2016 11th International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS)

Boolean Logic Gate Exploration for Memristor Crossbar

Lei Xie, Hoang Anh Du Nguyen, Mottaqiallah Taouil, Said Hamdioui, Koen Bertels
Laboratory of Computer Engineering, Delft University of Technology, the Netherlands
Email: {L.Xie, H.A.DuNguyen, M.Taouil, S.Hamdioui, K.L.M.Bertels}@tudelft.nl

***Abstract*—Emerging technologies are under research as alternatives for next-generation VLSI circuits. One of the promising candidates is memristor due to its scalability, high integration density, non-volatility, etc. Different design styles of memristor-based logic circuits have been proposed. This paper first overviews these design styles and compares them using several criteria. Subsequently, it selects a promising candidate to explore its potential logic gate space. Thereafter, it derives control voltage constraints used to ensure correct logic gate functionality. The newly obtained logic gates are verified by SPICE simulations, and finally the performance of the memristor gates are compared with CMOS gates. The results show that the memristor gates, with reasonable technology improvements, are comparable to CMOS gates or even outperform them.**

I. Introduction

As CMOS technology gradually scales down to its intrinsically physical device limits, it faces major challenges [1] such as saturated performance enhancement, increased leakage power consumption, reduced reliability, etc. To address such challenges, new technologies (e.g., memristors, nanotube, graphene transistors, etc. [2]) have been proposed as alternatives for next-generation VLSI circuits. Among these technologies, the memristor is one of the promising candidates. Massive memristors can be mapped on a crossbar architecture, where memristors are located between horizontal and vertical nanowire junctions [3]. The memristor crossbar is able to provide great scalability, high integration density, non-volatility, etc [3]. Several potential applications have been proposed including non-volatile memory [2,3], neuromorphic circuits [2,3] and novel computing paradigms for data-intensive applications (e.g., computation-in-memory [4,5]). To realize novel computing paradigms, it is pivotal to design fundamental logic circuits/gates.

Four types of memristor-based logic circuits have been proposed: (i) threshold logic [6], (ii) majority logic [6], (iii) material implication logic [7], and (iv) Boolean logic [8]; each logic type consist of different design styles. Among them, Snider Boolean logic (SBL) [9] is a promising candidate for memristor crossbar as shown later in this paper. The author of SBL proposed three primitive operations: copy, inversion and NAND. However, these operations are only a subset of all the possible logic gates that can be implemented by SBL. In addition, the author did not notice that these logic gates only function correctly when the applied control voltages satisfy certain constraints.

This paper explores all the possible logic gates for SBL. Thereafter, it derives the control voltage constraints by formulating the switching conditions of memristors during operation. The main contributions of this paper consist of:

- A brief overview of existing memristor-based logic circuits.
- The logic gate space exploration for SBL.
- The derivation of the control voltage constraints to ensure correct functionality of logic gates.
- A comparison between memristor and CMOS logic gates.

The remainder of this paper is organized as follows. Section II overviews memristor logic circuits and selects SBL as a promising candidate. Section III explores the SBL logic gate space. Section IV derives the control voltage constraints. Section V verifies new memristor gates and compares memristor and CMOS gates. Finally, Section VI concludes the paper.

II. Overview of Memristor Logic Circuits

This section first overviews the-state-of-art of memristor logic circuits. Thereafter, it selects SBL as a promising candidate to implement memristor crossbar logic circuits.

The memristor logic circuits are briefly overviewed by considering criteria related to *technology and design*.

Two **technology** criteria are considered: crossbar compatibility (Xbar) and computing technology requirement (Cmp.Tech.). Several memristor logic circuits have been proposed in particular for the crossbar architecture and therefore, they are compatible with crossbar (Y); in contrast, others may not be compatible (N). The memristor logic circuits consist of a computing (e.g., logic gates) part, and a CMOS control part (e.g., control logic, clock, etc.). Some logic circuits require only memristors (M) for the computing part, while others require both CMOS and memristors (CM). This is captured by computing technology requirement.

The considered **design** criteria are:

Logic Type: The logic type specifies the kinds of operations that are performed; e.g., Boolean logic includes operations such as AND, OR, etc. Five logic types have been implemented with memristors; they are Boolean logic (Bool), implication logic (IMP), threshold logic (TH), majority logic (MAJ) and hybrid logic (Hyb.). In hybrid logic, different logic types are merged (e.g., Boolean and implication logic are combined in [18]).

978-1-5090-0336-5/16/$31.00 ©2016 IEEE

TABLE I: Features of Memristor-Based Logic Circuits

Ref	Name	Abbr.	Technology		Design						
			Xbar	Cmp. Tech.	Logic Type	Logic Gate	Data Sig.	Usage	Syn.	No. Step	NV
[8]	Hybrid Transistor/Memristor Boolean Logic	HTMBL	N	CM	Bool	NAND	V	CFG	AS	S	N
[10]	Memristor Ratioed Logic	MRL	N	CM	Bool	AND,NAND,OR,NOR	V	CMP	AS	S	N
[11]	mLogic	mLogic	N	M	Bool	AND,NAND,OR,NOR,NOT	C	CMP	CLK	M	Y
[12]	Domain Wall Logic	DWL	N	M	Bool	AND,OR,NOT	C	CMP	CLK	M	Y
[6]	Memristive Programmable Logic Array	MPLA	Y	CM	Bool	AND,OR,NOT	V	CFG	AS	S	N
[13]	CMOS-like Configurable Memristor Logic	CCML	Y	M	Bool	AND,NAND,OR,NOR,NOT	V	CMP	AS	S	Y
[14]	Stateful Logic Pipeline Architecture	SLPA	Y	M	Bool	OR,NOR	R	CMP	CTR	M	Y
[9]	Snider Boolean Logic	SBL	Y	M	Bool	CPY,NOT,NAND	R	CMP	CTR	M	Y
[15]	BRS/CRS Crossbar Logic	BCCL	Y	M	IMP	CIMP,NIMP	V	CMP	CTR	M	Y
[7]	Material Implication Logic	MIL	Y	M	IMP	IMP	R	CMP	CLK	M	Y
[6]	Charge Sharing Threshold Gate	CSTG	N	CM	TH	TG	V	MEM	AS	S	N
[6]	Current Mirror Threshold Gate	CMTG	N	CM	TH	TG	V	MEM	AS	S	N
[16]	Programmable CMOS/Memristor Threshold Logic	PCMTL	N	CM	TH	TG	V	MEM	CLK	S	N
[6]	Memristive Programmable Threshold Logic Array	MPTLA	Y	CM	TH	TG	V	MEM	CLK	S	N
[17]	All Spin Logic	ASL	N	M	MAJ	MAJ,CPY,NOT	C	CMP	CLK	M	Y
[6]	Memristive Programmable Majority Logic Array	MPMLA	Y	CM	MAJ	MAJ	V	MEM	CLK	S	N
[18]	Memristive Stateful Logic	MSL	Y	M	Hyb.	IMP,CNIMP,AND,OR	R	CMP	CTR	M	Y

Logic Gate: Logic gates compute primitive operations used to build complex digital functions. For Boolean logic, the basic operations are AND, OR, NAND, NOR, and NOT. Note that not all the Boolean logic circuits can implement all gates. For implication logic, implication (IMP, $f{=}\overline{A}{+}B$ where A and B are inputs, and f the output), converse implication (CIMP, $f{=}A{+}\overline{B}$), nonimplication (NIMP, $f{=}A{\cdot}\overline{B}$), and converse nonimplication (CNIMP, $f{=}\overline{A}{\cdot}B$) gates have been proposed [19]. Threshold logic uses threshold gates (TG) and majority logic majority gates (MG).

$$f = \begin{cases} 1 & \sum_{i=1}^{n} w_i x_i \geq T \\ 0 & \text{Otherwise} \end{cases} \quad (1)$$

$$f = \begin{cases} 1 & \sum_{i=1}^{n} x_i > (n+1)/2 \\ 0 & \text{Otherwise} \end{cases} \quad (2)$$

Eq.1 and Eq.2 describe a TG and MG, respectively, where n presents the number of inputs, x_i the input, w_i the weight of input x_i, and T the threshold. Finally, some logic circuits may use a copy (CPY) operation to transmit data within the crossbar.

Data Signal (Data Sig.): The data in memristor circuits is presented either by a voltage (V) (i.e., different voltage levels), a current (C) (i.e., different current directions or values) or a resistance (R) (i.e., high or low resistance).

Memristor Usage (Usage): A memristor can be used as a digital computing switch (CMP), a resistive memory (MEM) or a configuration switch for signal routing in programmable logic array (CFG).

Synchronization (Syn.): Some logic circuits use a clock (CLK) or CMOS controller (CTR) for synchronization, while others operate asynchronously (AS).

Number of Steps: Logic circuits may complete the operation in a single (S) step or multiple steps (M).

Non-volatility (NV): The non-volatility indicates whether a logic circuit can store its results when it is powered down.

Table I summarizes the features of each design style. For instance, MRL uses hybrid technology to implement Boolean logic; it uses individual memristors based on a voltage; its basic gates are AND, NAND, OR and NOR gates where each memristor is used as a computing switch; MRL works asynchronously, computes the result within one cycle without having non-volatility. Note that Table I reflects relationships between different features; e.g., most hybrid CMOS/memristor circuits calculate the results within one cycle, while memristor-only circuits require several cycles.

SBL is one of the most promising candidates for memristor crossbar due to the following reasons. Technology-wise, SBL is compatible with memristor crossbar which provides great scalability and high integration density. Next, SBL's computing part is implemented only by memristor (no CMOS logic required) which potentially enables novel computing paradigms [4,15]. Design-wise, SBL is based on Boolean logic; this is preferred as it enables the resue of existing arithmetic designs, IP designs, and EDA flows. In addition, SBL is non-volatile, and therefore, has the potential to support lower-power designs as it can be powered-off when it is not used. Note that other criteria that are not used in this work (e.g., latency) may lead to different outcomes.

III. Logic Gate Space of Snider Boolean Logic

This section first describes the memristor model used by SBL. Thereafter, it presents the existing SBL gates. By investigating these gates, gate parameters are extracted which are subsequently used to explore the potential logic gate space.

A. Memristor Model

The current-voltage relation of the ideal memristor used in SBL [9] is shown in Fig. 1(a). In case the absolute value of the voltage across the device is greater than its threshold voltage V_{th}, the memristor switches from one resistive state to another. Otherwise, it stays in its current resistive state. Typically, a memristor requires two different V_{th} values to switch from high (R_H) to low (R_L) resistance (SET), and low to high resistance (RESET) [20]; see Fig. 1(b) and (c). The black squares at the top edge of the memristors present the positive terminal. For simplicity, we assume that this model has the same absolute V_{th} value for both SET and RESET.

B. Existing SBL Gates

The authors in SBL [9] proposed three different logic gates: copy (CPY), inversion (INV) and NAND as shown in Fig. 2.

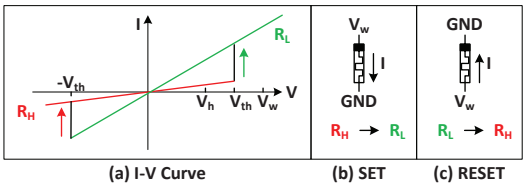

Fig. 1: Memristor model [9].

Each gate consists of one or multiple input memristors and an output memristor; the output memristor (surrounded by a dashed-line box in Fig. 2) is initialized to R_H prior to execution. For brevity, this initialization is not shown. The nanowire that connects the input and output memristors is floating; V_x presents the voltage of this floating nanowire and V_{om} the voltage across the output memristor. Both V_x and V_{om} are used to explain the working principle. In addition, INV and NAND gates requires an extra resistor R_s attached to the floating nanowire as shown in Fig. 2(c) and (e), respectively; R_s must satisfy $R_L \ll R_s \ll R_H$ to guarantee that V_x is close to the desired voltage for correct operation [9].

SBL logic gates use a high (R_H) and low (R_L) resistance to represent logic 1 and 0. To control these gates, three different voltages are required: V_w, V_h, and *GND* (G); see Fig. 1(a). V_w is used to program the resistance of a memristor; V_h is used to control primitive operations (e.g., NAND [9]), or used as half-select voltages to reduce the impact of sneak path currents [21]. The typical relationship between V_w, V_h, G and V_{th} is $0<V_h=\frac{V_w}{2}<V_{th}<V_w<2V_{th}$. This is the minimum requirement to ensure the functionality of logic gates.

CPY is given as an example as other logic gates work in a similar way. Voltages G and V_w are applied to the input and output memristors, respectively. In case the input is 1 (R_H), $V_x{=}V_w/2$, and hence, $V_{om}{=}V_w{-}V_x{=}V_w/2{<}V_{th}$ (see Fig. 2(a)). As a result, the output memristor stays at 1. In case the input is 0 (R_L), $V_x{\approx}0$ and $V_{om}{\approx}V_w{-}V_x{\approx}V_w{>}V_{th}$ (see Fig. 2(b)). As a result, the output memristor switches to 0.

C. Gate Parameters

To explore the logic gate space of SBL, several gate parameters are extracted from the SBL gates. These parameters are desribed next.

Structure: From the CPY and INV gates of Fig. 2(a) and (c), it is observed that the gates have a different structure due to R_s; R_s can be regarded as an extra terminal of the schematic. Hence, the schematic without R_s (see for example CPY in Fig. 2(a)) is referred to as *2 terminal (2T)*, while the schematic with R_s is referred to as *3 terminal (3T)* (e.g., INV gate in Fig. 2(c)).

Control Voltage: From the CPY and INV gates of Fig. 2(a) and (c), it is observed that the gates are controlled with different voltages; for instance, the *GND* and V_h are applied to input memristors of CPY and INV, respectively. The control voltages impact the functionality of the logic gates. For instance, Fig. 2(g) and (h) swap the input and output control voltages with respect to Fig. 2(a). As a result, this gate is not able to copy data from the input to output

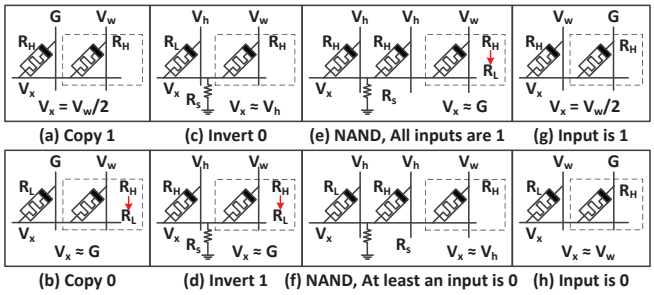

Fig. 2: Basic Operations

memristors. In general, we denote the control voltages using the tuple (V_{ci},V_{co}), where V_{ci} and V_{co} present the voltage applied to the input and output memristors, respectively; e.g., the control voltages of CPY in Fig. 2(a) is denoted by (G,V_w).

Fan-in (FI): From the INV and NAND gates of Fig. 2(c) and (e), it is observed that the fan-in (the number of inputs) leads to different functions; INV has a single fan-in (*SFI*), while NAND a multi fan-in (*MFI*). The number of inputs is denoted by n_i.

Fan-out (FO): All the existing SBL logic gates have only one output memristor; these gates are referred to as single fan-out (SFO) gates. Here, we extend the SFO gates to multi fan-out (MFO) gates by applying the same control voltages to multiple output memristors. The number of outputs is denoted by n_o.

Polarity: All input and output memristors of the existing SBL gates have their negative terminal connected to the floating nanowire, which is referred to as *negative polarity* (see also Fig. 2). Here, we also explore the case where all the positive termnals of the input and output memristors are connected to the floating nanowire; this is referred to as *positive polarity*. We limit the exploration to the case where either all positive or all negative terminals of both input and output memristors are connected to the vertical nanowires for manufacturing reasons [21].

Data Representation: All existing SBL gates use R_H and R_L to represent a logic 1 and 0, which are referred to as an *RH1* data representation. Here, we will also explore the case when R_H is used to represent logic 0 and R_L logic 1, which are referred to as *RH0*.

Based on the above gate parameters, we propose generic gate schematics for SBL as shown in Fig. 3(a) for 2T and (b) for 3T. In the figure, $M_{i,k}$ ($1{<}k{<}n_i$) and $M_{o,p}$ ($1{<}p{<}n_o$) present the resistances of the input and output memristors, respectively. These two schematics are simplified by their equivalent circuits obtained through Kirchhoff's current law; note that all the input/output memristors are driven by the same control voltages V_{ci}/V_{co}. The results are depicted in Fig. 3(c)-(d) where $M_{i,eq}$ and $M_{o,eq}$ present the equivalent resistance of input and output memristors, respectively. The equivalent circuits will be used to derive control voltage constraints in section IV.

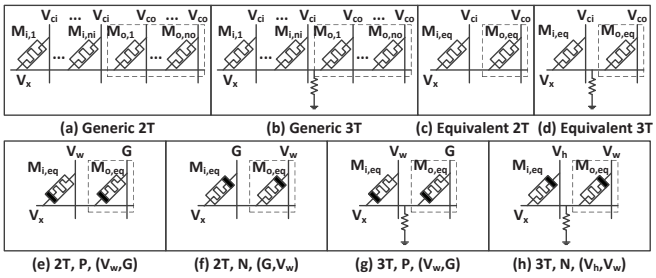

Fig. 3: 2T and 3T Logic Gates.

D. Logic Gate Space

The logic gate space is explored through an exhaustive search of all the gate parameters. The SFI/SFO gate space is first explored, and subsequently extended for MFI/MFO gates.

The SFI/SFO logic gate space is explored by using the following gate parameters: structure, polarity and control voltages; data representation is not considered (e.g., is set to RH1) as SFI/SFO logic gates can only either be CPY or INV gates; therefore, the data representation (RH1 or RH0) has no impact on them. This leads to total 24 (2 structures x 2 polarities x 3! control voltages) combinations. Among all the combinations, only four combinations are useful as shown in Fig. 3 (e)-(h); the first three gates in Fig. 3 (e)-(g) are CPY gates, while the last one in Fig. 3 (h) is an INV gate. Note that although no new functions are formed, our approach introduces two novel CPY gate implementations as shown in Fig. 3 (e) and (g). All the other combinations of the gate parameters ended up in useless gates, such as a gate shown in Fig. 2(g)-(h) where the output is always 0.

The MFI/MFO gate space is derived from the SFI/SFO gate space by extending it with the remaining gate parameters (i.e., fan-in, fan-out and data prepresentation) that are not considered in the SFI/SFO gate space exploration. Note that the fan-out does not impact the gate operations, whereas fan-in and data representation may do so. For instance, by *only* changing the SFI of the INV gate of Fig. 2(c) to a MFI gate the NAND gate of Fig. 2(e) is obtained. This can be explained by the parallel input memristors and their equivalent resistance $M_{i,eq}$; it can be either a relatively high resistance ($=\frac{R_H}{n_i}$) when all inputs are 1, or a relatively low resistance ($\approx\frac{R_L}{n_{iL}}$) when n_{iL} inputs are 0. From this, we observe that the parallel input memristors behave like an AND gate. Therefore, an AND gate can be realized by replacing the single fan-in of CPY in Fig. 4(b) with a multi fan-in. Similarly, in the case of RH0, the multi fan-in introduces OR and NOR gates. Note that the OR and NOR gates are the same as the AND and NOR gates except that they differ in data representation.

Fig. 5 shows the explored gate space with their gate parameters. These gates support basic computation and communication. In the figure, the original SBL gates proposed in [9] are highlighted by boxes. Compared to the original SBL, our approach (i) identifies new gates (such as AND, OR, NOR gates), (ii) shows different ways to design them (using 2T or 3T structure and different polarities), and (iii) introduces multi fan-out gates. Overall, these new gates have the potential to design logic circuits more efficiently than the original SBL gates.

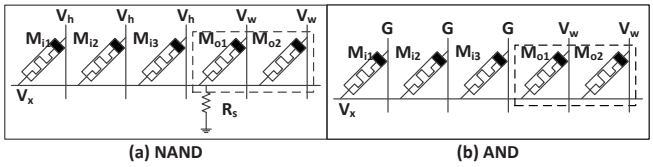

Fig. 4: Multi Fan-out Logic Gates.

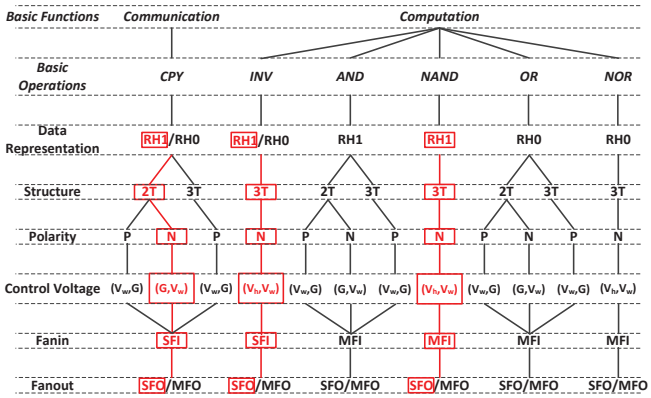

Fig. 5: Logic Gate Space.

IV. Constraints of Control Voltages

This section formulates the control voltage constraints of all the logic gates of Fig. 5. The logic gates only function correctly when their control voltages are in certain range. The voltage constraints are derived from the switching conditions of both input and output memristors. We assume that each memristor behaves as a resistor at each moment in time and ignore the resistance and capacitance of the nanowires.

The functionality of the gate is guaranteed when the switching conditions of the input and output memristors are satisfied. For all the logic gates, the input memristors should stay in their current resistive state independent of the gate function. The output memristors, which are initialized to R_H prior to evaluation, either stay at R_H or switch to R_L depending on the input values and gate type.

To illustrate the derivation of constraints, as an example we discuss the AND gate with the following gate parameters: RH1, 2T, P, MFI and MFO (see also Fig. 5). Fig.3(e) shows the equivalent circuit of this AND gate. In case all the inputs are 1 (R_H), the outputs of the AND gate must remain 1 (R_H) and $M_{i,eq}=\frac{R_H}{n_i}$ and $M_{o,eq}\approx\frac{R_H}{n_o}$. To ensure that all the input and output memristors stay logic 1 (R_H), the voltage across the input (V_{im}) and output (V_{om}) memristors should be less than V_{th} as shown in Eq.3.

$$V_{im} = V_x - V_{ci} = V_x - V_w < V_{th} \qquad (3)$$
$$V_{om} = V_x - V_{co} = V_x - 0 < V_{th}$$

In Eq.3, only the voltage of the floating nanowire V_x is unknown. To express V_x, Kirchhoff's current law is applied to

TABLE II: Control Voltage Constraints

2T Structure, Positive/Negative Polarity
$1+\frac{1}{r_R}\frac{n_o}{n_{iL}} < r_{wt} < min\{1+\frac{n_o}{n_i}, 1+\frac{n_{iL}}{n_o}\}$
3T Structure, Positive Polarity
$1+\frac{1}{r_R}\frac{n_o}{n_{iL}}+\frac{r_{Ls}}{n_{iL}} < r_{wt} < min\{1+\frac{n_o}{n_i}+\frac{r_{Hs}}{n_i}, 1+\frac{n_{iL}}{n_o+r_{Ls}}\}$
3T Structure, Negative Polarity
$1+\frac{n_i r_{ht}+n_o}{n_i+r_{Hs}} < r_{wt} < min\{1+\frac{r_R n_{iL} r_{ht}+n_o}{r_R n_{iL}+r_{Hs}}, 1+\frac{n_i}{n_o}+\frac{r_{Ls}}{n_o}+(1+\frac{r_{Ls}}{n_o})r_{ht}\}$

the circuit of Fig. 3(e), and solved for V_x as shown in Eq.4.

$$\frac{V_w - V_x}{M_{i,eq}} = \frac{V_x - 0}{M_{o,eq}} \Rightarrow V_x = \frac{M_{o,eq}V_w}{M_{o,eq}+M_{i,eq}} \quad (4)$$

Therefore, V_{im} and V_{om} can be rewritten as:

$$\Rightarrow V_{im} = -\frac{M_{i,eq}V_w}{M_{i,eq}+M_{o,eq}}; V_{om} = \frac{M_{o,eq}V_w}{M_{o,eq}+M_{i,eq}} \quad (5)$$

According to Eq.5, the condition V_{im}<V_{th}, which specifies that the input memristors must stay R_H, is always true as V_{im}<0<V_{th}. The condition V_{om}<V_{th} for the output memristors is rewritten in Eq.6 by substituting $M_{i,eq}$=$\frac{R_H}{n_i}$ and $M_{o,eq}$=$\frac{R_H}{n_o}$, respectively.

$$V_{om} = \frac{n_i}{n_i+n_o}V_w < V_{th} \quad (6)$$

In case at least one input of the AND gate is 0 (R_L), four conditions must be satisfied. All the inputs must stay at their current resistive states before and after the outputs switch, and the outputs must switch from logic 1 to 0 and stay at logic 0 after switching. Before the outputs switch, $M_{i,eq}$≈$\frac{R_L}{n_{iL}}$, $M_{o,eq}$=$\frac{R_H}{n_o}$ and V_x is obtained by substituting these equivalent resistances in Eq.4. To ensure that the input memristors stay at their current resistive states, $-V_{th}$<V_{im}<V_{th} (see Fig. 1(a)); to switch the output memristors to R_L, V_{om}>V_{th}. The relevant constraints are expressed in Eq.7 where r_R=$\frac{R_H}{R_L}$.

$$-V_{th} < V_{im} = -\frac{V_w}{1+r_R\frac{n_{iL}}{n_o}} < V_{th}, V_{om} = \frac{V_w}{1+r_R\frac{n_o}{n_{iL}}} > V_{th} \quad (7)$$

After the output memristors switch to R_L, the $M_{i,eq}$ remains the same ($M_{i,eq}$≈$\frac{R_L}{n_{iL}}$), while $M_{o,eq}$ changes to $\frac{R_L}{n_o}$. To ensure that the input memristors stay at their current resistive states, $-V_{th}$<V_{im}<V_{th} (see Fig. 1(a)); to prevent the output memristors from switching back to R_H, V_{om}>$-V_{th}$. Note that V_{om}>$-V_{th}$ is always true as V_{om}>0 (see Eq.5).

$$-V_{th} < V_{im} = -\frac{V_w}{1+\frac{M_{o,eq}}{M_{i,eq}}} = -\frac{V_w}{1+\frac{n_{iL}}{n_o}} < V_{th} \quad (8)$$

All above constraints are further reduced to Eq.9 where r_{wt}=$\frac{V_w}{V_{th}}$ and r_{ht}=$\frac{V_h}{V_{th}}$.

$$1+\frac{1}{r_R}\frac{n_o}{n_{iL}} < r_{wt} < min\{1+\frac{n_o}{n_i}, 1+\frac{n_{iL}}{n_o}\} \quad (9)$$

Note that n_{iL}=1 is the worst case condition. The constraints for other gates can be derived in a similar way. All the relevant constraints of each gate are summarized in Table II where r_{Hs}=$\frac{R_H}{R_s}$ and r_{Ls}=$\frac{R_L}{R_s}$.

TABLE III: Memristor Technology and Simulation Parameters

Technology						
F (nm)	D (nm)	R_L (Ω)	R_H (Ω)	V_{th} (V)	Area (F^2)	T_{sw} (ps)
50	8 (5)	200k (4M)	400M	1.5	4	527 (10.2)
Simulation						
V_w (V)		V_h (V)		R_s (Ω)		
1.95		0.975		2M (40M)		

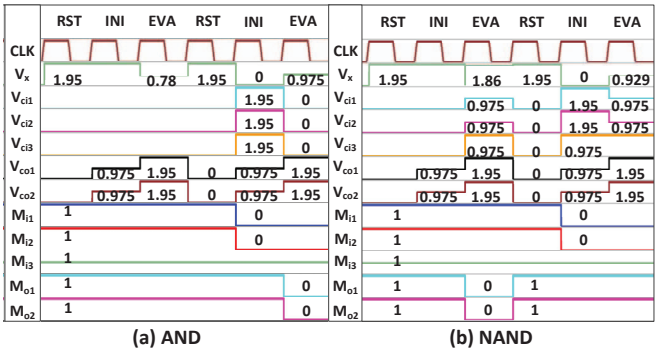

Fig. 6: Simulation Results of Logic Gates.

V. Simulation Results and Evaluation

This section first verifies the proposed logic gates and their constraints using SPICE simulations; subsequently it discusses the potentials of SBL gates by comparing them with CMOS gates in terms of area, delay and power/energy consumptions.

A. Simulation

The simulation model of the SBL gates contains a CMOS controller, voltage drivers, and a memristor crossbar. The controller is used to determine the control voltage that must be applied to each nanowire; it is based on a state machine consisting of four states: IDLE, RST (reset all the memristors to R_H), INI (initialize the input memristors to R_H or R_L) and EVA (evaluate the logic gates). The voltage drivers are implemented with three pass transistors and three voltage sources [21]. The memristor model (see Section III.A), controller and voltage drivers are described by Verilog-A modules, while the memristor crossbar by a SPICE netlist.

Table III summarizes both the memristor technology and simulation parameters; the technology parameters R_L, R_H, V_{th}, feature size (F), area (A) and thickness of TaO_x film (D) are extracted from the physical devices in [22], while the switching time (T_{sw}) is derived from the method used in [23]; the values of the technology parameters in brackets can be ignored for now and are explained later. The simulation parameters consist of the control voltages and R_s, which are determined by the proposed constraints in section IV.

To verify the proposed logic gates, the AND and NAND gates of Fig. 4 with three inputs and two outputs are used as examples. Fig. 6 shows the simulation results. In the waveform, the voltages $V_{ci\#}$, $V_{co\#}$, and V_x are shown in addition to the input and output values; R_H and R_L present logic 1 and 0.

The accuracy and effectiveness of the voltage constraints are mainly determined by the formulation of V_x. Therefore, the formulas in Section IV are used to calculate V_x, and the

TABLE IV: V_x at State EVA

Cases	AND		Cases	NAND	
	Cal. (mV)	SPICE (mV)		Cal. (mV)	SPICE (mV)
All 1	780	780	Not All 1	929.1	929
Not All 1, B	0.974	0.974	All 1 B	33.3	33.3
Not All 1, A	974.9	975	All 1, A	1856.5	1860

calculated results are compared with the simulation results as shown in Table IV. The table shows that for the AND and NAND gates the calcuated (cal.) and simulated (SPICE) values of V_x both before (B) and after switching (A) have an error of only a couple of mVs. The simulation results show that the functionality of gates perform correctly, and the constraints are effective to determine the values of control voltages.

B. Comparison with CMOS gates

To investigate the potentials of the SBL gates, the INV and 2-input NAND gates are compared to 32nm FinFET gates [3,24] in terms of area, delay, power/energy consumption during a single operation. The area of each SBL gate includes the sum of both the transistors (T) required for the control and memristors (M) used for computation; the voltage drivers are implemented using the same technology as CMOS gates; these voltage drivers can be shared by several (e.g., 5) SBL gates as they can use the same nanowires in the crossbar; the area of R_s (R) is not considered. In addition, we only consider the evaluation step (e.g., EVA in Fig. 6) and therefore, the controller is ignored. The delay of each SBL gate is $3{\cdot}T_{sw}$ as it requires three states for computation (see Fig. 4). The power consumption of the SBL gates is the sum of the power consumption of each memristor and R_s (which are calculated by $P = V^2/R$ where V is the voltage across the memristor, R the memristor's resistance at that moment; V is calculated by the voltage formulas in Section IV). Table V shows the evaluation results for both SBL and FinFET gates. For SBL gates, we consider two cases based on (a) existing and (b) future memristor technology. The future memristor technology parameters can be found by the numbers in the brackets of Table III. Comparing to the same type of FinFET gates, SBL gates perform worse in terms of delay and energy consumption, and may consume more power for exisiting memristor technology. For example, the SBL inverter consumes less power (652.9 nW) than the CMOS inverter (1067 nW) with the almost same area. However, the SBL inverter has longer (about 150x) delay than the CMOS inverter, and this leads to a higher energy consumption. In contrast, SBL gates with future technology parameters (e.g., with higher R_L and less thickness of TaO_x) can outperform FinFET gates in all metrics. For instance, the delay of the SBL inverter using future memristor technology is reduced to around 31ps, which is comparable to the CMOS inverter; therefore, the SBL inverter consumes around 28x less energy than CMOS. Overall, the SBL gates are potential to implement logic gates consume less area and power/energy with a slight delay panelty with reasonable technology improvements.

TABLE V: 32nm CMOS and Memristor Comparison

Inputs	Tech.	Gates	No. of Devices			Area (μm^2)	Delay (ps)	Power (nW)	Energy (fJ)
			T	M	R				
1	INV	CMOS	2	0	0	0.0794	10.6	1067	0.0113
		Existing	6*	2	1	0.0795	1581	652.9	0.3439
		Future					30.9	36.73	0.0004
2	NAND	CMOS	4	0	0	0.1590	14.9	194	0.0029
		Existing	9*	3	1	0.1014	1581	549.5	0.2895
		Future					30.9	31.13	0.0003

*These transistors can be shared among 5 gates.

VI. Conclusion

This paper first overviewed the existing memristor logic circuits. Thereafter, it explored the logic gate space for Snider Boolean logic, which is one of the promising candidates for logic circuit design style for memristor crossbars. Compared to CMOS gates, memristor-based gates show potentials to achieve competitive performance with reasonable technology improvements. The Snider Boolean logic gates can be used for both computation and communication; therefore, they are a promising alternative for future VLSI circuits.

References

[1] S. Borkar, "Design perspectives on 22nm cmos and beyond," in *DAC*. ACM, 2009, pp. 93–94.

[2] W. Zhao *et al.*, "Nanodevice-based novel computing paradigms and the neuromorphic approach," in *ISCAS*. IEEE, 2012, pp. 2509–2512.

[3] ITRS report. [Online]. Available: http://www.itrs.net

[4] S. Hamdioui *et al.*, "Memristor based computation-in-memory architecture for data-intensive applications," in *DATE*, 2015, p. 199.

[5] P.-E. Gaillardon *et al.*, "The programmable logic-in-memory (plim) computer," in *DATE*, 2016, p. 188.

[6] G. S. Rose *et al.*, "Leveraging memristive systems in the construction of digital logic circuits," *Proceedings of the IEEE*, vol. 100, pp. 2033–2049, 2012.

[7] J. Borghetti *et al.*, "Memristive switches enable stateful logic operations via material implication," *Nature*, vol. 464, pp. 873–876, 2010.

[8] J. Borghetti *et al.*, "A hybrid nanomemristor/transistor logic circuit capable of self-programming," *PNAS*, vol. 106, pp. 1699–1703, 2009.

[9] G. Snider, "Computing with hysteretic resistor crossbars," *Applied Physics A*, vol. 80, pp. 1165–1172, 2005.

[10] S. Kvatinsky *et al.*, "MRL:memristor ratioed logic," in *CNNA*. IEEE, 2012, pp. 1–6.

[11] D. Morris *et al.*, "mlogic: Ultra-low voltage non-volatile logic circuits using STT-MTJ devices," in *DAC*. ACM, 2012, pp. 486–491.

[12] J. A. Currivan *et al.*, "Low energy magnetic domain wall logic in short, narrow, ferromagnetic wires," *Magnetics Letters, IEEE*, vol. 3, pp. 3 000 104–3 000 104, 2012.

[13] I. Vourkas *et al.*, "A novel design and modeling paradigm for memristor-based crossbar circuits," *Nanotechnology, IEEE Transactions on*, vol. 11, pp. 1151–1159, 2012.

[14] K. Kim *et al.*, "Stateful logic pipeline architecture," in *ISCAS*. IEEE, 2011, pp. 2497–2500.

[15] E. Linn *et al.*, "Beyond von Neumann: logic operations in passive crossbar arrays alongside memory operations," *Nanotechnology*, vol. 23, p. 305205, 2012.

[16] L. Gao *et al.*, "Programmable cmos/memristor threshold logic," *IEEE Transactions on Nanotechnology*, vol. 12, pp. 115–119, 2013.

[17] B. Behin-Aein *et al.*, "Proposal for an all-spin logic device with built-in memory," *Nature nanotechnology*, 2010.

[18] E. Lehtonen *et al.*, "Memristive stateful logic," in *Memristor Networks*. Springer, 2014, pp. 603–623.

[19] A. N. Whitehead *et al.*, *Principia mathematica*. University Press, 1912, vol. 2.

[20] S. Kvatinsky *et al.*, "Team: Threshold adaptive memristor model," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, pp. 211–221, 2013.

[21] K.-H. Kim *et al.*, "A functional hybrid memristor crossbar-array/cmos system for data storage and neuromorphic applications," *Nano letters*, vol. 12, pp. 389–395, 2011.

[22] F. Miao *et al.*, "Anatomy of a nanoscale conduction channel reveals the mechanism of a high-performance memristor," *Advanced Materials*, vol. 23, pp. 5633–5640, 2011.

[23] M. Zangeneh *et al.*, "Performance and energy models for memristor-based 1T1R RRAM cell," in *GLVLSI*. ACM, 2012, pp. 9–14.

[24] C. Meinhardt *et al.*, "FinFET basic cells evaluation for regular layouts," in *LASCAS*. IEEE, 2013, pp. 1–4.