A Novel Program-Verify Algorithm for Multi-bit Operation in HfO$_2$ RRAM

F. M. Puglisi, Student Member, IEEE, C. Wenger, and P. Pavan, Senior Member, IEEE

Abstract—In this letter we propose a dispersion-aware program-verify (DAPV) algorithm to enable reliable multi-bit operations in HfO$_2$-based RRAM. The significant intrinsic dispersion of the resistive states, typically hindering multi-bit operations, is exploited to devise a program-verify scheme which enables multi-bit operations with unique properties of failure resilience and adaptability to degradation. We show that an appropriate choice of the algorithm parameters can minimize the average number of cycles needed to program the cell, enabling fast and reliable multi-bit operation. This maximizes the bit/cell ratio and minimizes the dispersion of targeted resistive states.

Index Terms—RRAM; Program-Verify; Variability; Resistive Switching, Multi-bit.

I. INTRODUCTION

RESISTIVE Random Access Memory (RRAM) based on HfO$_2$ allows fast low-power operation, and high-density [1-2], becoming a promising alternative non-volatile memory. Still, inherent stochastic features, i.e. randomness in the switching mechanism [3-5], hamper multi-bit operations. Major concerns are cycling variability [3-4], i.e. considerable dispersion of High-Resistive State (HRS) and Low-Resistive State (LRS) resistances, and resistance distributions degradation [6], i.e. changes in the stochastic response of the device over time. Controlling these phenomena, reducing RRAM reliability, requires employing program-verify (PV) schemes. In this letter, we propose a dispersion-aware PV (DAPV) scheme to minimize resistance dispersion and achieve reliable multi-bit operation. PV schemes for HfO$_2$ RRAMs reported in the literature, though efficient, are not tailored to prevent device failures [7-10]. Also, they are not designed to handle resistance distribution degradations. The proposed algorithm exploits the stochastic response of RRAM to the reset operation and the possibility of controlling the average HRS resistance via the reset pulse voltage to achieve unique properties of resilience to failures [11] and adaptability to degradation. The DAPV scheme enables 2 bit-cell operation, with better reliability compared to classical schemes [7-10].

II. DEVICE AND EXPERIMENTS

TiN/10nm-Ti/10nm-HfO$_2$/TiN RRAMs are tested in 1T1R

Francesco Maria Puglisi, and Paolo Pavan are with the Dipartimento di Ingegneria “Enzo Ferrari”, Università di Modena e Reggio Emilia, Via P. Vivarelli 10/1, 41125 Modena – Italy (phone: +39-059-2056524; fax: +39-059-2056329; e-mail: francescomaria.puglisi@unimore.it).

Christian Wenger is with IHP GmbH – Leibniz institute for innovative microelectronics, Im Technologiepark 25, 15236 Frankfurt (Oder), Germany.

Figure 1. R$_{LRS}$ and R$_{HRS}$ distributions for three different V$_{RESET}$. I$_C$ is 100µA. Only R$_{HRS}$ distribution depends on V$_{RESET}$. The colored and black vertical dashed lines represent the minimum of each R$_{HRS}$ distribution and the maximum R$_{LRS}$, respectively. The worst-case read window is indicated by an arrow for each V$_{RESET}$. In the inset, the worst-case (red squares) and the median (blue circles) read window vs. V$_{RESET}$ configuration – devices details are reported elsewhere [12]. The access nMOS is used to enforce a current compliance, I$_C$, to prevent current overshoot [13]. The preliminary forming operation is performed with I$_C$ ranging from 50µA to 100µA by applying a DC voltage sweep up to 3.5V (not shown). Forming results in the creation of a Conductive Filament (CF) [3-5]. Then, we perform 50 complete pulsed switching cycles (set and reset), and we repeat the experiment for different reset conditions. The reset operation, driving the device in HRS, is performed by applying a negative voltage pulse (−V$_{RESET}$), resulting in a partial oxidation of the CF (creation of a dielectric barrier) [3-5]. The set operation, driving the device in LRS, is performed by applying a 2V pulse, and leads to the restoration of the CF [3-5]. The RRAM resistance, R, is measured after each operation by applying a V$_{READ}=$100mV pulse. Set, reset, and read pulses are all 10µs wide, including 2µs of rise/fall times. The total pulse period is 20µs.

III. THE PROGRAM-VERIFY ALGORITHM

Figure 1 shows the probability distributions of both LRS and HRS resistances over 50 complete switching cycles when three different V$_{RESET}$ are applied, namely 1.3V, 1.5V and 1.8V. The LRS resistance (R$_{LRS}$=9kΩ) is normally distributed with no dependence on V$_{RESET}$, consistently with previous results and other studies [3-5, 11]. Conversely, the HRS resistance, R$_{HRS}$, is log-normally distributed [3-4] and strongly depends on V$_{RESET}$, resulting in a significant R$_{HRS}$ dispersion. The unambiguous identification of the device state after each set/reset operation requires no overlap between the tails of the R$_{LRS}$ and R$_{HRS}$ distributions across the whole memory lifetime. Hence, a correct monitor of the device reliability is the worst-
The algorithm ed by the algorithm when succeeding in programming. The HRS MAX target R span. This scenario
achieve “negative set failure” resilience. So, the application of consecutive reset pulses may cause the breakdown of the dielectric barrier within the CF and 2) is performed (given by the dielectric barrier pulses with reset (pulses with
Typically, it consists in delivering to the device a train of
The high electric field associated with two
physical dispersion distributions are overlapped, Fig. 1.
Indeed, the proposed scheme is resilient to the “negative set failure” but shows no adaptability to degradation.

A. Unconstrained Program-Verify (UPV)

A straightforward method to target a tight R<sub>IRS</sub> distribution (from R<sub>MIN</sub> to R<sub>MAX</sub>) while avoiding the “negative set failure” would be the following:

1. Choose V<sub>RESET</sub> and a target R span (R<sub>MIN</sub> - R<sub>MAX</sub>)
2. Reset the device
3. Check if the resistance, R, is R<sub>MIN</sub>&lt;R&lt;R<sub>MAX</sub>
4. If not, apply a set pulse and go back to 2

Notably, in classical schemes the set pulse is applied only when R&gt;R<sub>MAX</sub> [7-10], whereas another reset pulse is applied if R&lt;R<sub>MIN</sub>. Here, instead, a set pulse is applied every time the programmed R value is out of the target R span. This approach is more time-consuming due to the additional set pulse but circumvents the “negative set failure”. However it requires an accurate matching between the applied V<sub>RESET</sub> and the selected R span. For instance, Fig. 1 suggests it is unlikely to achieve 10<sup>5</sup>Ω&lt;R&lt;2·10<sup>7</sup>Ω when V<sub>RESET</sub>=1.3V. This choice will imply the algorithm failure (missing the target R span within 20 cycles). So, the algorithm parameters (V<sub>RESET</sub>, R<sub>MIN</sub>, and R<sub>MAX</sub>) must be wisely chosen to guarantee success and to minimize total programming time:

\[ T_{PROGR} = n(T_{RESET} + T_{SET} + T_{CHECK}) \]  

with \( n \) being the number of cycles, \( T_{RESET} \) (T<sub>SET</sub>) the reset (set) pulse period and \( T_{CHECK} \) the time to verify if R is within the targeted R span. However, if the R<sub>IRS</sub> distribution changes during device operation [6], even a good initial choice of the parameters may cause the algorithm to fail. As such, this scheme is resilient to the “negative set failure” but shows no adaptability to degradation.

B. Dispersion-Aware Program-Verify (DAPV)

A more robust approach is based on the real-time estimation of the most appropriate V<sub>RESET</sub> for the targeted R span. Indeed the R span which can in fact be targeted in the reset operation depends on V<sub>RESET</sub>, Fig. 1. If the algorithm misses the target R span by always exceeding (falling behind) it, then reducing (increasing) V<sub>RESET</sub> is required to achieve the target R span. The proposed scheme is summarized in the flow chart in Fig. 2. The counter “Iter” tracks the number of cycles, while “CountU” and “CountD” track how many consecutive times the programmed R exceeds (falls behind) the target R span. After each program operation (set and reset) and R evaluation, this scheme checks if V<sub>RESET</sub> must be changed for the next iteration. To this point, either CountU or CountD is compared to its maximum value, which is a parameter of the algorithm as well as the initial V<sub>RESET</sub>, R<sub>MIN</sub>, R<sub>MAX</sub>, and the maximum number of cycles Iter<sub>MAX</sub> (typically 20). Moreover, we also set

![Figure 2. Flow chart of the proposed DAPV algorithm. The algorithm parameters are Iter<sub>MAX</sub>, CountU<sub>MAX</sub>, CountD<sub>MAX</sub> and the initial V<sub>RESET</sub>. Orange boxes are comparison operation. Notably, only two comparisons per cycle are added, compared to the UPV.](image)

![Figure 3. R<sub>IRS</sub> and three R<sub>IRS</sub> distributions for 100 cycles as obtained with DAPV. For each target R span, the initial V<sub>RESET</sub> is varied by the algorithm to ensure successful writing and minimal programming time. The table reports the initial V<sub>RESET</sub> for the three target R<sub>IRS</sub> span, along with the median and average values over 100 algorithm runs of the effective V<sub>RESET</sub> which is used by the algorithm when succeeding in programming. The other parameters are set to Iter<sub>MAX</sub>=20, CountU<sub>MAX</sub>=CountD<sub>MAX</sub>=3.](image)
UPV, DAPV offers much better performance. Fig. 5 indicates that DAPV can program a cell in < 10 cycles with no failures, while UPV failure rate in the same conditions is 32%. Also, DAPV has a better success rate at low cycles than UPV, which reduces the possible endurance degradation resulting from the application of program-verify schemes. So, though being computationally heavier than UPV due to higher complexity (compared to UPV, two comparisons per cycle are added, Fig. 2), the DAPV shows lower average programming time.

V. CONCLUSIONS

We proposed a new dispersion-aware program-verify algorithm for HfO\textsubscript{2} RRAM. It exploits the intrinsically stochastic response of RRAMs to achieve an optimal control of resistance values. It allows reliable multi-bit operation with no failures and optimized programming time, while displaying the unique properties of failure resilience and adaptability to degradation.

REFERENCES