Physical Design Considerations of One-level RRAM-based Routing Multiplexers

Xifan Tang[1], Edouard Giacomin[2], Giovanni De Micheli[1]
and Pierre-Emmanuel Gaillardon[2]
[1]École Polytechnique Fédérale de Lausanne (EPFL), Vaud, Switzerland
[2] University of Utah, Salt Lake City, Utah, USA
Email: xifan.tang@epfl.ch

ABSTRACT

Resistive Random Access Memory (RRAM) technology opens the opportunity for granting both high-performance and low-power features to routing multiplexers. In this paper, we study the physical design considerations related to RRAM-based routing multiplexers and particularly the integration of 4T(ransistor)1R(RAM) programming structures within their routing tree. We first analyze the limitations in the physical design of a naive one-level 4T1R-based multiplexer, such as co-integration of low-voltage nominal power supply and high voltage programming supply, as well as the use of long metal wires across different isolating wells. To address the limitations, we improve the one-level 4T1R-based multiplexer by re-arranging the nominal and programming voltage domains, and also study the optimal location of RRAMs in terms of performance. The improved design can effectively reduce the length of long metal wires by 50%. Electrical simulations show that using a 7nm FinFET transistor technology, the improved 4T1R-based multiplexers improve delay by 69% as compared to the basic design. At nominal working voltage, considering an input size ranging from 2 to 32, the improved 4T1R-based multiplexers outperform the best CMOS multiplexers in area by 1.4×, delay by 2× and power by 2× respectively. The improved 4T1R-based multiplexers operating at near-Vt regime can improve *Power-Delay Product* by up to 5.8× when compare to the best CMOS multiplexers working at nominal voltage.

1. INTRODUCTION

Resistive Random Access Memory (RRAM) technology [1, 2, 3] has attracted intensive research interests in granting both high-performance and low-power features to routing multiplexers [4, 5]. Similar to pass-transistors or transmission gates in *on*/*off* state, RRAMs exhibiting *High Resistance State* (HRS)/*Low Resistance State* (LRS) can propagate/block signals. The benefits of RRAM-based multiplexers come from two aspects: (1) RRAMs can reduce the resistances and capacitances of the critical path, leading to high performance; (2) Once programmed, RRAMs are not affected by a reduction of the operating voltage, unlike pass-transistors or transmission gates whose conductance degrades with a reduction of V_{DD}. Therefore, RRAM-based multiplexers provide high-performance even when operating in the near-V_t regime [4, 5]. Previous works [4, 5, 6, 7, 8, 9] exploit RRAMs and 2T(ransistor)1R(RAM) programming structures to replace pass-transistors or transmission gates of CMOS multiplexers. Recently, 4T(ransistor)1R(RAM) programming structures [10] have been shown more efficient than 2T1R programming structures recently. The authors of [10] explain that both 2T1R and 4T1R programming structures have to employ a high programming voltage, different from nominal working voltage. This reveals a series of challenges at the physical design level, such as how to co-integration of low-voltage nominal power supply and high voltage programming supply, which have not been evaluated in previous works [4, 5, 6, 7, 8, 9, 10].

In this paper, we study the one-level 4T1R-based multiplexers by considering various physical design factors. We first investigate physical design implementation limitations of the naive design of a one-level 4T1R-based multiplexer and we propose an improved one-level 4T1R-based multiplexer, with an advanced physical design featuring: (1) a better granularity of the programming structures; (2) the protection of the datapath transistors from high programming voltage; (3) a 50% length reduction of the long metal wires across isolating wells. Electrical simulations show that, using a 7nm FinFET transistor technology, the modified 4T1R-based multiplexers improve delay by 69% as compared to the naive design. At nominal working voltage, considering an input size ranging from 2 to 32, the improved 4T1R-based multiplexers outperform the best CMOS multiplexers in area by 1.4×, delay by 2× and power by 2× respectively. Furthermore, the proposed 4T1R-based multiplexers operating at near-Vt regime can improve *Power-Delay Product* by up to 5.8× when compared to the best CMOS multiplexers working at nominal voltage.

The rest of this paper is organized as follows. Section 2 reviews on the background about RRAM technology and 4T1R-based programming structure. Section 3 introduces and analyzes a naive one-level 4T1R-based multiplexer at the physical design level. Section 4 proposes an improved one-level 4T1R-based multiplexer, overcoming difficulties in physical design. Section 5 presents the experimental results. Section 6 concludes this paper.

2. BACKGROUND AND MOTIVATION

In this part, we introduce the necessary background about RRAM technology, previous works on RRAM multiplexers and advancements in programming structures.

2.1 RRAM Technology

As one of the most promising emerging memory technology [11], *Resistive Random Access Memory* (RRAM) is envisaged to be integrated at low cost closely with conventional CMOS thanks to its *Back-End-of-the-Line* (BEoL) compatible fabrication process [3]. Indeed, RRAMs can be fabricated between

ISPD '17, March 19-22, 2017, Portland, OR, USA

DOI: http://dx.doi.org/10.1145/3036669.3036675

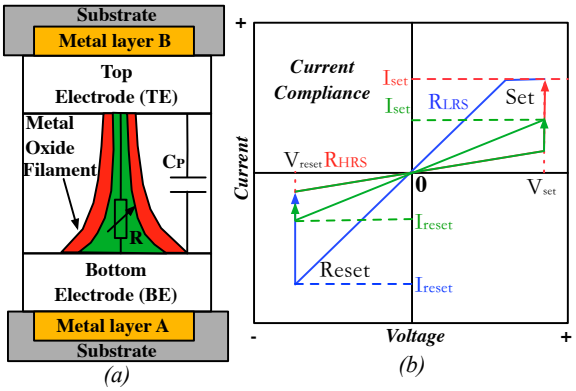

Figure 1: (a) RRAM structure and filamentary conduction; (b) I-V characteristics of set and reset processes.

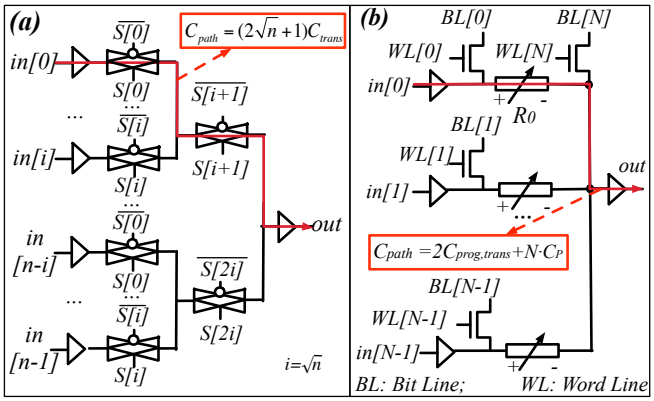

Figure 2: (a) Two-level CMOS multiplexer and (b) one-level 2T(ransistor)1R(RAM)-based multiplexer

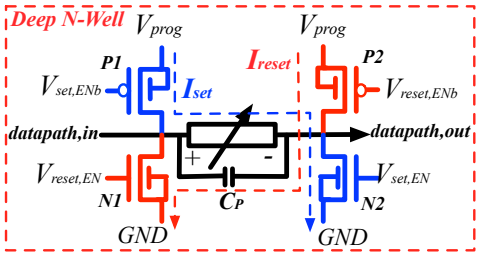

Figure 3: Schematic of a 4T(ransistor)1R(RAM)-based Programming Structure

the metal layers or even within the contact vias to the source or drain of a transistor, leading to a high co-integration density. The structure of a RRAM typically consists of three layers, where a transition metal oxide material stack is sandwiched between the top and bottom metal electrodes, as depicted in Fig. 1(a). Thanks to a filamentary switching mechanism, RRAMs can be switched between two stable resistance states: the *High Resistance State* (HRS) and the *Low Resistance State* (LRS). In addition to the resistive property, a RRAM also introduces a parasitic capacitance C_P. Depending on the employed materials, switching mechanisms of RRAMs are broadly classified to two categories: *Bipolar Resistive Switching* (BRS) and *Unipolar Resistive Switching* (URS). In this paper, we consider RRAM based on BRS only, which is a common choice in most literatures about RRAM-based circuits and systems [4, 5, 6, 7, 8, 9, 10].

Fig. 1(b) illustrates the I-V characteristics of a BRS RRAM. The switching between resistance states is triggered by applying a positive or negative programming voltage across the top and bottom electrodes. The minimum programming voltages required to trigger set and reset processes are defined as V_{set} and V_{reset}, respectively. The programming currents that are provided in set and reset processes are defined as I_{set} and I_{reset}, respectively. A current compliance on I_{set} is often enforced to avoid a permanent breakdown of the device, which is highlighted red in Fig. 1(b). Before being normally set/reset cycled, pristine RRAMs require a forming process to form their filament plug. Thanks to the filamentary conduction mechanism, the LRS resistance R_{LRS} can be dynamically adjusted by controlling the maximum I_{set}. For example, we show that a lower I_{set} leads to a smaller filament (highlighted green in Fig. 1(a)), resulting in a higher R_{LRS} (highlighted green in Fig. 1(b)) than the current compliance. Note that to reset a RRAM that is programmed with a I_{set} lower than current compliance, the required I_{reset} is also less than the maximum (see the green line in Fig. 1(b)). The tunable R_{LRS} is a unique feature of RRAM, which provides more flexibility in design space than other non-volatile memories, such as *Magnetic Random Access Memory* (MRAM) [12]. RRAMs can be scaled down effectively thanks to the filament mechanism. In advanced RRAM technology, an effective memory cell area can be as low as $4F^2$, where F is the feature size [13].

2.2 RRAM-based Multiplexer

RRAMs have attracted intensive research efforts on routing multiplexer designs in recent years [4, 5, 6, 7, 8, 9]. Major research opportunities lie in that RRAMs can be exploited to replace the pass-transistors or transmission gates in the multiplexers with different structures. When a RRAM is programmed to LRS, it can propagate signals as a pass-transistor/transmission gate in *on* state would do. In contrast, a RRAM in HRS can block signals as a pass-transistor / transmission gate in *off* state. Fig. 2 compares a two-level CMOS multiplexer [14] with a one-level N-input RRAM-based multiplexer [6, 7, 8]. The capacitance of the output node of RRAM-based multiplexer has a more pronounced non-linearity as compared to CMOS multiplexers because the parasitic capacitance of a RRAM C_P is much smaller than a transistor. With the reduction of parasitic capacitances and a smaller equivalent resistance than transistors, RRAMs can significantly improve the delay and power of multiplexers. Previous works [4, 5, 6, 7, 8, 9] typically employ 2T(ransistor)1R(RAM) programming structures, and neglect parasitics of a RRAM C_P in their evaluation. Indeed, [4, 5, 6, 7, 8, 9] treat RRAMs as an ideal capacitive load, which has been proved unrealistic in [10]. Fig. 3 illustrates the 4T(ransistor)1R(RAM)-based programming structure, where set and reset process of the RRAM are enabled by two pairs of *p*-type and *n*-type transistors, respectively. In order to set a RRAM into LRS, transistors **P1** and **N2** are turned *on* and transistors **P2** and **N1** are turned *off*, allowing a programming current I_{set}, highlighted blue in Fig. 3, to flow through the RRAM. In order to drive the set and reset currents, the programming voltage V_{prog} should be high enough and is potentially larger than the datapath signals, which is also true for 2T1R programming structure. Therefore, in physical design, a deep N-well (highlighted red in Fig. 3) is required to provide a different voltage domain for the programming structure. However, deep N-wells typically require large

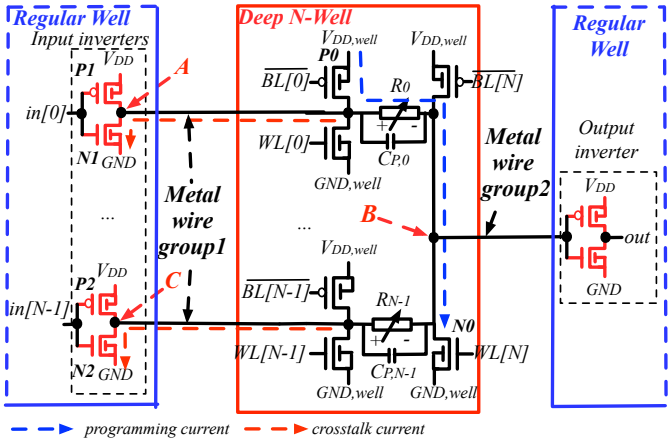

Figure 4: Circuit design and well arrangement of a naive $N:1$ one-level 4T1R-based multiplexer

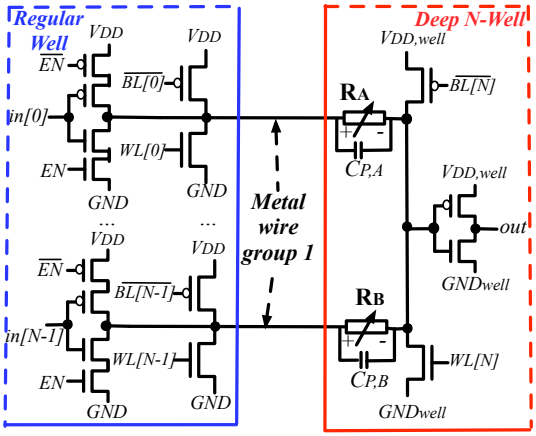

Figure 5: Circuit design and well arrangement of the improved one-level $N:1$ 4T1R-based multiplexer.

spacing between each other and also regular N-wells. This motivates us to take the parasitics into account and study the physical design aspects of integrating 4T1R programming structure into RRAM-based multiplexers, which has not been carefully studied yet to the best of our knowledge.

3. NAIVE 4T1R-BASED MULTIPLEXER

By adapting the circuit topology in Fig. 2, we illustrate in Fig. 4 a naive one-level $N:1$ multiplexer that can be programmed with 4T1R elements [10]. This naive one-level $N:1$ multiplexer consists of N pairs of 4T1R programming structures, which are controlled by $N+1$ Bit lines and $N+1$ Word lines. Note that all the RRAMs share a pair of programming transistors at the node B in Fig. 4, instead of using independent programming transistors. Sharing programming transistors can significantly reduce the parasitic capacitances at node B. All the RRAMs can be programmed in series. For instance, when a set process is required for RRAM R_0, control signals $\overline{BL[0]}$ and $WL[N]$ are enabled. Programming transistors **P0** and **N0** are turned *on* and drive a programming current (blue dash line in Fig. 4) flowing through RRAM R_0. Other programming transistors should be turned *off* during the programming period. However, such straightforward design in Fig. 4 encounters three limitations, as outlined next.

3.1 Limitation 1: Programming Currents Contribution from Datapath Transistors

Whether a RRAM can be programmed into a reasonable R_{LRS} highly depends on the amount of programming current that can be driven through the RRAM. In order to accurately control the programming current of a RRAM, only a pair of p-type and n-type transistors is turned *on* during programming. However, during programming, some datapath transistors in *on* state could inject or distribute the programming currents, leading to the achieved R_{LRS} to be out of the specification range. Take the example in Fig. 4, assume that RRAM R_0 is being programmed by enabling transistors **P0** and **N0**. Pull-down transistors of the input inverters, such as transistors **N1** and **N2**, could potentially be in *on* state, creating additional leakage paths, as highlighted by red dashed lines. This would disturb the V_{DS} of programming transistors and cause the programming current (blue dashed lines) to be smaller than expected, leading to a higher R_{LRS}. Note that not only pull-down transistors, but pull-up transistors of input inverters, such as **P1** and **P2**, can interfere with the programming current.

3.2 Limitation 2: Breakdown Threats of Datapath Transistors

To achieve a reasonable R_{LRS}, programming voltage $V_{DD,well}$ should be large enough to drive a high enough programming current. For instance, a programming voltage can be as high as $V_{DD,well} = 3.0V$ while the nominal voltage of the datapath transistors is only $VDD = 0.9V$ [10]. Such large gap between $V_{DD,well}$ and V_{DD} could cause the datapath transistors to breakdown during RRAMs' programming phases. Take the example in Fig. 4, the voltage of node A, V_A, can reach $V_{DD,well}$ while programming RRAM R_0, leading to the source-to-drain voltage of transistor **P1** being $V_{DD,well} - V_{DD}$. Assume that $V_{DD,well} = 3.0V$ and $V_{DD} = 0.9V$, both the gate-to-source voltage V_{GS} and source-to-drain voltage V_{DS} of transistor **P1** are $2.1V$, possibly leading transistor **P1** to breakdown. Note that not only transistor **P1** but also all the transistors belonging to the input and output inverters in Fig. 4 can be in a breakdown condition. While exposed to these conditions, even if datapath transistors do not break down, their reliability, i.e., lifetime, would significantly degrade.

3.3 Limitation 3: Long Interconnecting Wires between Wells

Since RRAMs require a programming voltage which is higher than the nominal one, a deep N-well isolation (highlighted red in Fig. 4) is required for the programming structures, resulting in three N-wells as shown in Fig. 4. In physical designs, a large spacing is required between a deep N-well and a regular N-well, which introduces long interconnecting wires. As illustrated in Fig. 4, two groups of long interconnecting wires have to be employed: one is between input inverters and programming structures while the other is between programming structures and output inverters. The long metal wires introduce parasitic resistances and capacitances to 4T1R-based multiplexers, potentially causing delay and power degradation.

4. IMPROVED 4T1R-BASED MULTIPLEXER

In this section, in order to address the limitations of the presented naive 4T1R-based multiplexer, we propose a robust one-level 4T1R-based multiplexer by employing tri-state

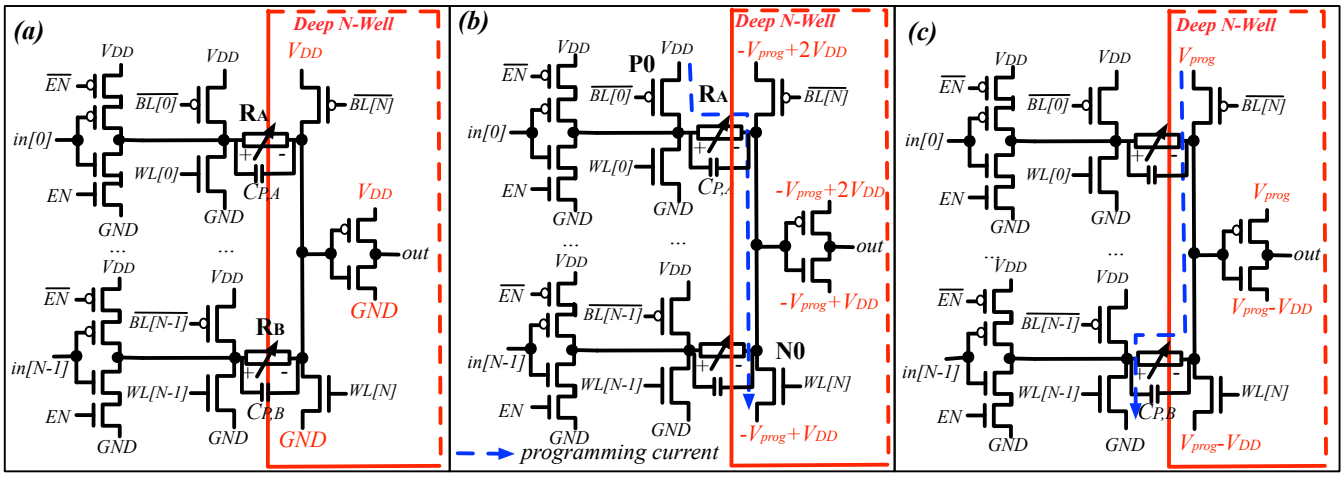

Figure 6: Improved one-level $N:1$ 4T1R-based multiplexer: (a) operating mode ($V_{DD,well} = V_{DD}$, $GND_{well} = GND$); (b) set process ($V_{DD,well} = -V_{prog} + 2V_{DD}$, $GND_{well} = -V_{prog} + V_{DD}$); (c) reset process ($V_{DD,well} = V_{prog}$, $GND_{well} = V_{prog} - V_{DD}$;

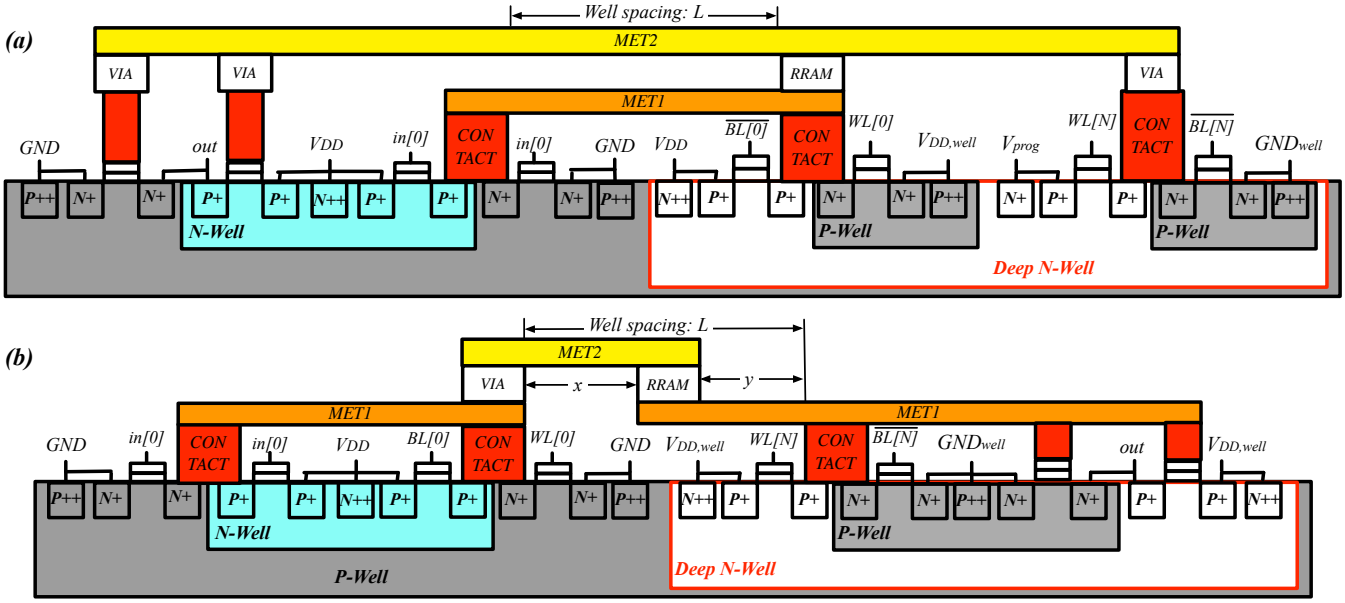

Figure 7: Cross-section of the layout of 4T1R multiplexers: (a) naive design; (b) improved design.

input and output inverters, and also rearranging the voltage domains and deep N-wells. We first introduce the improved design, and then discuss its advantages in physical design aspects.

4.1 Multiplexer Structure and Programming Strategy

Fig. 5 depicts the improved circuit designs, which are different from the naive circuit design (Fig. 4) in two aspects:
(a) the datapath input inverters are power-gated in order to eliminate the contribution of the datapath transistors in the programming phase;
(b) the two power domains (and the isolation deep N-well) are organized differently to Fig. 4.
Indeed, the input inverters and part of 4T1R programming structures are driven by a constant voltage domain V_{DD} and GND while the output inverter and the rest of 4T1R programming structures are driven by switchable voltage supplies $V_{DD,well}$ and GND_{well}. During operation, $V_{DD,well}$ and GND_{well} are configured to be equal to V_{DD} and GND respectively, as shown in Fig. 6(a). Note that the RRAM programming voltages are typically selected to be larger than V_{DD}, ensuring that RRAMs are not parasitically programmed during operation. When a set operation is triggered, input inverters are disabled and $V_{DD,well}$ and GND_{well} are switched to be $-V_{prog}+2V_{DD}$ and $-V_{prog}+V_{DD}$ respectively, as highlighted red in Fig. 6(b). During reset operations, input inverters are disabled and $V_{DD,well}$ and GND_{well} are switched to be V_{prog} and $V_{prog} - V_{DD}$ respectively, as highlighted red in Fig. 6(c). As such, the voltage difference across the RRAM during set or reset is $\pm V_{prog}$ and the working principle of the 4T1R programming structure can still be applied. Indeed, to enable the programming current path highlighted blue in Fig. 6(b), bit line $\overline{BL[0]}$ is configured to be GND and word line $WL[N]$ is configured to be $-V_{prog} + 2V_{DD}$ while other

programming transistors should be turned off by configuring $\overline{BL[i]} = VDD, WL[j] = GND, 1 \leq i \leq N-1, 0 \leq j \leq N-1$ and $\overline{BL[N]} = -V_{prog} + 2V_{DD}$.

4.2 Physical Design Advantages

The improved 4T1R-based multiplexer layout has two major advantages over the initial design in Fig. 4:

(1) the voltage drop across each datapath transistor can be limited to V_{DD}, allowing the use of logic transistors instead of I/O transistors (thicker oxides and higher breakdown voltage). Logic transistors occupy less area and introduce less capacitances than I/O transistors, potentially improving the footprint and delay of RRAM multiplexers. During the set and reset processes, the voltage drop of each transistor can be boosted from V_{DD} to $V_{DD,max}$, approaching the maximum reliable voltage without breakdown limitation. Boosted $V_{DD,max}$ leads to higher current density driven by transistors, further contributing to a lower R_{LRS} [10]. Note that the set and reset processes typically require short amount of time, i.e., typically $200ns$ for each RRAM [10]. Since programming does not occur many times (non-volatility), very low stress is applied on the transistors, further contributing to a robust operation.

(2) Only one connection between regular and deep N-Wells is necessary. As a result, only one group of long interconnecting wires is employed, potentially reducing the parasitics from metal wires. To be more illustrative, we depict in Fig. 7 and compare the cross-sections of the naive and improved designs at layout level. In each illustrative cross-section, we consider an input inverter $in0$, an output inverter, and a 4T1R programming structure. We assume that, in the naive design, input and output inverters can be accommodated with a regular N-well, so as to be more area efficient. However, even when the regular N-well is shared, long metal wires are still required because interconnections between datapath logics and programming structures have to include a large space between regular N-well and deep N-well. The length of metal wires $MET1$ and $MET2$ in Fig. 7(a) are dominated by the large well spacing L. Fig. 7(b) depicts the cross-section of the improved circuit in Fig. 5. Since RRAMs can be fabricated between metal lines, they can be located in any position between the two wells. Whatever location the RRAM is, there is only one long metal wire ($MET2$ and part of $MET1$) across two wells, while the other metal wires $MET1$ connect transistors inside the same well. Note that the length of interconnecting wires inside the same well is much smaller than those across two wells L. As a result, the length of metal wires in the naive design is dominated by $2 \cdot L$, while the improved design is dominated by L. Therefore, the improved design can reduce 50% the length of interconnecting wire than the naive design, contributing to smaller parasitic resistances and capacitances.

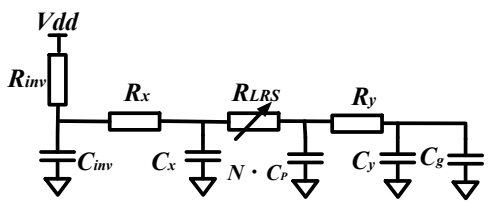

Figure 8: RC equivalent of a critical path of 4T1R-based multiplexer corresponding to the cross-section in Fig. 7(b).

4.3 Physical Position of RRAMs

As illustrated in Fig. 7(b), RRAMs are flexible in their location between the two wells. However, the choice of the location of RRAMs lead to different distribution of parasitics inside the 4T1R-based multiplexer, and further resulting in difference in performance. In this part, we study the impact of location of RRAMs on the performance, by using the Elmore Delay model [15]. We represent the distance between the RRAM and the regular N-well as $x \in [0, L]$, as shown in Fig. 7(b). We extract the critical path of the improved 4T1R-based multiplexer by considering the parasitics in Fig. 7(b) and depicts its equivalent RC model in Fig. 8. R_{inv} and C_{inv} represent the equivalent resistance and capacitance of an input inverter. (R_x, C_x) and (R_y, C_y) are the parasitic resistances and capacitances of the long metal wires, corresponding to (x, y) in Fig. 7(b) respectively. R_{LRS} denotes the resistance of a RRAM in LRS, and $N \cdot C_P$ is the total parasitic capacitances of RRAMs in a one-level 4T1R-based multiplexer. C_g is the gate capacitance of the output inverter. The Elmore Delay of the critical path is:

$$\begin{aligned} \tau &= R_{inv} \cdot C_{inv} + (R_{inv} + R_x)C_x \\ &+ (R_{inv} + R_x + R_{LRS}) \cdot N \cdot C_P \\ &+ (R_y + R_{inv} + R_x + R_{LRS})(C_y + C_g) \end{aligned} \tag{1}$$

Note that $R_x + R_y = x \cdot R_\square + y \cdot R_\square = L \cdot R_\square$ and $C_x + C_y = x \cdot C_\square + y \cdot C_\square = L \cdot C_\square$, where $R_\square$ and $C_\square$ are the square resistance and capacitance of a unit metal wire respectively. Equation 1 can be simplified:

$$\begin{aligned} \tau &= R_{inv} \cdot C_{inv} + L \cdot R_\square(C_g + L \cdot C_\square) \\ &+ (R_{inv} + R_{LRS})(\cdot N \cdot C_P + C_g + L \cdot C_\square \\ &+ (R_\square C_\square)x^2 + [R_\square(\cdot N \cdot C_P - L \cdot C_\square) - C_\square \cdot (R_{inv} + R_{LRS})]x \end{aligned} \tag{2}$$

The minimum delay τ_{min} is achieved when:

$$x_{opt} = \frac{L}{2} + \frac{R_{inv} + R_{LRS}}{2R_\square} - \frac{N \cdot C_P}{2C_\square} \tag{3}$$

Among parameters $L, N, R_{inv}, R_{LRS}, R_\square, C_P$ and $C_\square$, only N is the design parameter, while the others are all determined by a process technology. Equation 3 shows that x_{opt} decreases when N is increased. In other word, in large 4T1R-based multiplexer, RRAMs should be located close to the N-well.

4.4 Sharing deep N-Well between multiplexers

Deep N-wells can be efficiently shared between two cascaded 4T1R-based multiplexers, as illustrated in Fig. 9. The input inverters and part of programming structures of $MUX1$ in Fig. 9 can share a deep N-well with the output inverter and part of programming structures of $MUX0$. Note that the polarities of RRAMs of $MUX1$ are opposite to the RRAMs of $MUX0$, allowing simple programming strategies. As such, when set processes are required, $V_{DD,well}$ and GND_{well} are switched to $-V_{prog} + 2V_{DD}$ and $-V_{prog} + V_{DD}$ respectively; while during reset processes, $V_{DD,well}$ and GND_{well} are switched to V_{prog} and $V_{prog} - V_{DD}$ respectively; Otherwise, if all the RRAMs have had the same polarity, switching $V_{DD,well}$ and GND_{well} depends not only on the programming operation (either set or reset) but also on the location of multiplexers, requiring additional circuitry.

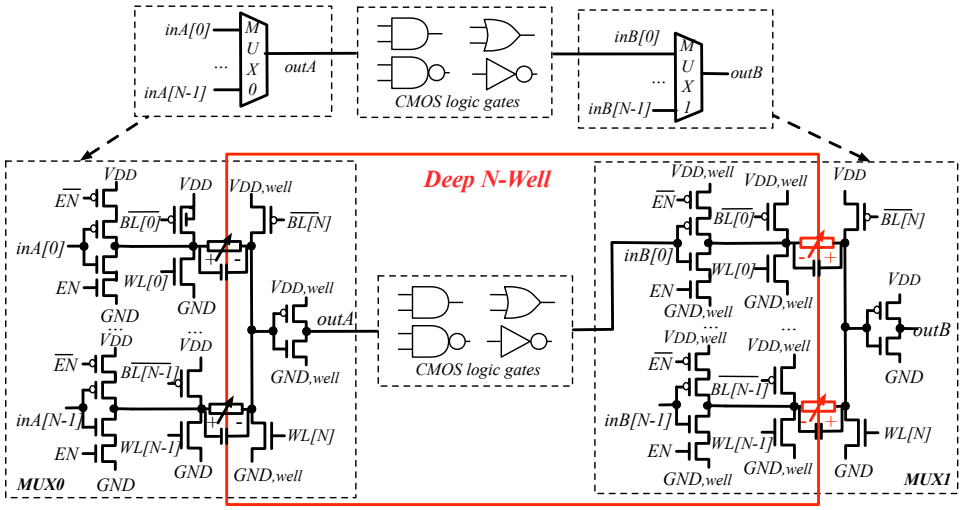

Figure 9: Cascading two improved one-level 4T1R-based multiplexers: share Deep N-Wells efficiently.

5. EXPERIMENTAL RESULTS

In this section, we first introduce our experimental methodology and then report area, delay and power results.

5.1 Experimental Methodology

In this paper, we consider a RRAM technology [5] with programming voltages $V_{set} = |V_{reset}| = 0.9V$ and a maximum current compliance of $I_{set} = |I_{reset}| = 500\mu A$. The lowest achievable on-resistance R_{LRS} of a RRAM is $1.6k\Omega$ while the off-resistance R_{HRS} is $23M\Omega$. The parasitic capacitance of a RRAM, C_P, is estimated to be $4.5aF$ by considering that the RRAMs are embedded in the *MET1* and *MET2* vias of our considered technology. The pulse width of a programming voltage in both set and reset processes is set to be $200ns$. The Stanford RRAM compact model [16] is used to model the considered RRAM technology. The ASAP 7nm FinFET design kit from ASU [17] is used in the circuit designs of datapath logics and 4T1R programming structures. Datapath circuits are built with standard logic transistors (regular-V_t), while the 4T1R programming structures employ I/O transistors for the naive design [10], and low-V_t transistors for the improved designs. The standard logic transistors have a nominal working voltage $V_{DD} = 0.7V$, and the I/O transistors can be overdriven to $1.8V$ while staying in their reliability limits. We compare area, delay and power of the naive and the improved 4T1R-based multiplexers to the CMOS multiplexers, by sweeping input size from 2 to 32. The baseline CMOS multiplexers are implemented with transmission gates. When input size $N \leq 12$, a one-level structure is considered, while when input size $N > 12$, a two-level structure is considered to guarantee the best performance. Input and output inverters, transmission gates are implemented with a pair of n-type and p-type FinFETs. Each of FinFET contains three fins. Area evaluations consider the layout area, while delay and power results are extracted from HSPICE [18] simulations.

5.2 Programming Transistor Sizing

As explained in [4], the sizing of programming transistors can significantly impact the delay of RRAM-based multiplexers. In this paper, we extend this study to the naive and improved 4T1R-based multiplexers in the specific context of FinFETs, by sweeping the number of fins from 1 to 3 in each FinFET. We selected a maximum of three fins, because, in the considered design kit, three fins allow the 4T1R structure to match the standard cell height, simplifying the layout considerations. Fig. 10 shows both delay and power difference of the improved 4T1R-based multiplexers ($x = L$) under various V_{DD}. A proper number of fins indeed can reduce the delay of 4T1R-based multiplexers by 14%-21% and also the power by 25% respectively. In terms of delay, the best number of fins is three for all the cases, which can be explained as follows: Three fins lead to lower achievable RRAM resistances than one or two fins, which, in turn, performs better in driving the large parasitic capacitances of long metal wires. Similar conclusions can be found for other 4T1R-based multiplexers in this paper. In the rest of this paper, we consider three fins for each FinFET in 4T1R-based multiplexers to achieve best delay metric.

5.3 Optimal RRAM Location

As shown in Equation 3, the location of RRAMs can influence the delay of 4T1R-based multiplexers. From the consider design kit, we extract process parameters $L = 0.8\mu m$, $R_{inv} = 4.5k\Omega$, $R_{\square} = 67.5\Omega/\mu m$ and $C_{\square} = 67.5aF/\mu m$. According to Equation 3, the best location of the RRAMs is $x_{opt} = L$ unless $N > 2080$. Therefore, in this part, we study only two locations for RRAMs : $x = 0$ and $x = L$. Fig. 11 compares the delay of naive and improved 4T1R-based multiplexers with different locations of RRAMs $x = 0$ and $x = L$. The improved design significantly reduces the delay by 35%-69% as compared to the naive design. Such large delay reduction comes from two aspects: (a) The input tri-state inverters guarantee a high programming current through RRAMs, resulting in a low R_{LRS}; (b) As the length of long metal wires is reduced by 50%, the parasitic resistances and capacitances of the improved design are smaller. Note that, in the naive design, the input inverters cause serious interference on the programming current when input size increases. Consequently, when input size is larger than 16, RRAM-based multiplexers cannot be programmed successfully. The best location of RRAMs is $x = L$, leading to a 5% delay improvement over $x = L$, which satisfies Equation 3. In the rest of this paper, we consider the improved design with $x = L$ in the comparison with CMOS multiplexers.

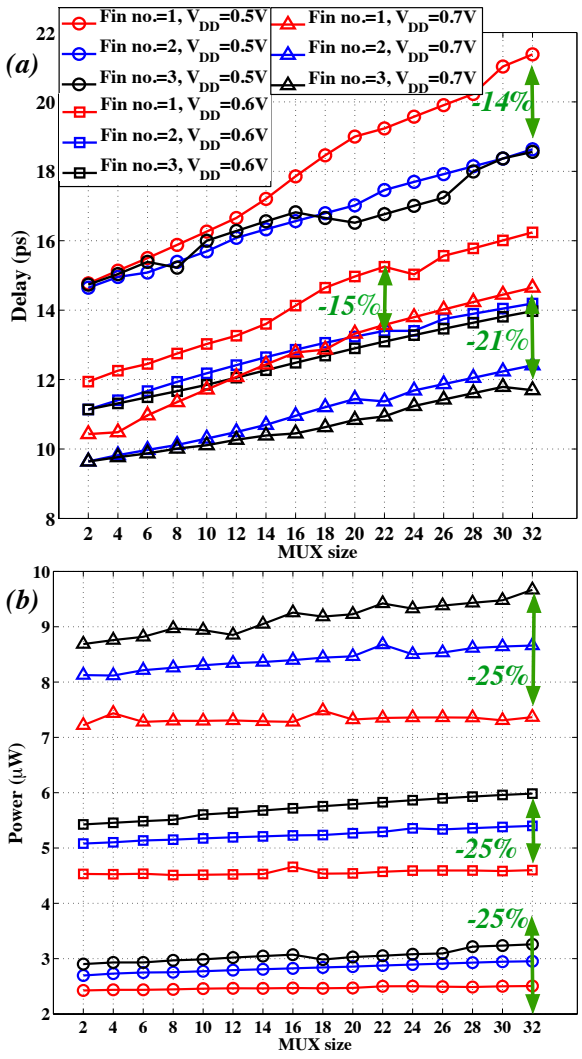

Figure 10: Best number of fins in each FinFET of improved 4T1R-based multiplexer ($x = L$) under different V_{DD} in terms of (a) Delay and (b) Power.

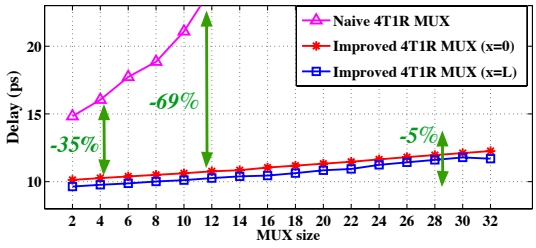

Figure 11: Delay comparison between naive and improved 4T1R-based multiplexers ($x = 0$ and $x = L$).

5.4 Area Results

In order to properly study the area of the 4T1R-based considering routing, well organization etc., and compare with the CMOS counterpart, we realized the layout of a 16-input two level CMOS multiplexer and a 4T1R-based one-level multiplexer. The CMOS multiplexer is built with two levels and must use SRAMs to store the configuration bits. As explained in the previous subsection, we can efficiently share the wells between different 4T1R-based multiplexers

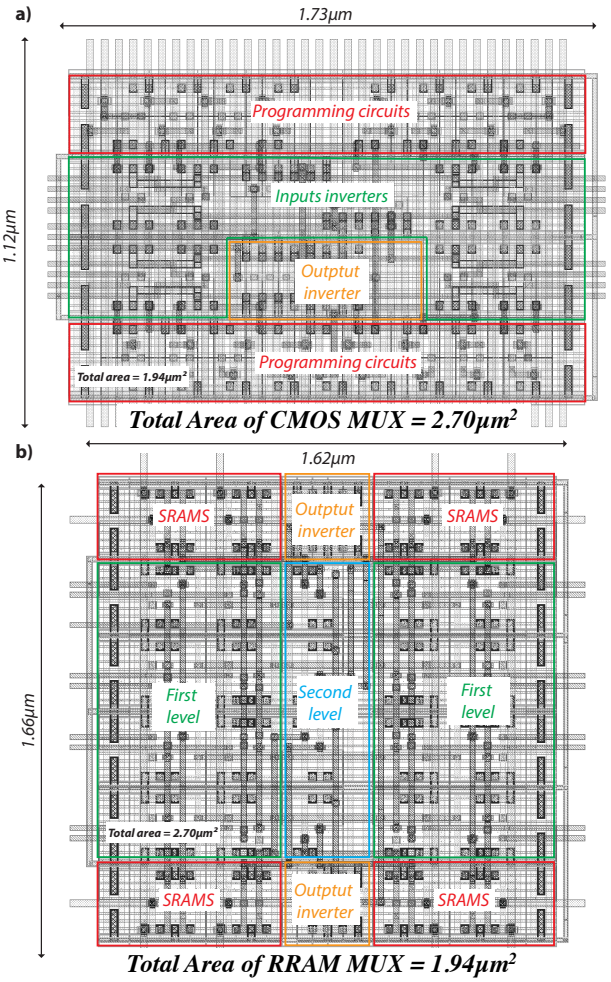

Figure 12: Layout of 16-input a 4T1R-based one-level multiplexer

leading to less area overhead. Therefore, the layout of the 16-input 4T1R-based one-level multiplexer only consists of the programming structures and input inverters of a first multiplexer and the output inverter of another multiplexer in a regular well. The output inverter and the associated programming structures will be located in a *deep N-well*, as well as the input inverters and associated programming structures of the other multiplexer. The space required by the topological design rule between the regular well and the *deep N-well* can be efficiently used to accommodate standard n-type transistors and route the multiplexers input signals. Fig. 12 depicts the layout organisation of the 16-input 4T1R-based one-level multiplexer. The input inverters are placed together in two stages so we can access to the multiplexer inputs from both sides through the horizontal lines (8 inputs in each side). The programming structures are placed above and under the input inverters and each associated $\overline{BL[N]}$ and WL[N] are accessible through the vertical metal lines. As a result, the 4T1R-based multiplexer area ($1.94um^2$) is 1.4× more efficient than its CMOS counterpart ($2.70um^2$).

5.5 Delay and Power Results

Fig. 13 compares the delay and power of the improved 4T1R-based multiplexers ($x = L$) and CMOS multiplexers under different V_{DD} respectively. Thanks to the significant

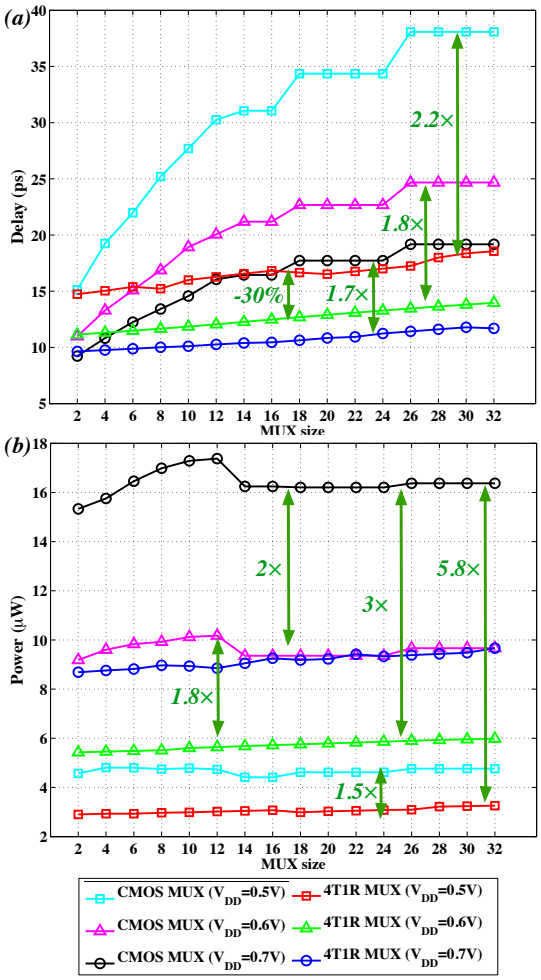

Figure 13: (Comparison between the improved 4T1R-based multiplexers ($x = L$) and CMOS multiplexer under different V_{DD}: (a) delay; (b) power.

reduction on the capacitances in critical paths, at nominal voltage, the 4T1R-based multiplexers improves delay significantly by 1.7× as compared to their CMOS counterparts. Since the resistances of RRAMs are independent from working voltages, at near-V_t regime, the delay improvements of the 4T1R-based multiplexers increase to 1.7× and 2.2× respectively. Note that the 4T1R-based multiplexers operating at $V_{DD} = 0.6V$ is still 30% more delay efficient than the CMOS multiplexers at $V_{DD} = 0.7V$. The reduction on the capacitances in critical paths also contributes to a significant improvement in power consumption. Compared to CMOS multiplexer, the 4T1R-based multiplexers improve the power by $1.5 - 2\times$ under various V_{DD}. More importantly, such power improvements are achieved without delay loss. Take the example of the 4T1R-based multiplexers operating at $V_{DD} = 0.5V$, their delays are similar to the CMOS multiplexers at $V_{DD} = 0.7V$, while the power consumption is reduced by 5.8×.

6. CONCLUSIONS

In this paper, we first investigate the naive design of a one-level 4T1R-based multiplexer and addresses its limitations from a physical design standpoint. We propose an improved one-level 4T1R-based multiplexer with advanced physical design considerations: (1) a better granularity of the programming structures; (2) the protection of the datapath transistors from high programming voltage; (3) a 50% length reduction of the long metal wires across isolating wells. Electrical simulations show that, using a 7nm FinFET transistor technology, the modified 4T1R-based multiplexers improve delay by 69% as compared to the naive design. At nominal working voltage, considering an input size ranging from 2 to 32, the improved 4T1R-based multiplexers outperform the best CMOS multiplexers in area by 1.4×, delay by 2× and power by 2× respectively. Furthermore, the proposed 4T1R-based multiplexers operating at near-Vt regime can improve *Power-Delay Product* by up to 5.8× when compared to the best CMOS multiplexers working at nominal voltage.

7. ACKNOWLEDGMENTS

This work was supported by the Swiss National Science Foundation under the project number 200021-146600.

8. REFERENCES

[1] R. Waser *et al.*, *Nanoionics-based Resistive Switching Memories*, Nature Materials, Vol. 6, 2007, pp. 833-840.

[2] H. Akinaga *et al.*, *Resistive Random Access Memory (ReRAM) Based on Metal Oxides*, Proceedings of the IEEE, Vol. 98, No. 12, 2010, pp. 2237 - 2251.

[3] H.-S. P. Wong *et al.*, *Metal-Oxide RRAM*, Proceedings of the IEEE, Vol. 100, No. 6, 2012, pp. 1951-1970.

[4] X. Tang *et al.*, *A High-performance Low-power Near-Vt RRAM-based FPGA*, IEEE ICFPT, 2014, pp. 207-215.

[5] X. Tang *et al*, *Accurate Power Analysis for Near-V_t RRAM-based FPGA*, IEEE FPL, 2015, pp. 174-177.

[6] S. Tanachutiwat *et al.*, *FPGA Based on Integration of CMOS and RRAM*, IEEE TVLSI, Vol. 19, No. 11, 2010, pp. 2023-2032.

[7] P.-E. Gaillardon *et al.*, *Emerging Memory Technologies for Reconfigurable Routing in FPGA Architecture*, IEEE ICECS, 2010, pp. 62 - 65.

[8] J. Cong and B. Xiao, *FPGA-RPI: A Novel FPGA Architecture With RRAM-Based Programmable Interconnects*, IEEE TVLSI, Vol. 22, No. 4, 2014, pp. 864-877.

[9] P.-E. Gaillardon *et al.*, *GMS: Generic Memristive Structure for Non-Volatile FPGAs*, IEEE/IFIP VLSI-SoC, 2012, pp. 94-98.

[10] X. Tang *et al.*, *A Study on the Programming Structures for RRAM-Based FPGA Architectures*, IEEE Transaction on Circuits And Systems I (TCAS-I): Regular Papers, Vol. 63, No. 4, pp. 503-516.

[11] G.W. Burr *et al.*, *Overview of Candidate Device Technologies for Storage-Class-Memory*, IBM J. R&D, Vol. 52, No. 4/5, July/Sept. 2008.

[12] J. Zhu, *Magnetoresistive Random Access Memory: The Path to Competitiveness and Scalability*, Proceedings of the IEEE, Vol. 96, No. 11, pp. 1786 - 1798, 2008.

[13] Y. S. Chen *et al.*, *Highly Scalable Hafnium Oxide Memory with Improvements of Resistive Distribution and Read Disturb Immunity*, IEEE IEDM, 2009, pp.1-4.

[14] E. Lee *et al.*, *Interconnect Driver Design for Long Wires in Field-Programmable Gate Arrays*, Journal of Signal Processing Systems, Springer, Vol. 51, No. 1, April 2008.

[15] W.C. Elmore, *The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers*, Journal of Applied Physics, Vol. 19, No. 1, 1948, pp. 55-63.

[16] J. Jiang *et al.*, *Verilog-A Compact Model for Oxide-based Resistive Random Access Memory*, IEEE SISPAD, 2014, pp. 41-44.

[17] L.T. Clark *et al.*, *ASAP7: A 7-nm FinFET Predictive Process Design Kit*, Microelectronics Journal, vol. 53, pp. 105-115, July 2016.

[18] Synopsys, *HSPICE User Guide: Simulation and Analysis*, Version I-2013.12, December 2013.