# Experience on Material Implication Computing With an Electromechanical Memristor Emulator

S. Zuin[1], M. Escudero-López[1], F. Moll[1], A. Rubio[1]
(1) Department of Electronic Engineering
Universitat Politècnica de Catalunya (UPC), BarcelonaTech
Barcelona, Spain
antonio.rubio@upc.edu

I. Vourkas[2], G. Ch. Sirakoulis[3]
(2) Department of Electrical Engineering
Pontificia Universidad Católica de Chile, Santiago, Chile
(3) Department of Electrical and Computer Engineering
Democritus University of Thrace, Xanthi, Greece
iovourkas@uc.cl, gsirak@ee.duth.gr

***Abstract*—Memristors are being considered as a promising emerging device able to introduce new paradigms in both data storage and computing. In this paper the authors introduce the concept of a quasi-ideal experimental device that emulates the fundamental behavior of a memristor based on an electro-mechanical organization. By using this emulator, results about the experimental implementation of an unconventional material implication-based data-path equivalent to the i-4004 are presented and experimentally demonstrated. The use of the proposed quasi-ideal device allows the evaluation of this new computing paradigm, based on the resistance domain, without incorporating the disturbance of process and cycle to cycle variabilities observed in real nowadays devices that cause a limit in yield and behavior.**

***Keywords—Memristor devices; Imply; Material Implication function; Unconventional Computing.***

## I. INTRODUCTION

The memristor (M) is a new circuit element postulated by Prof. Leon Chua in 1971 [1] extending the conventional set of well-known resistance (R), inductance (L) and capacitance (C) elements. Conceptually, it is a passive element that holds a nonlinear relationship between flux linkage and electric charge. This implies that memristor resistance evolves with the previous history of charge to which it has been exposed exhibiting an interesting nonvolatile characteristic. Memristor-based research activity was boosted in 2008 when Hewlett Packard connected experimental nanodevices with this, for the time being, theoretical principle [2], opening a fascinating new field of research and applications.

Memristor devices are a promising alternative for storage devices because they inherently behave as nonvolatile memory elements, with the corresponding impact on power consumption reduction and with the unconventional characteristic of storing data (logic states) in form of electrical resistance. Moreover, it is back end of line (BEOL) [3] compatible, scalable device and can achieve higher density [3] levels of stored data than with conventional technology (MOS devices, 6T and dynamic cells). It is also expected that memristive devices speed could match in a near future the one of conventional CMOS devices, focusing on a scenario of hybrid technology. Together with the clear application as storage device both academia and industry are focusing efforts to orient memristors towards unconventional ways to process data, exploring beyond-von Neumann architectures [4]. Among all computing techniques with memristors that are being explored [5], material implication-based [6] is especially interesting because it exploits the store/process capability in a straightforward way. Material implication (in many cases indicated as logic implication and notated as $p \rightarrow q$) is based on the IMPLY operation, a binary function declared in Table I.

TABLE I. IMPLY TRUTH TABLE

| Case | $p$ | $q$ | $q' = p \rightarrow q$ |
|---|---|---|---|
| 1 | 0 | 0 | 1 |
| 2 | 0 | 1 | 1 |
| 3 | 1 | 0 | 0 |
| 4 | 1 | 1 | 1 |

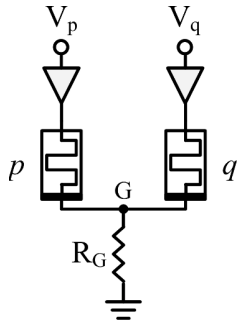



Fig. 1 Circuit implementation of the IMPLY function, where the logic variables $p$ and $q$ correspond to the resistance states of the two memristors.

The interest of this function when memristive circuits are considered, comes from the fact that when the two logical states (1, 0) are represented by the resistance of a memristor the IMPLY function can be implemented with the simple circuit shown in Fig. 1, where $p$ and $q$ (see Table I) correspond to the logic states of the two memristors of the circuit.

In Fig. 1 the IMPLY function is performed when synchronized pulses are applied to $V_p$ (of amplitude $V_{cond1}$) and $V_q$ (of amplitude $V_{cond2}$), when the following conditions (1) are verified ($R_{off}$ and $R_{on}$ stand for high and low resistance state, respectively). It can be shown that any function can be

performed using the same universal circuit topology, applying an appropriate sequence of voltage pulses $V_p$ and $V_q$. The IMPLY function together with the FALSE function (the reset of memristors to the high resistance state) form a universal logic set.

$$V_{cond1} < V_{set}$$
$$V_{cond2} > V_{set} \quad (1)$$
$$R_{off} > R_G > R_{on}$$

Nevertheless, due to the fact that nowadays memristor device technology is still not mature enough and also that the first devices to become commercially available are quite expensive, the development of hardware emulators has drawn a lot of attention from researchers [7]-[9]. Emulators in fact facilitate the experimental exploration of memristive dynamics in circuits and systems.

In this context, here we present our memristor emulator, a physical device of very simple implementation which behaves as an ideal memristor, practically without any deviation/variability and being easily parameterizable, based on an electromechanical principle. More specifically, section II introduces the basics about the electrical behavior of a memristor, first from a theoretical point of view and later in the case of a modern real device, showing the deviations of that last one and commenting the problems that causes when the IMPLY function is implemented with it. Next, section III introduces our memristor emulator. Section IV deals with details of the physical construction of the emulator and shows a demonstration assembly composed by seven such emulators, being an extension of the IMPLY circuit shown in Fig. 1. In this circuit the pulsed voltages of the seven devices are generated with a FPGA board and a specific level shifter circuit. Section V shows how the basic IMPLY circuit can be used to implement a generic computing data path (including the memristors memory and processing data owing to the IMPLY function). It is then shown how the ten key instructions of the i-4004 processor [10] can be implemented with this structure and our emulator, showing with detail the required pulses sequence for one of them (INC A). Finally in section VI, conclusions and discussion of the work are presented.

## II. Behavior of an Ideal and a Real Memristor

### A. The Ideal Memristor

A memristor is a polarised (nonreversible) dipole element with a resistance between its terminals dependable of the past. Fig. 2 shows the ideal behaviour of a memristor using the developed emulator. The figure shows the *i-v* characteristic of a memristor, where clearly two different resistive states can be observed, $R_{on}$ (low resistance high conductance state) and $R_{off}$ (high resistance low conductance state). In the example of Fig. 2, $R_{on} \approx 1100\Omega$ and $R_{off} \approx 120.000\Omega$. The transitions from one resistance level to the other are achieved by a voltage higher than $V_{set}$ (transition from off to on, in the example around 7V) and lower than $V_{reset}$ (transition from on to off, in the example -2.5V) thresholds. In the margin between these two voltage levels the device remains at a constant non-volatile resistance, i.e. it demonstrates linear on and off states (being on or off depending on the history of the applied voltage).

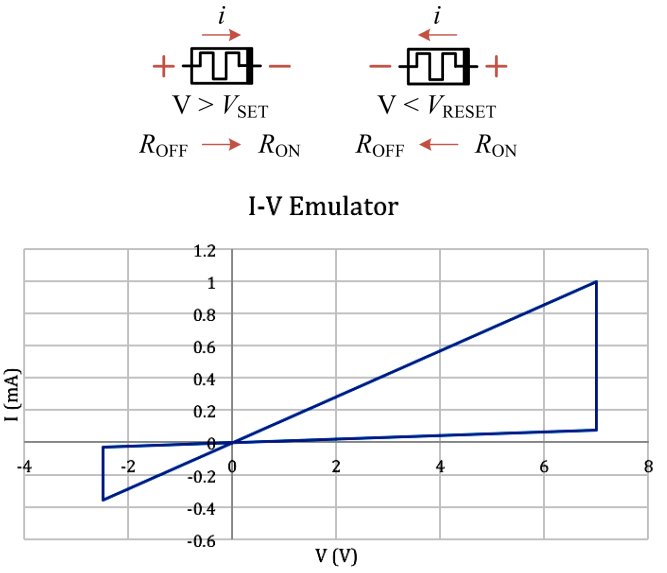



Fig. 2 Memristor symbol and switching convention, along with the *i-v* characteristic of an ideal memristor. The graph shows the two resistance states (on and off) of the device as well as the threshold voltages at which the state changes occur.

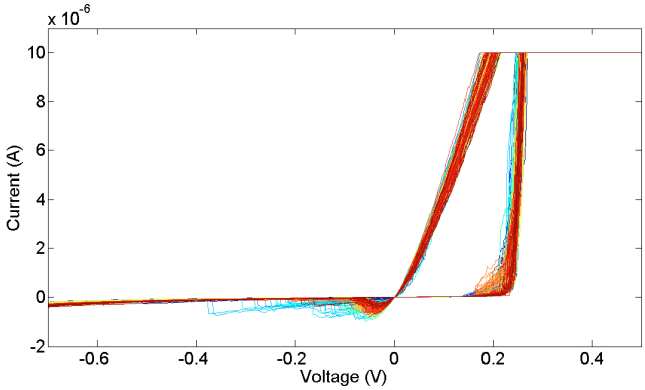



Fig. 3 Characteristic *i-v* for an experimental memristor, showing variability from cycle to cycle.

### B. Modern Real Memristor Devices

Fig. 3 shows the corresponding *i-v* characteristic of a real device. In the case shown we are considering a chalcogenide memristor manufactured by the KNOWM company [11], [12]. The different colored curves correspond just to repetitive cycles of set and reset switching, where it can be clearly observed that the experimental device exhibits an important time-varying variability in the four parameters of the device ($R_{on}$, $R_{off}$, $V_{set}$ and $V_{reset}$).

For the device shown in Fig. 3 (that has different parameters than the one shown in Fig. 2), the $R_{off}$ keeps quite stable but $R_{on}$ exhibits a significant random variability (with a drift between 15.000Ω and 10.000Ω). Additionally $V_{set}$ shows a significant variability (from 1.7V to 2.2V) as well as $V_{reset}$ (from -0.1V to -0.01V, approximately). It has been earlier

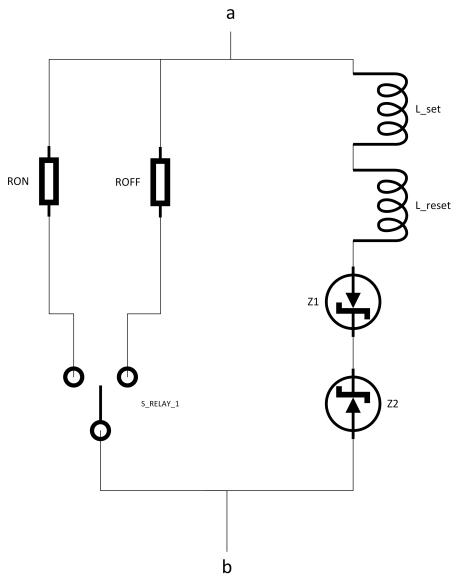



Fig. 4 Proposed emulator circuitry.

analysed and shown that these state and voltage variabilities cause a drop in the yield and behavior of the IMPLY circuit in [13].

## III. An Electromechanical Memristor Emulator

In order to perform an experimental evaluation of the application of the material implication principle in computing, a physical quasi-ideal emulator circuitry has to verify the following conditions:

- It should be able to exhibit a binary (without intermediate states) and perfectly lineal resistance without variability (that is $R_{on}$ or $R_{off}$).
- The voltages at which the resistance state changes ($V_{set}$ and $V_{reset}$) have to be clearly defined exhibiting a neglectible cycle to cycle variability.
- It has to exhibit an inherent nonvolatility property.
- It is desirable that the device be passive.

In this context, Fig. 4 shows the proposed emulator circuit. Specifically, a and b represent the two terminals of the emulator circuit, i.e. of the memristor. The basis of the proposal is an electromechanical device TQ latching (bistable) relay. The relay keeps the position of the contacts without the need of any external source of energy. The connection is changed when a pulse of current passes through the set and reset coils in one or other sense.

The main circuit is the branch b-relay-resistors-a. The two resistors are conventional resistors with resistances of value $R_{on}$ and $R_{off}$. The other branch, b-zeners-coils-a, corresponds to the *set* and *reset* functions of the memristor. The two anti-series zeners cause a potential barrier that defines the $V_{set}$ and $V_{reset}$ threshold voltages. When $V_{ab}$ is in-between the set and reset voltages (the zeners are off) the branch does not conduct current. The values of the $R_{on}$ and $R_{off}$ of the emulated

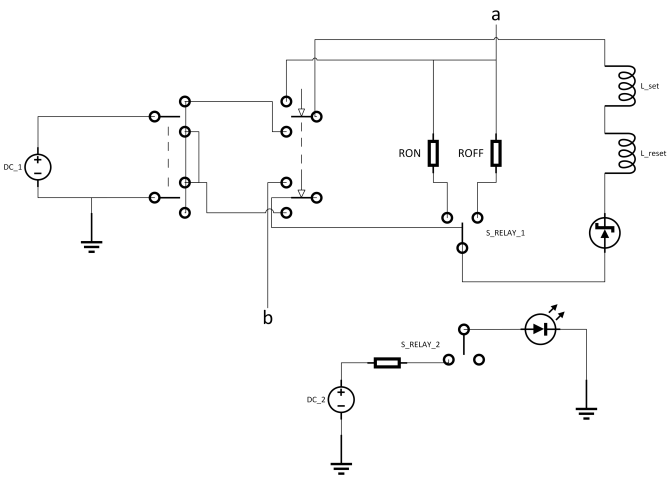



Fig. 5 Complete emulator circuitry including display and pre-setting facilities.

memristor are given by the two resistors and the transition voltages by the following ($V_a$-$V_b$ criteria):

$$V_{set} = V_{Z2} + V_\gamma \qquad V_{reset} = -V_{Z1} - V_\gamma \tag{2}$$

where $V_\gamma$ is the forward voltage of diodes. It is possible to adapt the design to any set and reset voltages selecting the appropriate zener diodes. In cases where the $V_{reset}$ is near to zero (like in the physical memristor shown in Fig. 3) it is possible to use just one zener, being in this case:

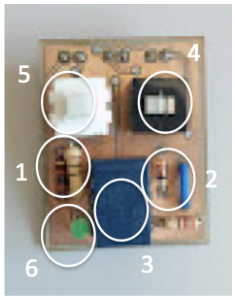
$$V_{set} = V_{Z2} \qquad V_{reset} = -V_\gamma \tag{3}$$

Taking advantage of the fact that the relay device (Panasonic TQ 2 coils 2 circuits relay) has two independent circuits, the second one can be used to optionally display the state of the memristor with a green led and with the inclusion of a switch, a push button and an external power supply, the option to pre-set the emulator of any of the two resistance states (on or off) is achieved. These two complementary circuits do not affect the basic circuit shown in Fig. 4. Fig. 5 shows the scheme of the complete emulator including display and pre-setting.

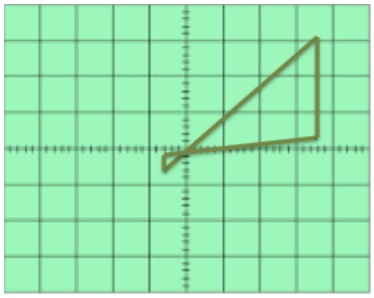
## IV. Physical Construction of the Emulator and Description of a 7-Memristors Imply Circuit Set-Up

Fig. 6(a) shows the physical aspect of the developed emulator implementation (25×25mm) on a printed circuit board. Detail 1 (circled) shows the two $R_{on}$ and $R_{off}$ resistors, 2 the zener diode, 3 the relay, 4 the on and off selector switch, 5 the setting button and 6 the state indication led. Fig. 6(b) shows the *i-v* characteristic of the emulator.

We next introduce our scheme for the computation between two registers in an unconventional data path unit by using the IMPLY principle. Let's consider two registers A and B (in order to simplify the physical implementation we will consider 2-bit registers). The structure of the circuit is universal for any




(a)



(b)

Fig. 6 (a) Physical implementation of our memristor emulator. (b) *i-v* characteristic of the emulator (vertical axis: 25mA/div., horizontal axis: 2volts/div.).

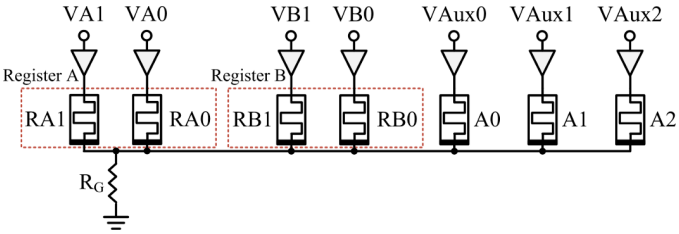



Fig. 7 Universal data path implementation for a two registers data path by using the IMPLY principle and the scheme of Fig. 1.

set of instructions between the registers when including the necessary auxiliary memristors. Our approach is straightforward. We will consider an extended IMPLY circuit as the one shown in Fig. 1 but with 7 memristors (2 per register, RA1/RA0 for register A, RB1/RB0 for register B and 3 auxiliary A0, A1, and A2, enough for the set of instruction of section V) for storage of intermediate results (see Fig. 7).

The seven pulsed signals are independent and are generated with a sequencer implemented with an FPGA board with the appropriate level shifter buffers. Fig. 8 presents the complete set-up implementation. In the experiment the following settings have been used: $R_{off}$ = 1kΩ, $R_{on}$ = 100Ω, $R_G$ = 220Ω, $V_{set}$ = 7V, $V_{cond1}$ = 9V, $V_{cond2}$ = 6.5V, $V_{clear}$ (to reset the memristors) = -2V. The FPGA generates, for each of the 7 pulsed voltages, a 2-bit code of the applied voltage ($V_{cond1}$, $V_{cond2}$, $V_{clear}$, high impedance), decoded by the level shifter buffer board.

## V. Application of the Universal Data Path Circuitry to an I-4004-like Instruction Set

We will now consider the following set of instructions: AND (A,B), OR (A,B), NAND (A,B), NOR (A,B), XOR (A,B), R (A), R (B) INC (A), INC (B), DEC (A), DEC (B),

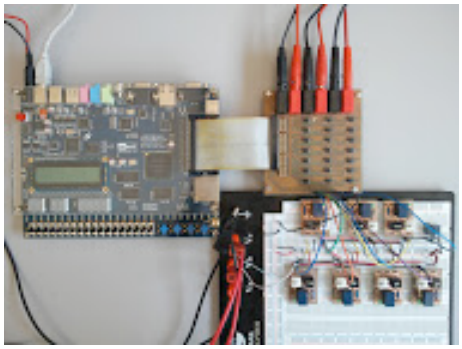


Fig. 8 Global implementation of a universal data path for two 2-bit registers. It can be identified the protoboard with 7 emulators, the drivers-specific board and the ALTERA D-2 FPGA board as a sequence generator.

ADD (A,B) and SUB (A,B). It has to be observed that in this structure there is no privileged register (as was the case in the i-4004 with the register A or accumulator), any register (A and B) can be source and/or destination of the instructions, R corresponds to rotate, so we consider 10 instructions.

### A. Application of Sequences and Performances

The electromechanical relays allow working with pulses at a cadence of 250Hz. Table II shows the summary of time performances of execution of each one of the 10 instructions.

TABLE II. Instruction Requirements (@250Hz)

| Instruction | Time of execution (ms) | # of pulses applied | # of internal operations |
|---|---|---|---|
| AND | 160 | 40 | 10 |
| OR | 96 | 24 | 6 |
| NAND | 192 | 48 | 12 |
| NOR | 256 | 64 | 16 |
| XOR | 352 | 88 | 22 |
| R | 112 | 28 | 7 |
| INC | 320 | 80 | 20 |
| DEC | 320 | 80 | 20 |
| ADD | 560 | 140 | 35 |
| SUB | 832 | 208 | 52 |

### B. The Increment Instruction (INC).

In this subsection we show in detail as a matter of example the sequences of actions and pulses corresponding to the instruction INC. In the shown example we apply INC to register A, not being used consequently register B, and auxiliary memristors A0, A1 and A2, the latter being where the carry is stored at the end of the operation. The sequence of the 20 required operations are given by:

---

False A2, A0

RA0 → A0

RA1 → A0

A0 → A2

False A0, A1

RA0 → A0

RA1 → A1

RA0 → RA1

A0 → A1

False A0

A1 → A0

RA1 → A0

False A1, RA1

A0 → A1

A1 → RA1

False A1, A0

RA0 → A0

A0 → A1

False RA0

A1 → RA0

For readability reasons, Fig. 9 shows the 20 operations expressed correspondingly as pulsed voltages on the respective memristors. The different operations are performed just by modifying the sequence of pulses and not the architecture. Able to work at a pulsed frequency of 250Hz, the execution time of the instructions goes from 96ms in the case of the OR to 208ms in the case of the SUB. Therefore, IMPLY logic can be considered as an interesting approach to unconventional processors. Fig. 10 shows the evolution of the memristors for these 20 sequential operations. At the beginning the state of A1, A0 is 10 (green, red) and after the 20$^{th}$ operation it is 11 (green, green), i.e. 10+1, showing the intermediate states of memristor for the intermediate operations.

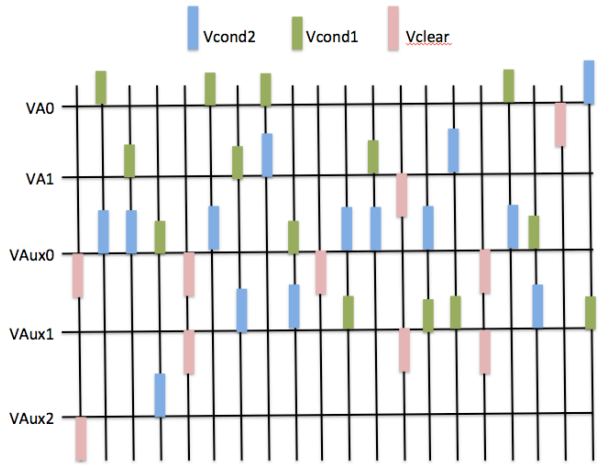



Fig. 9 Sequence of voltage levels applied to the five memristors used in the INC instruction.

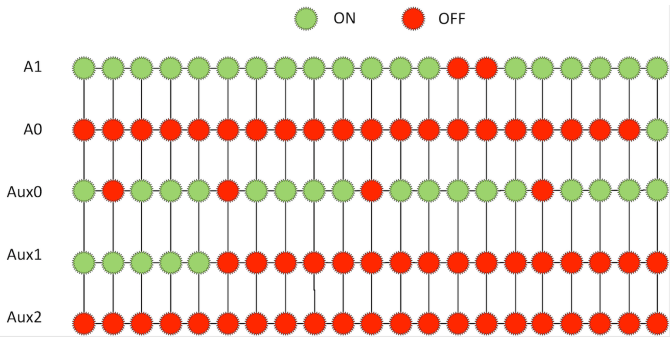



Fig. 10 State and evolution of the five memristors for the sequence of 20 pulses that perform the INC(10) =11.

## VI. Conclusions

This paper presented a laboratory implementation of a two-register data path unit by using seven physical memristor emulators in a straightforward implementation of the material implication or IMPLY logic function. The result is a simple enough, satisfactory and universal way to implement a set of instructions that in the case of the present article were inspired from the basic set of instructions of the i-4004 microprocessor. The ten basic considered instructions can be implemented just modifying the pulsed voltages sequence. Evidently the sequence is independent of the data and also the architecture is unique for the ten instructions and for any other that could be selected. The memristor used consists in an electromechanical emulator, a quasi-ideal and easily customizable device based on a bistable relay. Such quasi-ideal memristor emulator allows to experiment without considering process and cycle to cycle variability limitations, present in modern nanodevices. Such variability introduces yield limitations and imposes in many cases memristor state refreshing. The experimentation shows the benefits of the IMPLY as an unconventional universal processing unit.

## Acknowledgment

This work has been funded by the Spanish MINECO and ERDF through project Maragda (TEC2013-45638-C3-2-R). This work has been also partially supported by CONICYT FONDECYT Postdoctorado No. 3160042/2016 Chilean government research grant.

## References

[1] L. Chua, “If it’s pinched it’s a memristor,” Semicond. Sci. Technol., vol. 29, (104001), 2014.

[2] D.B. Strukov, G.S. Snider, D.R. Stewart, and R.S. Williams, “The missing memristor found,” Nature, vol. 453, pp. 80-84, May 2008.

[3] E. Korczynski, “Knowm first to deliver configurable artificial neural networks using bi-directional learning memristors,” Semiconductor Manufacturing and Design, Solid State Technology, semimd.com.

[4] I. Vourkas and G.Ch. Sirakoulis, Memristor-Based Nanoelectronic Computing Circuits and Architectures: Foreword by Leon Chua, ser. Emergence, Complexity and Computation, New York, NY, USA: Springer-Verlag, 2015.

[5] I. Vourkas, and G.Ch. Sirakoulis, “Emerging Memristor-based Logic Circuit Design Approaches: A Review,” IEEE Circuits and Systems Magazine, vol. 16, no. 3, pp. 15-30, 2016.

[6] J. Borguhetti, 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, pp. 873-876, April 2010.

[7] D. Biolek, “Memristor Emulators,” in Memristor Networks, A. Adamatzky and L. Chua, Eds., Springer Int. Publishing, 2014, pp. 487-503.

[8] C. Yang, H. Choi, S. Park, M. Pd Sah, H. Kim, and L.O. Chua, “A memristor emulator as a replacement of a real memristor,” Semicond. Sci. Technol., vol. 30, (015007), 2015.

[9] I. Vourkas, V. Ntinas, A. Abusleme, G.Ch. Sirakoulis, and A. Rubio, “A Digital Memristor Emulator for FPGA-Based Artificial Neural Networks,” 2016 IEEE Int. Verification and Security Workshop (IVSW), Sant Feliu de Guixols, Cataluña, Spain, July 4-6, pp. 54-57.

[10] W. Aspray, “The Intel 4004 microprocessor,” IEEE Annals of the History of Computing, vol. 19, no. 3, pp. 4-15, August 2002.

[11] R. Colin Johnson, “Memristor computer emulates brain functions”, EE Times, 1/20/2016.

[12] Bio Inspired Technologies, “Neuro-bit: the world's first commercially available memristor,” [Online]. Available: http://www.bioinspired.net/ [Accessed 15 December 2015].

[13] X. Fang and Y. Tang, “Circuit analysis of the memristor stateful implication gate,” Electronic Letters, vol. 49, no. 20, pp. 1282-1283, September 2013.