Reconfigurable ECC for Adaptive Protection of Memory

Abhishek Basak[1], Somnath Paul[2], Jangwon Park[3], Jongsun Park[3], and Swarup Bhunia[1]
[1]Case Western Reserve University, Cleveland, OH, USA [3]Korea University, Seoul, Korea
[2]Intel Corporation, Portland, OR, USA
Email: axb594@case.edu

***Abstract*–**Post-silicon healing techniques that rely on built-in redundancy (e.g. row/column redundancy) remain effective in healing manufacturing defects and process variation induced failures in nanoscale memory. They are, however, not effective in improving robustness under various run-time failures. Increasing run-time failures in memory, specifically in case of low-voltage low-power memory, has emerged as a major design challenge. Traditionally, a uniform worst-case protection using Error Correction Code (ECC) is used for all blocks in a large memory array for runt-time error resiliency. However, with both spatial and temporal shift in intrinsic reliability of a memory block, such uniform protection can be unattractive in terms of either ECC overhead or protection level. We propose a novel *Reconfigurable ECC* approach, which can adapt, in space and time, to varying reliability of memory blocks by using an ECC that can provide the right amount of protection for a memory block at a given time. We show that such an approach is extremely effective in diverse applications.

1. Introduction

Post-silicon calibration and healing has emerged as an effective approach for recovering from manufacturing defects or process variation induced failures in digital, analog and RF circuits/systems [1-6]. In case of nanoscale memory, aggressive area optimization in the quest of higher integration density has made them highly vulnerable to defects as well as run-time failures. Built-in redundancy (e.g. in row/column) has provided a well-adopted healing approach for memory to adapt to hard defects [16]. However, tolerance to runtime failures in memories, especially large on-chip caches, has emerged as one of the key challenges in present day system-on-chip or microprocessor design [7]. Such failures can affect random or contiguous bit positions. For example, soft errors in static random access memory (SRAM) cells caused by striking of high-energy alpha and neutron particles, cause bit flipping in contiguous locations [8]. On-chip caches are also becoming increasingly vulnerable to random errors. These can be caused by – 1) supply voltage noise, 2) thermal noise, or 3) temporal degradation due to aging effects [8]. Aging effects such as Bias Temperature Instability (BTI) [9] can considerably degrade the read stability for SRAM over time, leading to read failures. The situation is worsened, owing to increasing process variations in nanoscale CMOS technologies. Due to inter and intra-die variations, different sections of a memory array may move to different process corners [10]. This leads to a distribution of vulnerability to runtime failures across memory blocks in different chips and within a chip. Fig. 1(a) illustrates this scenario, with simulations for a 2 MB processor cache in 45 nm technology (blocks of size 1-8 KB) showing 7-10X variations in reliability across memory blocks [10]. Cumulatively, these effects result in multi-bit upsets (MBU) in a memory codeword.

Runtime errors have been conventionally addressed by error correction code (ECC), typically parity or low-cost Hamming codes, such as SECDED, which are capable of single error correction and double error detection. However, simple SECDED protection has proved insufficient for multiple-bit failures in a codeword. To address MBUs, a combination of SECDED and bit-interleaving has emerged as the widely accepted choice [11]. Bit-interleaving distributes the contiguous errors into different words and facilitates correction using SECDED. However, it comes at significant energy overhead due to reads performed on unwanted words. On the other hand, existing approaches for multiple bit random failure tolerance either address only static (not runtime) failures during manufacturing test or can substantially compromise storage space in the memory [10]. Besides, a uniform ECC protection for all memory blocks fails to account for the distribution of vulnerability across blocks and can provide overly pessimistic results if ECC allocation is based on the worst-case block vulnerability.

In this paper, we propose a reconfigurable ECC approach, which assigns the right correction capability to individual memory blocks at a time based on their relative vulnerabilities to runtime failures. This is achieved by incorporating ECC encode/decode logic with varying error correction capability during design and selecting them on demand during actual operation. The issue of extra check bits, for increasing error correction capabilities is taken into account by storing them in the "ways" of an associative cache, similar to [13], incurring minor performance loss (~4%). To enhance the effectiveness of a multi-bit tolerance scheme, we propose using shortened Bose-Chaudhuri-Hocquenghem (BCH) cyclic code [13] with zero

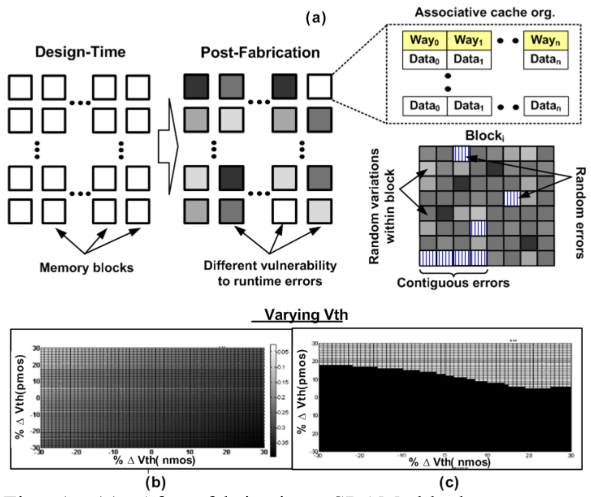

Fig. 1. (a) After fabrication, SRAM blocks move to different reliability corners due to intra- and inter-die variations; (b) Variation of read stability & (c) soft error vulnerability with variation in transistor threshold voltage.

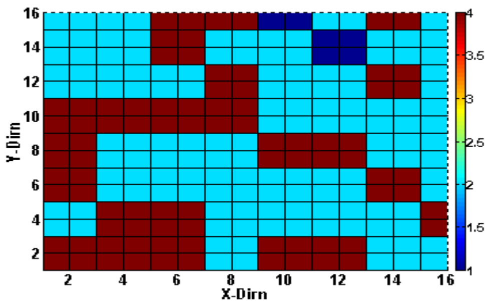

Fig. 3. Spatially variable ECC allocation for memory blocks on a die at the worst-case inter-die corner.

padding for variable ECC protection. In particular, the paper makes the following contributions:

1) It proposes a "variability-aware" ECC allocation, which matches the reliability of a block with appropriate level of ECC protection, and hence, prevents pessimistic uniform ECC. Variable ECC allocation also allows adaptation to scaled operating voltage for low-power operation and temporal reliability degradation.

2) The proposed approach uses a BCH encoding. BCH has much lesser number of check bits than other schemes, hence requiring less cache storage space and a relatively small IPC overhead.

3) A circuit-architecture co-design approach has been employed for the encoder/decoder logic to minimize power/performance overhead. The paper explores application of the scheme for tolerating random and contiguous runtime errors in a unified framework

2. ADAPTIVE PROTECTION

Memory cells, which have moved to low threshold (V_t) or high V_t corners post-fabrication, due to process variations, are susceptible to read, write, access, and hold failures [16]. A SRAM cell was simulated at 45nm technology node for effects on read stability and soft-error tolerance with variations of the nominal V_t (+/- 30%). The results are shown in Fig. 1(b) and 1(c), respectively. It can be noted that a weak-PMOS and a strong-NMOS makes the SRAM cell most vulnerable toward flipping during the read operation. The light colored regions in Fig. 1(c) denote the cells, which have flipped due to a charged particle strike. The soft-error immunity for the SRAM cell is severely degraded when the PMOS becomes weak under inter/intra-die variations. As shown in Fig. 2, the entire spectrum for PMOS and NMOS V_t variation in an SRAM cell can be divided into three regions based on the extent of ECC protection they should receive. Cells with strong PMOS and nominal NMOS (ΔV_t <+/- 10%) are most resilient to failure mechanisms considered above and hence assigned the least ECC protection ($t = t_{min}$, where t denotes the number of bit errors that may be corrected for a given block). Cells with PMOS and NMOS V_t variation less than +/-10 percent of the nominal value should be assigned a higher ECC protection ($t= t_{med}$) due to their vulnerability toward soft error. Cells falling outside the above ΔV_t regions should be assigned the maximum protection ($t =t_{max}$) to tolerate runtime failures arising from system noise, soft errors, and aging effects.

As the proposed ECC assignment is performed after fabrication, it necessitates generation of a reliability map of the cache blocks during the manufacturing test. The baseline vulnerability (used to calculate t_{min}) is **determined by the amount of parametric variation (**in terms of read/write margins) present in the most reliable memory block. Based on the reliability map, values of

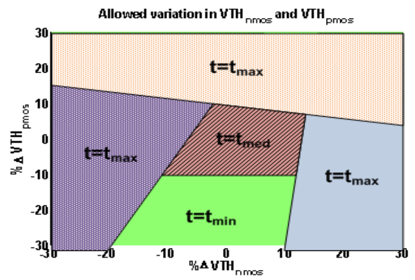

Fig. 2. The spectrum of NMOS and PMOS variation is divided into multiple reliability regions with corresponding error correction (t) requirement.

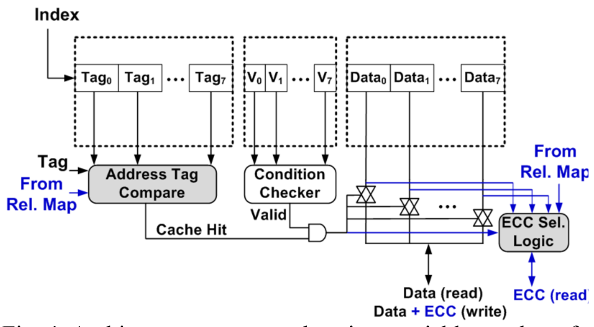

Fig. 4. Architecture to store and retrieve variable number of check bits from one or more "ways" of an associative cache.

$t_{min} < t < t_{max}$ for other memory blocks are determined by their relative vulnerabilities to parametric failures. These t values are encoded (e.g. into 2 bits) and stored on chip in a small non-volatile memory. Based on the encoded values, variable number of ways per set are dedicated to storage of ECC bits by asserting appropriate select signals to the ECC selection logic. During runtime, the ECC selection logic is responsible for distinguishing the data from ECC bits.

3. Spatially Varying ECC

Based on the reliability map, variable numbers of check bits are allocated to different memory blocks. We have simulated a 2MB L2 cache, divided into 256 memory blocks of 8 KB each. 1000 instances of the model were simulated with an inter-die variation of σ_{inter} = 25%, a systematic and random intra-die variation (σ_{sys} and σ_{intra}) of 20 and 15 % per instance. Based on the V_t distribution of the memory blocks, variable ECC was assigned to the memory blocks for each instance. Fig. 3 shows the ECC assignment for the instance which has the maximum number of memory blocks with $t=t_{max.}$ Proposed approach allows appropriate ECC allocation to different dies based on their specific inter-die corners.

Cache size (S), associativity (A), block size (B), and the maximum number of bit errors that can be tolerated (t_{max}) serve as the input specifications for the variable ECC allocation approach. During the design phase, based on $t_{max,}$ the number of required check bits (m) and the number of ways (w) required to store the check bits are calculated. Suitable modifications in memory architecture to interchangeably use these 'w' ways for both data and ECC storage are incorporated into the memory, as shown in Fig. 4. In the modified architecture, along with the address tag, an enable signal from the reliability map would serve as an input to the *Address Tag Compare* logic. This ensures that during read, data out is not selected from any of the ECC ways. In the event of cache hit on any other data way, the hit and the enable signal, can be used to separate the ECC bits from the data bits using the ECC selection logic.

4. Temporally Varying ECC

As a temporally varying ECC, a reconfigurable ECC architecture, which can dynamically change the error correction capability according to failure rate in a codeword, can be employed. The proposed temporally-varying reconfigurable ECC offers three different error correction capability (1/2/4-bit), and the hardware resources among multiple ECC blocks can be shared for minimizing hardware area.

A. Reconfigurable ECC encoder

The reconfigurable ECC encoder architecture is composed of Galois field adders and dividers. Since BCH encoders of different error correction capability have different generator polynomial ***g(x)*** for dividing message polynomial, we have three division parts (1/2/4-bit) in the reconfigurable encoder. The area overhead of the division module is modest due to relatively smaller encoder area (~5 %) than that of the decoder.

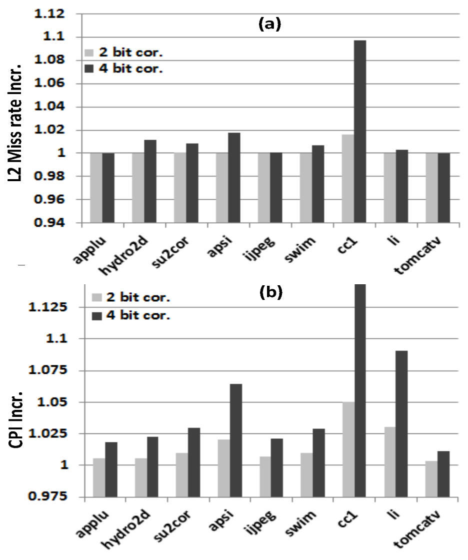

Fig. 5. Increase in (a) L2 cache miss rate, and (b) cycles per instruction (CPI), for 2 and 4-bit error correction.

B. Reconfigurable ECC decoder architecture

The reconfigurable BCH decoder architecture is based on 4-bit correction BCH decoder with associated control logic. When the decoder is operating in 1 or 2-bit correction mode, a significant portion (unused modules) of the decoder can be simply turned off to save power.

The reconfigurable BCH decoder is composed of syndrome generator, key equation solver, and Chien search modules [10]. Since the syndrome generator for 1-bit or 2-bit correction BCH can be expressed as a subset of syndrome generation for 4-bit correction BCH, the syndrome generator hardware can be implemented in a scalable fashion, which means that only 25% or 50% of syndrome generators can be utilized for 1-bit or 2-bit correction mode, respectively, with simple control logic for turning off unused modules.

For the key equation solver (KES) module design, the simplified inverse-less Berlekamp Massy (SiBM)

Table I: Synthesis results of BCH Decoder

BCH Type	1Bit Cor. BCH	2Bit Cor. BCH	4Bit Cor. BCH	Reconfi. BCH
Combinational area	4747	9933	20605	20911
Non-combinational area	801	1458	2556	2937
Total area	5548	11392	23161	23850
Critical path	6ns	6ns	6ns	6ns
# of Cycles	3	4	6	3/4/6

TABLE II: RECONFIGURABLE BCH POWER CONSUMPTION

Correction Type	Power Consumption (mW)	Power Saving Over [4bit]
1bit	1.40	64.74%
2bit	2.23	43.83%
4bit	3.97	0%

algorithm [14] is used. The KES is composed by 2t identical processing elements (PEs), where t is the maximum correctable errors. After KES module supporting 4-bit correction is designed with 8 PEs, the architecture can be directly used for 1/2-bit correction BCH with unnecessary 6 or 4 PEs turned off. The scalable Chien search module has similar structure as the syndrome generator. For 4-bit correction BCH Chien search, four sub-blocks, which are mainly composed of Galois field constant multipliers, operate in parallel.

5. IMPLEMENTATION RESULTS

Conventional 1-bit, 2-bit, and 4-bit correction BCH decoders and the proposed reconfigurable decoders are implemented using 65-nm standard-cell CMOS library, and Table I shows the implementation results. In terms of hardware area, the reconfigurable decoder is larger than 4-bit correction BCH by 1.4%, which is mainly due to additional control overhead. Using more than 1,000 random input data, power consumptions of the different BCH decoder architectures are measured with spice simulations. As shown in Table II, when the reconfigurable decoder is used as 1-bit or 2-bit correctible BCH, power saving is around 64.7% and 43.8 %, respectively, over the 4-bit correctible decoder. Compared to the conventional 2-bit correction BCH decoder, our 2-bit correctible reconfigurable decoder consumes only about 8% of more power. The smaller power overhead is achieved by optimizing the control logic required for reconfiguration.

Fig. 5 shows the simulation results on cache miss rate and clock-cycles per instruction (CPI) when the proposed temporally varying ECC scheme is applied to 2 MB 8-way L2 cache (with each way storing 64B of data) in a 64-bit Alpha processor. The simulation is performed with Simplescalar [15] using SPEC1995 benchmark, and it is assumed that the L2 cache access latency is 12 clock cycles. According to the simulation results, when the proposed varying ECC is used as 2-bit or 4-bit correctible scheme, it has minor performance degradation for most of the benchmarks compared to baseline 1-bit correctible ECC scheme.

6. CONCLUSION

We have presented an adaptive protection scheme for nanoscale memory array that provides right amount of protection to each memory block under spatially and temporally varying reliability. Area, performance, and energy overhead for the proposed scheme are minimized by appropriate choice of ECC and joint circuit/architecture-level optimizations of the encoding/decoding hardware. Proposed adaptive error-resilience approach is amenable for efficient dynamic adaption in operating point (e.g. voltage). Future investigation will include application of reliability-aware address mapping and combination with bit-interleaving to reduce ECC overhead and enhance error protection.

7. ACKNOWLEDGEMENT

The work is supported through the National Science Foundation grants #CCF-0964514 and #ECCS-1002237.

REFERENCES

[1] S. Narasimhan, K. Kunaparaju, S. Bhunia, "Healing of DSP Circuits Under Power Bound Using Post-Silicon Operand Bitwidth Truncation", IEEE Trans. on Circuits and Systems I, Vol. 59, No. 9, 2012.
[2] S. Narasimhan, S. Paul, R. S. Chakraborty, F. Wolff, C. Papachristou, D. Weyer, and S. Bhunia, "System Level Self-Healing for Parametric Yield and Reliability Improvement under Power Bound," NASA/ESA Conference on Adaptive Hardware and Systems, 2010.
[3] A. Goyal, M. Swaminathan, A. Chatterjee, "Self-correcting, self-testing circuits and systems for post-manufacturing yield improvement", IEEE International Midwest Symposium on Circuits and Systems, 2011.
[4] A. Goyal, M. Swaminathan, A. Chatterjee, "RF Substrate yield improvement using package-chip co-design and on-chip calibration", IEEE Electrical Design of Advance Packaging and Systems, 2010.
[5] P. K, Lala, B. K. Kumar " An Architecture for self-healing digital systems", Journal of Electronic Testing: Theory and Applications, Vol. 19, No. 5, 2003.
[6] J. Lee et al., "Self-Healing Design in Deep Scaled CMOS Technologies", Journal of Circuits, Systems and Computers, vol. 21, no. 6, 2012.
[7] S.S. Mukherjee et. al., "Cache Scrubbing in Microprocessors: Myth or Necessity?," Int'l Symp. Dependable Computing, 2004.
[8] J. Maiz et. al., "Characterization of Multi-Bit Soft Error Events in Advanced Srams", Int'l Electron Devices Meeting, 2003.
[9] K. Roy et. al., "NBTI Induced Performance Degradation in Logic and Memory Circuits: How Effectively Can We Approach a Reliability Solution?", Asia and South Pacific Design Automation Conference, 2008.
[10] S. Paul, F. Cai, X. Zhang, and S. Bhunia, "Reliability-Driven ECC Allocation for Multiple Bit Error Resilience in Processor Cache", IEEE Trans. on Computer, 2011.
[11] N. Quach, "High Availability and Reliability in the Itanium Processor," IEEE Micro, 2000.
[12] Z. Chisti et. al., "Improving Cache Lifetime Reliability at Ultra-Low Voltages," Int'l Symp. Microarchitecture, 2009.
[13] S. Lin & D. Costello, Error Control Coding, second ed. Prentice Hall, 2004.
[14] W. Liu et. al., "Low-Power High-Throughput BCH Error Correction VLSI Design for Multi-Level Cell NAND Flash Memories," IEEE SIPS, 2006.
[15] Simplescalar Toolset V3.0, http://www.simplescalar.com/, 2010.
[16] S. Mukhopadhyay et al., "Modeling of Failure Probability and Statistical Design of SRAM Array for Yield Enhancement in Nanoscaled CMOS," IEEE Trans. on CAD, vol. 24, no. 12, 2005.