable enabled, only DCR enabled, and a combination of DCR+LD. In all three cases, bit bypass is used to protect the tags, and less than 1% of cache lines are disabled. Enabling the RR techniques (BB and DCR+LD) decrease V_{min} by an average of 25% in the L2 cache.

The correction capability of ECC is not used in Figure 9 to decrease V_{min}, and ECC is used only to confirm that

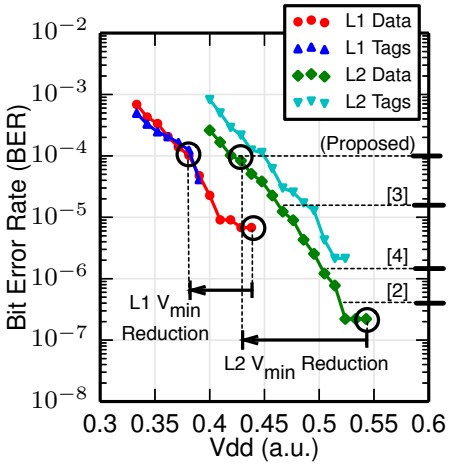

Fig. 7: Measured SRAM bitcell failure rate versus voltage for the L1 cache (6T bitcells) and L2 cache (8T bitcells).

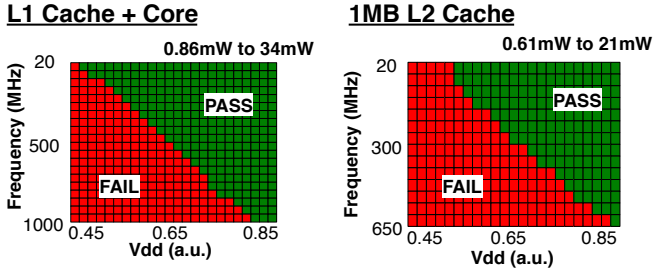

Fig. 8: Baseline L1 and L2 cache shmoo plot for benchmarks running on the processor without any of the reprogrammable redundancy techniques enabled.

all SRAM errors are identified during testing. Analysis of the SRAM failure maps predicts that using ECC correction alone would have achieved a 15% V_{min} reduction, but with a 7% area overhead. However, this result assumes that a BIST does not exist to identify the first double-bit failure, and V_{min} is limited by the worst array for a given yield target. By using BIST and setting a unique V_{min} for each chip, a 20% V_{min} reduction is possible. 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 as well, but only at voltages with extremely low frequencies that lie well below the energy-optimal point. In comparison to circuit techniques that report 130-200mV of V_{min} reduction for 5-7% area overhead [10], [11], the proposed RR techniques can achieve comparable effectiveness with less overhead. However, direct comparison is difficult as some reported modern bitcells are not designed to be used without SRAM assist, and the reported V_{min} reduction is optimistic. RR protects against all failure mechanisms, while circuit assist techniques must carefully tune assists to trade-off between different failure mechanisms.

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. Reprogrammable redundancy is particularly attractive because it can be used in conjunction with simple SECDED 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.

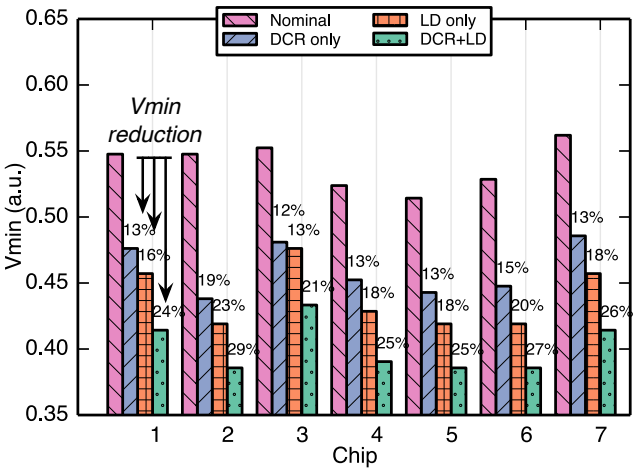

Fig. 9: Enabling the proposed reprogrammable redundancy reduces the minimum operating voltage of the L2 cache by 25%.

Acknowledgements

The authors would like to thank Yunsup Lee, Andrew Waterman, Henry Cook, Stephen Twigg, Brian Richards, James Dunn, Scott Liao, and Jonathan Chang for their contributions. This work was funded in part by BWRC, ASPIRE, DARPA PERFECT Award Number HR0011-12-2-0016, Intel ARO, and a fabrication donation by TSMC.

References

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

[2] M. Huang *et al.*, "An Energy Efficient 32-nm 20-MB Shared On-Die L3 Cache for Intel Xeon Processor E5 Family," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 8, pp. 1954–1962, Aug 2013.

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

[4] S. Sawant *et al.*, "A 32nm Westmere-EX Xeon enterprise processor," in *IEEE International Solid-State Circuits Conference (ISSCC) Digest of Technical Papers*, Feb 2011, pp. 74–75.

[5] T. Mahmood, S. Kim, and S. Hong, "Macho: A failure model-oriented adaptive cache architecture to enable near-threshold voltage scaling," in *Proceedings of the 2013 IEEE 19th International Symposium on High Performance Computer Architecture (HPCA)*, 2013, pp. 532–541.

[6] A. R. Alameldeen *et al.*, "Energy-efficient Cache Design Using Variable-strength Error-correcting Codes," in *Proceedings of the 38th Annual International Symposium on Computer Architecture (ISCA)*, 2011, pp. 461–472.

[7] A. Waterman *et al.*, "The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Version 2.0," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2014-54, May 2014. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-54.html

[8] Y. Lee *et al.*, "A 45nm 1.3GHz 16.7 double-precision GFLOPS/W RISC-V processor with vector accelerators," in *European Solid State Circuits Conference (ESSCIRC-2014)*, Sept 2014, pp. 199–202.

[9] R. Naseer and J. Draper, "DEC ECC design to improve memory reliability in sub-100nm technologies," in *IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, 2008, pp. 586–589.

[10] J. Chang *et al.*, "A 20nm 112Mb SRAM in High-κ metal-gate with assist circuitry for low-leakage and low-V MIN applications," in *IEEE International Solid-State Circuits Conference (ISSCC) Digest of Technical Papers*, 2013, pp. 316–317.

[11] T. Song *et al.*, "A 10nm FinFET 128Mb SRAM with assist adjustment system for power, performance, and area optimization," in *IEEE International Solid-State Circuits Conference (ISSCC) Digest of Technical Papers*, Jan 2016, pp. 306–307.