Circuit Designs of High-Performance and Low-Power RRAM-Based Multiplexers Based on 4T(ransistor)1R(RAM) Programming Structure

Xifan Tang, *Student Member, IEEE*, Edouard Giacomin, Giovanni De Micheli, *Fellow, IEEE*, and Pierre-Emmanuel Gaillardon, *Senior Member, IEEE*

***Abstract*— Routing multiplexers based on pass-transistors or transmission gates are an essential components in many digital integrated circuits. However, whatever structure is employed, CMOS multiplexers have two major limitations: 1) their delay is linearly related to the input size; 2) their performance degrades seriously when operated in near-V_t regime. *Resistive Random Access Memory* (RRAM) technology brings opportunities of overcoming these limitations by exploiting the properties of RRAMs and associated programming structures. In this paper, we propose new one-level, two-level and tree-like multiplexers circuit designs using 4T(ransistors)1R(RAM) elements and we compare them to naive one-level multiplexers. We consider the main physical design aspects associated with 4T1R-based multiplexers, such as the layout implications using a 7 nm FinFET technology, and the co-integration of low-voltage nominal power supply and high-voltage programming supply. Electrical simulations show that using a 7 nm FinFET transistor technology, the proposed 4T1R-based multiplexers reduce delay by 2× and energy by 2.8× over naive 4T1R and 2T1R counterparts. At nominal working voltage, considering an input size ranging from 2 to 50, the proposed 4T1R-based multiplexers reduces Area-Delay and Power-Delay products by 2.6× and 3.8× respectively, as compared to best CMOS multiplexers. In the near-V_t regime, the proposed 4T1R-based multiplexer demonstrates 2× larger delay efficiency over the best CMOS multiplexer. The proposed 4T1R-based multiplexers operating at near-V_t regime can still achieve up to 22% delay improvement when compared to best CMOS multiplexers working at nominal voltage.**

***Index Terms*— Circuit design, high-performance, low-power, multiplexer, resistive memory.**

I. Introduction

ROUTING multiplexers are essential components in *Application Specific Integrated Circuits* (ASICs) and *Field Programmable Gate Arrays* (FPGAs). In CMOS technology, multiplexers are typically implemented with pass-transistors or transmission gates [1]. Depending on the structural topology, multiplexers can be categorized into three types: one-level, two-level and tree-like. Two-level multiplexers are most widely used because of their best area-delay product [2]. However, no matter which topology is used, the core structure always relies on pass-transistors or transmission gates to propagate/block signals. As a result, the performance of CMOS routing multiplexers is sensitive to the operating voltage. This is at the origin of a strong limitation of CMOS routing multiplexer: their delay degrades seriously when operating voltage decreases.

Resistive Random Access Memories (RRAMs) [3]–[5] have been intensively exploited to replace the pass-transistors or transmission gates in routing multiplexers [7]–[12]. When exhibiting High Resistance State (HRS)/Low Resistance State (LRS), RRAMs can propagate/block signals similar to pass-transistors or transmission gates in on/off state. Previous works [7]–[12] commonly employ 2T(ransistor)1R(RAM) programming structures in order to program RRAMs between HRS and LRS. Two major advantages of RRAM-based multiplexers have been predicted from previous works: (1) Large delay reductions can be achieved because equivalent resistances of RRAMs in LRS can be smaller than pass-transistors or transmission gates [7]–[12]; (2) Both power reduction and high level of performance can be achieved in the near-V_t regime because the equivalent resistances of RRAMs are independent from the operating voltage as opposed to pass-transistors or transmission gates whose conductance degrades with a reduction of V_{DD} [11] [12]. However, 2T1R programming structures have recently been proved much less efficient than 4T(ransistor)1R(RAM) programming structure [6]. Whether the predicted advantages can be strengthened by 4T1R programming structure has not been carefully studied yet. Additionally, previous works [7]–[12] did not evaluate how to co-integrate datapath and programming transistors at the physical design level. This paper intends to fill this gap.

Compared to our previous work that focused on RRAM programming structures [6], this paper studies how to efficiently co-integrate 4T(ransistor)1R(RAM) programming structure into routing multiplexers. The contributions of this paper are: (1) We analyze the limitations of naive design of

Manuscript received October 24, 2016; accepted December 7, 2016. Date of publication December 23, 2016; date of current version April 20, 2017. This work was supported by the Swiss National Science Foundation under the project number 200021-146600. This paper was recommended by Associate Editor Y. Pu.

X. Tang and G. De Micheli are with the Integrated Systems Laboratory, School of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne (EPFL), Vaud, Switzerland (e-mail: xifan.tang@epfl.ch).

E. Giacomin and P.-E. Gaillardon are with the Laboratory for NanoIntegrated Systems, Electrical and Computer Engineering department, University of Utah, Salt Lake City, USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2016.2638542

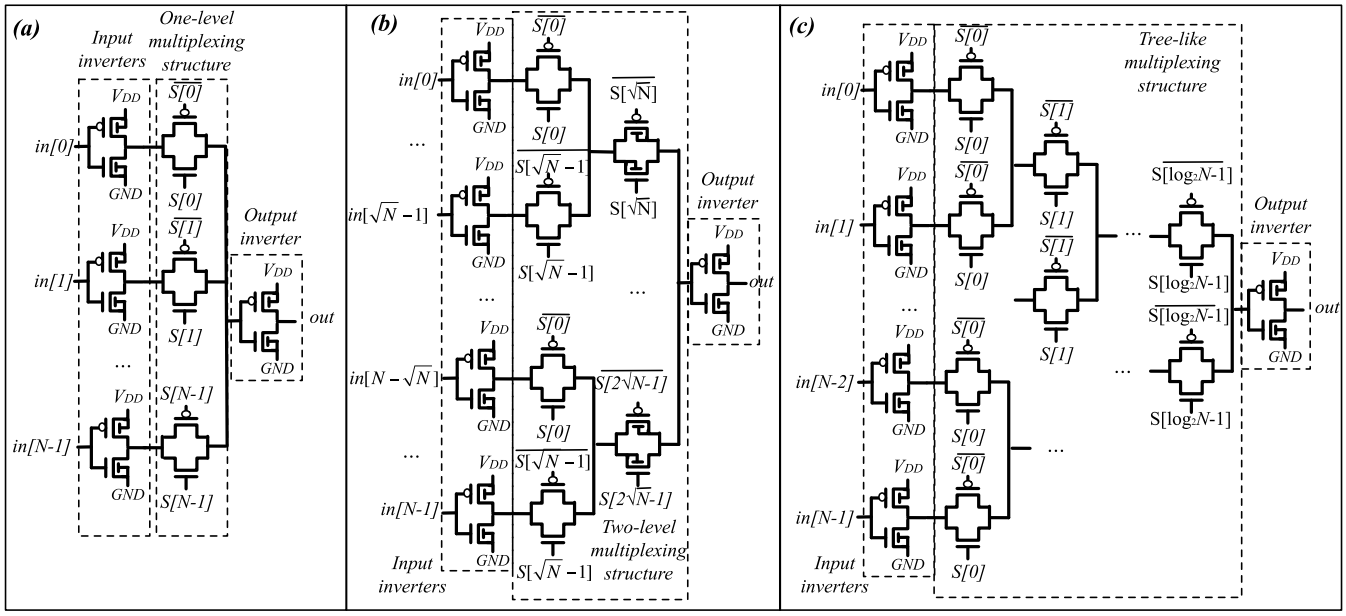

Fig. 1. N-input CMOS multiplexer structures: (a) one-level; (b) two-level; (c) tree-like.

one-level 4T1R multiplexer, which lead to RRAM programming failures as well as transistor breakdown; (2) To address the limitations, we propose novel one-level, two-level and tree-like multiplexer circuit designs using 4T(ransistor)1R(RAM) elements, and also consider the main physical design implications, such as the parasitic capacitance of RRAMs and programming transistor sizing technique using the recently released ASU 7 nm FinFET PDK [13]; (3) We investigate the impact of RRAM process variations on the performance of the improved 4T1R-based multiplexers. Electrical simulations show that naive 2T(ransistor)1R(RAM) and 4T(ransistor)1R(RAM)-based multiplexers cannot be programmed when input size is larger than 34. The proposed 4T1R multiplexers guarantee successful RRAM programming and improves delay by 2× and energy by 2.8× over naive 4T1R and 2T1R counterparts. At a nominal operating voltage and by considering input size ranging from 2 to 50, 4T1R-based routing multiplexers reduces Area-Delay and Power-Delay products by 2.6× and 3.8× respectively, as compared to the best CMOS FinFET multiplexers. In the near-V_t regime, 4T1R-based multiplexers demonstrate 2× delay efficiency over the best CMOS FinFET multiplexers. More interestingly, 4T1R-based multiplexers, when operated in near-V_t regime, can achieve up to 22% delay improvement compared to the best CMOS FinFET multiplexers working at nominal voltage.

The rest of this paper is organized as follows. Section II gives some generalities about RRAM technology and reviews RRAM-based programming structures and multiplexer designs. Section III shows basic one-level, two-level and tree-like 4T1R-based multiplexer designs and discusses their limitations. Section IV proposes improved 4T1R-based multiplexers. Section V presents some experimental results and Section VI analyzes the impact of process variations. Section VII concludes this paper.

II. Background and Motivations

In this section, we first present some generalities about multiplexer circuit design, and we review RRAM technology and the recent progress made on RRAM-based programming structures.

A. Multiplexer Designs

As a common component in digital circuits, CMOS routing multiplexers are typically implemented by pass-transistors or transmission gates, whose gate signals are controlled by control lines. CMOS multiplexers can be built with different structures, in order to trade off area, delay and power. Fig. 1 shows three most popular multiplexer structures: one-level, two-level and tree-like, implemented by transmission gates. Details of multiplexer designs implemented with pass-transistors can be found in [2]. In this paper, we only consider transmission gates in CMOS multiplexers because they perform better in area-delay-power product [17]. Table I shows an analytical comparison on area, delay and power among the different multiplexers in Fig. 1. One-level multiplexers lead to the smallest area and perform best in delay and power when input size N is small, i.e., less than $N \leq 8$. However, when input size N is large, one-level multiplexers require a large number of control lines and their delay and energy numbers, which are linear to N, increase significantly. Two-level multiplexers are the best choices when input size N is large. The number of control lines, delay and energy of two-level multiplexers are linear to the square root of N, leading to significant delay and energy reduction as compared to one-level multiplexers.

In addition, the area of two-level multiplexers is only slightly larger than their one-level counterparts, even when N is large. Therefore, in terms of area-delay-power product, two-level multiplexers lead to the best results when N is large (i.e., $N > 8$). Tree-like multiplexers are only preferred when

TABLE I
ANALYTICAL COMPARISON ON AREA, DELAY AND ENERGY OF N-INPUT CMOS MULTIPLEXERS

Multiplexer	Area[1]	Delay[2]	Energy[3]
One-level	$N \cdot Area_{trans}$	$R_{trans} \cdot N \cdot C_{trans}$	$0.5 \cdot \alpha \cdot N \cdot C_{trans} \cdot V_{DD}^2$
Two-level	$(N + [\sqrt{N}]) \cdot Area_{trans}$	$R_{trans} \cdot (3[\sqrt{N}] + 1) \cdot C_{trans}$	$0.5 \cdot \alpha \cdot (2[\sqrt{N}] + 1) \cdot C_{trans} V_{DD}^2$
Tree-like	$(2N - 2) \cdot Area_{trans}$	$R_{trans} \cdot \frac{1}{2}(3[log_2 N]^2 - [log_2 N]) \cdot C_{trans}$	$0.5 \cdot \alpha \cdot \frac{1}{2}(3[log_2 N]^2 - [log_2 N]) \cdot C_{trans} \cdot V_{DD}^2$

[1] Area of input and output inverters are not included here. [2] Elmore delay model [22] is considered here.
[3] Only the switching energy of multiplexer structures is considered here. α is the switching activity.
* $Area_{trans}$, R_{trans} and C_{trans} are the area, equivalent resistance and source/drain capacitances of a transmission gate.

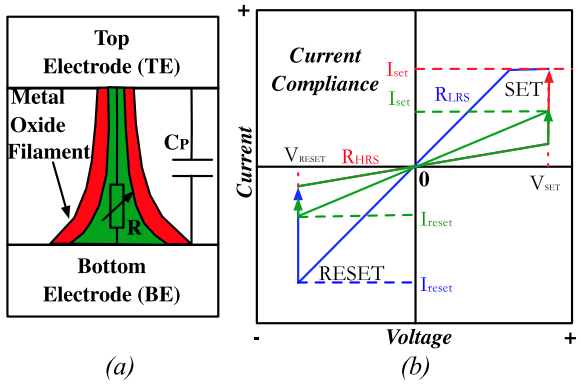

Fig. 2. (a) RRAM structure and filamentary conduction; (b) I-V characteristics of set and reset processes.

there is a tight constraint on the number of control lines. Due to their large number of stages, tree-like multiplexers perform worse in area, delay and power than others. Note that, in this simple analytical study, we focus only on the multiplexer core and do not consider the peripheral circuitries. Table I reveals a bottleneck of CMOS multiplexers: no matter which topology is used, the core structure always relies on pass-transistors or transmission gates to propagate/block signals. Consequently, the performance of CMOS routing multiplexers degrades seriously when operating voltage decreases.

B. RRAM Technology

As one of the most promising emerging memory technology [14], *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 [5]. Indeed, RRAMs can be fabricated between 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. 2(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 [7]–[12].

Fig. 2(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 programming voltage should be large enough to create a strong electric field inside the metal oxide in order to generate or dilute the filament. In addition to a programming voltage, the size of filament is also strongly impacted by the programming current. Therefore, a proper combination of programming voltage and current contributes to either a switching from HRS to LRS, called **set** process, or a switching from LRS to HRS, called **reset** process. 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. 2(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 I_{set} to be fed. For example, we show that a lower I_{set} conducts a smaller filament (highlighted green in Fig. 2(a)), resulting in a higher R_{LRS} (highlighted green in Fig. 2(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. 2(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) [15].

In addition, set and reset processes require a minimum pulse width to stabilize the filamentary conduction, which is defined as the writing time. The maximum number of writing operations that RRAMs can afford is expressed by the endurance. The maximum time period when RRAMs can maintain the resistance state without degradation is expressed by the data retention. RRAMs can be scaled down effectively thanks to filament mechanism. In advanced RRAM technology, a effective memory cell area can be as low as $4F^2$, where F is the feature size [16].

C. Previous Works on RRAM-Based Multiplexer

RRAMs have attracted intensive research efforts on routing multiplexer designs in recent years [7]–[12]. Earlier works [18]–[21] used RRAM-based non-volatile memory

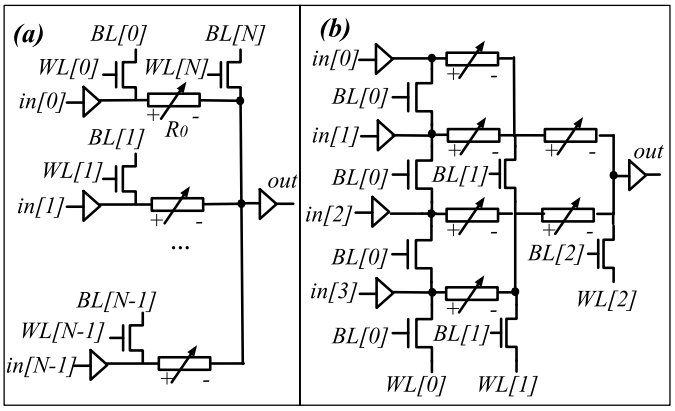

Fig. 3. Early designs of RRAM-based multiplexers: (a) A N-input one-level structure [9]; (b) An illustrative example of two-level and tree-like 4:1 structure [10].

structures to replace the configuration memories in the routing structures. These modifications grant non-volatility to the FPGA and enable *instant-on normally-off* operations. However, the multiplexer structures in [18]–[21] were still based on CMOS multiplexers, leading to no improvements on performance. More 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. Since the on-resistance of a RRAM, i.e., R_{LRS}, can be made smaller than the equivalent resistance of a pass-transistor/transmission gate, RRAMs can significantly reduce the delay of multiplexers.

Since a low R_{LRS} is a source of the performance improvement brought by RRAM-based multiplexers, how to program RRAMs from HRS to LRS efficiently becomes one of the most critical problems. The performance of RRAM-based multiplexers is determined by not only a low R_{LRS} but also the parasitic capacitances of programming transistors [11]. Previous works [7]–[12] typically employ 2T(ransistor)1R(RAM) programming structures. Fig. 3(a) shows a one-level N-input RRAM-based multiplexer [7]–[9] and Fig. 3(b) presents an illustrative example of a two-level/tree-like 2T1R-based multiplexer [10], whose input size is 4. RRAM-based multiplexers in Fig. 3 depend on n-type transistors to provide high programming current, in order to achieve a low R_{LRS}. For instance, when $WL[0] = WL[N] =' 1'$, $BL[0] =' 1'$ and $BL[N] =' 0'$, RRAM R_0 is programmed to LRS. However, previous works [7]–[11] treat RRAMs as capacitive loads rather than resistive loads, which has been proved to be unrealistic in [6]. Recent work [6] proposes a 4T(ransistor)1R(RAM) programming structure, which improve by 1.4× the programming current density compared to 2T1R structures, leading to a lower R_{LRS}. As shown in Fig. 4, a 4T1R programming structure employs two pairs of p-type and n-type transistors to set and reset a RRAM. For example, a set process is enabled when transistors **P1** and **N2** are turned on, while a reset process is enabled when transistors **P2** and **N1** are turned on. Note that both 2T1R and 4T1R programming structures

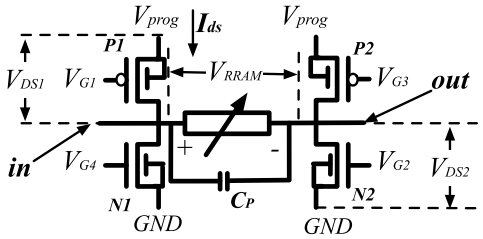

Fig. 4. Schematic of a 4T1R programming structure.

have to use a programming voltage V_{prog} which is larger than the nominal V_{DD} used by datapath transistors, in order to drive enough programming current for the RRAMs [6]. To the best of our knowledge, how to integrate 4T1R programming structure efficiently into RRAM-based multiplexers has not been carefully studied yet. Integration of 4T1R programming structure faces a few physical design challenges, such as a) how to avoid crosstalk currents between datapath transistors and programming structures; and b) how to protect the datapath transistors from reliability issues under high V_{prog}. This paper intends to provide analysis and solutions to these challenges.

III. Basic 4T1R-Based Multiplexer

In this section, we propose a naive multiplexer structure using 4T1R elements and discuss a few limitations of the structure.

A. Multiplexer Structure and Programming Strategy

By following the general topology shown in Fig. 3, a basic one-level N : 1 multiplexer can be developed with 4T1R elements. The resulting one-level N-input RRAM-based multiplexer is illustrated in Fig. 5 and consists of N pairs of 4T1R programming structures, which are controlled by $N + 1$ Bit lines and $N + 1$ Word lines. Since RRAMs require a programming voltage which is higher than the nominal one, a Deep N-well isolation (highlighted red in Fig. 5) is required for the programming structures, resulting in two power domains. Instead of providing each RRAM with four independent programming transistors, all the RRAMs can share a pair of programming transistors (controlled by $\overline{BL[N]}$ and $WL[N]$ respectively) at node B. As a result, each RRAM can be individually programmed with either positive or negative voltage polarity. For example, we can first set RRAM R_0 by enabling $\overline{BL[0]}$ and $WL[N]$. Note that the rest of bit lines and word lines should be off, to ensure the programming current (highlighted blue in Fig. 5) flows only through transistor **P0**, RRAM R_0 and transistor **N0**. Then we can turn off $\overline{BL[0]}$ and $WL[N]$, and turn on $\overline{BL[N]}$ and $WL[N-1]$ to reset RRAM R_{N-1}. Sharing programming transistors in the multiplexer structure is flexible enough from a reconfiguration standpoint. In practice, in a N-input multiplexer, only one RRAM is in LRS while the others are in HRS. Each time a multiplexer is reconfigured, one RRAM is reset from LRS to HRS and another is set from HRS to LRS, implying two steps (one reset process and one set process). Note that set and reset process have to be executed sequentially because

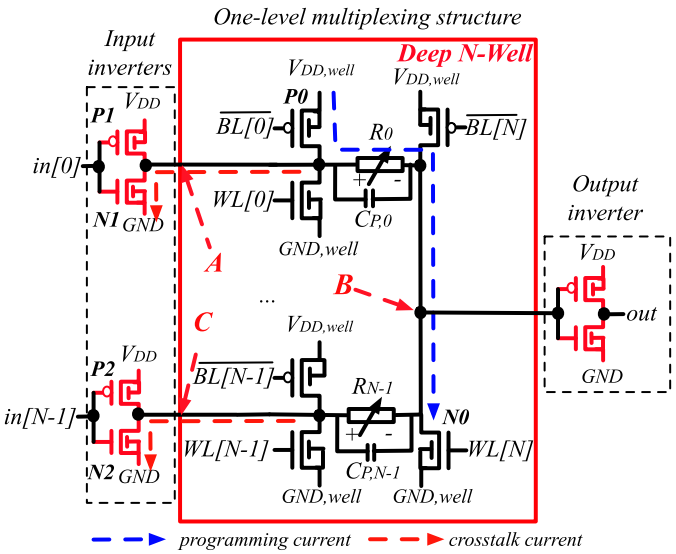

Fig. 5. Schematic of a basic one-level N-input 4T1R-based multiplexer.

set and reset processes require different programming voltages at node B. Whether the multiplexer has shared programming transistors or employs independent programming transistors for each RRAMs, we always need two steps (one reset process and one set process) in each reconfiguration. More importantly, sharing programming transistors can significantly reduce the parasitic capacitances at node B in Fig. 5, leading to large delay and power improvements. Independent programming transistors cause that the total parasitic capacitance at node B includes N pairs of programming transistors. In contrast, sharing programming transistors lead to that the total parasitic capacitance at node B includes only a pair of programming transistors.

B. Limitations

Such straightforward design suffers from two possible limitations due to the co-integration of both datapath and programming channels.

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 specifications. Take the example in Fig. 5, assume that RRAM R_0 is being programmed by enabling transistors **P0** and **N0**. Datapath transistors **N1** and **N2** could potentially be in *on* state, sinking part of the programming current, as highlighted by red dashed lines. This would 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, such as **N1** and **N2**, but pull-up transistors of input inverters, such as **P1** and **P2**, can interfere with the programming current. Such interference becomes serious as input sizes increases, which can significantly reduce the programming current passing through RRAMs and even cause failure in configuring RRAMs.

2) Breakdown Threats of Datapath Transistors:: To achieve a reasonable R_{LRS}, programming voltages $prog\_VDD$ should be large enough to drive a high enough programming current. For instance, [6] considers a programming voltage as high as $prog\_VDD = 3.0V$ while the nominal voltage of the datapath transistors is only $VDD = 0.9V$. Such large gap between $prog\_VDD$ and V_{DD} could cause the datapath transistors to breakdown during RRAMs' programming phases. Take the example in Fig. 5, the voltage of node A, V_A, can reach $prog\_VDD$ while programming RRAM R_0, leading to the source-to-drain voltage of transistor **P1** being $prog\_VDD - V_{DD}$. Assume that $prog\_VDD = 3.0V$ 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. 5 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. Therefore, there is a strong need to study how to properly integrate 4T1R programming structures into RRAM-based multiplexers without area and delay overhead while guaranteeing robust operations.

IV. Improved 4T1R-Based Multiplexer

In this section, we address the limitations of the previously introduced naive 4T1R-based multiplexers by employing power-gated inverters and rearranging the power domains. In addition to the one-level 4T1R-based multiplexers, we also investigate two-level and tree-like multiplexer structures, similar to baseline CMOS multiplexers.

A. One-Level Multiplexer Structure

In order to address the identified limitations, we present, in Fig. 6(a), an improved one-level N-input 4T1R-based multiplexer, which is different from the one in Fig. 5 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. 5. 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

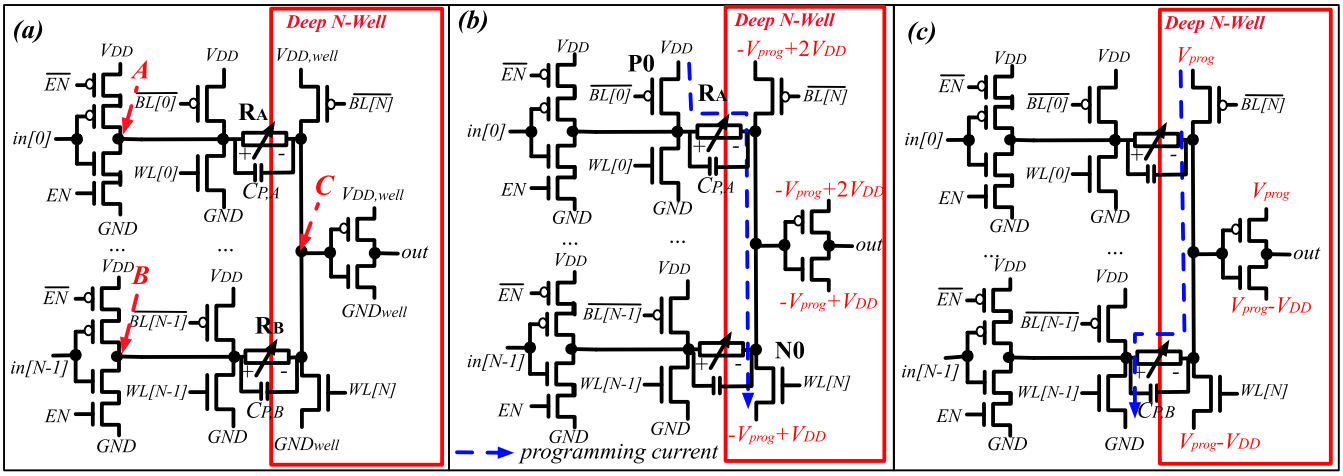

Fig. 6. Improved one-level N-input 4T1R-based multiplexer: (a) operating mode ($V_{DD,well} = V_{DD}$, $GND_{well} = GND$); (b) set process ($V_{DD,well} = -V_{prog} + 2V_{DD}$, $GND_{well} = -V_{prog} + V_{DD}$); (c) reset process ($V_{DD,well} = V_{prog}$, $GND_{well} = V_{prog} - V_{DD}$.

TABLE II
VOLTAGES ARRANGEMENTS FOR OPERATION, SET AND RESET EXAMPLES IN FIG. 6(a)–(c)

Control lines/ Voltages	Operation Fig. 6(a)	Set process Fig. 6(b)	Reset process Fig. 6(c)
$\overline{BL[0]}$	V_{DD}	GND	V_{DD}
$\overline{BL[i]}$, $1 \leq i \leq N-1$	V_{DD}	V_{DD}	V_{DD}
$\overline{BL[N]}$	V_{DD}	$-V_{prog} + 2V_{DD}$	$V_{prog} - V_{DD}$
$WL[i]$, $0 \leq i \leq N-2$	GND	GND	GND
$WL[N-1]$	GND	GND	V_{DD}
$WL[N]$	GND	$-V_{prog} + 2V_{DD}$	$V_{prog} - V_{DD}$
$\overline{EN}$	GND	V_{DD}	V_{DD}
EN	V_{DD}	GND	GND
$V_{DD,well}$	V_{DD}	$-V_{prog} + 2V_{DD}$	V_{prog}
GND_{well}	GND	$-V_{prog} + V_{DD}$	$V_{prog} - V_{DD}$

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 $\overline{BL[N]} = -V_{prog} + 2V_{DD}$. Table II summaries the voltages involved in the different operations.

The improved 4T1R-based multiplexer has a major advantage over the initial design in Fig. 5: 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

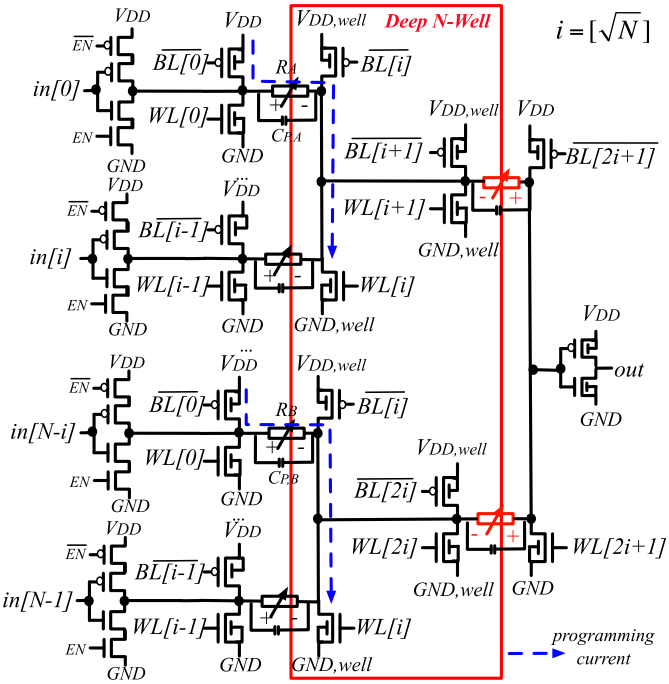

Fig. 7. Schematic of a robust two-level N-input 4T1R-based multiplexer.

breakdown limitation. Boosted $V_{DD,max}$ leads to higher current density driven by transistors, further contributing to a lower R_{LRS} [6]. Note that the set and reset processes typically require short amount of time, i.e., typically 200 ns for each RRAM [6]. Since programming does not occur many times (non-volatility), very low stress is applied on the transistors, further contributing to a robust operation.

B. Two-level and Tree-like multiplexer Structure

Based on the circuit topology of CMOS multiplexers shown in Fig. 1, we also develop N-input 4T1R-based multiplexers implemented with two-level and tree-like structures. The resulting structures are depicted in Figs. 7 and 8 respectively.

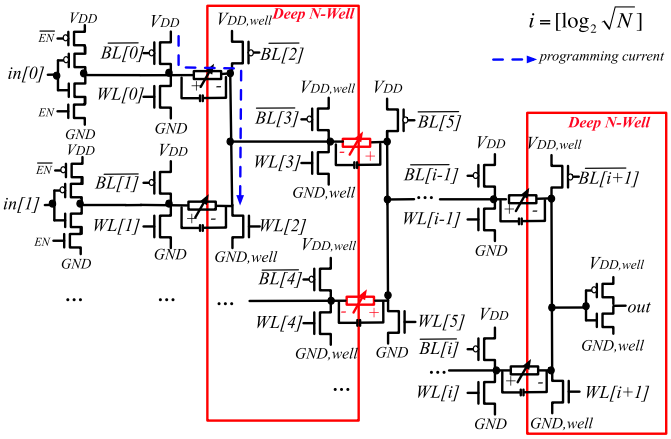

Fig. 8. Schematic of a robust tree-like N-input 4T1R-based multiplexer.

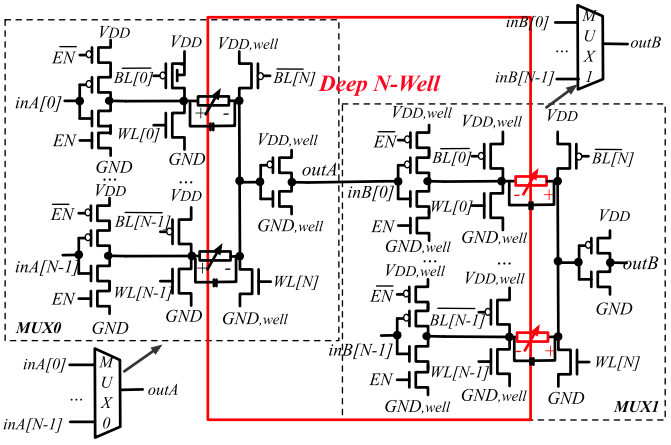

Fig. 9. Cascading two N-input one-level 4T1R-based multiplexers: share Deep N-Wells efficiently.

The two-level and tree-like structures are implemented by cascading elementary one-level multiplexer structures similar to the one shown in Fig. 5. Note that even in two-level and tree-like 4T1R multiplexers, only one DNW is needed, as highlighted red in Figs. 7 and 8 respectively. To simplify the programming strategies, RRAMs in the even levels have opposite polarities than those in the odd levels. Take the example in Fig. 7, the polarities of RRAMs in the second level, highlighted in red, are opposite to the first level. 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 type of process (either set or reset) but also on the number of levels (either even or odd), requiring additional circuitry. In addition, DNWs also 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 DNW 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 a similar programming strategy as highlighted above.

The number of bit lines and word lines can be reduced, as the 4T1R programming structures belonging to the same level can efficiently share control lines, allowing RRAMs to be programmed simultaneously. Take the example of Fig. 7, all the multiplexer structures from the first stage can be connected to bit lines $\overline{BL[j]}, 0 \le j \le \sqrt{N}$ and word lines $WL[j], 0 \le j \le \sqrt{N}$. RRAMs that are controlled by $\overline{BL[0]}$ and $WL[\sqrt{N}]$, i.e., R_A and R_B in Fig. 7, can be programmed simultaneously, which is resembling to the control sharing in a CMOS multiplexer tree. RRAMs belonging to different stages have to be programmed sequentially. A two-level or tree-like 4T1R-based multiplexer requires $2m$ steps (m reset processes and m set processes) to program all the RRAMs, where m represents the number of stages. In contrast, a one-level 4T1R-based multiplexer, consisting of fewer RRAMs, only need two steps, implying less reconfiguration time and programming energy.

C. Limitations on the Programming Voltage V_{prog}

During set and reset processes, the necessary programming voltage V_{prog} is determined by the source-to-drain voltage drop across the programming transistors and the programming threshold voltage of the RRAMs. The V_{DS} of the programming transistors should be large enough in order to drive sufficient programming current, but should also be selected under the breakdown conditions. Therefore, there exists a limit for V_{prog} to be respected. For instance, in the set example of Fig. 6(b), V_{prog} can be expressed as the sum of the voltages across RRAM A and the programming transistors **P0** and **N0**:

$$\begin{cases} V_{DS,P0}+V_{DS,N0}+V_{set,min}=V_{prog}, \\ V_{DS,P0}=V_{DS,N0}\le V_{DD,max}, \end{cases} \tag{1}$$

where $V_{set,min}$ is minimum programming voltage to trigger a set process for a RRAM. Note that the V_{DS} of the programming transistors should be the same to guarantee the best achievable current density [6]. Similarly, for the reset example in Fig. 6(c), one can derive a similar set of constraints with transistors **P1** and **N1**:

$$\begin{cases} V_{DS,P1}+V_{DS,N1}+V_{reset,min}=V_{prog}, \\ V_{DS,P1}=V_{DS,N1}\le V_{DD,max}, \end{cases} \tag{2}$$

where $V_{set,min}$ is minimum programming voltage to trigger a reset process for a RRAM.

In addition to the limitations mentioned above, the use of different wells also constrains V_{prog} as the diode across *P-Well* and *Deep N-Well* should be reversely biased, as illustrated in Fig. 10(a) and (b). During the reset process in Fig. 10(a), diode D_0 is always reversely biased because the voltage of *P-Well* is GND and the voltage of *Deep N-Well* is $V_{prog} > GND$. However, during the set process in Fig. 10(b), diode D_1 is reversely biased only when:

$$(-V_{prog}+2V_{DD})-GND \ge 0. \tag{3}$$

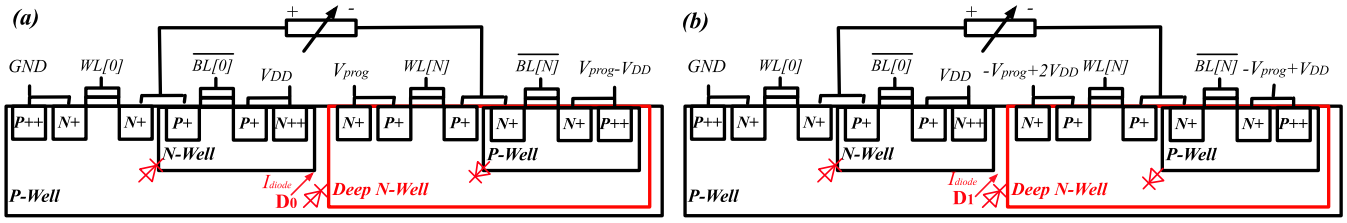

Fig. 10. Cross-section of the layout of a 4T1R programming structure: (a) during reset process; (b) during set process.

TABLE III
ANALYTICAL COMPARISON ON AREA, DELAY AND SWITCHING ENERGY OF N-INPUT 4T1R-BASED MULTIPLEXERS

Multiplexer	Area[1]	Delay[2]	Energy[3]
One-level	$N \cdot Area_{trans}$	$R_{LRS} \cdot (C_{trans} + N \cdot C_P)$	$0.5 \cdot \alpha \cdot C_{trans} + (N \cdot C_P) \cdot V_{DD}^2$
Two-level	$(N + [\sqrt{N}]) \cdot Area_{trans}$	$4R_{LRS} \cdot (C_{trans} + [\sqrt{N}] \cdot C_P)$	$0.5 \cdot \alpha \cdot 4 \cdot (C_{trans} + [\sqrt{N}] \cdot C_P) V_{DD}^2$
Tree-like	$(2N - 2) \cdot Area_{trans}$	$\frac{1}{2}(3[log_2 N]^2 - [log_2 N]) R_{LRS}$ $\cdot (C_{trans} + C_P)$	$0.5 \cdot \alpha \cdot \frac{1}{2}(3[log_2 N]^2 - [log_2 N])$ $\cdot (C_{trans} + C_P) V_{DD}^2$

[1] Area of input and output inverters are not included here. [2] Elmore delay model [22] is considered here.
[3] Only the switching energy of multiplexer structures is considered here. α is the switching activity.
* R_{LRS} is the equivalent resistance of a RRAM in LRS. C_P is smaller than C_{trans}.

If we boost V_{DD} to $V_{DD,max}$ during set and reset process, the constraint becomes:

$$(-V_{prog} + 2V_{DD,max}) - GND \geq 0. \tag{4}$$

By combining (1), (2) and (5), we obtain:

$$\begin{cases} V_{prog} \leq 2V_{DD,max} + V_{set}, \\ V_{prog} \leq 2V_{DD,max} + V_{reset}, \\ V_{prog} \leq 2V_{DD,max}. \end{cases} \tag{5}$$

As a result, the upper bound for V_{prog} can be expressed as:

$$V_{prog} \leq 2V_{DD,max} \tag{6}$$

As discussed in [6], a larger V_{prog} leads to a higher programming current and a lower R_{LRS}. In this paper, we consider $V_{prog} = 2V_{DD,max}$ for the electrical simulations.

D. Analytical Comparison between 4T1R multiplexers

Note that the two-level and tree-like 4T1R-based multiplexers reduce the number of control/programming lines significantly but does not reduce the number of required RRAMs. An analytical comparison of the area, delay and energy between 4T1R-based multiplexers is shown in Table III, and will be verified by electrical simulations in Section V. In CMOS technology, two-level multiplexers produce the best area-delay-power product because their structure reduces not only the number of control lines but also the parasitic capacitances introduced in the critical path. Since the parasitic capacitances of a RRAM is typically smaller than a transistor, the delay and power of one-level 4T1R-based multiplexers scale better with the number of inputs N than CMOS multiplexers. When the input size is small and total capacitance is dominated by programming transistors, the delay and power of one-level 4T1R-based multiplexers are better than two-level and tree-like structures. When the input size is large enough, the total capacitance is dominated by C_P and two-level 4T1R-based multiplexers become better in delay and power.

V. EXPERIMENTAL RESULTS

In this section, we will verify the conclusions drawn by our analytical comparison with electrical simulations and further evaluate the performance of the proposed multiplexers. We first explain our experimental methodology. Then, we show and comment the transient behavior of 4T1R-based multiplexers, and finally we compare the area, delay and power between different 4T1R-based and CMOS multiplexer topologies.

A. Experimental Methodology

We consider a RRAM technology [12] with programming voltages $V_{set} = |V_{reset}| = 0.8V$ 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 200 ns. Stanford RRAM compact model [25] is used to model the considered RRAM technology. The ASAP 7 nm FinFET design kit from ASU [13] is used in the circuit designs of datapath logics and 4T1R programming structures. Datapath circuits are built with standard logic transistors, while the 4T1R programming structures employ low-V_t transistors. The standard logic transistors have a nominal working voltage $V_{DD} = 0.7V$, and can be overdriven to $0.9V$ while staying in their reliability limits. Transmission gates are implemented with a pair of n-type and p-type FinFETs. Input and output inverters are minimum sized. Delay and power results are extracted from HSPICE [26] simulations. The datapath V_{DD} is swept from 0.5 V to 0.7 V with a step 0.1 V, in order to study the trade-off between delay and power in sub/near-V_t regime. The programming voltage V_{prog} is selected to

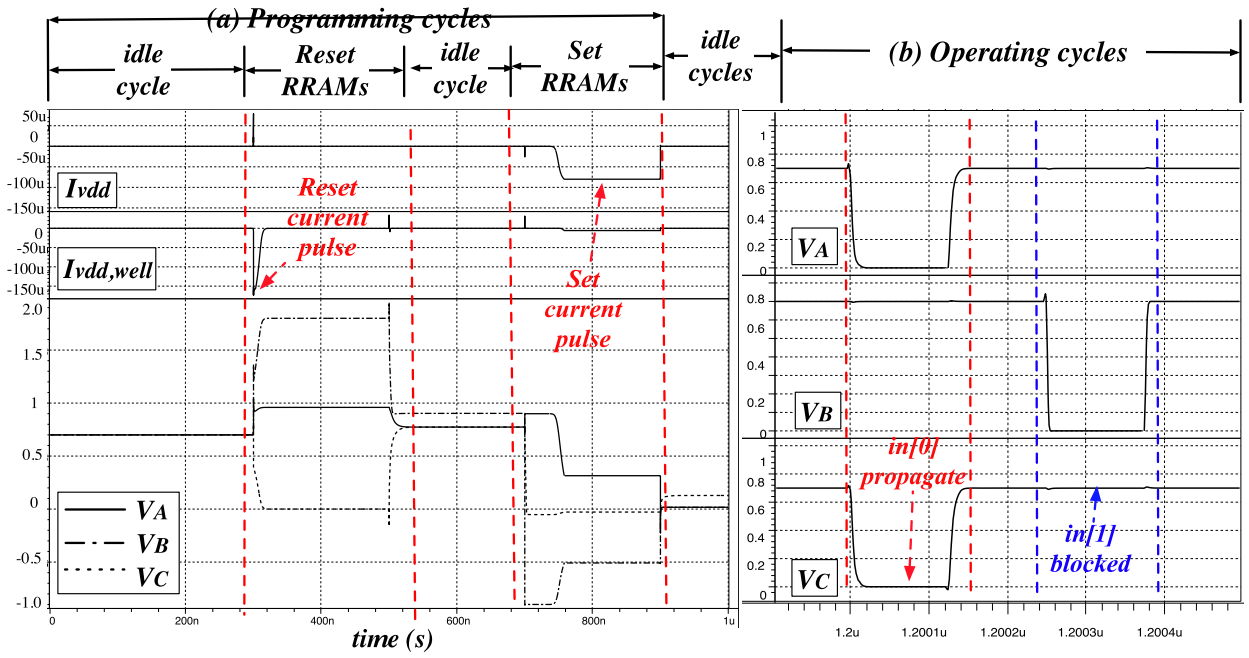

Fig. 11. Transient analysis of a 2-input 4T1R-based multiplexer in Fig. 6(a): (a) signal waveforms of programming phase; (b) signal waveforms of operation.

be 1.8 V, respecting to the physical design limits, discussed in Section IV-C.

The comparison baseline is selected from the CMOS multiplexer topologies in Fig. 1 in terms of best *Power-Delay Product* (PDP). When input size N is lower or equal than 12, we consider one-level CMOS multiplexers as baseline. When input size N is larger than 12, our baseline becomes a two-level CMOS multiplexer. As for 4T1R-based multiplexers, we consider one-level, two-level and tree-like structures for comparison on area, delay and power.

B. Transient Analysis

In order to validate the analytical comparisons in Table III, we perform transient simulations for 4T1R-based multiplexers, which consist of two phases: (1) the programming phase, where set and reset operations are made to validate the RRAM programming strategy; and (2) the datapath operation phase, where we verify if the multiplexer is functionally correct. Without loss of generality, we focus on a representative example: a 2-input one-level 4T1R-based multiplexer (consider $N = 2$ in Fig. 6). Such transient analysis was conducted for every 4T1R-based multiplexer. Before programming, we initialize a 4T1R-based multiplexer in Fig. 6 as follows: RRAMs R_A and R_B are formed and configured to HRS and LRS respectively. During the programming phase depicted in Fig. 11(a), R_B is first reset to HRS by a reset procedure, then R_A is set to LRS by a set cycle. Fig. 11(a) illustrates that both R_A and R_B can be set or reset successfully according to the changes in programming currents I_{vdd0} and I_{vdd1}. Between the programming phase and operating cycles, there are a few idle cycles during which programming transistors are all turned off. After then, input pulses are generated sequentially to the four inputs, as shown in Fig. 11(b). We see that the multiplexer is functionally correct, as $in[0]$ is propagated to the output while $in[1]$ is blocked. Transient analysis also verifies that RRAMs can be programmed correctly without interfering each other.

TABLE IV
PDP OF THE CONSIDERED 16-INPUT MULTIPLEXERS WITH VARYING NUMBER OF FINS IN EACH PROGRAMMING TRANSISTOR

PDP (aJ)	CMOS	4T1R-based		
#. of FinFETs	Two-level	One-level	Two-level	Tree-like
1	83.7	**48.5**	**51.3**	**69.5**
2	263.1	54.4	62.6	95.8
3	343.5	60.3	76.5	122.0

C. Best Number of Fins for Each Programming Transistor

As explained in [11], the sizing of programming transistors can significantly impact the delay and power number of RRAM-based multiplexers. In this paper, we extend this study to the specific FinFET context. For each 4T1R-based multiplexer structure, we sweep the number of fins per programming transistors from 1 to 3, in order to identify the optimal fin number in terms of *Power-Delay Product* (PDP). We consider three fins as the upper limit because in the considered design kit, three fins allow the 4T1R structure to match the standard cell height, simplifying some layout considerations. Table IV summarizes the PDP results for 16-input 4T1R-based multiplexers, implemented with one-level, two-level and tree-like structures. Note that all the 4T1R-based multiplexers achieve best PDP when the number of fin is one, as the representative example shown in Table IV. In the rest of paper, we consider all 4T1R-based multiplexers with one fin per programming transistor.

D. Area Comparison

In order to properly study the physical area of the proposed structure, i.e., considering routing, well organization etc., and

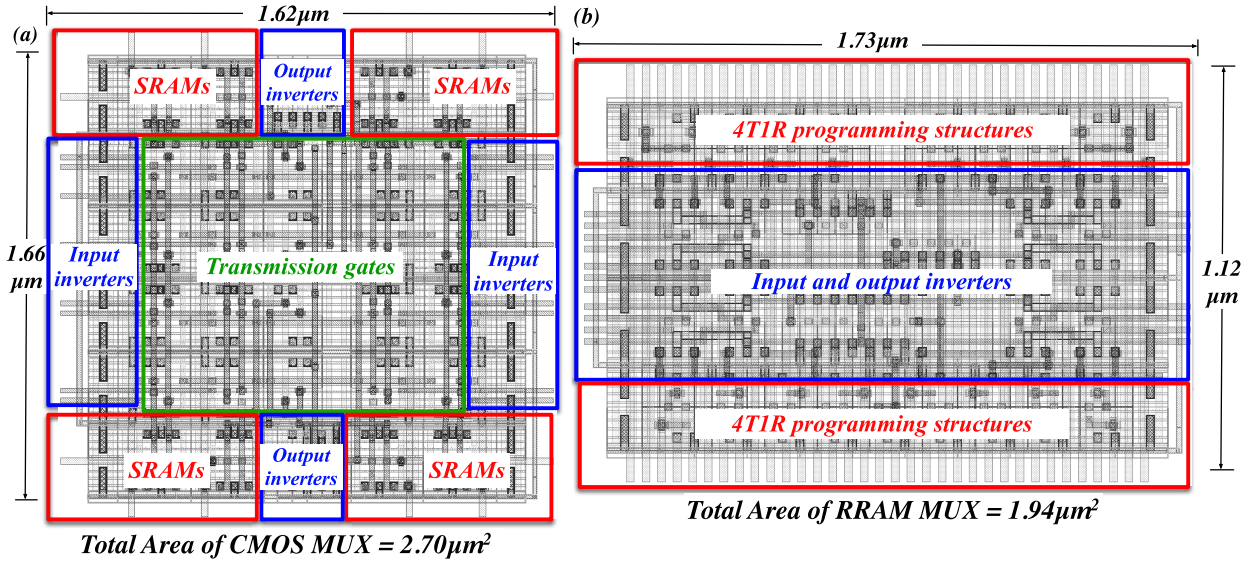

Fig. 12. Layout of 16-input multiplexers: (a) CMOS two-level structure; and (b) 4T1R-based two-level structure.

draw fair area comparisons with regular CMOS, we realized the layouts of a 16-input two-level CMOS multiplexer and a 16-input two-level 4T1R-based multiplexer, as depicts in Fig. 12(a) and (b) respectively. Since the different wells can be efficiently shared among multiplexers as shown in Fig. 9, the layout of 4T1R-based multiplexer consists of the programming structures and input inverters (*MUX0* in Fig. 9) in a regular well. The output and associated programming structure of another multiplexer (*MUX1* in Fig. 9) can be shared in this same well. The output inverter and associated programming structure of *MUX0* will be located in a *deep N-well* which also contains programming structure and input inverters of another multiplexer. CMOS multiplexers must employ SRAMs to store their configuration bits, while 4T1R-based multiplexers eliminate the use of SRAMs as their configuration bits are stored in RRAMs. To access either the SRAMs or the RRAMs, we assume a memory bank organization, i.e., using parallel word lines and bit lines. Since CMOS and 4T1R-based multiplexers have similar number of configuration bits, the area of their memory banks are similar and are not included in their layouts. The benefit on removing SRAMs leads to that a 4T1R-based multiplexer ($1.94\mu m^2$) is 28% smaller than its CMOS counterpart ($2.70\mu m^2$). We believe that the area comparison between 16-input multiplexers is representative and also its conclusive trend is also valid for multiplexers with other sizes.

E. Delay Improvements

Fig. 13(a) compares the delay of CMOS multiplexers, naive 4T1R and 2T1R-based multiplexers and the improved 4T1R-based multiplexers with the different structures under analysis. Note that when input size is larger than 34, RRAM programming of the naive 4T1R multiplexers is regarded as a failure because programming structures cannot drive enough current through RRAMs due to serious interference from input inverters. As a result, the RRAM LRS becomes too high and the multiplexer performance degrades significantly. Similarly, the naive 2T1R multiplexers cannot be properly programmed when input size is 8 because 2T1R programming structure leads to 50% smaller current density. In contrast, the improved 4T1R-based multiplexers with one-level, two-level and tree-like structures can guarantee RRAM configuration successful. In addition, the improved 4T1R-based multiplexers reduce delay by 2× over the naive 4T1R and 2T1R-based multiplexers. When the input size is smaller than 50, one-level structure performs better in delay than two-level and tree-like structures due to its smaller parasitic capacitances. When the input size becomes larger than 50, a two-level structure becomes the best choice in delay. One-level structures and two-level 4T1R-based multiplexers achieve up to 34% and 27% delay improvements respectively, as compared to their CMOS counterparts. Note that even when the input size is small, i.e., $N = 2$, one-level 4T1R-based multiplexers have similar performance than CMOS implementations. The performance gap between one-level and two-level structures reduces to zero when input size is 50. Similar to CMOS multiplexers, when input size is larger than 50, a two-level structure should be used for best performance.

We also investigate the performance of the multiplexers in the near-V_t regime. As illustrated in Fig. 13(b), CMOS multiplexers suffer from 2.25× delay degradation when V_{DD} decreases from 0.7 V to 0.5 V. However, because, unlike transistors, the resistances of RRAMs are not affected by a reduction of V_{DD}, one-level 4T1R-based multiplexers keep a high-performance-level even in the near-V_t regime. When $V_{DD} = 0.6V$, one-level 4T1R-based multiplexers improve delays by up to 2×, as compared to CMOS multiplexer. Note that, when compared to CMOS multiplexers operating at $V_{DD} = 0.7V$, one-level 4T1R-based multiplexers operating with $V_{DD} = 0.6V$ outperform up to 22% in delay.

F. Energy and Power Benefits

Fig. 14(a) shows the energy efficiency of naive 4T1R and 2T1R-based multiplexers and 4T1R-based multiplexers

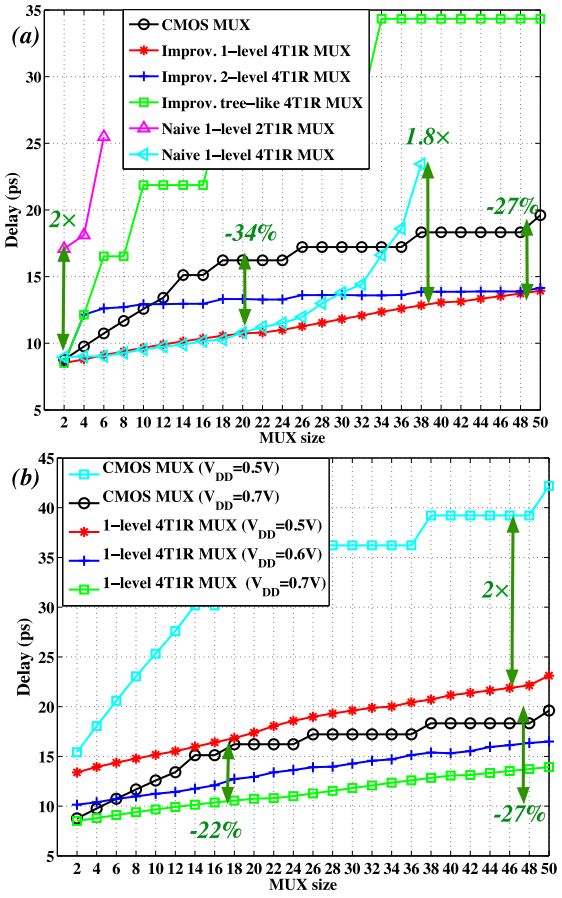

Fig. 13. Delay comparison between CMOS and 4T1R-based multiplexers: (a) delay improvements of one-level, two-level and tree-like structures ($V_{DD} = 0.7\ V$); (b) delay efficiency of one-level structure at near V_t regime.

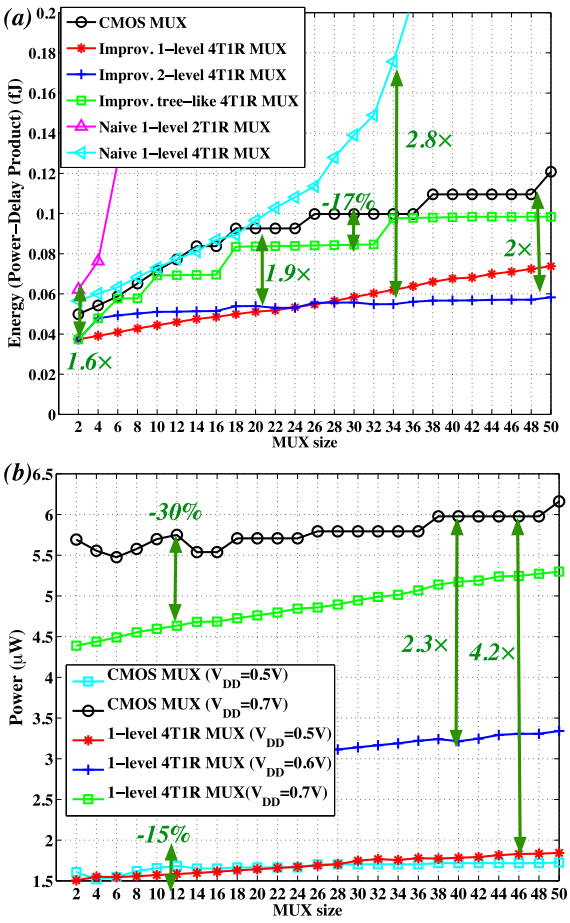

Fig. 14. Power comparison between CMOS and 4T1R-based multiplexers: (a) energy improvements of one-level, two-level and tree-like structures ($V_{DD} = 0.7V$); (b) power reduction of one-level structure at near V_t regime.

with different improved structures. Note that naive 4T1R and 2T1R-based multiplexers consumes 2.8× more energy than the improved one-level 4T1R-based multiplexers due to serious leakage current from input inverters. When the input size is smaller than 24, a one-level structure multiplexer performs better in terms of energy consumption, bringing up to 2.5× reduction compared to CMOS multiplexers, thanks to the smaller parasitic capacitances. When the input size is larger than 24, a two-level structure becomes the best choice. 4T1R-based multiplexers are not only efficient in energy but also in power, as shown in Fig. 14(b). At nominal $V_{DD} = 0.7V$, one-level 4T1R-based multiplexers reduce power by 30% as compared CMOS multiplexers. In near-V_t regime, i.e. $V_{DD} = 0.5\ V$, the power reduction can reach up to 4.2× as compared to CMOS multiplexers at nominal $V_{DD} = 0.7V$. Note that such power reduction is achieved along with significant delay improvements.

G. Area-Delay and Power-Delay Products Analysis

To explore the inherent trade-offs with area, delay and power, we compare *Area-Delay Product* (ADP) and *Power-Delay Product* (PDP) of CMOS and 4T1R-based multiplexers, as shown in Fig. 15. Similar to CMOS multiplexers, we select the best structure for 4T1R-based multiplexers with varying input sizes, in terms of best PDP. When input size is lower or equal than 24, we consider one-level structure. When input size is larger than 24, we consider two-level structure. Since 4T1R-based multiplexers reduce both area and delay significantly, *Area-Delay Product* (ADP) of 4T1R-based multiplexers can be up to 2.6× more efficient than CMOS multiplexers than CMOS multiplexers, as illustrated in Fig. 15(a). Since 4T1R-based multiplexers are more delay and power efficient than CMOS multiplexers in near-V_t regime, *Power-Delay Product* (PDP) of 4T1R-based multiplexer improves over 3.8× the one of CMOS multiplexers, as shown in Fig. 15(b). $V_{DD} = 0.5\ V$ guarantees the best PDP for 4T1R-based multiplexers. In summary, 4T1R-based multiplexers are delay and power efficient at both nominal V_{DD} and near-V_t regime.

VI. Impact of Process Variations of RRAMs

RRAMs are more susceptible to device variations than transistors. As their mechanism is physically stochastic, there is a large observed cycle-to-cycle variability [5]. The variations on RRAM parameters, such as V_{set} and V_{reset}, could lead to a degradation of RRAM-based multiplexers performance. Therefore, it is necessary to understand, for a given technology node, what is the range of variations that the RRAM multiplexers can

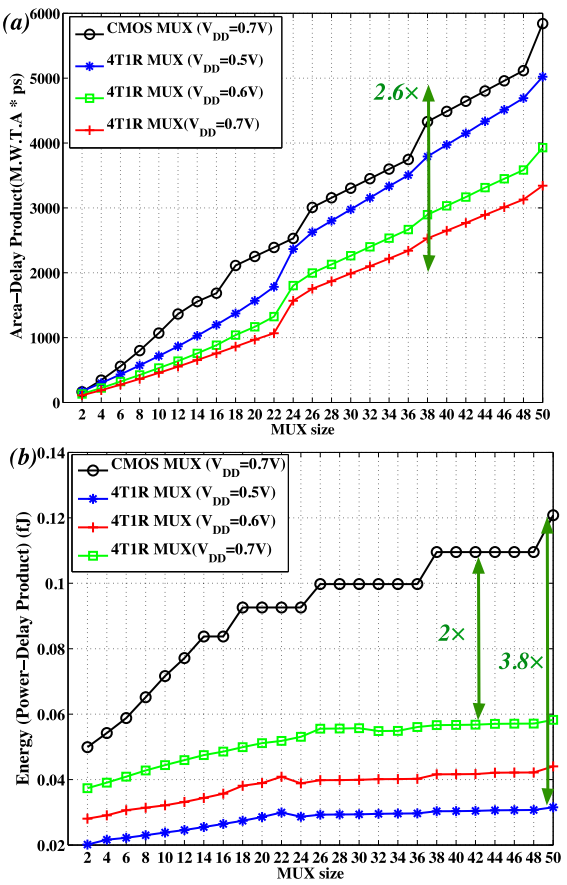

Fig. 15. Comparison between CMOS multiplexers and 4T1R-based multiplexers: (a) Area-Delay Product; (b) Power-Delay Product.

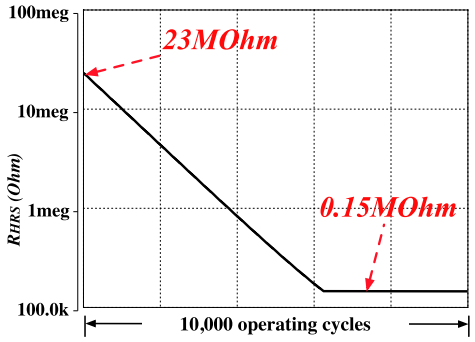

Fig. 16. R_{HRS} degradation when $V_{set} = 0.6\ V < V_{DD} = 0.7\ V$.

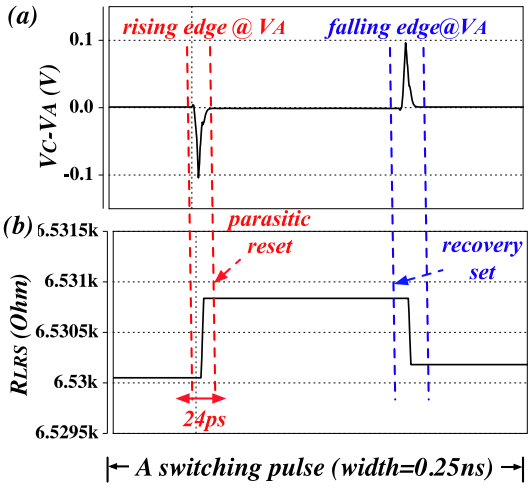

Fig. 17. (a) Voltage across a RRAM in LRS (V_A and V_C in Fig. 6(a)) during operation; and (b) R_{LRS} degradation when $V_{reset} = 0.1\ V$.

tolerate without significant degradation in delay and power. In this section, we study the effect of two representative RRAM parameters: V_{set} and V_{reset}, coupled with a 7 nm FinFET technology.

A. Impact of Variations on V_{set}

Process variations on V_{set} may cause $V_{set} < V_{DD}$, where RRAMs could be parasitically set during operation. Take the example in Fig. 11(b), during regular operation (highlighted in red), where $V_A = GND, V_B = V_{DD}$ and $V_C = GND$, the voltage drop across RRAM R_B could be large enough to trigger a set process. The RRAM R_B in HRS could be gradually set to LRS after a certain amount of time. In this part, we consider a RRAM technology whose $V_{set} = 0.6\ V$ is smaller than $V_{DD} = 0.7\ V$. Using electrical simulations, we run a fatigue test for a 2-input RRAM multiplexer by running 10 thousands operating cycles, whose input waveforms are similar to the one shown in Fig. 11(b). Fig. 16 illustrates the degradation trend of R_{HRS} of RRAM R_B, where R_{HRS} decreases gradually from $23M\Omega$ to $0.15M\Omega$ and then no further degradation is observed. The lower bound of R_{HRS} degradation remains to be $0.15M\Omega$ even when 100 000 and $1M$ operating cycles are further applied. The existence of a lower bound of R_{HRS} can be explained as following: The voltage at node C in Fig. 6(a) is dependent on the resistance of R_B,

$$V_C = V_{DD} \cdot \frac{R_{R_B}}{R_{R_A} + R_{R_B}}, \tag{7}$$

where R_{R_A} and R_{R_B} represent the resistances of RRAM R_A and R_B in Fig. 6(a) respectively. As R_{R_B} degrades, V_C decreases as well, leading to the voltage drop across RRAM R_B decreases. When the voltage drop across RRAM R_B is reduced to $V_{set} = 0.6\ V$, the parasitic set process is stopped. The lower bound of degradation is independent from the number of operating cycles but is related to R_A and V_{set}. Note that the degradation on R_{HRS} could cause significant leakage overhead [12]. In this paper, we consider a 15% margin between nominal V_{DD} and V_{set}. Additionally, the excellent performance of 4T1R-based multiplexers in near-V_t regime allows the use of low V_{DD}, i.e., $= 0.5V$, further increasing the margin to 60%. We believe such margin is sufficient to resist V_{set} variations.

B. Impact of Variations on V_{reset}

A parasitic reset process could also happen to a RRAM in LRS when the voltage drop across RRAM $|V_{RRAM}| < |V_{reset}|$. However, during normal operation, the voltage drop across a RRAM is typically smaller than $0.1\ V$, as shown in Fig. 17(a), and the duration of such voltage drop is as short as $24\ ps$.

Hence, as long as V_{reset} varies to be above $max\{V_C - V_A\}$, i.e., $= 0.1\ V$, a parasitic reset process can be fully avoided. Using electrical simulation, we consider $V_{reset} = 0.4\ V, 0.5\ V$ and $0.6\ V$ in the same torture test as described in Section VI-A, and the resistances of RRAM in LRS remains unchanged in all the conditions. Even if V_{reset} is smaller than $max\{V_C - V_A\}$, R_{LRS} degradation is much less serious than R_{HRS}. Fig. 17(b) illustrates that when V_{reset} is below $0.1\ V$, the parasitic reset caused by a rising edge of V_A ($V_C > V_A$) can be partly recovered by a falling edge of V_A ($V_C < V_A$), resulting in a 0.5Ω R_{LRS} degradation per operation cycle. However, as compared to nominal $V_{reset} = 0.8\ V$ considered in this paper, process variation can be well controlled to ensure $V_{reset} > 0.1\ V$ and thus parasitic reset can be fully avoided.

VII. Conclusion

In this paper, we proposed new one-level, two-level and tree-like multiplexers circuit designs using 4T(ransistors)1R(RAM) elements and we compared them to naive one-level multiplexers. We studied the physical design aspects of the 4T1R-based multiplexers, e.g., layout for 7 nm FinFET technology and the co-integration of low-voltage nominal power supply and high-voltage programming supply. Electrical simulations show that using a 7 nm FinFET transistor technology and by considering input size ranging from 2 to 50, the proposed 4T1R-based multiplexers reduce delay by $2\times$ and energy by $2.8\times$ over the naive 4T1R and 2T1R counterparts. At nominal working voltage, the proposed 4T1R-based multiplexers reduces *Area-Delay* and *Power-Delay* products by $2.6\times$ and $3.8\times$ respectively, as compared to the best CMOS multiplexers topologies. The proposed 4T1R-based multiplexers operated at near-V_t regime can achieve up to 22% delay improvement along with $2.3\times$ power reduction, as compared to best CMOS multiplexers working at nominal voltage.

References

[1] J. M. Rabaey *et al.*, *Digital Integrated Circuits*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 2002.

[2] E. Lee, G. Lemieux, and S. Mirabbasi, "Interconnect driver design for long wires in field-programmable gate arrays," *J. Signal Process. Syst.*, vol. 51, no. 1, pp. 57–76, Apr. 2008.

[3] R. Waser and M. Aono, "Nanoionics-based resistive switching memories," *Nature Mater.*, vol. 6, no. 11, pp. 833–840, Nov. 2007.

[4] H. Akinaga *et al.*, "Resistive random access memory (ReRAM) based on metal oxides," *Proc. IEEE*, vol. 98, no. 12, pp. 2237–2251, Dec. 2010.

[5] H.-S. P. Wong *et al.*, "Metal-oxide RRAM," *Proc. IEEE*, vol. 100, no. 6, pp. 1951–1970, Jun. 2012.

[6] X. Tang, G. Kim, P.-E. Gaillardon, and G. D. Micheli, "A study on the programming structures for RRAM-based FPGA architectures," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 63, no. 4, pp. 503–516, Apr. 2016.

[7] S. Tanachutiwat, M. Liu, and W. Wang, "FPGA based on integration of CMOS and RRAM," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 11, pp. 2023–2032, Nov. 2011.

[8] P.-E. Gaillardon, M. H. Ben-Jamaa, G. B. Beneventi, F. Clermidy, and L. Perniola, "Emerging memory technologies for reconfigurable routing in FPGA architecture," in *Proc. IEEE ICECS*, Dec. 2010, pp. 62–65.

[9] J. Cong and B. Xiao, "FPGA-RPI: A novel FPGA architecture with RRAM-based programmable interconnects," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 4, pp. 864–877, Apr. 2014.

[10] P.-E. Gaillardon, D. Sacchetto, S. Bobba, Y. Leblebici, and G. D. Micheli, "GMS: Generic memristive structure for non-volatile FPGAs," in *Proc. IEEE/IFIP VLSI-SoC*, Oct. 2012, pp. 94–98.

[11] X. Tang *et al.*, "A high-performance low-power near-Vt RRAM-based FPGA," in *Proc. IEEE ICFPT*, Dec. 2014, pp. 207–215.

[12] X. Tang, P.-E. Gaillardon, and G. D. Micheli, "Accurate power analysis for near-V_t RRAM-based FPGA," in *Proc. IEEE FPL*, Sep. 2015, pp. 174–177.

[13] L. T. Clark *et al.*, "ASAP7: A 7-nm FinFET predictive process design kit," *Microelectron. J.*, vol. 53, pp. 105–115, Jul. 2016.

[14] G. W. Burr, B. N. Kurdi, J. C. Scott, C. H. Lam, K. Gopalakrishnan, and R. S. Shenoy, "Overview of candidate device technologies for storage-class memory," *J. Res. Develop.*, vol. 52, nos. 4–5, Jul. 2008.

[15] J.-G. Zhu, "Magnetoresistive random access memory: The path to competitiveness and scalability," *Proc. IEEE*, vol. 96, no. 11, pp. 1786–1798, Nov. 2008.

[16] Y. S. Chen *et al.*, "Highly scalable hafnium oxide memory with improvements of resistive distribution and read disturb immunity," in *Proc. IEEE IEDM*, Dec. 2009, pp. 1–4.

[17] C. Chiasson *et al.*, "Should FPGAs abandon the pass-gate?" in *Proc. FPL*, 2013, pp. 1–8.

[18] Y. Liauw *et al.*, "Nonvolatile 3D-FPGA with monolithically stacked RRAM-based configuration memory," in *Proc. ISSCC*, 2012, pp. 406–408.

[19] P.-E. Gaillardon *et al.*, "Phase-change-memory-based storage elements for configurable logic," in *Proc. IEEE FPT*, Dec. 2010, pp. 17–20.

[20] Y.-C. Chen *et al.*, "Non-volatile 3D stacking RRAM-based FPGA," in *Proc. IEEE FPL*, Aug. 2012, pp. 367–372.

[21] K. Huang, R. Zhao, W. He, and Y. Lian, "High-density and high-reliability nonvolatile field-programmable gate array with stacked 1D2R RRAM array," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 1, pp. 139–150, Jan. 2016.

[22] W. C. Elmore, "The transient response of damped linear networks with particular regard to wideband amplifiers," *J. Appl. Phys.*, vol. 19, no. 1, pp. 55–63, 1948.

[23] K. A. El-Ayat *et al.*, "A CMOS electrically configurable gate array," *IEEE J. Solid-State Circuits*, vol. 24, no. 3, pp. 752–762, Jun. 1989.

[24] J. Greene *et al.*, "A 65nm flash-based FPGA fabric optimized for low cost and power," in *Proc. ACM FPGA*, 2011, pp. 87–95.

[25] J. Jiang *et al.*, "Verilog-A compact model for oxide-based resistive random access memory," in *Proc. IEEE SISPAD*, Sep. 2014, pp. 41–44.

[26] *HSPICE User Guide: Simulation and Analysis, Version I-2013.12*, Synopsys, Mountain View, CA, USA, Dec. 2013.

Xifan Tang (S'13) received the B.Sc. degree in microelectronics from Fudan University, Shanghai, China, in 2011, and the M.Sc. degree in electrical engineering from the École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, in 2013, where he is currently working toward the Ph.D. degree. His current research interests include computer-aided design for programmable architecture and emerging technologies. He is the recipient of 2015 Chinese Government Award for Outstanding Self-Financed Students Abroad.

Edouard Giacomin will receive his Electrical Engineer degree from CPE Lyon, France, in 2016. He is currently doing his M.S. thesis at the University of Utah, Salt Lake City, UT, USA, designing a test chip for ReRAM-based FPGA structures. He will work toward the Ph.D. degree at the University of Utah starting January 2017, working towards ReRAM-based FPGA design.

Giovanni De Micheli (F'94) is Professor and Director of the Institute of Electrical Engineering and of the Integrated Systems Centre at EPF Lausanne, Switzerland. He is program leader of the Nano-Tera.ch program. Previously, he was Professor of Electrical Engineering at Stanford University. Prof. De Micheli is a Fellow of ACM and a Member of the Academia Europaea. His research interests include several aspects of design technologies for integrated circuits and systems, such as synthesis for emerging technologies, networks on chips and 3D integration. He is also interested in heterogeneous platform design including electrical components and biosensors, as well as in data processing of biomedical information. He is author of *Synthesis and Optimization of Digital Circuits* (McGraw-Hill, 1994) and the coauthor and/or coeditor of eight other books and of over 700 technical articles. His citation h-index is 88 according to Google Scholar. He is member of the Scientific Advisory Board of IMEC (Leuven, B), CfAED (Dresden, D) and STMicroelectronics. Prof. De Micheli is the recipient of the 2016 IEEE/CS Harry Goode award for seminal contributions to design and design tools of Networks on Chips, the 2016 EDAA Lifetime Achievement Award, the 2012 IEEE/CAS Mac Van Valkenburg award for contributions to theory, practice and experimentation in design methods and tools and of the 2003 IEEE Emanuel Piore Award for contributions to computer-aided synthesis of digital systems. He received also the Golden Jubilee Medal for outstanding contributions to the IEEE CAS Society in 2000, the D. Pederson Award for the best paper on the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS in 1987, and several Best Paper Awards, including DAC (1983 and 1993), DATE (2005) and Nanoarch (2010 and 2012). He has been serving IEEE in several capacities, namely: Division 1 Director (2008-9), cofounder and President Elect of the IEEE Council on EDA (2005-7), President of the IEEE CAS Society (2003), Editor in Chief of the IEEE Transactions on CAD/ICAS (1997-2001). He has been Chair of several conferences, including Memocode (2014) DATE (2010), pHealth (2006), VLSI SOC (2006), DAC (2000) and ICCD (1989).

Pierre-Emmanuel Gaillardon (S'10–M'11–SM'16) received the Electrical Engineer degree from CPE-Lyon, France, in 2008, the M.Sc. degree in electrical engineering from INSA Lyon, France, in 2008, and the Ph.D. degree in electrical engineering from CEA-LETI, Grenoble, France, and the University of Lyon, France, in 2011. He is an Assistant Professor in the Electrical and Computer Engineering (ECE) department at the University of Utah, Salt Lake City, UT, USA, and he leads the Laboratory for NanoIntegrated Systems (LNIS). Prior to joining the University of Utah, he was a Research Associate at the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, within the Laboratory of Integrated Systems (Prof. De Micheli). and a Visiting Research Associate at Stanford University, Palo Alto, CA, USA. Previously, he was Research Assistant at CEA-LETI, Grenoble, France. Prof. Gaillardon is recipient of the C-Innov 2011 best thesis award and the Nanoarch 2012 best paper award. He is an Associate Editor of the IEEE TRANSACTIONS ON NANOTECHNOLOGY. He has been serving as TPC member for many conferences, including DATE'15-16, DAC'16, Nanoarch'12-16, and is reviewer for several journals and funding agencies. He will serve as Topic co-chair "Emerging Technologies for Future Memories" for DATE'17. He is a senior member of the IEEE. The research activities and interests of Prof. Gaillardon are currently focused on the development of reconfigurable logic architectures and digital circuits exploiting emerging device technologies and novel EDA techniques.