Optimization Opportunities in RRAM-based FPGA Architectures

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

***Abstract*—*Static Random Access Memory* (SRAM)-based routing multiplexers, whatever structure is employed, share a common limitation: their area, delay and power increase linearly with the input size. This property results in most SRAM-based FPGA architectures typically avoiding the use of large multiplexers. *Resistive Random Access Memory* (RRAM)-based multiplexers, built with one-level structure, have a unique advantage over SRAM-based multiplexers: their ideal delay is independent from the input size. This property allows RRAM-based FPGA architectures to use larger multiplexers than their SRAM-based counterparts, without generating any delay overhead. In this paper, by carefully considering the properties of RRAM multiplexers, we assess that current state-of-art architectural parameters for SRAM-based FPGAs cannot preserve optimality in the context of RRAM-based FPGAs. As a result, we propose that in RRAM-based FPGAs, (a) the routing tracks should be interconnected to *Look-Up Table* (LUT) inputs via a one-level crossbar, instead of through *Connection Blocks* and local routing; (b) the *Switch Blocks* should employ larger multiplexers; (c) length-2 wires should be used instead of length-4 wires. When operated in nominal voltage, the proposed RRAM-based FPGA architecture reduces area by 26%, delay by 39% and channel width by 13%, as compared to a SRAM-based FPGA with a classical architecture. When operated in the near-V_t regime, the proposed RRAM-based FPGA architecture improves *Area-Delay Product* by 42% and *Power-Delay Product* by $5\times$ as compared to a classical SRAM-based FPGA at nominal voltage.**

I. Introduction

Advances in *Resistive Random Access Memories* (RRAMs) technology have attracted intensive research interests in exploring RRAM-based *Field Programmable Gate Array* (FPGA) architectures. Previous works [1]–[6] focus on replacing *Static Random Access Memory* (SRAM)-based multiplexers in the classical FPGA architectures with RRAM-based multiplexers. Since RRAM-based multiplexers are naturally more delay-efficient than SRAM-based multiplexers, RRAM-based FPGAs can achieve a 7%-15% gain in area, a 45%-58% reduction in delay and a 20%-58% reduction in power, when compared to SRAM-based FPGAs [1]–[4]. However, very limited works focus on the architectural repercussions of this technology and study novel RRAM-based architectures that can fully unlock the potential of RRAM-based multiplexers.

The delay of SRAM-based multiplexers, whatever structure is employed, increases linearly with the input size. This limitation forces most SRAM-based FPGA architectures to use multiple levels of small crossbars, instead of large multiplexers, to avoid any area, delay and power overhead [7]. However, RRAM-based multiplexers have a unique advantage over SRAM-based multiplexers: their ideal delay is independent of the input size. The advantage of RRAM-based multiplexer reshapes the traditional interconnection topology in classical SRAM-based architecture. As a result, RRAM-based FPGA architectures should privilege one-level crossbars, consisting of large multiplexers, as much as possible. The paradigm shift in the interconnection topology also requires to rethink the optimal architectural parameters, which have been well determined for classical SRAM-based architectures [7]. Hence, there is a strong need to investigate the right RRAM-based FPGA architectures which can exploit the full potential of RRAM-based multiplexers, and determine the associated optimal architectural parameters.

In this paper, we propose a new RRAM-based FPGA architecture that fully leverages the advantage of RRAM-based multiplexers. Three architectural enhancements are proposed: (a) The routing tracks should be interconnected to *Look-Up Table* (LUT) inputs via a one-level crossbar, instead of through *Connection Blocks* and local routing; The Connection Block connectivity parameter $F_{c,in}$ should be increased. (b) The *Switch Block* connectivity parameter F_s should be increased; (c) The Best single wire length L should be smaller. We study the best values of $F_{c,in}$, F_s and L in terms of area, delay, power and channel width. Architecture-level simulations show that a proposed RRAM-based FPGA should employ ($F_{c,in} = 0.33$, $F_s = 6$ and $L = 2$) to achieve best performance, which is different from those of classical SRAM-based architectures. Averaged over the twenty biggest MCNC benchmarks, the proposed RRAM-based FPGA architecture when operated in nominal voltage, reduces area by 26%, delay by 39% and channel width by 13%, as compared to a SRAM-based FPGA with a classical architecture. When operated in the near-V_t regime, the proposed RRAM-based FPGA architecture improves *Area-Delay Product* by 42% and *Power-Delay Product* by $5\times$ as compared to a classical SRAM-based FPGA at nominal voltage.

The rest of this paper is organized as follows. Section II reviews the background of SRAM-based and RRAM-based FPGAs. Section III introduces advantages of RRAM-based multiplexers. Section IV proposes three architectural enhancements. Section V presents the experimental results, while Section VI concludes the paper.

II. Background

Modern SRAM-based FPGAs typically consist of an array of *tiles*, which are interconnected by routing tracks [7]. Each tile consists of a *Configurable Logic Block* (CLB), two *Connection Blocks* (CBs) and a *Switch Block* (SB). A CLB contains N *Basic Logic Elements* (BLEs), which are tightly interconnected by a fully-connected local routing architecture. Inside a BLE, there is a fracturable *Look-Up Table* (LUT) [8], a *Flip-Flop* (FF) and a BLE output selector (2:1 multiplexer). CBs connect routing tracks to CLB inputs, while SBs interconnect routing tracks.

Advancements in RRAM technology have attracted intensive research efforts on replacing SRAM-based routing multiplexers with RRAM-based implementations [1]–[6]. When a RRAM is programmed to LRS/HRS, it propagates/blocks signals, similarly to a transmission gate in *on/off* state. Thanks to its non-volatility, a RRAM can combine the functionalities of a transmission gate and a SRAM into one device. More details about RRAM technology can be found in [9]. Previous works [1]–[6] predicted that using RRAM-based multiplexers, FPGAs can benefit from a 7%-15% area reduction, a 45%-58% performance improvement, and a 20%-58% power saving, as compared to SRAM-based counterparts.

However, these improvements are obtained by simply replacing the SRAM-based multiplexers in classical FPGA architectures with RRAM-based multiplexers. Very limited work studies the potential of novel RRAM-based FPGA architecture that exploits the features of RRAM-based multiplexers. In addition, most RRAM-based researches overlook the challenges in physical designs, i.e., consider a ideal RRAM, which may cause architecture-level improvements less realistic. Recent work [6] has carefully studied the physical design details of RRAM programming structures, and proposes an efficient 4T(ransistor)1R(RAM) programming structure. Therefore, it is worthy to investigate the architectural enhancements for RRAM-based FPGAs, by considering realistic RRAM-based multiplexer designs.

978-1-5090-5859-4/17/$31.00 © 2017 IEEE

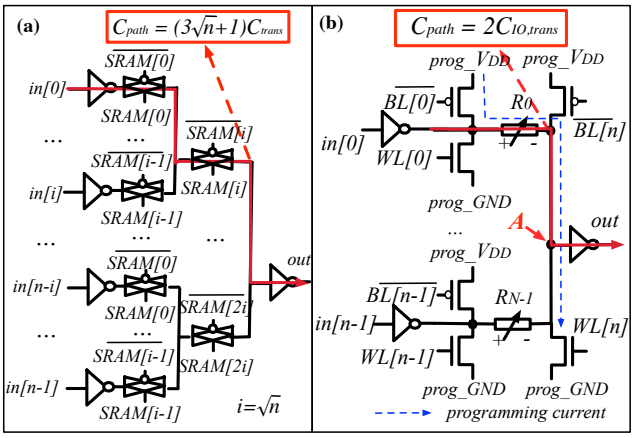

Fig. 1. Schematics and path capacitances of (a) SRAM-based and (b) RRAM-based multiplexers.

TABLE I. Delay comparison between SRAM-based and RRAM-based routing multiplexers.

MUX Location	Input size	Fan-out	SRAM MUX (ps)	RRAM MUX (ps)	Improv-ements
Local Routing	80	1	57.7	19.2	67%
BLE output selector	2	70	38.8	42.2	-11%
CB	48	60	76.0	48.2	36%
SB	4	124[1]	57.8	49.6	14%

* Output buffers are considered and sized according to the fan-outs
[1]The fanout includes the parasitics of a long routing metal wire.

III. Advantages of RRAM-based Routing Multiplexers

Fig. 1(a) and (b) illustrate the schematics of SRAM-based and RRAM-based multiplexers, respectively. We consider a two-level structure for the SRAM-based multiplexers because it guarantees the best area-delay product compared to one-level or multi-level structures [10]. The RRAM-based multiplexer is built with a one-level structure and 4T1R programming elements exploiting I/O transistors [6]. Note that, in a RRAM-based multiplexer, programming structures are efficiently shared by the RRAMs. As a result, the total capacitance on the critical paths of a RRAM-based multiplexer is significantly smaller than a SRAM-based multiplexer, especially when n is large, as highlighted in Fig. 1. Table I compares the SPICE-simulated delay of SRAM-based and RRAM-based multiplexers by considering realistic sizing and loads in their FPGA architectural context. In this paper, we consider a commercial CMOS 40nm technology, whose nominal working voltage is $V_{DD} = 0.9V$. We use the Stanford RRAM model [14], which physically models an ideal memory, with the following parameters: $R_{LRS} = 2k\Omega$, $R_{HRS} = 27M\Omega$, $I_{set} = 500\mu A$, $V_{set} = V_{reset} = 1.2V$, which are sufficient to guarantee that the RRAM-based circuits are as power efficient as SRAM-based circuits [5]. In high fan-in and low fan-out condition, where the delay is dominated by the multiplexing structure, RRAM-based multiplexer can achieve 67% reduction in delay, thanks to its smaller capacitances. In contrast, when fan-in is low and fan-out is high, the delay is dominated by the output buffer and the RRAM-based multiplexer has a moderate 11% performance decrease. This outstanding performance in high fan-in and low fan-out motivates us to study which FPGA architectural choices will be influenced by the features of RRAM-based multiplexer. Circuit design details of RRAM-based multiplexers were partly covered in [6] and are out of the scope of this paper.

IV. Proposed RRAM-based Routing Architecture

In this part, we propose three architectural enhancements of RRAM-based FPGAs exploiting the advantages of RRAM-based multiplexers.

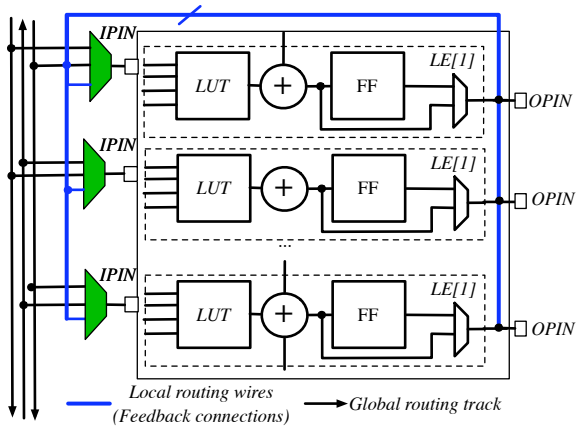

Fig. 2. Proposed CB and CLB architectures.

A. Enhancement 1: A Unified Connection Block

In SRAM-based FPGA architectures, a routing track has to pass through a CB multiplexer and a local routing multiplexer before reaching a LUT input. Such routing architecture efficiently reduces the number of CB multiplexer to be used but requires tapered buffers at the outputs of the CB multiplexers, in order to drive the high fan-out of the local routing. The use of large tapered buffers potentially increases the delay from a routing track to a LUT input. This situation is extremely inefficient for RRAM-based FPGAs since a tapered buffer may be less delay efficient than the RRAM-based multiplexer itself. Therefore, we propose that RRAM-based FPGA should use a one-level RRAM-based crossbar to provide interconnections between routing tracks and LUT inputs, as illustrated in Fig. 2. Note that feedback connections are also resolved by the unified connection block. The proposed routing architecture is well suited to RRAM-based multiplexers for three reasons: (a) Each CB multiplexers now has a unique fan-out, and tapered buffers can be avoided; (b) Both routing and feedback delays can be significantly reduced because only one large multiplexer interconnects a routing track to a LUT input; (c) The number of inputs of a CLB is increased, which can potentially lead to a total area reduction even for SRAM-based FPGAs [11].
The proposed routing architecture requires to redefine the best fraction of routing tracks that can be reached by each CB multiplexer, $F_{c,in}$. Note that in the classical architecture ($F_{c,in} = 0.15$), all the nets mapped to the inputs of a CLB are different because the local routing can connect a net from a CLB input to multiple LUTs. The proposed architecture may have a net mapped to multiple CLB inputs due to the absence of local routing. Therefore, we need to increase $F_{c,in}$ to allow more CLB inputs to be reached by a single routing track, to compensate the potential loss in routability. In an FPGA tile, the LUT inputs are connected to the right and bottom sides of a CLB. Each LUT has $K/2$ input connected to the right/bottom side of a CLB. To ensure that different LUT inputs can be connected from a common routing track, $F_{c,in}$ should be at least $2/K$. Fig. 3 depicts such an example when $K = 6$. Input $in0$ of $LUT0$ and input $in0$ of $LUT1$ can be reached by the same track $Track0$. Note that any two inputs of the same LUT do not share a routing track.

B. Enhancement 2: Increase Capacity of SB MUXes

Since a RRAM-based multiplexer is more delay-efficient than a SRAM-based multiplexer, the connection flexibility parameter of *Switch Block* (SB) F_s can be increased. In SRAM-based FPGAs, $F_s = 3$ promises the best area-delay product [12], where each routing track on one side of a SB can reach three other routing tracks on different sides of a SB. Indeed, a larger F_s can improve the routability but it may produce area and delay overhead coming from the larger SB

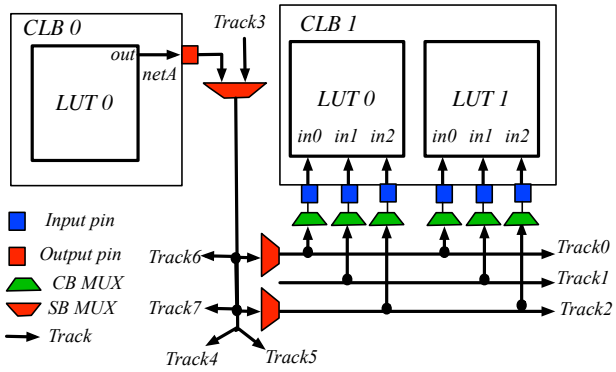

Fig. 3. An illustrative example of the proposed routing architecture($K = 6$) with $F_{c,in} = 0.33$ and $F_s = 6$.

multiplexers. However, considering RRAM-based routing architecture, the delay overhead is no longer a concern thanks to the advantage of the RRAM-based multiplexers. Therefore, a larger F_s, i.e. $F_s = 6$, can considered, where a routing track can drive six different tracks. Fig. 3 also illustrates such an example where $netA$ is routed through with $Track3$. If $F_s = 3$, $Track3$ can only drive $Track0$, $Track4$ and $Track6$. When $Track0$ is not available, the output of $LUT0$ has to seek for another routing track by increasing the channel width. If $F_s = 6$, $Track3$ can reach both $Track0$ and $Track2$. When $Track0$ is occupied by another net, $Track3$ can easily use $Track2$ to route $netA$.

C. Enhancement 3: Smaller Best Length Wire < 4

In FPGA architectures, a length-L wire is a wire that spans across L CLBs [7]. When only one type of wires is allowed to be used in an FPGA, the type of length-L wires that produces best area-delay product is called the *best single wire length*. Commercial FPGAs typically provide different types of wires, i.e., length-1 for short connections and length-8 for long connections. However, the best single wire length is useful to decide which type of wires should be predominant within the architecture. Length-4 wires are the best choice for classical SRAM-based FPGA architectures ($F_{c,in} = 0.15, F_s = 3$) [7]. Length-4 wires can produce the best performance on average because the delay of a SB multiplexer is higher than a long metal wire across a logic block. However, RRAM-based multiplexers are more delay efficient and can be even faster than a long metal wire, potentially twisting the cost function. We use Elmore delay [13] to estimate the delay per logic block of a Length-L wire:

$$T_{delay,wire}/L = \frac{1}{L}\sum_{i=0}^{L-1} R_i \sum_{j=i}^{L-1} C_j$$
$$= L \cdot \frac{R_m C_m}{2} + \frac{1}{L} \cdot (T_{del} + R_o C_o - 2R_m C_{SB} - 2R_m C_{CB})$$
$$+ R_m(C_{SB} + C_{CB} - C_m) + R_o(C_m + C_{SB} + C_{CB})$$

where R_m and C_m are the resistance and capacitance of a metal wire spanning a logic block, respectively; T_{del} represents the intrinsic delay of a SB multiplexer; R_o and C_o denote the equivalent resistance and capacitance of the tapered buffer that drives the metal wire, respectively; C_{SB} and C_{CB} are the equivalent input capacitance of each SB and CB, respectively. In the proposed RRAM-based routing architecture, where T_{del} decreased thanks to the RRAM-based multiplexer, $T_{delay,wire}/L$ will definitely decrease. Therefore, the best single wire length of the proposed routing architecture will be smaller than 4.

V. Experimental Results

In this part, we first introduce the experimental methodology and then study the effect of each architectural enhancements.

TABLE II. Average area, delay, power, channel width and *Area-Delay Product* (ADP) improvements of the proposed SRAM-based and RRAM-based FPGA architectures for sweeping $F_{c,in}, F_s$ and L.

(a) Sweeping $F_{c,in}$

SRAM FPGA[1]	Area	Delay	Power	Chan. W	ADP
$F_{c,in} = 0.25$	6%	-4%	13%	43%	3%
$F_{c,in} = 0.33$	5%	-1%	10%	-27%	5%
$F_{c,in} = 0.5$	-9%	-8%	10%	-20%	-18%
RRAM FPGA[2]	Area	Delay	Power	Chan. W	ADP
$F_{c,in} = 0.25$	12%	-8%	14%	-43%	5%
$F_{c,in} = 0.33$	12%	3%	6%	-30%	15%
$F_{c,in} = 0.5$	2%	-9%	8%	-22%	-7%

(b) Sweeping F_s

SRAM FPGA[1]	Area	Delay	Power	Chan. W	ADP
$F_s = 3$	5%	-1%	10%	-27%	5%
$F_s = 6$	8%	5%	9%	-9%	12%
$F_s = 9$	5%	-5%	-4%	10%	1%
RRAM FPGA[2]	Area	Delay	Power	Chan. W	ADP
$F_s = 3$	12%	3%	6%	-30%	15%
$F_s = 6$	11%	7%	6%	-13%	17%
$F_s = 9$	7%	8%	4%	-7%	14%

(c) Sweeping L

SRAM FPGA[1]	Area	Delay	Power	Chan. W	ADP
$L = 1$	-6%	-20%	7%	25%	-13%
$L = 2$	11%	3%	8%	11%	14%
$L = 4$	8%	5%	10%	-9%	12%
RRAM FPGA[2]	Area	Delay	Power	Chan. W	ADP
$L = 1$	5%	-1%	-4%	23%	3%
$L = 2$	15%	11%	0%	13%	23%
$L = 4$	11%	7%	6%	-13%	17%

[1] Baseline is the classical SRAM-based arch.
[2] Baseline is the classical RRAM-based arch.

A. Methodology

All the investigated tile-based FPGA architectures follow a Stratix IV-like CLB architecture [15], featured by $K = 6, N = 10, F_{c,out} = 0.1$ and $W = 320$. All the baseline architectures have 40 inputs for each CLB ($I = 40$). Because the local routing is removed in the proposed architecture, we provide 60 inputs for each CLB ($I = K \cdot N = 60$). We use the VTR flow [17] to evaluate the area, delay, power and channel width of the investigated FPGA architectures. The largest twenty MCNC benchmarks [16] are optimized at the logic level by ABC [18] and then packed, placed and routed by VPR7 [17]. We add a 30% slack to the minimum routable channel width W_{min}, in order to simulate a low-stress routing [7]. For a fair comparison, the maximum routing iterations are set to 50 for the classical architecture. Since our proposed architecture removes local routing and more nets have to be routed by the global router, we use 100 routing iterations. Instead of directly comparing between SRAM-based and RRAM-based FPGAs, we focus on studying the effects of the proposed architectural enhancements on both SRAM-based and RRAM-based FPGAs. Both SRAM-based and RRAM-based implementations of the proposed architecture are investigated and their benefits are examined by comparing to the baseline architectures implemented with SRAMs and RRAMs, respectively. For each architectural enhancement, we study its impact on area, delay, power and channel width. Since each architectural enhancement involves different routing architecture parameters, such as $F_{c,in}$, F_s and L, for a fair comparison, we vary a single parameter in each comparison and find a reasonable value for each parameter. Once we find the best value of one parameter, we set it to this value and vary another. We believe that such methodology helps to identify where RRAM FPGAs can be improved beyond SRAM FPGAs.

B. Impact of the Unified Connection Block and Best $F_{c,in}$

In Table II(a), we sweep $F_{c,in} = \{0.15, 0.25, 0.33, 0.5\}$ to examine its best value for the proposed architecture in both SRAM-based and RRAM-based context. The SRAM-based proposed architecture with $F_{c,in} = 0.33$ produces a slightly better area-delay product (-5%) than the classical architecture, but performs worse (+1%) in delay. In

contrast, the RRAM-based proposed architecture with $F_{c,in} = 0.33$ reduces delay by 3% and area-delay product by 15%, when compared to the classical architecture. In either SRAM-based or RRAM-based FPGAs, the proposed architecture with $F_{c,in} = 0.33$ produces the best ADP. Note that we see a 5%-12% area reduction in both SRAM-based and RRAM-based proposed architectures when $F_{c,in} = 0.33$, which is close to the conclusion of literature [11]. Power and channel width overheads are observed in both SRAM-based and RRAM-based proposed architectures, because their routability is lower than their baselines due to the absence of local routing. However, these overheads can be potentially eliminated because the routability can be significantly improved when we increase F_s and decrease L. In terms of the best ADP, we consider $F_{c,in} = 0.33$ for the proposed FPGA architectures in the rest of this paper.

C. Impact of an Increased F_s

In Table II(b), we sweep $F_s = \{3, 6, 9\}$ to determine its best value for the proposed architecture in both SRAM-based and RRAM-based context. The proposed RRAM-based architectures can benefit larger delay reduction (-7%) than SRAM-based (-5%), because RRAM-based multiplexers are more delay efficient for the unified connection block. However, $F_s > 3$ introduces larger SB multiplexers, which potentially increase the area of both SRAM-based and RRAM-based proposed architectures. Therefore, $F_s = 6$ produces the best ADP for both SRAM-based and RRAM-based proposed architectures. Note that, even when $F_s = 9$, RRAM-based proposed architecture leads to a 8% delay reduction thanks to its RRAM-based multiplexer, while, the SRAM-based proposed architecture has a 5% delay overhead. As a large F_s boosts the routability, a 20% channel width reduction is achieved in both SRAM-based and RRAM-based proposed architectures, as compared to those with $F_s = 3$. In terms of the best ADP, we consider $F_s = 6$ for the proposed FPGA architectures in the rest of this paper.

D. Impact of a Decreased L

In Table II(c), we sweep $L = \{1, 2, 4\}$ to determine its best value for the proposed architecture in the context of SRAM-based and RRAM-based topologies. In SRAM-based architectures, whatever F_s is, length-4 wires achieve the best delays. However, with the RRAM-based architectures, the proposed architecture with length-2 wires promises the best delay (-11%) and also the best ADP (-23%). As L is reduced from 4 to 2, we see a 23% channel width reduction because short wires are more flexible and promise better routability. Length-4 wires guarantee the best power results but requires large area, due to the lower flexibility and the larger channel width. Conversely, length-1 wires have the smallest channel width but more SB multiplexers have to be used in long routing paths. Therefore, we see significant area and power overhead. In terms of the best ADP, $L = 2$ is the best single wire length for the proposed FPGA architecture.

In summary, we determine that $F_{c,in} = 0.33, F_s = 6, L = 2$ produce the best performances for the proposed FPGA architecture. As shown in Fig. 4, the proposed RRAM-based FPGA at nominal voltage improves area by 26%, delay by 39%, and channel width by 13% without any power consumption overhead, as compared to a classical SRAM-based FPGA. Furthermore, when operated in the near-V_t regime, the proposed RRAM-based FPGA at $V_{DD} = 0.7$ can achieve 42% and 5× improvement on *Area-Delay Product* and *Power-Delay Product* respectively, as compared to a classical SRAM-based FPGA running at a nominal voltage.

VI. Conclusion

In this paper, we first addressed the unique advantage of RRAM-based routing multiplexers: their ideal delay is independent from input size. To exploit this advantage, we propose three architectural enhancements for RRAM-based FPGAs: (a) the traditional CB and local routing are replaced with a unified CB, leading to ultra-fast interconnection from routing tracks to LUT inputs; CB connectivity parameter $F_{c,in}$ should be at least 0.33 to ensure routability. (b) the SB connectivity parameter F_s can be increased to achieve routability improvements without delay overhead; (c) the best single wire length L is reduced, leading to better routability. We study the best values of $F_{c,in}$, F_s and L in terms of area, delay, power and channel width. Experimental results show that a RRAM-based FPGA with architectural enhancements should employ ($F_{c,in} = 0.33$, $F_s = 6$ and $L = 2$) to achieve best ADP. When operated at nominal voltage, the proposed RRAM-based FPGA architecture reduces area by 26%, delay by 39% and channel width by 13%, as compared to a SRAM-based FPGA with a classical architecture. When operated in the near-V_t regime, the proposed RRAM-based FPGA architecture improves *Power-Delay Product* by 5× as compared to a classical SRAM-based FPGA at nominal voltage.

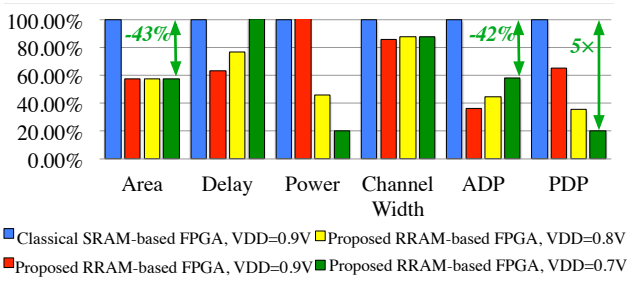

Fig. 4. Normalized average area, delay, power, channel width, ADP and PDP of classical SRAM-based and proposed RRAM-based architectures.

Acknowledgment

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

References

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

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

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

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

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

[6] X. Tang *et al.*, *A Study on the Programming Structures for RRAM-Based FPGA Architectures*, IEEE TCAS-I, Vol. 63, No. 4, 2016, pp. 503-516.

[7] V. Betz *et al.*, *Architecture and CAD for Deep-Submicron FPGAs*, Springer Publishers, 1998.

[8] M. Hutton *et al.*, *Improving FPGA Performance and Area Using an Adaptive Logic Module*, IEEE FPL, 2004, pp. 135-144.

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

[10] E. Lee *et al.*, *Interconnect Driver Design for Long Wires in Field-Programmable Gate Arrays*, JSPS, Vol. 51, No. 1, April 2008.

[11] W. Feng *et al.*, *Designing Efficient Input Interconnect Blocks for LUT Clusters Using Counting and Entropy*, ACM TRETS, Vol. 1, No. 1, Article 6, March 2008.

[12] G. Lemieux *et al.*, *Directional and Single-Driver Wires in FPGA interconnect*, ICFPT, 2004, pp. 41-48.

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

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

[15] Altera Corporation, *Stratix IV device handbook version SIV5V1-1.1*, July 2008.

[16] S. Yang, *Logic Synthesis and Optimization Benchmarks User Guide Version 3.0*, MCNC, Jan. 1991.

[17] J. Rose *et al.*, *The VTR Project: Architecture and CAD for FPGAs from Verilog to Routing*, FPGA, 2012, pp. 77-86.

[18] University of California in Berkeley, *ABC: A System for Squential Synthesis and Verification*, http://www.eecs.berkeley.edu/~alanmi/abc/