Flow-based Computing on Nanoscale Crossbars: Design and Implementation of Full Adders

Zahiruddin Alamgir, Karsten Beckmann and Nathaniel Cady
SUNY Polytechnic Institute
4403 NanoFab East
257 Fuller Road
Albany, NY 12203
Email: ncady@sunypoly.edu

Alvaro Velasquez and Sumit Kumar Jha
Computer Science Department University of Central Florida
4000 Central Florida Blvd
Orlando, FL 32765
Email: jha@eecs.ucf.edu

***Abstract*—We present the design and implementation of a full adder circuit that exploits the natural flow of current through nanowires and More-than-Moore nano-devices in two dimensional crossbars. We evaluate the speed and energy efficiency of our design and compare it to equivalent one-bit adder designs using CMOS and nanoscale memristors. Our memristive full adder circuit has been shown to be an order of magnitude faster and more energy-efficient than equivalent CMOS designs. Our circuit is an order of magnitude more compact that equivalent CMOS designs. We also argue that our design occupies less area and is faster than competing memristor designs.**

***Index Terms*—Memristor, Crossbars, Adders, Sneak Paths, Digital Computing, Nanowires, Nanoscale Devices.**

I. Introduction

The exponential growth in the performance of computing systems over the last forty years has climaxed with a 10,000-fold increase in individual processor speeds over the last two decades of the 20th century [1]. During this period, three things have remained invariant:

(a) The increase in performance has come largely from the shrinking of the feature size of CMOS devices [2] reducing the voltage requirement of the circuit and permitting an increase in the clock frequency without exceeding the power budget.
(b) John Von Neumann's computer architecture [3] has been successfully implemented using hierarchical memory units and one or more processor units communicating using standardized on-chip interconnects and off-chip buses. This Von Neumann gap between processors and memories has traditionally been bridged using increasingly complex memory hierarchies.
(c) There has been a clear separation between the software and hardware stacks [4] achieved using a series of optimizing compilers for standardized languages with different back-ends [5] and a strongly preserved backward-compatibility of the x86 instruction set over several generations of processors. Thus, the increasing design complexity involved in rapidly improving computing performance, such as deep and speculative pipelining, not yet been achieved.

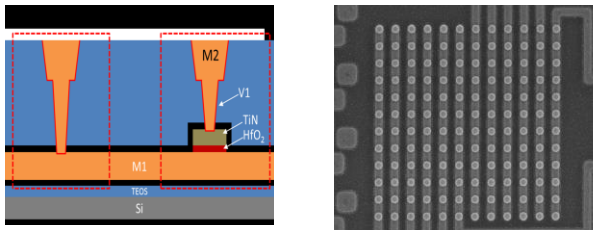

Fig. 1 (left) Schematic cross-section of a memristor used in this work. (right) SEM View of a 12x12 memristor crossbar array. Crossbar arrays were fabricated at SUNY Polytechnic Institute on 300 mm wafers using a modified IBM 65nm 10LPe process flow

The worldwide information technology industry has grown to a size of more than $3 trillion largely because of the exponential growth in computing capacity and the sustained consistency of the programming model across multiple generations of processors. The end of Dennard scaling threatens this exponential growth and the societal expectations of faster cyberinfrastructure from scientists and the public at large.

In this paper, we respond to the crisis caused due to the end of Dennard scaling by demonstrating that the flow of current through nanoscale crossbars can be used to perform addition in a fast, energy-efficient, and compact manner. We evaluate the speed and energy efficiency of our design and compare it to equivalent one-bit adder designs using CMOS and nanoscale memristors. Our memristive full adder circuit has shown to be an order of magnitude faster and more energy-efficient that the equivalent CMOS design. Our circuit is also an order of magnitude more compact that equivalent CMOS designs. Our results suggest a promising direction towards the development of next generation, energy efficient, and compact computing devices.

II. Related Work

A. Addition Using Memristive 'Stateful' Logic Operation

Borghetti et al. [6] have shown that, in addition to being used as memory elements, memristors can be used as conditionally switchable logic device via so-called "material implication" or IMP operation. For example, two memristors P and Q are connected electrically through a horizontal nanowire to a load R_G. IMP is performed by two simultaneous voltage pulses, V_{COND} and V_{SET}, applied to switches P and Q, respectively, to execute conditional switching on Q, depending on the state of P. They have demonstrated that NAND logic can be executed using three memristors by repeated use of the IMP operations. Lehtonen et al. [7] showed mathematically that addition can be performed using this concept of logic-in-memory. A set of input memristors is driven to a required set of initial states and a Boolean function f of these inputs is computed sequentially or step-by-step in another set of memristors known as working memristors. As sum and carry are Boolean functions, addition can be performed using memristors through a number of such sequential computations. This scheme requires 3N+5 devices and 88N+48 cycles where N stands for number of bits. Kvatinsky et al. [8] presented a specific modification in the crossbar structure to introduce two improved adder designs: a parallel adder and a serial adder. Their methods have been shown (through simulation) to reduce the number of devices and the number of cycles. In their approach, the serial adder requires 3N+3 devices and 29N cycles and the parallel adder requires 9N devices and 5N+18 cycles of operation to perform addition. Algorithmic insight into the synthesis of nanoscale crossbars [9] and networks of crossbars [10], including fault-tolerant designs, from functional specifications have also been studied earlier. This paper focusses on the experimental characterization of one-bit full adder.

B. Addition Using Complementary Resistive Switches

Complementary resistive switching (CRS) cells are made up of two anti-serially connected ReRAM cells. Siemen et al. came up with a scheme of two multi-bit adders using CRS based logic-in-memory process [11]. This adder has reduced the number of devices and steps. CRS based adders perform addition functions on a CRS passive crossbar array by using simple consecutive signal sequences. The system consists of many crossbar arrays and a control unit to send signal to word-lines and bit-lines. The calculations take place in several steps and intermediate steps are read out or stored information is used for the next steps. In final step, information is merged and final sum bits are calculated. This group also introduced two types of adder circuits. The first one is a pre-calculation adder, requiring 2(N +1) devices and 2(N +1) +2 cycles of operations. This type needs more than one wordline configured in different arrays, such as one wordline for sum calculation and other wordlines for auxiliary calculations. The second adder is a toggle cell adder that needs only one wordline in one array and requires N+ 2 devices and 4N+ 5 steps.

C. Addition Using CMOS

The most common CMOS adder circuit is 28 transistor (28T) adder. To fulfill different constraints such as area, power and speed, several CMOS based adders are being proposed such as Transmission Gate (TG) adder, Pass Transistor Logic (PTL) adder, and Gate Diffusion Input (GDI) adder. Each of these designs has its own merits and shortcomings. For example, conventional 28T adder circuit has high noise margin and reliable operation at low voltage. But the large number of transistors causes higher power consumption and a larger area. A full adder can be designed by implementing multiplexers and XORs as transmission gates. This type of adder has the fastest speed, but suffers from higher power dissipation compared to 28T adder.

All of the memristor-based adder schemes described above have only been demonstrated by means of simulation. This is perhaps because memristor-based adder operation requires an additional control unit to send control signals and to store memory in case of intermediate destructive reading. No comparison of latency, power or energy consumption with CMOS based adders have been described to date. As the adder schemes described above operate sequentially in several steps, it can be argued that latency could be substantially higher than that of CMOS adder circuit.

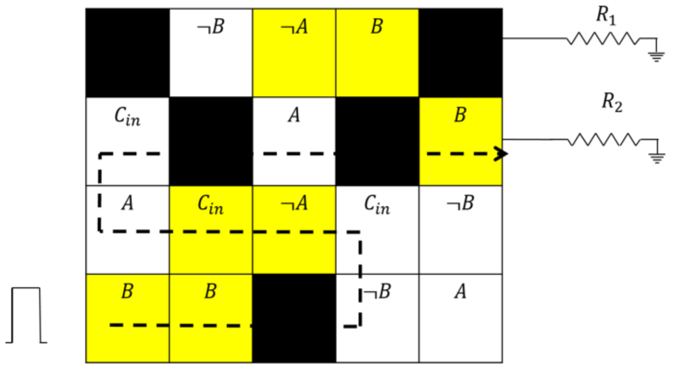

Fig. 2. The optimized design of a one-bit adder using sneak paths in a nanoscale memristor crossbar array. The black filled squares represent turned-off memristors and other squares denote a literal or its negation. The dashed blue arrow indicates the flow of sneak current from the lowest row to the third row when the inputs are: $A = 1$, $B = 0$, and $C = 1$.

III. Design And Implementation of the Full Adder

Our design for a one-bit adder is shown in Figure. 2. Here, the black filled squares represent turned-off (high resistance) memristors and other squares denote a literal or its negation. The input bits are A and B, and the carry-in bit is C_{in}. A Boolean value *'False'* or *'0'* corresponds to a turned-off memristor, and the Boolean value *'True'* or *'1'* corresponds to a turned-on memristor. The crossbar design can compute *Sum* and C_{out} by applying a voltage at the bottom wire. A large current will flow along the first row when *Sum* is *'True'* or *'1'* and a negligible amount of current would flow when *Sum* is *'False'* or *'0'*. Similarly, a large current will flow along the second row when *'C_{out}'* is *'1'* and negligible current will flow when *'C_{out}'* logic level is *'0'*. As a result, If we add high resistive loads R_1 and R_2 (R_1=R_2=50 kΩ) at the output of the first and the second row, voltage drop across R_1 (V_{R1}) will be much higher than the voltage drop across R_2 (V_{R2}) when *Sum* is *'1'* and C_{out} is *'0'* and vice versa. When both *Sum* and C_{out}

are *'0'* or both are *'1'*, voltage drops across both loads will be negligible or high, respectively.

To clarify how the sneak current flows to differentiate logic levels, we present a specific example where inputs were *A=1, B=0, C_{in}=1*. Each block represents a memristor at the crossbar junction: black boxes are turned off memristors, yellow boxes are memristors that are turned off for this input combination, and white boxes are turned on (low resistance state) memristors according to the design explained above. In Figure 2, the dotted line shows the flow of sneak current when a voltage is applied at the bottom row. In this case, the sneak current flows through the C_{out} nanowire and the voltage drop across R_2 is much higher than the voltage across R_1. As a result, C_{out} would be evaluated as '*High'*.

In order to study the performance of our full adder designs experimentally, crossbar arrays (12×12) were implemented on a 300 mm wafer platform in a back-end-of-the-line (BEOL) process. The IBM 65 nm 10LPe process was used as the mainframe to integrate a custom build ReRAM layer between metal 1 (M1) and metal 2 (M2). Metal 1 was fabricated in a damascene process where 100 nm lines were etched into the SiO_2 insulating layer and subsequently filled with a line consisting of TaN/Ta followed by electroplated Cu. After a planarization step the HfOx layer was deposited in a reactive PVD process and subsequently served as the active part of the memristor. A subsequent TiN deposition created the top electrode and in a RIE step followed by a wet etch the 12×12 memristor elements were defined in the size of 100×100 nm^2 on top of M1. By depositing SiO_2 the active layer was encapsulated and the M2 insulation was fabricated. A final dual-damascene step created the 100 nm wide M2 trenches and V1 holes to connect to the created TiN top electrode. M2/V1 got fabricated by putting down the liner (TaN/Ta) and again electroplating Cu. A final planarization step removed the excess copper resulting in insulated M2 lines. For our adder circuit implementation, we used 4×5 sub-matrix in a corner of our 12×12 matrix array.

A. Experimental Verification

The adder operation was performed in two steps. In the first step, the array was configured to one of the input combinations. In a commercial realization, a control circuit might be used to change one input state to another input state. In the second step, pulse voltages of 0.1 volt (~100 mV) and 0.2 volt (~200 mV) were applied in the bottom row and the voltages across R_1 and R_2 for all the input configurations of the 1-bit adder were measured. In all the cases, output voltages corresponded to the expected logic levels. The output voltage for output logic level *'1'* was at least 20 times higher than that for output logic level *'0'*. For example, when the inputs are *A=1, B=1, C=0,* the output logic levels are: *Sum = 0, C_{out} = 1*. In this case, V_{R1} = 1.77 mv and V_{R2} = 98.04 mV were found. In this case, $V_{R2}/V_{R1} = 55$. As a result, the probability of noise margin error would be very low. The experimental results are summarized in the Table 1.

B. Speed and Energy-Efficiency of Computation

Any arithmetic operation such as addition, in broad view, requires four basic steps in Arithmetic Logic Unit (ALU) in a microprocessor – a) loading the address of the data, b) moving the content of the memory location into accumulator, c) adding the content of the memory location, and d) moving the accumulator content to memory location. These events proceed with clock cycles. The first two steps can be considered as pre-computing steps. The power and propagation delay of pre-computing step of memristive adder depend on switching power and speed. It can be noted that, Torrezan et al. have shown sub nanosecond switching of tantalum oxide memristor [12, 13]. This fast switching could be attributed to the reasons that ionic species move a fairly short distance (1 nm or less) during switching and Joule heating significantly enhances the mobility of the ionic species [14]. Hence, we will compare the delay and power of computation step of CMOS adder and the proposed memristive adder.

TABLE I
OUTPUT CURRENTS FOR DIFFERENT INPUT COMBINATIONS

Input			Output Voltages (mV)		Output Logics	
A	*B*	*C_{in}*	*Sum(V_{R1})*	*C_{out} (V_{R2})*	*Sum*	*Cout*
0	0	0	0.94	1.33	0	0
0	0	1	95.53	0.74	1	0
0	1	0	98.50	0.79	1	0
0	1	1	1.04	98.43	0	1
1	0	0	98.95	0.84	1	0
1	0	1	4.58	98.42	0	1
1	1	0	1.77	98.04	0	1
1	1	1	97	99.04	1	1

The speed of computation depends on the delay between the input and output signals. The delay measurement was limited by the electrical testing instrument currently available in our laboratories. As an alternative, Cadence Spectra circuit simulator (SPICE level simulator), tightly integrated with virtuoso custom design platform, was used to design a schematic of a 4×5 crossbar array and a 28T CMOS adder and to simulate propagation delay and other metrics of both type of adders. Our crossbar array was fabricated on the IBM 65 nm CMOS 10LPe Low Power technology platform. Line resistances and other parasitic capacitances such as lateral capacitance, fringe capacitance etc. were extracted for simulation following IBM 10LPe design specifications. The on state memristor resistance values were set to 1 kΩ, and the turned off memristor resistances were set to 1 MΩ. The 28T adder was designed based on the same technology platform and optimal transistor size was used to maintain optimal delay. For CMOS, current and voltage drops are negligible due to a pull-up, pull-down network. But in crossbar adders, the

TABLE II
PERFORMANCE METRICS COMPARISON

	28T(1GHz)	Crossbar (1Volt input)	Crossbar (0.2 Volt input)
Propagation Delay (ps)	39	1.4	1.3
Power (µw)	4.68	22.5	1
PDP (x1E-17 Joules)	182.52	31.5	1.3

voltage drops occur along the sneak path. Hence, we added a higher resistance (50 kΩ) as the load in simulation so that sufficient voltage can be obtained at output node to drive the next stage. All the input combinations were simulated using 1 V, and 200 mV voltage pulses with 300 ps pulse width having 50% duty cycle. For 28T adder 980 MHz (∼1 GHZ) clock pulse and 1.2 volts bias were used. For CMOS the average power was calculated using $P_{avg} = V_d \int_0^T i(t)dt$ where T is the time period when all possible input combinations took place. For the crossbar adder, the same expression is used and the results are averaged over all possible input combinations. Propagation delay of the crossbar adder was found to be very low compared to the CMOS adder in the simulation. For the crossbar adder, power consumption for a 1 V input pulse was higher than that of the CMOS adder. And, in the case of a 200 mV input pulse, the memristive crossbar adder provided extremely low power operation. So for 100 mV input voltage, the power consumption would be even lower. In terms of Power Delay Product (PDP), multiplication of power and propagation delay, which is energy per operation, the memristive crossbar adder seems to be a better candidate than the CMOS implementation.

C. Area

As memristors have excellent scalability, crossbar based adders have an advantage over CMOS adders in terms of total circuit area. The dimensions of our 4 × 5 crossbar array were 640 nm × 1000 nm and so the area of the crossbar adder circuit was 0.64 μm^2. The conventional 28T CMOS full adder circuit based on 65 nm technology has an area of 10 μm^2, which is approximately 15 times higher than our memristor crossbar based adder. Again, 10 nm × 10 nm HfO_2 based ReRAM has been reported [15]; thus, crossbar based adders of significantly reduced area are possible. As a full adder circuit is a building block of other arithmetic operations, such as subtraction, multiplication, and division in microprocessors, the scalability of memristor crossbar arrays can be leveraged to create arithmetic and logic circuits of much reduced area using non-volatile memory storage device.

ACKNOWLEDGMENT

The authors would like to thank the Air Force Research Laboratory for financial support under contract FA8750-11-1-0008 (PI Nathan Cady). Sumit Kumar Jha acknowledges support from the Royal Bank of Canada and the Oak Ridge National Laboratory/UT Battelle, LLC under contract#4000136396. The research is also supported by National Science Foundation Software & Hardware Foundations #1438989 and Exploiting Parallelism & Scalability #1422257 projects. Alvaro Velasquez acknowledges support from the National Science Foundation Graduate Research Fellowship.

REFERENCES

[1] A. A. Chien and V. Karamcheti, “Moore’s law: The first ending and a new beginning,” *Computer*, vol. 46, no. 12, pp. 0048–53, 2013.

[2] C. A. Mack, “Fifty years of moore’s law,” *Semiconductor Manufacturing, IEEE Transactions on*, vol. 24, no. 2, pp. 202–207, 2011.

[3] J. Von Neumann and G. Brucks, “Preliminary discussion of the logical design of an electronic computing instrument,” 1946.

[4] G. Martin, “Will hardware and software be codesigned?” *IEEE Design & Test of Computers*, pp. 70–73, 2011.

[5] F. Brandner, V. Pavlu, and A. Krall, “Automatic generation of compiler backends,” *Software: Practice and Experience*, vol. 43, no. 2, pp. 207–240, 2013.

[6] J. Borghetti, G. S. Snider, P. J. Kuekes, J. J. Yang, D. R. Stewart, and R. S. Williams, “Memristive switches enable stateful logic operations via material implication,” *Nature*, vol. 464, no. 7290, pp. 873–876, 2010.

[7] E. Lehtonen and M. Laiho, “Stateful implication logic with memristors,” in *Proceedings of the 2009 IEEE/ACM International Symposium on Nanoscale Architectures*. IEEE Computer Society, Conference Proceedings, pp. 33–36.

[8] S. Kvatinsky, G. Satat, N. Wald, E. G. Friedman, A. olodny, and U. C.Weiser, “Memristor-based material implication (imply) logic: Design principles and methodologies,” *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 22, no. 10, pp. 2054–2066, 2014.

[9] *Proceedings of the 2015 IEEE/ACM International Symposium on Nanoscale Architectures, NANOARCH 2015, Boston, MA, USA, July 8-10, 2015*. IEEE, 2015. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7164206

[10] A. Velasquez and S. K. Jha, “Parallel computing using memristive crossbar networks: Nullifying the processor-memory bottleneck,” in *9th International Design and Test Symposium, IDT 2014, Algeries, Algeria, December 16-18, 2014*. IEEE, 2014, pp. 147–152. [Online]. Available: http://dx.doi.org/10.1109/IDT.2014.7038603

[11] A. Siemon, S. Menzel, R. Waser, and E. Linn, “A complementary resistive switch-based crossbar array adder,” *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on*, vol. 5, no. 1, pp.64–74, 2015.

[12] A. C. Torrezan, J. P. Strachan, G. Medeiros-Ribeiro, and R. S. Williams, “Sub-nanosecond switching of a tantalum oxide memristor,” *Nanotech- nology*, vol. 22, no. 48, p. 485203, 2011.

[13] J. P. Strachan, A. C. Torrezan, G. Medeiros-Ribeiro, and R. S. Williams, “Measuring the switching dynamics and energy efficiency of tantalum oxide memristors,” *Nanotechnology*, vol. 22, no. 50, p. 505402, 2011.

[14] J. J. Yang and R. S. Williams, “Memristive devices in computing sys- tem: Promises and challenges,” *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 9, no. 2, p. 11, 2013.

[15] B. Govoreanu, G. Kar, Y. Chen, V. Paraschiv, S. Kubicek, A. Fantini, I. Radu, L. Goux, S. Clima, and R. Degraeve, “10nm^2 hf/hfo$_x$ crossbar resistive ram with excellent performance, reliability and low-energy op- eration,” in *Electron Devices Meeting (IEDM), 2011 IEEE International*. IEEE, 2011.