Aging-resilient SRAM design: an end-to-end framework

Xuan Zuo and Sandeep K. Gupta
Ming Hsieh Department of Electrical and Computer Engineering,
University of Southern California, Los Angeles, CA, USA
{xzuo, sandeep}@usc.edu

***Abstract*—The performance of transistors degrades due to aging. Bias temperature instability (BTI) is the most prominent aging mechanism in nano-scale CMOS technologies. Aging degradation causes lifetime failures and lowers the quality of shipped chips. We have developed an end-to-end SRAM design framework to maximize the aging resilience under the given constraints. Specifically, we analyze the impact of aging in SRAM peripheral circuits, including address decoder, precharge, write circuit and sense amplifiers (SAs). We explore the efficiency of error-correcting codes (ECC) to combat aging by quantifying the area and delay overheads of ECC and estimating the lifetime yield and DPPM of SRAMs with ECC, respectively. We also calculate the soft error resilience when ECC is used to repair aging failures. After comparing approaches based on cell sizing and ECC in terms of overheads, lifetime yield and DPPM, we can choose either one or a combination of these approaches to identify the optimal design against aging under the given constraints. We integrate our methods into an existing SRAM compiler, CACTI [1], to provide the end-to-end capability to designers.**

I. Introduction

The performance of a transistor degrades due to aging. Bias temperature instability (BTI) is the most prominent aging effect in nano-scale CMOS [2]. BTI aging degrades the stability of SRAM cells over time [2]. Thus, some SRAM cells functioning properly at fabrication may fail with usage. Aging degradation leads to field failures and results in low quality (typically described in terms of defective parts per million (DPPM)) for the shipped chips over their desired lifetimes.

Aging can be slowed down via design at architecture and circuit levels. Existing design techniques for aging mitigation at architecture-level include workload balancing between different cores [3], dynamic voltage scaling, proactive use of SRAM redundancies [4], periodically flipping data bits stored in SRAMs [5], shutting down idle cache blocks [6], and so on. Certain circuit-level design strategies, such as adaptive body bias [7], can be used to compensate for the effects of aging so that the circuit is more robust against transistor aging. However, it does not take into account differential aging [8]. Also to have the ability to adjust body bias voltages of individual transistors, this method requires a standalone threshold voltage sensing circuit to estimate aging degradation. All these methods require significant changes at the architecture-level or expensive changes at the circuit-level.

In contrast, in our prior work [8], we consider different stress conditions for different transistors in SRAM cells and propose sizing approaches in SRAM cells to optimize the lifetime yield-per-area under the tight constraints on aging quality loss [8].

In this paper, we develop an end-to-end SRAM design framework to maximize the aging resilience under the given constraints to provide designers the capability for optimal design of aging-resilient SRAMs. In addition to transistor sizing, we study the use of error-correcting codes (ECC) to improve the aging resilience. We also analyze the impact of aging in SRAM peripheral circuits. By quantifying the area and delay overheads of ECC and estimating the lifetime yield and DPPM of SRAMs with ECC, we explore the efficiency of ECC to combat aging. After comparing approaches based on transistor sizing in SRAM cells and ECC in terms of overheads, lifetime yield and DPPM, we can choose either one or a combination of these approaches to identify the optimal design against aging under the given constraints.

The rest of this paper is organized as follows. In Section II, we introduce the background material, namely BTI aging and transistor sizing approach. In Section III we analyze the aging degradation for SRAM peripheral circuits. In Section IV, we quantify the overheads of ECC and calculate DPPM and lifetime yield of SRAM using different ECC schemes. In Section V, we present our end-to-end SRAM design framework for lifetime yield-per-area optimization. We present design results for two example cases in Section VI. Finally, we present our conclusions in Section VII.

II. BTI aging and SRAM cell sizing approach

BTI is the major reliability hazard in nano-scale CMOS. Negative bias temperature instability (NBTI) and positive bias temperature instability (PBTI) cause threshold voltage (Vth) degradation for stressed pMOS and nMOS transistors, respectively [9]. Logic values (0's and 1's) are typically stored in SRAM array with different probabilities. Different transistors in an SRAM cell suffer different stress conditions, thus their Vth value degrade by different amounts. BTI aging degrades the stability of SRAM cells over time [2].

The noise margin degradation caused by aging in SRAMs depends on the workload applied to the cells. Some single-purpose IoT systems have a specific workload, while more general-purpose systems have a broader range of workloads. The workload can be characterized based on the different types of applications. We use four typical workloads to demonstrate our design approach ahead, i.e., **Gaussian 1**, **Gaussian 2** (a

978-1-7281-5359-9/20/$31.00 ©2020 IEEE

Gaussian distribution with a larger standard deviation compared to Gaussian 1), **Skew distribution** (Signal probability distribution in data caches extracted from [10]), **Unknown** (specific distribution is unknown and can be arbitrary in general; this is the worst-case in terms of sizing).

We proposed a transistor sizing approach to mitigate stability degradation caused by aging in SRAM cells [8]. After studying the impact of aging on the stability of SRAM cells, we identify which noise margin to increase. Via exploring the relationship between transistor sizing and noise margins, as well as the associated trade-offs, we identify how to size the transistors to increase the aging resilience for different workloads. We use two ways to size the transistors, i.e., skew-size and up-size. Skew-size is asymmetric sizing and is more efficient and effective for asymmetric workload, where one of the read noise margin degradation is larger than the other one. Up-size is to increase the size of both pull-down transistors and is more expensive. Up-size is used when the workload is symmetric or only using skew-size cannot achieve the target DPPM,

Lifetime yield of an SRAM is the probability that an SRAM is able to function correctly throughout its desired lifetime. **Aging quality loss** of an SRAM is the probability that an SRAM functions correctly at fabrication but fails during its desired lifetime due to aging.

Our design target for SRAM is to optimize for lifetime yield-per-area under a given aging quality loss target (measured in **DPPM**). The SRAM cell sizing approach is effective at improving the aging resilience for various workloads without power overhead.

III. Aging analysis for SRAM peripheral circuits

In Section II, we know that BTI aging degrades the stability of SRAM cells. The transistor sizing approach can mitigate the stability degradation of SRAM cells. To develop an end-to-end SRAM design framework to increase the aging resilience, we need to study the impact of aging in SRAM peripheral circuits. In this section, we summarize our study of the impact of aging on peripheral circuits of SRAM, including address decoder (AD), write circuit, precharge circuit and sense amplifiers (SAs). In the next section, we study the impact of using ECC to improve the aging resilience. While most of our subsequent framework for design of aging-resilient SRAMs focuses on sizing (presented earlier in [8]) and ECC, the impact of aging on delay of peripheral circuitry is used to ensure that the final designs we select do meet the user constraints on SRAM delay.

After analyzing the circuit structures and stress conditions, we understand the impact of aging in AD, SAs and so on. In particular, BTI aging causes delay degradation in AD, write circuit, precharge circuit and current-controlled latch sense amplifier [11] (SA1). Aging causes both delay degradation and stability degradation in latch based sense amplifier with pass transistors [11] (SA2). The delay degradation in the write circuit is not our concern since it is not (typically) in the critical path. The same sizing approach as for SRAM cells can be used to mitigate the stability degradation in SA2.

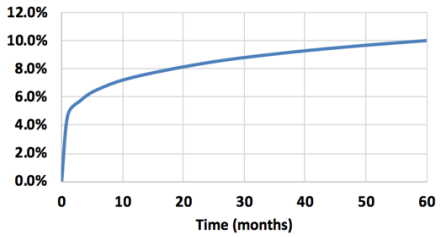

Fig. 1: Read delay degradation caused by aging over the lifetime

However, we can simply choose SA1 or other stability degradation immune sense amplifiers to avoid stability degradation caused by aging in such sense amplifiers. Via SRAM aging analysis, we can conclude that aging causes delay degradation in peripheral circuits and stability degradation in cells. We quantify the delays of various SRAM components as well as the corresponding aging-induced delay degradations. To avoid failure caused by delay degradation, we can leave a sufficient margin in the timing of control signals or increase VDD. We can also redesign the peripheral circuits if the timing constraint is tight. The overall read delay degradation is the sum of address decoder delay degradation, cell discharge delay degradation, and SA delay degradation. The cell delay degradation caused by aging is small compared to that of the peripheral circuits. The decoder delay accounts for a large proportion of the read delay and hence decoder delay degradation is the major component in the overall read delay degradation, especially for large address decoders. Fig. 1 shows the read delay degradation caused by aging for a 12-bit address SRAM over the lifetime. If the timing constraint is tight, we can redesign the address decoder to reduce the delay. Alternatively, we can either use high VDD at the beginning of the operation or increase VDD gradually in the field to avoid failures caused by delay degradation.

IV. Using ECC to repair aging failures

A. ECC background

ECC is a powerful technique used in memories to repair failures caused in a limited number of cells in arbitrary locations. In particular, Single-Error-Correction (SEC) code is one of the most popular codes used in memories but can only correct a single bit error [12]. Bose-Chaudhury-Hocquenghem Double-Error Correction (BCH DEC) [12] can correct two-bit errors. Conventionally, ECC is used to recover from soft errors. In this section, we explore the capability of ECC to repair both soft errors and aging failures. Adding ECC to memory incurs area, latency, and power overheads. To analyze the efficiency of ECC for combating aging, we quantify the overheads of ECC and calculate the DPPM and lifetime yield of SRAM using different ECC schemes. We also estimate the soft error resilience (SER) to ensure that the designs using ECC to combat aging also meet the soft error resilience constraint.

B. DPPM and lifetime yield estimation with ECC

Our design objective is to optimize the lifetime yield-per-area of SRAM array while also satisfying a given DPPM target. Thus, we first compute DPPM and lifetime yield when

ECC is used to correct aging failures. We define D as the length of the data (in bits) corrected by ECC. For an SRAM array consisting of N cells, R = N/D. We assume the failures of SRAM cells caused by aging are independent from cell to cell, and this assumption is valid for all the cases (workloads) studied in this paper. If we use SEC code to handle aging failures, the aging quality loss of an SRAM array (Q_{ag-SEC}) can be calculated as follows:

$$Q_{ag-SEC} = 1 - \prod_{m=1}^{R}[\prod_{i=1}^{D}(1 - P_{f,ag}^{c,m,i}) + \sum_{i=1}^{D} P_{f,ag}^{c,m,i} \prod_{j=1,j\neq i}^{D}(1 - P_{f,ag}^{c,m,j})]$$

where $P_{f,ag}^{c}$ is the aging failure rate of an SRAM cell, i.e., the probability that an SRAM cell functions properly at fabrication but fails during its desired lifetime due to aging. This equation calculates the probability of an SRAM array with at least one block (D-bit word) having two or more cell failures caused by aging. SEC code can correct one failure in each block. Thus, a chip will function correctly if every block only contains either zero or single failure. If any of the blocks contains more than one aging failures during its desired lifetime, the SRAM array fails due to aging. **DPPM** of the SRAM array = Aging quality loss $\times 10^6$.

The lifetime yield of an SRAM array ($Y_{life-SEC}$) can be calculated as follows when we use SEC code to correct aging failures:

$$Y_{life-SEC} = \prod_{i=1}^{N}(1 - P_{f}^{c,i}) \prod_{m=1}^{R}[\prod_{i=1}^{D}(1 - P_{f,ag}^{c,m,i}) + \sum_{i=1}^{D} P_{f,ag}^{c,m,i} \prod_{j=1,j\neq i}^{D}(1 - P_{f,ag}^{c,m,j})]$$

where P_{f}^{c} is the failure rate of an SRAM cell and is used in the experiment to take into account the yield at the time of fabrication.

DEC code can correct two aging failures in each block. Thus, SRAM fails due to aging only when any of the blocks contains more than two aging failures during its desired lifetime. Thus, the aging quality loss of an SRAM array with DEC code (Q_{ag-DEC}) can be calculated as follows:

$$Q_{ag-DEC} = 1 - \prod_{m=1}^{R}[\prod_{i=1}^{D}(1 - P_{f,ag}^{c,m,i}) + \sum_{i=1}^{D} P_{f,ag}^{c,m,i} \prod_{j=1,j\neq i}^{D}(1 - P_{f,ag}^{c,m,j}) + \frac{1}{2}(\sum_{i=1}^{D}\sum_{j=1,j\neq i}^{D} P_{f,ag}^{c,m,i} P_{f,ag}^{c,m,j} \prod_{k=1,k\neq i,k\neq j}^{D}(1 - P_{f,ag}^{c,m,k}))]$$

The lifetime yield of an SRAM array when we use DEC code to correct aging failures ($Y_{life-DEC}$) is calculated as follows:

$$Y_{life-DEC} = \prod_{i=1}^{N}(1 - P_{f}^{c,i}) \prod_{m=1}^{R}[\prod_{i=1}^{D}(1 - P_{f,ag}^{c,m,i}) + \sum_{i=1}^{D} P_{f,ag}^{c,m,i} \prod_{j=1,j\neq i}^{D}(1 - P_{f,ag}^{c,m,j}) + \frac{1}{2}(\sum_{i=1}^{D}\sum_{j=1,j\neq i}^{D} P_{f,ag}^{c,m,i} P_{f,ag}^{c,m,j} \prod_{k=1,k\neq i,k\neq j}^{D}(1 - P_{f,ag}^{c,m,k}))]$$

C. Calculation of soft error resilience when using ECC to repair aging failures

The probability that a soft error occurs, at a single cell during the time interval [0, t] (P_{sf}), is calculated as follows,

$$P_{sf}(t) = 1 - exp(-\nabla(t))$$

where $\nabla(t) = F_b \times \frac{t \times 24hrs}{10^9 hrs}$. F_b is fit per bit.

When ECC is used to repair both aging failures and soft errors, the bit error (to be corrected by ECC) occurs during the time interval [0,t] with probability

$$P_{er}(t) = 1 - (1 - P_{ag})(1 - P_{sf}(t))$$

where P_{ag} is the error probability due to aging, P_{sf} is the error probability due to soft errors. Obviously, $P_{er}(t) = P_{ag} + P_{sf}(t) - P_{ag}P_{sf}(t) > P_{sf}(t)$. That is, when ECC is used to repair both aging failures and soft errors, ECC essentially deals with a more error-prone situation even if $P_{sf}(t)$ is assumed unchanged. Or equivalently, because ECC is used to repair aging failures together with soft errors, the same ECC, in general, has a smaller chance to recover from soft errors compared with conventional scenarios where it is solely used to correct soft errors. To characterize the potential degradation, we study the soft error resilience of ECC after using ECC to repair aging failures, which is the probability that both soft error and aging failures can be repaired.

The soft error resilience of each word for SEC after aging repair can be computed as,

$$P_{SEC-resilience}(t) = \prod_{i=1}^{D}(1 - P_{f,ag}^{c,i})exp(-\nabla(t)) + \sum_{i=1}^{D}[1 - exp(-\nabla(t)) + P_{f,ag}^{c,i}exp(-\nabla(t))] \prod_{j=1,j\neq i}^{D}(1 - P_{f,ag}^{c,j})exp(-\nabla(t))$$

The soft error resilience of each word for DEC after aging repair can be computed as,

$$P_{DEC-resilience}(t) = \prod_{i=1}^{D}(1 - P_{f,ag}^{c,i})exp(-\nabla(t)) + \sum_{i=1}^{D}[1 - exp(-\nabla(t)) + P_{f,ag}^{c,i}exp(-\nabla(t))] \prod_{j=1,j\neq i}^{D}(1 - P_{f,ag}^{c,j})exp(-\nabla(t)) + \frac{1}{2}(\sum_{i=1}^{D}\sum_{j=1,j\neq i}^{D}[1 - exp(-\nabla(t)) + P_{f,ag}^{c,i}exp(-\nabla(t))][1 - exp(-\nabla(t)) + P_{f,ag}^{c,j}exp(-\nabla(t))] \prod_{k=1,k\neq i,k\neq j}^{D}(1 - P_{f,ag}^{c,k})exp(-\nabla(t))$$

In Table I, the DPPM and lifetime yield for SRAMs using D0 are listed with different ECC schemes. D0 is the 6T SRAM cell optimized for the yield-per-area at the time of fabrication. In the table, the number associated with each ECC, e.g. SEC-512, is the length of the data (in bits) corrected by ECC. For G1 and Skew workloads, any of the ECC schemes in the table can achieve the target DPPM, value of 50. When the DPPM is small, the lifetime yield is close to the yield at fabrication. Thus, the lifetime yield-per-area is determined by the area overhead. For simplicity, we first ignore the aging in the ECC circuit. For G2 workload, SEC-64 or SEC with smaller code length or any DEC can achieve target DPPM. For Unknown workload, DEC-512 or DEC with smaller code length can be used to achieve target DPPM.

We can add extra ECC only for aging resilience to avoid sacrificing the soft error resilience of the design. The better approach for real design is to use the existing ECC for soft error directly to handle aging failure, especially for the cases with small aging failure rates (such as for G1 workload). This results in infinitesimal degradation of soft error resilience. However, for cases with large aging failure rates, the probability that multiple blocks have at least one aging failure in each block is high. In such cases, using existing ECC for soft error to correct aging failure decreases the soft error resilience to unacceptably low levels. Thus, we also estimate the soft error resilience when using ECC to combat aging. If the existing ECC for soft error cannot meet the soft error or aging constraint, a stronger ECC circuit needs to be used to handle the aging failure. Table III shows the ECC schemes considering both aging repair and soft error resilience for different workloads.

D. Characterize the ECC implementation overheads

To optimize the design in terms of lifetime yield-per-area, it is necessary to estimate the area overhead of ECC. The major components of ECC implementation are encoder and decoder logic, and storage for check bits. For example, SEC-512 requires 11-bit check bits for every 512-bit. DEC almost doubles the number of check bits as compared with SEC.

Table I: Lifetime yield and DPPM comparison for SRAMs using a cell optimized for yield-per-area at the time of fabrication (D0) under four different workloads with different ECC schemes

Workload	DPPM (SEC-512)	DPPM (SEC-64)	DPPM (SEC-16)	DPPM (DEC-1024)	DPPM (DEC-512)	Lifetime yield (SEC-512)	Lifetime yield (DEC-512)
Gaussian 1	0.00067	0.00091	0.00069	0.00067	0.00067	0.96643	0.96643
Gaussian 2	*57.90*	7.11	0.552	0.0010	0.00039	0.96637	0.96643
Skew	2.07	0.26	0.019	0.00018	0.00021	0.96643	0.96643
Unknown	*18330.4*	*2283.6*	*544.3*	*56.1*	6.53	0.94872	0.96642

Table II shows the number of check bits needed to implement SEC and DEC and the corresponding area overheads at different correctable data lengths. The encoder and decoder of SEC are constructed as XOR trees. BCH DEC are cyclic codes and usually implemented by multi-bit Linear Feedback Shift Registers (LFSR). Thus, the delay and area overheads are larger than those of SEC. A single-cycle implementation of DEC decoders incurs 55% to 69% latency penalty compared to SEC codes. Thus, for similar area overhead, we first choose SEC.

We consider that the extra area of ECC implementation is caused by encoder and decoder circuits, and the memory cells required to store the check bits. In Table II, the area overhead of storage of check bits is calculated by the ratio of the area of the memory cells needed to store check bits and the area of correctable data cells. The encoder and decoder area overheads are the ratio of the area of encoder and decoder circuits to the total area of data cells in SRAM array. For a fixed error-correcting capability, encoder and decoder complexities increase with the length of the data corrected by ECC, while the check bits area overheads decrease dramatically. For SEC, the decoder and encoder logic occupy a much smaller area compared to the storage of check bits. The major area overhead for SEC is the storage bits. For a fixed code length, the complexities of encoder/decoder logic and check bit array both increase with the error-correcting capability. However, the encoder/decoder logic complexities increase much faster than check bit array. For example, the area overhead of check bits array for data length of 512 grows from 2.15% to 3.91%, while the area overhead of encoder/decoder logic grows from 0.05% to 4.09%. For DEC-512, the encoder and decoder logic dominate the area.

We synthesize the ECC encoder and decoder circuits using Design Compiler with 45nm PDK library and report the area of encoder and decoder circuits for various correctable data lengths and error correction capabilities. Then we characterize the area of ECC decoder and encoder circuits in terms of correctable data length, error correction capability. The soft error rate is 5×10^6 Fit/Mb.

The cell area is calculated based on a parametric layout of an SRAM cell. We modify CACTI to add the bit cell area as a function of transistor size and add ECC area including check bits storage, and decoder and encoder circuits. The peripheral circuits and interconnect area can be estimated using CACTI. Thus, the modified CACTI is able to estimate the overall area

Table II: Number of check bits and area overhead (compared to the total area of data cells in SRAM array) for SEC and DEC for different correctable data lengths

	SEC			BCH DEC		
The length of data (in bits) corrected by ECC	**# of check bits**	**Storage of check bits area overhead (%)**	**Encoder + decoder area overhead (%)**	**# of check bits**	**Storage of check bits area overhead (%)**	**Encoder + decoder area overhead (%)**
16	6	37.5	0.0012	10	62.5	0.0048
32	7	21.87	0.0025	12	37.5	0.0174
64	8	12.5	0.0052	14	21.87	0.066
128	9	7.03	0.011	16	12.5	0.26
256	10	3.91	0.024	18	7.03	1.028
512	11	2.15	0.050	20	3.91	4.09

Table III: Area overhead (relative to entire SRAM, computed using modified CACTI) comparison for sizing and ECC approach under four different workloads

Workload	**Area overhead for sizing approach**	**Area overhead for ECC used only for aging**	**SEC-512 was already available (SER>0.9)**	**DEC-256 was already available (SER>0.9)**
Gaussian 1 (G1)	1.0222	1.0181 (SEC-512)	1	1
Gaussian 2 (G2)	1.0443	1.0659 (DEC-512)	1.0470 (DEC-512)	1
Skew	1.0329	1.0181 (SEC-512)	1	1
Unknown	1.0665	1.0659 (DEC-512)	1.0470 (DEC-512)	1

of SRAM including ECC implementation. Table III shows the area overhead comparison of ECC approach and the sizing approach for different workloads. For calculating the area overhead in the second and third columns in Table III, the base is the area of entire SRAM using D0 without ECC. Thus the area overhead for the sizing approach is the ratio of the area of entire SRAM using cells optimized through sizing approach without ECC to the base respectively. The area overhead in the third column is the ratio of the area of entire SRAM using D0 with appropriate ECC schemes to the base respectively. The base is the area of entire SRAM using D0 with SEC-512 for the fourth column and using DEC-256 for the fifth column. We assume the original ECC is replaced by the new ECC scheme for area overhead calculation.

From Table III, we see that the area overhead of ECC approach is lower than that of the cell sizing approach for G1, skew, and Unknown workloads. SRAM design under G1 and Skew workloads can use SEC-512 to achieve the target DPPM. The overall area overhead is only 1.81%. However, ECC encoder and decoder circuits cause delay overheads. From synthesis results, encoders for SEC incur a latency penalty from 0.3ns-0.8ns for various correctable data lengths. The delay overhead for decoders for SEC is more than two times of that of encoders. The delay penalty for multi-cycle implementation of BCH DEC code is dramatically high. Thus, under a tight timing constraint, the sizing approach is chosen over ECC approach to combat aging.

V. Design approach

A. An end-to-end SRAM design framework for lifetime yield-per-area optimization

We have studied cell sizing and ECC approaches to increase the aging resilience. Each approach has different overheads. According to the constraints given by customers, we can achieve the optimal SRAM design to maximize the lifetime yield-per-area under the DPPM constraint via one of or the combination of the two approaches. We propose an end-to-end SRAM design framework to optimize the lifetime yield-per-area under the given constraints as follows:

Given access time, power, area, DPPM, and soft error resilience constraints. *Assume ECC exists for soft error.*

Step 1: Use CACTI to find candidate designs under the given constraints on access time, power, and area. The base cell D0 is the cell optimized for the yield-per-area at the time of fabrication.

Step 2: Estimate the DPPM and SER of SRAM with the existing ECC scheme for various workloads depending on the application.

If both DPPM and SER meet the target, report the design. If not, go to step 3.

Step 3: For the given DPPM for SRAM, using SRAM sizing approach to achieve given DPPM constraint while optimizing lifetime yield-per-area. The redesigned cell is called new cell. DPPM is estimated without ECC. Perform binary search across cell designs between the original cell and the new cell. Estimate SER and DPPM with ECC. Choose the cell with the smallest area overhead satisfied DPPM and SER constraints. Report the area overhead. This area overhead bounds the ECC approach.

Step 4: Explore ECC approach to achieve target DPPM. The SRAM cell is the base cell D0 optimized for the yield-per-area at the time of fabrication.

a) Starting with the original ECC, reduce the data length to half.

If the starting ECC is SEC. Simply calculate the area overhead of check bits.

If check bit area overhead of SEC with half-size data blocks is larger than that of DEC with the largest data length, we move to code with higher error correction capability (DEC).

b) Estimate the DPPM and SER of SRAM with the new ECC scheme. Repeat 4a) if DPPM or SER does not satisfy the target. We choose SEC over DEC for similar area overhead when both can satisfy the target DPPM because SEC code results in a lower delay penalty.

Report the area overhead of ECC using CACTI. Report the access time and power of design with the new ECC.

If the access time or power exceeds the constrains, report the design in step 3 as optimal design. End the process. We need to add delay degradation caused by aging to the access time when compare with user access time constraints.

Compare the area overhead with the area overhead in step 3. If the area overhead of ECC is smaller than that of the cell sizing approach, we choose this ECC scheme. Report the

Table IV: Design results of 2MB SRAMs with 6T cells under four different workloads

	Workload	Cell	Add ECC	DPPM	Area overhead
Case 1:	Access time < 5ns, SEC-512 exists, Soft error resilience > 0.9				
	G1	D0	No	0.00067	1
	G2	D2	No	48.78	1.0435
	Skew	D0	No	2.07	1
	Unknown	D4	No	31.52	1.0653
Case 2:	Access time < 10ns, no ECC exists				
	G1	D0	SEC-512	0.00067	1.0181
	G2	D2	No	48.78	1.0443
	Skew	D0	SEC-512	2.07	1.0181
	Unknown	D0	DEC-512	6.53	1.0659

design as optimal design. Otherwise, report the design in step 3 as the optimal design. End the process.

If we assume ECC does not exist for soft error, the design flow can be simplified. We can remove Step 2 and binary search in Step 3. We also do not need to evaluate soft error resilience since we assume original ECC does not exist because soft errors are not important. In Step 4, instead of starting with the original ECC, we starting with the largest possible correctable data length (the size of cache line) and the lowest error correction capability (SEC) and estimate the DPPM of the design.

VI. Experiment results

For all the experimental evaluations, we use the aging model proposed in [13]. The desired lifetime in the experiments is 60 months. We adopt the probability collective method proposed in [14] to estimate the failure rate and aging failure rate. Our design objective is to maximize the lifetime yield-per-area of a 2MB SRAM with 6T SRAM cells under target DPPM (i.e., 50). The block size is 64B.

Table IV shows the design results for two study cases. Case 1 has a small access time specification and DEC cannot be adopted in this case. In case 1, we assume SEC-512 already exists for soft error and need to consider soft error resilience. D0 is the SRAM cell design optimized for yield-per-area at the time of fabrication. D2 and D4 are the cell designs produced by the transistor sizing approach for G2 and Unknown workloads. For G1 and Skew, the existing ECC can achieve target DPPM and soft error resilience. For G2, cell design D2 with up-sized pull-down transistors are used. Both target DPPM and soft error resilience are satisfied with minimum area overhead. For Unknown, D4 (larger pull-down transistors compared with D2) is adopted, since DEC cannot be used due to access time constraint. We assume a larger access time limitation for case 2. We do not need to evaluate soft error resilience since we assume original ECC does not exist because soft errors are not important. DEC-512 is adopted for Unknown workload to achieve minimum area overhead without exceeding the access time constraint.

VII. Conclusion

We developed an end-to-end SRAM design framework to maximize the aging resilience under the given constraints. Specifically, we analyzed the impact of aging in SRAM peripheral circuits, including address decoder, precharge, write circuit and sense amplifiers. We conclude that aging causes delay degradation in peripheral circuits and stability degradation in cells. We explored the efficiency of ECC to combat aging by quantifying the area and delay overheads of ECC and estimating the lifetime yield and DPPM of SRAMs with ECC, respectively. We also calculated the soft error resilience when ECC is used to repair aging failures. We find that ECC is efficient for repairing aging failures for workloads with small aging failure rates without sacrificing the soft error resilience. After comparing approaches based on cell sizing and ECC in terms of overheads, lifetime yield and DPPM, we can choose one or a combination of these approaches to identify the optimal design against aging under the given constraints. To provide the end-to-end capability to designers, we integrated our cell sizing approach and our ECC approach into an existing SRAM compiler, CACTI. Our new compiler provides the design with the optimal lifetime yield-per-area under given constraints.

References

[1] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "Cacti 6.0: A tool to model large caches," *HP Laboratories*, pp. 22–31, 2009.

[2] T. T.-H. Kim, W. Zhang, and C. H. Kim, "An sram reliability test macro for fully automated statistical measurements of degradation," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 59, no. 3, pp. 584–593, 2012.

[3] J. Abella, X. Vera, and A. Gonzalez, "Penelope: The nbti-aware processor," in *40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007)*, pp. 85–96, IEEE, 2007.

[4] J. Shin, V. Zyuban, P. Bose, and T. M. Pinkston, "A proactive wearout recovery approach for exploiting microarchitectural redundancy to extend cache sram lifetime," in *ACM SIGARCH Computer Architecture News*, vol. 36, pp. 353–362, IEEE Computer Society, 2008.

[5] S. V. Kumar, K. Kim, and S. S. Sapatnekar, "Impact of nbti on sram read stability and design for reliability," in *7th International Symposium on Quality Electronic Design (ISQED'06)*, pp. 6–pp, IEEE, 2006.

[6] A. Ricketts, J. Singh, K. Ramakrishnan, N. Vijaykrishnan, and D. K. Pradhan, "Investigating the impact of nbti on different power saving cache strategies," in *Proceedings of the conference on design, automation and test in Europe*, pp. 592–597, European Design and Automation Association, 2010.

[7] H. Mostafa, M. Anis, and M. Elmasry, "Adaptive body bias for reducing the impacts of nbti and process variations on 6t sram cells," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 58, no. 12, pp. 2859–2871, 2011.

[8] X. Zuo and S. K. Gupta, "Asymmetric sizing: An effective design approach for sram cells against bti aging," in *VLSI Test Symposium (VTS), 2017 IEEE 35th*, pp. 1–6, IEEE, 2017.

[9] S. Natarajan *et al.*, "A 32nm logic technology featuring 2 nd-generation high-k+ metal-gate transistors, enhanced channel strain and 0.171μm 2 sram cell size in a 291mb array," in *Electron Devices Meeting, 2008. IEDM 2008. IEEE International*, pp. 1–3, IEEE, 2008.

[10] C.-C. Chen, T. Liu, and L. Milor, "System-level modeling of microprocessor reliability degradation due to bias temperature instability and hot carrier injection," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 8, pp. 2712–2725, 2016.

[11] S.-M. Kang, Y. Leblebici, and C. Kim, "Cmos digital integrated circuits: analysis & design," tech. rep., McGraw-Hill Higher Education, 2014.

[12] R. H. Morelos-Zaragoza, *The art of error correcting coding*. John Wiley & Sons, 2006.

[13] S. Bhardwaj, W. Wang, R. Vattikonda, Y. Cao, and S. Vrudhula, "Predictive modeling of the nbti effect for reliable design," in *Custom Integrated Circuits Conference, 2006. CICC'06. IEEE*, pp. 189–192, IEEE, 2006.

[14] F. Gong, S. Basir-Kazeruni, L. Dolecek, and L. He, "A fast estimation of sram failure rate using probability collectives," in *Proceedings of the 2012 ACM international symposium on International Symposium on Physical Design*, pp. 41–48, ACM, 2012.