In-place Repair for Resistive Memories Utilizing Complementary Resistive Switches

Amirali Ghofrani, Miguel Angel Lastras-Montaño, Yuyang Wang, Kwang-Ting Cheng
Electrical and Computer Engineering Department
University of California, Santa Barbara
{ghofrani, mlastras, wyy, timcheng}@ece.ucsb.edu

ABSTRACT

Recent advances in resistive memory technologies have demonstrated their potential to serve as next generation random access memories (RAM) which are fast, low-power, ultra-dense, and non-volatile. However, owing to their stochastic filamentary nature, several sources of hard errors exist that could affect the lifetime of a resistive RAM (ReRAM).

In this paper, we propose a novel mechanism to protect resistive memories against hard errors through the exploitation of a unique feature of bipolar resistive memory elements. Our solution proposes an unorthodox use of *complementary resistive switches* (a particular implementation of resistive memory elements) to provide an "in-place spare" for each memory cell at negligible extra cost. The in-place spares are then utilized by our repair scheme to extend the lifetime of a resistive memory. Our repair scheme detects data errors during regular memory accesses and triggers repair using the in-place spares at a page-level granularity. We show that in-place spares can be used along with other memory reliability and yield enhancement solutions, such as error correction codes (ECC) and spare rows.

We develop a statistical model to evaluate our method's effectiveness on extending ReRAM's lifetime. Our analysis shows that the in-place spare scheme can roughly double the lifetime of a ReRAM system. Alternatively, our method can yield the same lifetime as a baseline ReRAM, with either significantly fewer spare rows or a lighter-weight ECC, both of which can save on energy consumption and area.

CCS Concepts

•**Hardware** → **Memory and dense storage;**

1. INTRODUCTION

CMOS-based memory technologies cannot keep up with the ever-increasing demand for denser and lower-power memories, as technology scaling results in increasing leakage and degraded reliability of memory elements. As an alternative, emerging metal-oxide valence-change resistive memory technology, generally referred to as memristors [1], have demonstrated great potential as the next generation random access memories.

A memristor is a two-terminal passive programmable resistor, that maintains its resistance in the absence of an electric field. High/low resistances are used to represent logic value 0/1. Memristors exhibit a set of desirable characteristics, including low-power operation [2], fast switching speed [3], possible elimination of the access-transistor per memory cell [4], and CMOS compatibility [5]. Ultra-high density memristive memory arrays can be realized as memristor's feature size can be shrunk to a sub-10nm scale [6]. Multiple layers of such arrays can be stacked on top of each other to further increase the density [7]. Minor modifications to the device stack, can further provide a double-memristor cell in place of a regular memristor, also known as complementary resistive switches (CRS) [8].

However, different sources of error could affect memristive devices, owing to their stochastic filamentary nature [9]. Physical defects and endurance problems could lead to "hard errors", which are permanent failures of memory cells [10]. This is in contrast to "soft errors", that are random recoverable errors due to causes such as retention failures [11] or write time variation [12]. Hard errors are commonly addressed by adding redundancy, e.g., by remapping a row with defective bits to a healthy spare row. The existing redundancy-based repair schemes come with the area overhead of the spares, as well as the area and performance overhead of the remapping logic [13]. Alternatively, error correction codes (ECC) have been proposed to detect and correct few erroneous bits, hard or soft, by encoding the data and adding parity bits [14]. However, ECC incurs considerable overhead in terms of area and energy. The energy overhead becomes even more prominent when the ECC is applied to ultra-low-power ReRAMs.

In this paper we propose a novel use of complementary resistive switches (CRS) to provide a zero-area-overhead *in-place spare* for each bit. A repair mechanism is also presented to activate the in-place spares. The proposed method extends the lifetime of memristive memory modules with minor modifications to the memory architecture. We derive a statistical model to evaluate the effectiveness of the proposed method for lifetime improvement of ReRAMs. Our model incorporates the impact of the ECC and the spare rows on ReRAM's lifetime. We also explore the possibility of using the in-place spares to yield a similar lifetime as a baseline ReRAM, for the objective of minimizing spare rows or using a lighter-weight ECC. We quantify the possible reduction in the area and energy consumption of a ReRAM if the in-place spare scheme is employed.

The rest of the paper is organized as follows: Section 2 covers the necessary backgrounds on memristors. Section 3 discusses the origins of errors in memristive devices, as well as the solutions employed in conventional memory technologies. The use of in-place spares is detailed in Section 4. Section 5 presents a statistical model and evaluates the effectiveness of the proposed method. Section 6 concludes the paper.

2. BACKGROUND

2.1 Memristive Devices

A memristor, or a memristive device, is a passive programmable resistor, experimentally found by the HP Labs in 2008 [15]. Mem-

ISLPED '16, August 08-10, 2016, San Francisco Airport, CA, USA

DOI: http://dx.doi.org/10.1145/2934583.2934590

ristors are typically fabricated as a stack of thin layer(s) of non-conductive switching oxide, sandwiched between conductive metallic electrodes, as shown in Figure 1a.

Most devices need a *forming* step, in which a large *forming* voltage, V_{form}, is applied on the device[16]. The forming step breaks the oxide and migrate a large number of oxygen ions to the cathode, resulting in one or several filaments of oxygen vacancies inside the oxide layer [10]. These oxygen vacancies are conductive. Figure 1b shows a memristor after forming.

The resistance of the device can be reversibly switched between a high-resistance OFF state and a low-resistance ON state, by applying negative or positive voltage (current) pulses respectively. Applying a negative pulse mobilizes oxygen ions to recombine with the oxygen vacancies. The recombination partially ruptures the conductive filament and switches the device into an OFF state, as depicted in Figure 1c (i.e. a RESET operation). A positive pulse regenerates the oxygen vacancies and rebuilds the filaments (i.e. a SET operation). Memristors typically exhibit a very high OFF to ON resistance ratio [17].

Figure 1e shows a typical electrical characteristics of a memristor. Applying write voltages above a memristive write threshold, $\pm V_{thm}$, changes the resistance of the device, while applying smaller voltages has negligible effect on its state [17]. The resistance value of a memristor is read by applying a small read voltage and monitoring the resulting current.

2.2 Complementary Resistive Switches

In 2010, Linn *et al.* [18] proposed the concept of complementary resistive switches (CRS) by anti-serially stacking two memristors sharing a common electrode, as shown in Figure 2a. Simpler CRS structures were proposed later by removing the common electrode and having two layers of the same oxide material but with different stoichiometries (e.g. Ta_2O_5 and TaO) [8], as illustrated in Figure 2b. The latter structure makes the cost and complexity of fabricating a CRS similar to that of a regular memristor.

The CRS was proposed to store data based on the combined state of the top and bottom memristors, M_t and M_b, rather than the overall device resistance. A CRS represents logic 0 (CRS-0) when the $\{M_t, M_b\}$ pair is in the $\{ON, OFF\}$ state. Similarly an $\{OFF, ON\}$ configuration represents a logic 1 (CRS-1). With M_t and M_b being in series, both configurations exhibit a very high resistance which leads to lower current requirements and reduced power consumption [19].

Figure 2c illustrates a typical I-V characteristics of a CRS device, that exhibits two types of switching: CRS and memristive switching. Applying a voltage pulse above a CRS write threshold, V^+_{thc}, results in a *CRS switching* which forms a *strong* conductive filament in M_b while turning M_t OFF. Hence, a CRS is switched to an $\{OFF, ON\}$ configuration (i.e. transition ① in Fig 2c). With such a strong filament in M_b, applying voltages in the *memristive write region* (i.e. $[V^-_{thc}, V^-_{thm}]$ and $[V^+_{thm}, V^+_{thc}]$) provides a regular memristive write access to the top device M_t without affecting M_b: The top device exhibits a regular *memristive switching* behavior, and can switch to OFF (i.e. transition ②) or ON states (i.e. transition ③). Blue and red lines in Figure 2c show memristive and CRS switchings respectively.

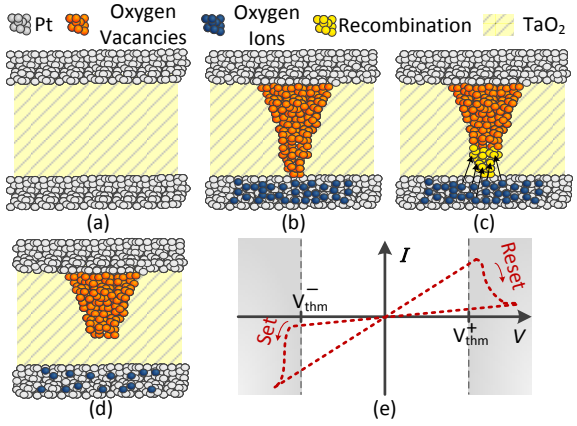

Figure 1: A possible realization of a memristor. a) A memristive device stack before forming, b) A formed memristive device in the ON state. c) RESET process d) a memristor in an OFF state e) a typical electrical characteristics of a memristor.

Similarly, applying voltage pulses below V^-_{thc} switches a CRS into an $\{ON, OFF\}$ state and forms a strong filament in M_t (i.e. transition ④). With a strong filament in M_t, applying a voltage pulse in the memristive write region switches M_b to ON and OFF states, i.e. transitions ⑤ and ⑥ respectively. Note that M_t and M_b are anti-serially connected, thus they require opposite voltage polarities to switch.

In a nutshell, when one of the devices in a CRS stack is set to a strong ON state via a CRS switching, the other device can be written to exclusively, with regular memristive write accesses. Such voltage-range-controlled state transitions in CRS devices present a unique yet uninvestigated feature of such devices: to provide a dual-memristor memory cell with individual accesses to each of the constituent devices at the exact same footprint of a regular memristor. We explore this feature to provide a spare for each memory bit and extend the lifetime of ReRAMs.

3. FAILURE MECHANISMS AND SOLUTIONS

3.1 Soft Errors

Soft errors in memristive devices are recoverable data errors generally due to an "unintended" formation/rupture of the conductive filament inside a memristor. *Retention failure* [11] is an example, which occur when a weak conductive filament is ruptured due to the stochastic movement of the conductive particles, causing an ON→OFF flip.

The cycle-to-cycle variation in the write time of memristors, i.e. the time required to switch a device, is another source of soft errors in ReRAMs. Memristive devices could have ultra-slow write cycles for which the duration of the applied write pulse is not enough to switch the device [12]. An adaptive write mechanism can be used to address this issue, which monitors the state of the target cells during a write operation, and report any unsuccessful bit-write to trigger a re-write [20].

3.2 Hard Errors

Hard errors are due to permanent structural transformations inside a memristive cell. Several mechanisms lead to stuck-at-ON (*S@ON*) hard errors in memristors. *Extra vacancy attributed* failures result in an "irreversible" increase in the diameter of the conductive filament. The *depletion of the cathode from oxygen ions* is another reason behind *S@ON* failures that reduces the recombination probability of oxygen vacancies and oxygen ions and decreases a memristor's OFF resistance [10].

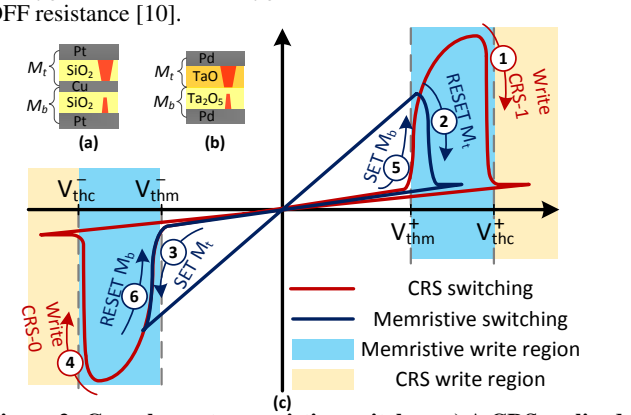

Figure 2: Complementary resistive switches. a) A CRS realized by two anti-serially memristors stacked on top of each other. b) Simple CRS structure with removed middle electrode. c) Typical I-V diagram and state transitions.

As for stuck-at-OFF ($S@OFF$) failures, *over-reset* phenomena has been reported in which over-stressing the device with consecutive RESET operations could lead to a complete dissolution of the conductive filaments inside the device. An over-reset device cannot be recovered with regular SET operations [3].

While $S@OFF$ hard-errors might be fixed by applying high-voltage pulses (i.e. an extra forming step), $S@ON$ errors are harder to address. In a $S@ON$, a memristor is shorted and has a very low electrical resistance that is comparable to that of the voltage drivers' transistors. Hence, even in the case of applying a high-voltage RESET pulse, the effective voltage on the device would be small due to the large IR-drop on the line drivers' transistors, and thus cannot reverse the failure.

To the best of our knowledge, there is no comprehensive study on the relative error rate of $S@ON$ and $S@OFF$ errors in ReRAMs. However, the abundance of studies on $S@ON$ failures [10, 21, 22], as well as the reversibility of some $S@OFF$ errors (e.g. by another forming step)[3] suggests a higher error rate for $S@ON$ failures.

3.3 Potential Solutions

Soft errors are commonly addressed by the use of error correction codes (ECC). An ECC encodes the data and adds parity bits which enables the correction of T bits of errors during a read access. An ECC can also detect up D faulty bits, where D is greater than T. There is a myriad of ECCs in the literature, providing various levels of protection against errors. Among the most commonly used ECCs for memories are Hamming codes [23], that offer single-error-correction (i.e. $T = 1$), and Bose-Chaudhuri-Hocquenghen (BCH) codes [24], that are a family of ECC with multi-bit error correction capability.

Memory scrubbing is another method to address soft errors [25]. A scrubbing controller periodically reads data words, check them for errors through the use of ECC, and write the corrected data back in case of an error.

ECC can also be utilized to address hard errors. *Scrambling* methods are applied to distribute the faulty bits into different code-words such that the number of faulty-bits per code-word is less than the correction capability of the adopted ECC [26]. However, using ECC to correct hard errors reduces its effective correction capability against random soft errors. Moreover, ECC comes with noticeable energy overhead as it surcharges an encoding/decoding phase to each memory access. This is in addition to the area overhead of the parity bits and the ECC logic. The overhead increases with the ECC strength: stronger ECCs can correct more errors, but also incur more overhead.

Redundancy-based repair schemes are used to specifically address hard errors. Such repair schemes detect hard errors and use embedded "spare rows" to replace faulty rows [27, 14]. The row replacement is accomplished with the help of a remapping logic which relies on a content addressable memory (CAM). A CAM stores the addresses of the faulty rows along with the addresses of existing spares to replace them. The remapping logic uses the CAM to redirect future accesses to faulty rows to their corresponding spares [13]. To support the use of spare-rows, an additional access to the remapping CAM is necessary for each memory access, which adds a performance penalty to all memory accesses. Increasing the number of spares provides greater protection against hard errors, at the cost of increased area and performance penalty due to a larger CAM.

4. MOTIVATION AND PROPOSAL

Existing solutions to extend the lifetime of memories and protect them against failures, such as spare rows and ECC, come with considerable energy and area overhead. This overhead becomes even more noticeable for emerging ReRAM technologies which are ultra-small and ultra low power. Hence, low-cost solutions that can help reduce such overheads will be attractive and valuable. To this end, we explore the use of complementary resistive switches, to provide "virtually-free" in-place spares per each memory element to extend the lifetime of a ReRAM.

4.1 CRS Devices as In-place Spares

A complementary resistive switch can be used to realize dual-memristor memory elements. It is shown that the unique electrical characteristics of a CRS provides exclusive write accesses to each of the two constituent devices by controlling the range of applied write voltages [19]. Furthermore, an exclusive read access to either of the devices in a CRS stack can be realized by keeping the other device in an ON state. Note that a CRS read operation reads the "total" resistance of the constituent devices that are in series. Hence, keeping either of the devices in an ON state makes it transparent to the read operation, in view of the orders of magnitude difference between the ON and OFF resistance values of a memristive device [17].

Inspired by the possibility of such an exclusive read and write accesses, we propose the use of the extra memristor in a CRS cell as a spare. For clarity, we consider the top memristor in a CRS stack, M_t, as the *spare*, and the bottom one, M_b, as the *primary* device. The idea is to first utilize the primary device as the active memory element, and then use the spare, upon the failure of the primary device. The primary device is "activated" by applying a CRS write pulse below V_{thc}^{-}, as shown in Figure 3a. Such pulse initializes the $\{M_t, M_b\}$ pair to an $\{ON, OFF\}$ state and keeps the spare in an ON state that is transparent to read or write accesses. When activated, the primary device can be written to through regular write accesses without affecting the spare. That is, M_b can be switched between ON and OFF states (i.e. $\{ON, ON\}$ and $\{ON, OFF\}$ CRS configurations), as shown in Figure 3b, until it fails due to a hard error. If the primary device fails into a $S@ON$ state (which is more likely to happen than $S@OFF$, as discussed in Section 3.2), the memory element can be *repaired* by activating the spare device. To this end, a one-time CRS write pulse above V_{thc}^{+} sets the CRS to an $\{OFF, ON\}$ state (Figure 3c). From there on, the spare device becomes the active memory element while the $S@ON$ primary device is transparent to the memory accesses.

A $S@OFF$ failure of the primary device could render the spare useless, as in that case, the whole CRS cell becomes $S@OFF$. In section 5 we examine the effect of $S@OFF$ failure rates and show that even under a pessimistic assumption that the $S@ON$ and $S@OFF$ error rates are equal, our method can still improve the memory lifetime considerably.

The use of such "in-place" spares provides two main advantages over the conventional redundancy-based repair schemes such as spare-rows: 1) No area-overhead is incurred, as the spare devices are fabricated on top of the primary devices and as part of the same device stack, and 2) in contrast to the spare-rows, such in-place spares exist at the exact same address as the failed memory element. Hence, there is no need for address remapping to activate the spares, and thus the overhead associated with the remapping logic can be avoided.

In order to differentiate the proposed use of a CRS as a dual-memristor-cell (DMC) with in-place spares, from the original CRS concept, hereafter we refer to a CRS stack as DMC.

4.2 Architectural Modifications

The anti-serially connected memristors in a DMC are accessed with opposite polarities: while applying a positive write pulse switches an active primary device into an OFF state, the same pulse would turn ON the spare device if it is activated. Hence, the memory management system needs to track which device in a DMC is active to ensure that for a write operation, the right voltage polarity is applied to the active device.

We propose the use of a "polarity bit", *pbit*, to track the active device in a DMC. The *pbit* is accessed prior to each write operation to select the proper write voltage polarity. To minimize the overhead of bookkeeping, we use only one *pbit* per block: either all DMCs in a block use the primary device as their active device, or all of them

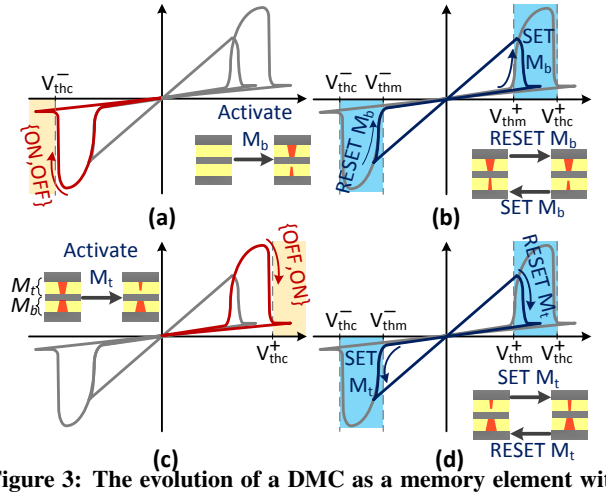

Figure 3: The evolution of a DMC as a memory element with an in-place spare: a) Activation of the primary device with a V_{thc}^{-} pulse. b) Use of the primary device as a regular memristor with ON $\leftrightarrow$ OFF switching. c) Spare activation with a V_{thc}^{+} pulse. d) The spare is used as the active memristor.

use the spare.

To avoid the performance penalty of accessing the *pbit*, we set the block size to that of a *page* to take advantage of the *paging system* commonly implemented in the OS [25]. The OS maintains validity, permission, and address translation information for fixed-length contiguous blocks of memory that are called *pages*, in a *page table entry*. Page table entries are loaded into an extremely fast CAM called *translation look-aside buffer* (TLB), and are accessed as a part of each memory operation. Hence, by storing the DMC polarity data, i.e. *pbit*, at a page-level granularity, the *pbit* can be stored in the corresponding page table entry and accessed with no extra performance penalty during a write operation. Figure 4 highlights the minor modifications made to the datapath of a ReRAM to track the page polarity in green.

4.3 Repair Mechanism

The ECC can only correct up to T bits of errors per word-line. To extend the lifetime of a ReRAM, we propose a repair scheme that employs in-place spares to repair word-lines with more than T bits of errors. Our repair scheme reuses commonly adopted reliability improvement mechanisms, i.e. ECC and the adaptive write mechanism, to detect the number of errors during regular memory accesses: an adaptive write mechanism reports the exact number of bit-errors during a write operation, while ECC can detect up to D bits of errors during a read access, where D is larger than T.

Knowing the number of bit-failures, N_e, our repair scheme triggers the replacement of a word with a spare, as soon as N_e exceeds the correction capability of the ECC. Possible spare rows are utilized first to replace a defective word, W_f. The "address remapping" logic is configured to remap W_f to an available spare row. Once the spare rows are exhausted, the repair mechanism is triggered and the in-place spares are activated for the whole page.

The activation process of the in-place spares consists of three phases: For each word in a page, 1) the word is read and stored in a buffer, 2) spare devices are activated by applying a V_{thc}^{+} voltage pulse to the device stack, and 3) the buffered values are written back to the spare-activated word-line. The repair controller also updates the *pbit* and resets the remapping logic for the spare-activated page. Figure 4 shows a ReRAM equipped with in-place spares.

5. ANALYSIS AND RESULTS

5.1 Viability Model

In order to evaluate the effect of the in-place spares on extending the lifetime of a ReRAM, we derive a statistical model to assess

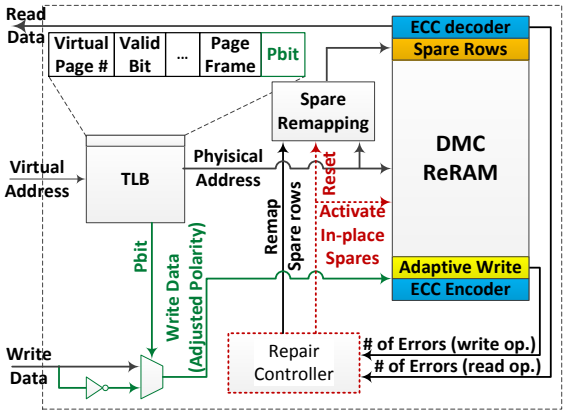

Figure 4: A DMC ReRAM. Green components keep track of the DMC page polarity: *pbit* is read from a page table entry to select the write voltage polarity. Dashed red components enable the repair: The existing repair controller is extended to activate in-place spares for the whole page once all spare rows are exhausted.

the *viability* of a ReRAM system in the presence of hard and soft errors. A viability function, $V(t)$, is defined as the probability that by time t, a ReRAM system has not yet experienced a failure, i.e. an error that cannot be corrected by ECC or repaired by spares. We use the Poisson distribution to model the probability of a $S@ON$ (or $S@OFF$) bit-failure at time t, $P_{S@1(or\ 0)}(t)$, with a fixed error rate, λ_1 (or λ_0) [27]:

$$P_{S@1(0)}(t) = 1 - e^{-\lambda_{1(0)}t} \tag{1}$$

Similarly, a fixed error rate λ_s is assumed for soft errors. We further consider a correction rate, μ, to model soft error mitigation mechanisms such as scrubbing. Equation 2 derives the probability of having a faulty bit due to a soft error, $P_{SE}(t)$:

$$P_{SE}(t) = \frac{\lambda_s}{\mu + \lambda_s}(1 - e^{-(\mu+\lambda_s)t}) \tag{2}$$

With the use of ECC, a B-bit word-line (that has B_D data bits and B_P parity bits) is viable as long as the total number of hard and soft bit-errors per word does not exceed T. Note that the number of parity bits depends on the ECC correction capability. $V_W(t,t_a)$ captures the viability of a word-line:

$$V_W(t,t_a) = \sum_{i=0}^{T}\sum_{j=0}^{T-i}\sum_{k=0}^{T-i-j}\binom{B}{i}P_{S@0}(t)^i(1-P_{S@0}(t))^{B-i} \cdot\binom{B-i}{j}P_{S@1}(t-t_a))^j(1-P_{S@1}(t-t_a))^{B-i-j} \cdot\binom{B-i-j}{k}P_{SE}(t)^k(1-P_{SE}(t))^{B-i-j-k} \tag{3}$$

where i, j, and k represent the number of $S@OFF$, $S@ON$, and soft errors respectively, and t_a denotes the activation time of the in-place spares. Note that activating the in-place spares *resets* $S@ON$ errors in a DMC ReRAM. Hence, for the calculation of the $S@ON$ bit-failure probability, the time origin is adjusted accordingly. The t_a equals 0 when measuring the viability of a word with regular memristors or a DMC word but before the activation of the in-place spares.

Considering S spare rows per page, a page with W words remains viable as long as the number of faulty words in the page does not exceed S. Equation 4 captures the page viability, $V_{page}(t,t_a)$,

Table 1: Simulation parameters

Parameter	Description	value
λ_s	Soft error rate	10^{-12}
λ_1	stuck at ON error rate	10^{-10}
λ_0	stuck at OFF error rate	$\rho\lambda_1$
ρ	$S@ON$ to $S@OFF$ error ratio	{1,10,100}
μ	Soft error correction rate	10^{-11}
T	ECC Correction capability	{0,1,2}
W	# of words per page	1024
S	# of spare words per page	[0-64]
B_D	# of data bits per word	{64,128,256}

assuming hot spare rows:

$$V_{page}(t,t_a) = \sum_{i=0}^{S} \binom{W+S}{i} V_w(t,t_a)^{W+S-i}(1-V_w(t,t_a))^i \quad (4)$$

The viability of a regular ReRAM page is obtained by setting t_a equal to 0 in Equation 4. In case of a DMC memory, the page viability both before and after the activation of the in-place spares should be considered, as given in Equation 5:

$$V_{DMC}(t) = V_{page}(t,0) + \int_0^t -V'_{page}(t_a,0)V_{page}(t,t_a)dt_a \quad (5)$$

The first term in Equation 5 represents the viability of a page prior to the in-place spare activation. The activation of the in-place spares at time t_a provides an additional viability, $V_{page}(t,t_a)$. However, t_a is a random variable in the $[0,t]$ range. Hence, the viability component due to the spare activation is integrated over this range, with regard to the probability distribution function of t_a, that is $-V'_{page}(t,0)$.

The lifetime of a memristive page can be derived based on the viability function, according to Equation 6:

$$Lifetime = \int_0^\infty -tV'_{DMC}(t)dt \quad (6)$$

Note that while our calculations employ a Poisson distribution for hard and soft errors, other distributions can be applied by customizing Equations 1 and 2. Furthermore, a ReRAM with no spare rows and/or ECC, can be modeled simply by setting S and/or T to 0, respectively. Table 1 summarizes the employed parameters and their exemplar values.

5.2 Effect of In-place Spares on Lifetime

Figure 5 illustrates the viability of a DMC ReRAM page (dashed lines) versus that of a baseline regular ReRAM (solid lines). Results are shown for an exemplar case of $T=2$, $B_D=64$, $W=1024$, $S=8$, $\lambda_s=10^{-12}$, $\lambda_1=10^{-10}$, $\mu=10^{-11}$, and for different $S@ON$ to $S@OFF$ error rate ratios, ρ. To quantify the viability improvements,

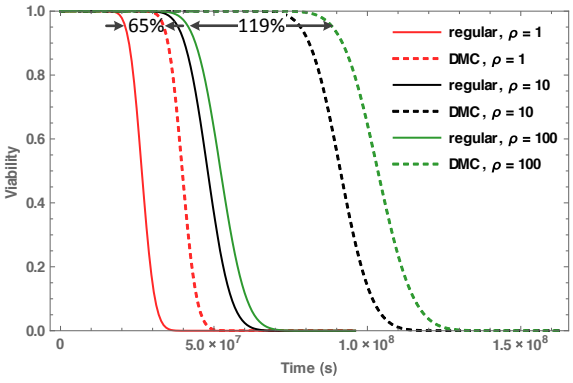

Figure 5: Viability of a DMC ReRAM versus a regular ReRAM for different $S@ON$ to $S@OFF$ error rate ratios.

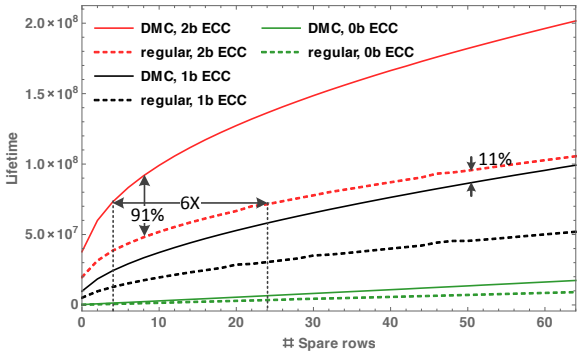

Figure 6: The effect of in-place spares, ECC correction capability, and the number of spare rows on the lifetime of a ReRAM page.

we consider the time at which a memory page shows 99% viability, $t_{99\%}$. For $\rho=100$ (green lines), a DMC ReRAM extends $t_{99\%}$ by 119%. Even with equal $S@ON$ and $S@OFF$ error rates (red lines), $t_{99\%}$ is still improved by over 65%.

Figure 6 illustrates the effect of the in-place spares on the lifetime of a ReRAM page as a function of S and T, while ρ is set to 10. For example, with $S=8$ and $T=2$, use of the in-place spares can increase the lifetime by 91%.

Figure 6 demonstrates the possibility of using in-place spares to reduce the number of spare rows. For example, with $T=2$, a DMC ReRAM page with four spare rows provides the same lifetime as a regular memristive page with 24 spare rows.

The use of in-place spares also provides an opportunity to use lighter-weight ECCs in a ReRAM system to save on the area and energy requirements of the ECC, while maintaining a similar ReRAM lifetime. For an exemplar ReRAM page with 48 spare rows, i.e. 4% row redundancy, a DMC ReRAM page protected by a single-error-correcting (SEC) ECC exhibits a lifetime that is only 11% short of that of a regular page with a double-error-correcting (DEC) ECC.

5.3 Energy and Area Reduction

Figure 7 illustrates the possible reduction in area and power consumption by using the in-place spares to reduce the number of spare rows. This reduction is mainly attributed to the reduction in the size of the CAM module in the remapping logic. The horizontal axis shows the number of spares in pairs of $\{S_{DMC}, S_{Reg}\}$, where S_{Reg} is the necessary number of spare rows in a regular ReRAM, to provide a lifetime similar to that of a DMC ReRAM with S_{DMC} spare rows. The vertical axis shows the area and power overhead of a CAM to support S_{DMC} and S_{Reg} spare rows, respectively. For example, a DMC ReRAM page with six spare rows, provides the same lifetime as a regular ReRAM page with 36 spare rows. Hence, with smaller number of spare rows required, the power and area requirements of the remapping CAM can be reduced by 81% and 83%, respectively. Power and area numbers are obtained by synthesizing different CAM sizes with Synopsys design compiler at a

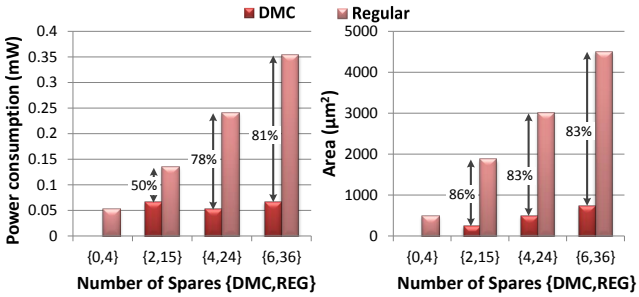

Figure 7: Power consumption and area requirements of the remapping logic CAM. A DMC ReRAM provides a similar lifetime with significantly lower number of spare rows, thus saves on the CAM's area and power consumption.

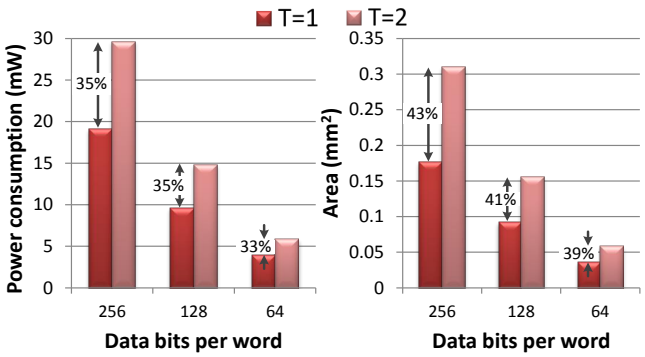

Figure 8: Power consumption and area requirements of a SEC and a DEC BCH ECC logic. The in-place spares enable employing a lighter-weight ECC, thus saving on area and power.

45nm CMOS technology node targeting a 200ps latency.

Figure 8 shows the potential of the in-place spares to reduce the area and energy consumption of a ReRAM system by enabling lighter-weight ECCs while maintaining a similar lifetime. SEC and DEC BCH encoder/decoders are synthesized using Synopsys design compiler, targeting a 400ps latency in a 45nm CMOS technology. For 128-bit data words, reducing the a BCH's correction capability from two to one reduces the area and power consumption of the ECC logic by 41% and 35%, respectively. The savings improve further for words with more data bits. Note that reducing the ECC complexity results in greater savings compared to the savings resulting from reducing the number of spare rows.

6. CONCLUDING REMARKS

We propose a novel use of complementary resistive switches, to provide a dual-memristor-cell (DMC) with an in-place spare for ReRAM at negligible extra cost. The in-place spares can repair stuck-at-ON defects that are prevailing in a ReRAM system. Unlike conventional redundancy-based schemes, the proposed method incurs no area overhead due to the spares and does not require a remapping logic.

We present a statistical model to evaluate the effectiveness of the proposed method on extending the ReRAM's lifetime. The use of the in-place spares can roughly double the lifetime of ReRAMs. Alternatively, a DMC ReRAM exhibits a similar lifetime to a regular ReRAM, but with a lighter-weight ECC. The reduction in the complexity of the ECC can save an average of 39-43% on the area and 33-35% on the energy consumption of a BCH ECC module. Similarly, use of a DMC ReRAM can save on power and area by reducing the number of spare rows required to attain a given lifetime, which results in ≈6X reduction in the area overhead and power consumption of a CAM module inside the remapping logic.

7. REFERENCES

[1] L. Chua, "Resistance Switching Memories Are Memristors," *Applied Physics A: Materials Science & Processing*, vol. 102, no. 4, pp. 765–783, 2011.

[2] J. P. Strachan, A. C. Torrezan, M. F. Miao, M. D. Pickett, J. J. Yang, W. Yi, G. Medeiros-Ribeiro, and R. S. Williams, "Measuring the Switching Dynamics and Energy Efficiency of Tantalum Oxide Memristors," *Nanotechnology*, vol. 22, p. 505402, November 2011.

[3] H. Y. Lee *et al.*, "Evidence and Solution of Over-RESET Problem for HfO_x-based Resistive Memory with sub-ns Switching Speed and High Endurance," in *Electron Devices Meeting (IEDM), 2010 IEEE International*, pp. 19–7, IEEE, 2010.

[4] J. J. Yang, M.-X. Zhang, M. D. Pickett, M. Feng, J. P. Strachan, W.-D. Li, W. Yi, D. A. A. Ohlberg, B. J. Choi, W. Wu, J. H. Nickel, G. Medeiros-Ribeiro, and R. S. Williams, "Engineering Nonlinearity into Memristors for Passive Crossbar Applications," *Applied Physics Letters*, vol. 100, no. 11, p. 113501, 2012.

[5] J. Rofeh, A. Sodhi, M. Payvand, M. A. Lastras-Montaño, A. Ghofrani, A. Madhavan, S. Yemenicioglu, K.-T. Cheng, and L. Theogarajan, "Vertical Integration of Memristors onto Foundry CMOS Dies using Wafer-Scale Integration," in *Electronic Components and Technology Conference (ECTC) , 2015 IEEE 65th*, pp. 957–962, May 2015.

[6] C. Ho, C.-L. Hsu, C.-C. Chen, J.-T. Liu, C.-S. Wu, C.-C. Huang, C. Hu, and F.-L. Yang, "9nm Half-Pitch Functional Resistive Memory Cell with 1 uA Programming Current Using Thermally Oxidized sub-Stoichiometric WOx Film," in *International Electron Devices Meeting (IEDM)*, pp. 19.1.1–19.1.4, IEEE, 2010.

[7] A. Kawahara, R. Azuma, Y. Ikeda, K. Kawai, Y. Katoh, Y. Hayakawa, K. Tsuji, *et al.*, "An 8 Mb Multi-Layered Cross-Point ReRAM Macro with 443 MB/s Write Throughput," *Solid-State Circuits, IEEE Journal of*, vol. 48, no. 1, pp. 178–185, 2013.

[8] Y. Yang, P. Sheridan, and W. Lu, "Complementary Resistive Switching in Tantalum Oxide-based Resistive Memory Devices," *Applied Physics Letters*, vol. 100, no. 20, p. 203112, 2012.

[9] C. Xu, D. Niu, Y. Zheng, S. Yu, and Y. Xie, "Impact of Cell Failure on Reliable Cross-Point Resistive Memory Design," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 20, no. 4, p. 63, 2015.

[10] B. Chen, Y. Lu, B. Gao, Y. H. Fu, F. F. Zhang, P. Huang, Y. S. Chen, *et al.*, "Physical Mechanisms of Endurance Degradation in TMO-RRAM," in *2011 International Electron Devices Meeting*, 2011.

[11] B. Gao, H. Zhang, B. Chen, L. Liu, X. Liu, R. Han, J. Kang, Z. Fang, H. Yu, B. Yu, and D.-L. Kwong, "Modeling of Retention Failure Behavior in Bipolar Oxide-Based Resistive Switching Memory," *Electron Device Letters, IEEE*, vol. 32, pp. 276–278, March 2011.

[12] S. Yu, X. Guan, and H.-S. P. Wong, "On the Switching Parameter Variation of Metal Oxide RRAM; Part II: Model Corroboration and Device Design Strategy," *IEEE Transactions on Electron Devices*, vol. 59, no. 4, pp. 1183–1188, 2012.

[13] M. Lee, L.-M. Denq, and C.-W. Wu, "A Memory Built-in Self-repair Scheme based on Configurable Spares," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 30, no. 6, pp. 919–929, 2011.

[14] T.-H. Wu, P.-Y. Chen, M. Lee, B.-Y. Lin, C.-W. Wu, C.-H. Tien, *et al.*, "A Memory Yield Improvement Scheme Combining Built-in Self-repair and Error Correction Codes," in *Test Conference (ITC), 2012 IEEE International*, pp. 1–9, IEEE, 2012.

[15] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The Missing Memristor Found," *Nature*, vol. 453, no. 7191, pp. 80–83, 2008.

[16] F. Miao, J. J. Yang, J. Borghetti, G. Medeiros-Ribeiro, and R. S. Williams, "Observation of Two Resistance Switching Modes in TiO_2 Memristive Devices Electroformed at Low Current," *Nanotechnology*, vol. 22, no. 25, p. 254007, 2011.

[17] J. J. Yang, D. B. Strukov, and D. R. Stewart, "Memristive Devices for Computing," *Nature Nanotechnology*, vol. 8, pp. 13–24, January 2013.

[18] E. Linn, R. Rosezin, C. KÃijgeler, and R. Waser, "Complementary Resistive Switches for Passive Nanocrossbar Memories," *Nature Materials*, vol. 9, pp. 403–406, 2010.

[19] M. A. Lastras-Montaño, A. Ghofrani, and K.-T. T. Cheng, "HReRAM: A Hybrid Reconfigurable Resistive Random-Access Memory," *Proceedings Design, Automation, and Test in Europe (DATE), IEEE*, 2015.

[20] A. Ghofrani, M.-A. lastras-montaño, S. Gaba, M. Payvand, W. Lu, L. Theogarajan, and K.-T. Cheng, "A Low-Power Variation-Aware Adaptive Write Scheme for Access-Transistor-Free Memristive Memory," *ACM Journal on Emerging Technology in Computing Systems*, vol. 12, pp. 3:1–3:18, Aug. 2015.

[21] D. Strukov, "Endurance write speed tradeoffs in nonvolatile memories," *arXiv preprint arXiv:1511.07109*, 2015.

[22] B. Gao, H. Zhang, B. Chen, L. Liu, X. Liu, *et al.*, "Modeling of Retention Failure Behavior in Bipolar Oxide-based Resistive Switching Memory," *Electron Device Letters, IEEE*, vol. 32, no. 3, pp. 276–278, 2011.

[23] R. W. Hamming, "Error Detecting and Error Correcting Codes," *Bell System technical journal*, vol. 29, no. 2, pp. 147–160, 1950.

[24] R. C. Bose and D. K. Ray-Chaudhuri, "On a Class of Error Correcting Binary Group Codes," *Information and control*, vol. 3, no. 1, pp. 68–79, 1960.

[25] B. Jacob, S. Ng, and D. Wang, *Memory Systems: Cache, DRAM, Disk*. Morgan Kaufmann, 2010.

[26] S.-K. Lu, H.-C. Jheng, H.-W. Lin, M. Hashizume, and S. Kajihara, "Built-In Scrambling Analysis for Yield Enhancement of Embedded Memories," in *Test Symposium (ATS), 2014 IEEE 23rd Asian*, pp. 137–142, IEEE, 2014.

[27] G. Mayuga, Y. Yamato, T. Yoneda, M. Inoue, and Y. Sato, "An ECC-based Memory Architecture with Online Self-repair Capabilities for Reliability Enhancement," in *Test Symposium (ETS), 2015 20th IEEE European*, pp. 1–6, IEEE, 2015.