GMS: Generic Memristive Structure for Non-Volatile FPGAs

Pierre-Emmanuel Gaillardon[1], Davide Sacchetto[2], Shashikanth Bobba[1], Yusuf Leblebici[2], Giovanni De Micheli[1]
[1] LSI, EPFL, Lausanne, Switzerland [2] LSM, EPFL, Lausanne, Switzerland
pierre-emmanuel.gaillardon@epfl.ch

Abstract—**The invention of the memristor enables new possibilities for computation and non-volatile memory storage. In this paper we propose a *Generic Memristive Structure* (GMS) for 3-D FPGA applications. The GMS cell is demonstrated to be utilized for steering logic useful for multiplexing signals, thus replacing the traditional pass-gates in FPGAs. Moreover, the same GMS cell can be utilized for programmable memories as a replacement for the SRAMs employed in the look-up tables of FPGAs. A fabricated GMS cell is presented and its use in FPGA architecture is demonstrated by the area and delay improvement for several architectural benchmarks.**

I. INTRODUCTION

Future deeply scaled circuits will see their performances limited by the physical limitations of the materials. To keep pushing the performance of computation and the density of storage, the microelectronics industry envisages using more efficient state variable than the electronic charge. In this sense, the memristor is an attractive candidate for both computation and memory, thanks to its programmable resistive state. When considering the *Resistive RAM* (ReRAM) memories, which can be classified as memristors, excellent scalability and programming time can be obtained if compared to traditional Flash. This is related to the fact that ReRAMs are simple two-terminal devices [1, 2].

While a lot of research effort targets high density ReRAM-based standalone memories [3], the focus of this work is the usage of ReRAMs for *Field-Programmable Gate Arrays* (FPGAs). The reason behind this choice is that in reconfigurable logic, up to 40% of the area is dedicated to the storage of configuration signals [4]. Traditionally, the configuration data is serially loaded in SRAM cells, distributed throughout the circuit [5]. As a consequence, circuit power on is limited by slow serial configuration. To overpass SRAM volatility and loading time, Flash NVM have been proposed [6]. Nevertheless, the use of a hybrid CMOS-Flash technology results in high fabrication cost. Conversely, ReRAMs are fabricated within the *Back-End-of-the-Line* (BEoL) metal lines, moving the configuration memory to the top of the chip and reducing the area utilization [7]. Similarly, the ReRAM cells can be utilized in combination with *Through-Silicon-Via* (TSVs), enabling 3-D stacked FPGA architectures [8].

With the recent development of ReRAM technology, a number of novel FPGA building blocks and architectures have been proposed in the past few years. For example, routing structures based on ReRAMs have shown promise. In [9], a cross point for switchboxes, using the RRAMs as non-volatile switches, is proposed to route signals through low-resistive paths, or to isolate them by means of high-resistive paths. The concept of routing elements based on ReRAM switches was then exploited in [10, 11] for timing optimization in FPGAs.

In this paper, we propose a complete proof of concept of a ReRAM-based *Generic Memristive Structure* (GMS) circuit for FPGAs from technology development to architectural evaluation. The main idea is to replace the pass-transistors in SRAM-based FPGAs by ReRAMs. Hence, the ReRAMs store the information in their resistive states and can be used either to route signals through low-resistive paths, or to isolate them by means of high-resistive paths. Such a functionality is used to build either routing *Multiplexers* (MUXs) or configuration nodes. In order to keep the programming complexity as per SRAM-based FPGAs, we propose an efficient methodology based on the Generic Memristive Structure complementary programming. The proposed methodology has been validated by electrical measurements on a fabricated GMS device. Finally, the impact of the GMS MUXs and configuration memories is studied at the system level over a set of complex benchmarks. We show that the GMS-based FPGA reduces area by 7%, while the low on-resistance of ReRAMs provide a gain of 58% in delay compared to SRAM-based counterpart.

The paper is organized as follows. Section II presents the FPGA architecture and describes the motivation of this work. Section III describes fabrication of ReRAMs. Then, in Section IV, the novel GMS cell design and the associated programming technique are presented. The GMS is then used as a building block for MUXs and configuration memories in Section V. The GMS impact is studied at circuit level in Section VI. Architectural benchmarking is detailed in Section VII. Finally, in Section VIII, we present the conclusion.

II. ARCHITECTURAL BACKGROUND AND MOTIVATION

FPGAs are regular circuits formed by several identical reconfigurable logic blocks called *Configurable Logic Blocks* (CLBs) that are surrounded by reconfigurable interconnect lines [5]. As depicted in Fig. 1, every CLB is formed by a set of *N Basic Logic Elements* (BLEs). A BLE is simply a *K*-input *Look-Up-Table* (LUT), whose output can be routed to any other LUT input with or without being saved in a flip-flop. Every CLB has *I* inputs coming from other CLB outputs. All design parameters *N*, *K* and *I* can be set by the FPGA architect depending on the targeted system granularity.

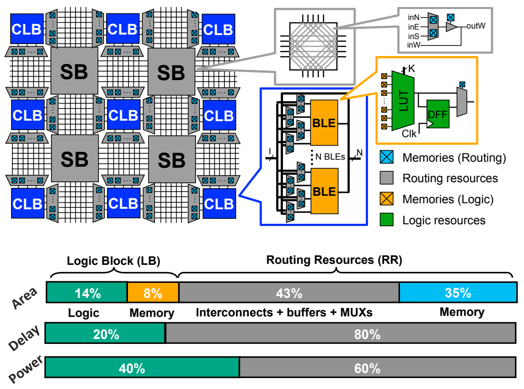

Figure 1. Baseline FPGA architecture [5] (top) and FPGAs area/delay/power repartition per resources [4] (bottom).

Programmable interconnections between the different blocks are realized by a massive number of multiplexers configured by memory cells. Fig. 1 depicts also the area/delay/power breakdown of the various components of a baseline SRAM-based island-style FPGA. It is noteworthy that the customizable resources play a major role in FPGA performance, contributing to more than 80% of the total area and delay. For this reason, the FPGA architecture can be improved by working on memories and their efficient use in routing operations.

III. ReRAM Technology

Many different ReRAM technologies are currently investigated. In this section, we will draw some generalities and present the fabrication flow.

A. *Generalities*

Oxide memory technologies base their working principle on the change in resistance state due to a modification of the conductivity. Different physical mechanisms can be identified in the switching of ReRAMs [1]. In the following, we will focus only on the *Bipolar Resistive Switching* (BRS) mechanism [2]. The BRS mechanism is related to the O_2 vacancy redistribution in TiO_2 layers upon application of a voltage across the oxide, causing a resistance change from low to high and viceversa depending on the voltage polarity. In the following, the electrode on top of the ReRAM structure is defined as the positive electrode.

B. *Experimental Process Flow*

The fabrication flow of the test structures started from bulk-Si wafers passivated by a 100-nm thick Al_2O_3 layer deposited by *Atomic Layer Deposition* (ALD) (Fig. 2-a). Next horizontal 70-nm thick *Bottom Electrode* (BE) lines were patterned by lift-off and e-beam evaporation (Fig. 2-b). Then, for some devices a 10-nm thick TiO_2 layer was deposited with ALD (Fig. 2-c). For other devices a 50-nm thick TiO_2 was deposited by reactive sputtering of a Ti target in $O_2/N_2/Ar$ atmosphere at room temperature. Finally, vertical *Top Electrode* (TE) lines of the crossbar were defined with a second lift-off together with contact areas for electrical characterization (Fig. 2-d). The fabrication method was demonstrated for crossbar arrays with half-pitch down to 100-nm. In Fig. 2-e, a 64-bits crossbar memory cell with 200-nm half-pitch is shown. In Table I several electrode combinations of top and bottom electrode materials are reported.

C. *Experimental Measurements*

To achieve consistent BRS, a forming step with low current compliance (<100uA) was performed. A typical forming voltage above +3.5 V was found for a top electrode voltage, while the bottom electrode was grounded. Then consistent BRS with different R_{ON} and R_{OFF} for the different devices was measured (Fig. 2-f). Forming operation is not suited for highly distributed memory applications, such as FPGAs. Nevertheless, forming-free ReRAMs can be fabricated by the use of different methods, and in this study the two devices with TiO_2 deposited by reactive sputtering switch without the need of a forming step (Table I). This can be attributed to the more defective structure of sputtered TiO_2, which typically consists of an heterogeneous mixture of different phases.

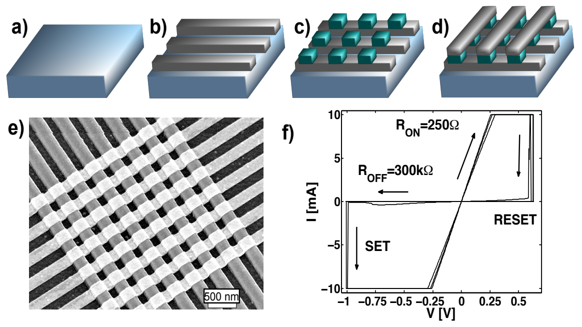

Figure 2. a) Si wafer coated with 100-nm ALD Al_2O_3 insulation layer. b) Horizontal Al metal lines deposited with lift-off defined by e-beam lithography. c) 10-nm thick TiO_2 layer deposited with ALD. d) Vertical Al metal lines deposited as per b) forming the top electrodes. e) A 64-bit crossbar prototype array with 200-nm half-pitch. f) ReRAM bipolar switching (500-nm half-pitch cell). After the forming process SET and RESET occurs at negative and positive top electrode voltages.

TABLE I
Measured Electrical Characteristics For Different ReRAMs

TE	TiO_2 deposition	BE	R_{ON} [Ω]	R_{OFF} [Ω]	$V_{FORMING}$ [V]	V_{SET} [V]	V_{RESET} [V]
Al	ALD	Al	250	300k	+3.5	-1	+0.7
Al	ALD	Ni	180	1k	-7.25	-2	+1.9
Pt	ALD	TiN	200	1k	+10	+0.80	-1.00
W	ALD	W	20	150	+4	+2	-1.2
Pt	Sputtering	Pt	100	1M	NA	+1.8	-1.3
Cu	Sputtering	Pt	5k	500M	NA	-4.2	+5

D. *Storage Element Integration Flow*

One of the big advantages of ReRAM technology is CMOS-compatibility. The materials involved in ReRAMs are deposited at low temperature and can be integrated into the *Back-End-of-The-Line* (BEoL). As an illustration, a schematic cross section of a co-integrated ReRAM-CMOS transistor is shown in Fig. 2-a. As in standalone NOR arrays, illustration includes a storage node and a selector transistor in series in the 1-Transistor 1-Resistor (1T1R) configuration. The memory element may be fabricated either just after the Si contact formation step or after the first steps of interconnections (*e.g.* on top of Metal 1 interconnect level).

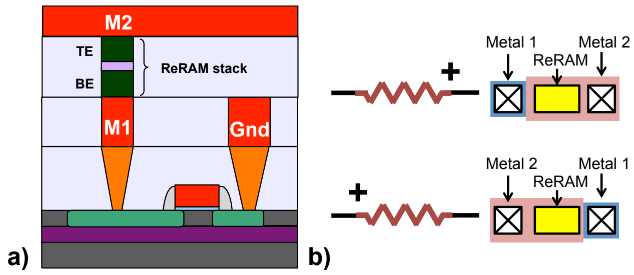

Figure 3. (a) Cross sectional schematic showing the integration of a ReRAM integrated between the M1 and M2 interconnection levels in the back-end-of-line. The bottom electrode is thus directly connected to a MOSFET selector (bottom) forming a 1-Transistor 1-Resistor (1T1R) memory node. (b) ReRAM polarity selection by physical design.

IV. Generic Memory Structure

In order to simplify the programming scheme, a GMS structure consisting of two in series connected ReRAMs is introduced.

A. *GMS Concept*

As per the previous section, ReRAMs can be fabricated within the BEoL processing. Hence, it is possible to fabricate them between two metal layers (e.g. in between Metal 1 and Metal 2). Because of the BRS of the ReRAMs of this study, depending on the forming polarity, either the Metal 1 or the Metal 2 terminal can be utilized as the positive electrode of the memory, giving two possible configurations (see Fig. 2-b).

In the GMS, two ReRAMs are interconnected as shown in Fig. 4-a. The positive terminal of the top device is connected to the negative terminal of the bottom device. This arrangement enables complementary programming of the two ReRAMs composing a GMS. We call the concurrent programming of the GMS a complementary programming operation. A similar programming scheme was previously used for low power crossbars [12]. Fig. 4-b illustrates the programming of the top path (i.e. left to right arrow in the programming graph shown in Fig. 4-d). R_1 and R_2 are switched simultaneously to R_{OFF} and to R_{ON} respectively. This operation is achieved by grounding the common right terminal and biasing the left terminal to V_{th} (which corresponds to the SET voltage $-V_{th}$ for R_1 and to the RESET voltage $+V_{th}$ for R_2). Programming the bottom path (see Fig. 4-c) is done by inverting V_{th} and *Gnd* (which corresponds to the RESET voltage for R_1 and to the SET voltage for R_2). In addition to increasing the programming speed, only two voltages are needed (*Gnd* and V_{th}), thanks to the complementary scheme.

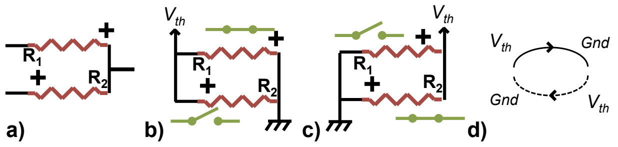

Figure 4. GMS complementary programming.

B. *Experimental Validation*

The complementary programming operation has been validated by electrical measurements, while the MUX performances have been extracted by electrical simulations. Fig. 5 depicts the resistance values of R_1 and R_2 of an GMS-based MUX21. Resistances are read at V_{READ}=+0.1V.

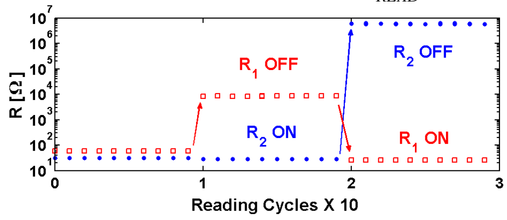

Figure 5. Complementary switching operation for the ReRAM-based GMS.

After a preliminary forming step, R_1 and R_2 are set to R_{ON}. The devices are then read for 10 cycles, showing a stable non-volatile resistance. Hence R_1 and R_2 are switched using the complementary programming operation presented in the previous section. During the first write operation SET and RESET events are induced on R_2 and R_1, respectively (see Fig. 6-c) by applying a voltage pulse for 500us. After reading the resistance values for another 10 cycles, again validating the non-volatility of the resistance states, a second complementary switching operation is performed as depicted in Fig. 6-b. Now the resistance states of R_1 and R_2 switch complementary, as seen in the reading sequence of Fig. 7. Note that the resistance values of R_1 and R_2 do not exactly match. This is due to the different ReRAM geometries and to the large variability of the cells utilized for the demonstrator. Nevertheless, improved variability of one order of magnitude has been demonstrated for ReRAM prototypes fabricated with industrial methods [2].

V. GMS-based FPGA Design

In this section, the operation of a novel multiplexer design and a configuration memory based on GMS is discussed.

A. *GMS-Based Multiplexer*

1) *Overall Structure*

Fig. 6-a illustrates a 4 to 1 MUX based on CMOS transmission-gates arranged in two cascaded stages. In this MUX, a unique path is configured between an input D_X and the output Y. The path is selected by the signals S_X. Adjacent paths are complementary addressed by inverted signals SN_X. The selection signals are permanently driven to ensure a constant path selection. Inspired by this structure, the ReRAM-based MUX (depicted in Fig. 6-b) uses the unique low on-resistance property of the ReRAM memories to create high-performance non-volatile switches. Thus, it is possible to replace the pass-gates by the ReRAM to obtain a non-volatile MUX. The path selection operation is achieved by programming to low resistance state all the individual memories that belong to the desired path. The others paths are deselected by programming their memories to high resistance state.

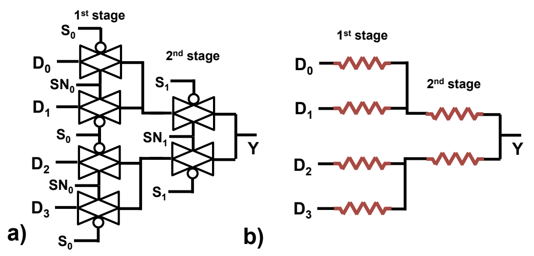

Figure 6. 4 to 1 multiplexers based on a) pass-gates and b) ReRAMs

2) *Configuring the MUX Network*

For each ReRAM composing the MUX structure, a *high-* or a *low-* resistive state must be programmed. This individual selection and write operation leads to an increase in the programming complexity. In order to simplify this, a complementary programming scheme for ReRAM network is proposed here. Complementary programming is explained for a 4 to 1 MUX, however it can be generalized to a generic MUX. A two stage 4 to 1 MUX and its programming circuit are shown in Fig. 9-a. By enabling the V_P signal ($V_P = 1$), all the input and output nodes of stage *i* are shorted to the output of the configuration flip-flops Q_i and Q_{i+1} respectively. In the example of the Fig. 9, nodes *n1* to *n4*, *n5* and *n6*, and *n7* are connected to Q_1, Q_2 and Q_3 respectively. In order to avoid cross programming between different stages, each stage is configured sequentially. Hence, for a two stage MUX network, we need two steps for configuring a unique path. As an example, we illustrate the two steps required to configure the path connecting input D_1 to the output Y. All the configuration transistors are first turned *Off* ($V_P = 0$). A set of digital logic levels is then serially fed to the shift register. Voltages are chosen to configure the stage in the desired state without interfering with the other stages. Fig. 9-b presents the configuration scheme for programming the two stages sequentially. In the first step (Step 1), the first stage is configured to enable the ReRAM connected to the input D_1. This is achieved by applying $Q_1=Gnd$ and $Q_2=V_{th}$. Programming of the successive stages is disabled by applying the same voltage at Q_2 to the upstream stages (i.e. $Q_3=V_{th}$). It has to be noted that with this operation also the ReRAM connected to the input D_3 is enabled. After storing the desired voltages in the registers, the configuration transistors are turned *On* ($V_P = 1$) and the basic elements of the stage are programmed in the desired state. After the ReRAMs programming, all the configuration transistors are turned *Off.* In the next step, the procedure is repeated for the second stage, which is configured by enabling the ReRAM path between nodes n5 and n7. This is achieved by applying $Q_2=V_{th}$ and $Q_3=Gnd$. At this point, the first stage is kept static, without any programming, by applying the same voltage as Q_2 to the downstream stages (i.e. $Q_1=V_{th}$).

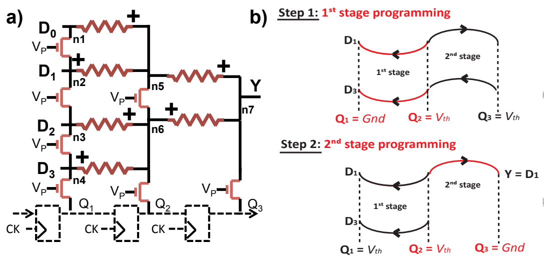

Figure 7. a) 4 to 1 multiplexer with programming circuits and b) associated programming diagram to configure Output to input D1.

B. *GMS-Based Configuration Memory*

In this section, we present an elementary circuit used to move most of the configuration part of reprogrammable circuits to the fabrication back-end, reducing their impact on fabrication front-end occupancy. Such a memory node is dedicated to drive LUT inputs. The memory node is based on a unique GMS node and provides intrinsically the retained information as a voltage level. Furthermore, it allows an efficient layout by sharing lines.

1) *Overall Structure*

The basic memory node is presented in Figure 8-a. The circuit consists of 2 ReRAMs connected in a voltage divider configuration between 2 fixed voltage lines (L_A and L_B). The memories are operated in a complementary manner, in order to improve reliability. The output is designed to place a fixed voltage on a conventional standard cell input. Read operations are intrinsic with the structure, while programming is an external operation to perform on the cell.

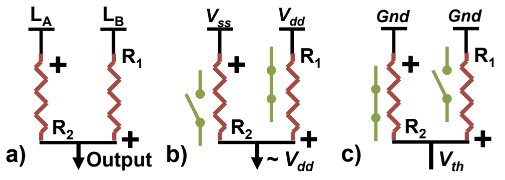

Figure 8. (a) ReRAM-based memory node. (b) Node in read configuration. (c) Node in write configuration.

2) *Read operation*

A voltage divider is implemented in this topology to intrinsically realize the conversion from a bit of data stored as resistance level to a voltage level. Figure 8-b presents a configuration example where the node stores a '1'. Voltage lines L_A and L_B are respectively connected to V_{ss} and V_{dd}. For the sake of illustration, consider that the resistive memory R_1, connected to the V_{dd} line, is configured to the low resistivity state. The other memory R_2, connected to V_{ss}, is in the high resistivity state. As a consequence, a voltage divider is configured and the output node is charged close to the voltage of the branch with a high conductivity. The logic levels depend on R_{ON} and R_{OFF} as per the voltage divider structure.

It is also worth noticing that in continuous read operation, a current will be established through the resistors. This leads to a passive current consumption through the structure, which is highly dependent on R_{OFF}. This static current can be reduced by the choice of a memory technology like $Cu/TiO_2/Pt$ (Table I) maximizing the R_{OFF} value, without any impact on the speed. Indeed, the configuration memories are not directly related to the data path and only drive static nodes.

3) Write Operation

Figure 8-c presents the programming phase of the node. First, the lines L_A and L_B are disconnected from the power lines and connected to the programming signals. The programming signals are chosen according to the GMS programming scheme. Fig. 9 presents the programming circuits required to program an array of GMS-based configuration memories. To provide individual access, each GMS has its own selection transistor. Thus, the different lines can be shared in a standalone-memory-type architecture, yielding a more efficient layout. The different modes and programming signals are selected by line-driving MUXs.

VI. Performance Characterization

In this section, evaluation of the GMS-based FPGA elementary blocks is proposed at the circuit level. The study focuses on the block-level metrics such as area, programming time and energy.

A. Methodology

To validate the RRAM-based building blocks, we characterized their performances metrics in terms of area and write time. The performance extraction is based on the node complexity expressed in terms of the basic elements that are required to realize the circuit. The area is extracted from basic layout considerations using CMOS 45-nm technology rules [13] and expressed in half-pitch to give values independent of lithography node. Timing and energy numbers are extracted from the ITRS [14]. Comparison with building blocks traditionally used in FPGA, such as CMOS SRAM 5T cells [5] and Flash memory elements [6], are then used to evaluate the structures. The associated numbers are also extrapolated from the ITRS [14]. Note that we are dealing with non-volatile memories. Hence, we will stress the comparison with regards to Flash.

B. Memory Performance Characterization

Table II shows some characterization results in terms of area, write time and programming energy for the proposed solution and traditional FPGA memory nodes. Note that this comparison only considers the storage node itself and is not including all the external programming circuitry. We observe that the proposed ReRAM cell is the most compact solution with a gain of 3x compared to Flash, even with the impact of the programming current on the access transistor. This advantage is due to the reduction of the memory front-end footprint to only one transistor, compared to 5 for the SRAM cell and 2 for the Flash solution (one pull-up transistor coupled to a floating gate transistor). In addition, ReRAMs offer a significant writing time and programming energy reduction for non-volatile memory technologies of 16x and 8x respectively. Finally, note that the leakage power depends mainly on the material. Materials with a high R_{OFF} should be priviledged for low power operation. Indeed, $Cu/TiO_2/Pt$ demonstrates a gain of 2 orders of magnitude compared to SRAMs.

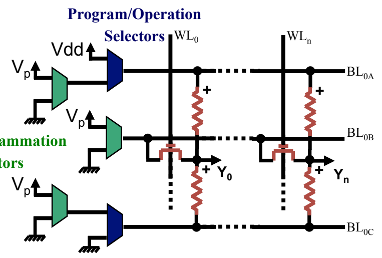

Fig. 9. Line sharing illustration in standalone-memory-like architecture.

TABLE II
GMS-Based Configuration Memory Performance Evaluation

	Cell	Area (F^2)	Write time (ns)	Prog. energy (pJ)	Leakage at 1V (nW)
SRAM	5T	196	0.2	5.10^{-4}	142
Flash cell	2T	84	1 000	100	210
ReRAM cell	1T2R	28	60	12	1 000 (Pt) 2 (Cu)
Flash vs. ReRAM	-	x 3	x 16.6	x 8.3	x 0.2 - x 105

C. Data Path Impact Characterization

Fig. 10 depicts the timing response of the 2 to 1 basic multiplexer. The elementary multiplexer structure is build with a unique $Pt/TiO_2/Pt$-based GMS. The timing response was obtained by electrical simulation and compared to a CMOS equivalent counterpart built in 45-nm technology [13]. We observe a timing improvement of 4.5 times as compared to CMOS. This remarkable delay reduction is due to the low on-resistance of the ReRAM technologies. For instance, at 45-nm the internal resistance of an *n*-type transistor is 3.8 kΩ (minimal size transistor extracted from 45-nm design kit [13]), while the ReRAM technology exhibits an on-resistance of hundreds of Ohms. Note that, in the proposed MUX design, the programming circuits are not related to the data path. Thus, only the ReRAM parameters impact the electrical performances. Finally, as the memories are directly used to perform the routing operation, no leakage power is dissipated by the MUX (i.e. no permanent leakage path exist in the structure), offering significant interest for power reduction.

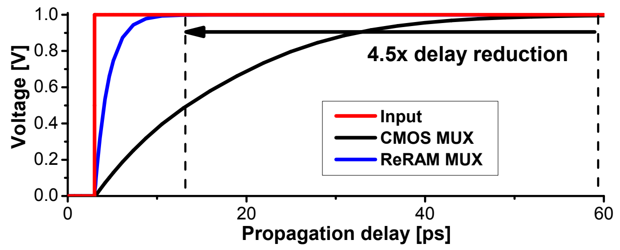

Figure 10. Electrical simulation of a GMS-based 2 to 1 MUX timing response.

VII. Architecural Impact

The demonstrated structure introduces compact ReRAM memories with low on-resistance in the data paths of the FPGAs. In this section, we will study the impact of the structure on an architectural perspective.

A. Methodology

A set of logic circuits taken from the MCNC benchmark were used, which have been synthesized using ABC [15]. The technology mapping was then performed with a library of 4-input LUTs (K=4) using ABC as well. Subsequently, the logic packing of the mapped circuit into CLBs was done with N=10 BLEs per CLB and I=22 external inputs using AA-PACK [16]. Finally, the placement and routing were carried out using VPR6.0 [16]. Each benchmark was first synthesized on an SRAM-based FPGA in the CMOS 45 nm process [13]. Then, the SRAM-based MUXs were replaced by their ReRAM counterparts (Pt/TiO_2/Pt). The impact of the circuits needed for the programming is taken into account for the evaluations.

B. Simulation Results

The benchmarks were mapped in CMOS SRAM-based and ReRAM-based FPGAs. The critical path delay estimation is shown in Fig. 11. The benchmarks showed an area reduction ranging from 4% to 8%, with 7% on average, coming from a slight reduction of the silicon surface occupied by the routing resources. Note that the complete set of programming resources have been considered. The simulations showed a critical path delay reduction ranging from 43% to 73%, with 58% on average. The reduction is the direct impact of the high performances MUXs, introduced throuhout the data path. This makes the ReRAM-based FPGA potentially faster than the SRAM-based counterparts. In addition, the leakage power of the CLBs is reduced by 10%, thanks to inexistence of leakage current in the MUX structure.

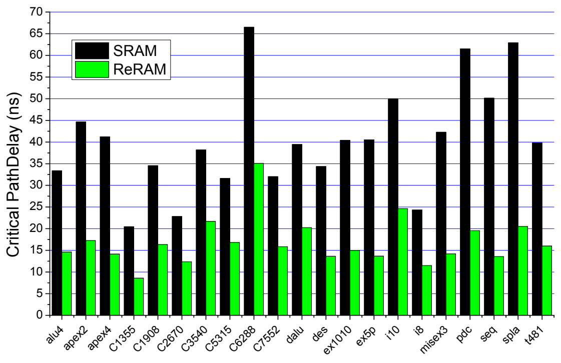

Figure 11. Delay estimation for FPGAs synthesized with ReRAM- and SRAM-based multiplexers.

VIII. Discussions

The bipolar resistive switching ReRAMs presented in this paper have been fabricated and electrically characterized in terms of R_{ON}/R_{OFF} ratio and read/write voltages for a different combination of metal electrodes with sputtered or ALD deposition of TiO_2. As reported in Table I, the cells with sputtered TiO_2 are the only ones showing resistive switching without the need of a forming step. ReRAM made of Pt/sputtered TiO_2/Pt has been chosen to carry out the architectural simulations for FPGA because of a better R_{ON}/R_{OFF} ratio and the compatibility with a ±2V programming voltages. The GMS cells are utilized to replace SRAM LUTs in FPGAs in a more compact way because ReRAMs are implemented into the BEoL. Moreover, the complementary switching mechanism of the GMS cells is utilized also as steering logic. In particular, the low R_{ON} memories improve the delay in the data paths of FPGAs. Last but not least, ReRAMs can be built in different flavors, depending on the objectives in terms of delay and power trade-off. For instance, the Cu/sputtered TiO_2/Pt ReRAM of Table I can be exploited for its large R_{OFF}. Such a material leads to a reduction of the CLB static power consumption of up to 69% compared to SRAM FPGAs.

IX. Conclusion

This paper introduced a novel design, called GMS, based on resistive memories, designed to replace traditional routing resources in reconfigurable logic circuits. Resistive RAM memories combined into a complementary switching GMS cells were used to reduce the footprint and to improve the electrical performances of the data path. The GMS cell can also be used to replace standalone memories, leading to more compact LUTs and steering logic, due to the BEoL integration of ReRAMs. After validating the working principle of the GMS complementary switching, the impact of ReRAMs on the area and the critical path delay of an FPGA was simulated. Thanks to ReRAMs, the area and the delay are reduced by 7% and 58% respectively due to the compactness and the low on-resistance of ReRAMs.

References

[1] G.W. Burr *et al.*, "Overview of candidate device technologies for storage-class-memory," IBM J. R&D, 52(4/5), 2008.

[2] Y.S. Chen *et al.*, "Challenges and Opportunities for HfO_X Based Resistive Random Access Memory," IEDM Tech. Dig., 2011.

[3] S.-S. Sheu *et al.*, "A 4Mb Embedded SLC Resistive-RAM Macro with 7.2ns Read-Write Random-Access Time and 160ns MLC-Access Capability," ISSCC Tech. Dig., 2011.

[4] M. Lin *et al.*, "Performance Benefits of Monolithically Stacked 3-D FPGA," IEEE TCAD, 26(2), 2007.

[5] V. Betz *et al.*, "Architecture and CAD for Deep-Submicron FPGAs", Kluwer Academic Publishers, 1999.

[6] K. J. Han *et al.*, "Flash-based Field Programmable Gate Array Technology with Deep Trench Isolation," IEEE CICC, 2007.

[7] Y.Y. Liauw *et al.*, "Nonvolatile 3D-FPGA With Monolithically Stacked RRAM-Based Configuration Memory," ISSCC Tech. Dig., 2012.

[8] D. Sacchetto *et al.*, "Resistive Programmable Through Silicon Vias for Reconfigurable 3D Fabrics, IEEE TNANO, 11(1):8-11, 2012.

[9] P.-E. Gaillardon *et al.*, "Emerging memory technologies for reconfigurable routing in FPGA architecture," ICECS Tech. Dig., 2010.

[10] S. Tanachutiwat, M. Liu, and W. Wang, "FPGA Based on Integration of CMOS and RRAM," IEEE TVLSI, 19(11):2023-2032, Nov. 2011.

[11] J. Cong and B. Xiao, "mrFPGA: A novel FPGA architecture with memristor-based reconfiguration," NANOARCH Tech. Dig., 2011.

[12] E. Linn, R. Rosezin, C. Kügeler and R. Waser, "Complementary resistive switches for passive nanocrossbar memories," Nature materials, 2010.

[13] http://www.eda.ncsu.edu/wiki/FreePDK/

[14] Emerging Research Devices, 2011 Edition, ITRS

[15] Berkeley Logic Synthesis and Verification Group, ABC, Release 70930. http://www.eecs.berkeley.edu/~alanmi/abc/

[16] J. Luu *et al.*, "Architecture description and packing for logic blocks with hierarchy, modes and complex interconnect," ACM FPGA Symp., 2011.