Aquesta és una còpia de la versió final d'un article publicat a IEEE Symposium Series on Computational Intelligence. "Proceedings SSCI 2016"

URL d'aquest document a UPCommons E-prints:

http://hdl.handle.net/2117/101683


DOI 10.1109/SSCI.2016.7850154
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 Engineering
Democritus University of Thrace, 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 electromechanical organization. By using this emulator, results about the experimental implementation of an unconventional material implication-based data-path equivalent to the i4004 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. It is BEOL [3] compatible, scalable, 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 academy and industry are focusing efforts to orient memristors towards unconventional ways to process data, exploring non-von Neumann architectures [4bis]. Among all computing techniques with memristors that are being explored, material implication-based [5] 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 function, a Boolean function declared in Table I.

![Fig. 1. Implementation of the IMPLY function, where logical states \( p \) and \( q \) correspond to the resistance state of the two indicated memristors.](image)

<table>
<thead>
<tr>
<th>Case</th>
<th>( p )</th>
<th>( q )</th>
<th>( q' ) = ( p \rightarrow q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

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 \( q \) (Table I) correspond to the logic states of the two memristors of the circuit.
Device. In the case shown we are considering a chalcogenide

$$V_q$$ (of amplitude $$V_{cond2}$$), when the conditions

$$V_{cond1} < V_{set}$$
$$V_{cond2} > V_{set}$$
$$R_{off} > R_G > R_{on}$$

are verified ($$R_{off}, R_{on}$$ stand for high, low resistance state).

It can be shown that any function can be performed using
the same topology (universal topology), applying an
appropriate sequence of voltage pulses $$V_p$$ and $$V_o$$. The IMPLY
function together with the FALSE function (the reset of
memristors to high resistance state) form an universal logic
set.

In this work section II introduces 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. Section III
introduces our memristor emulator, a physical device, that
based on an electromechanical principle behaves as an ideal
memristor, practically without any deviation and easily parameterizable. Section IV deals with details of the physical
construction of the emulator and shows a demonstration assembling composed by seven emulators, an extension of the
IMPLY circuit shown in Fig. 1. In this circuit the pulsed
voltage 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 thanks to the IMPLY function). It is shown
how the 10 key instructions of the i-4004 processor [6] can be
implemented with this structure showing with detail the
required pulses sequence for one of them (INC A). Finally in
section VI the conclusion and discussion of the work is
presented.

II. ELECTRICAL BEHAVIOR OF AN IDEAL AND A REAL
MEMRISTOR

A. Electrical Behavior of a 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. The figure
shows the I-V characteristic of a memristor, 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 $$R_{on} = 1100\Omega$$ and
$$R_{off} = 120,000\Omega$$. The transition from one resistance level to the
other is given by $$V_{set}$$ (transition from off to on, in the example
around 7V) and $$V_{reset}$$ (transition from on to off, in the example
-2.5V). In the margin between these two voltage levels the
device keeps at a constant nonvolatile resistance state (on or off
depending on the history of voltage).

B. Electrical behavior of a modern real memristor device

Fig. 3 shows the corresponding I-V characteristic of a real
device. In the case shown we are considering a chalcogenide

memristor manufactured by KNOWM company [7]. The
different colored curves correspond just to repetitive cycles of
set and reset, where it can be clearly observed that the
experimental device exhibits an important time-varying
variability in the four parameters of the device ($$R_{on}, R_{off}, V_{set}
and V_{reset})$$.  

![I-V Emulator](image)

Fig. 2. Symbol and electrical characteristic of an ideal memristor.
The characteristic shows the two resistance states (on and off) of
the device as well as the voltages at which the state changes occur.

![Characteristic I-V](image)

Fig. 3. Characteristic I-V for an experimental memristor,
showing variability from cycle to cycle.

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 approximatelly).

It has been earlier analysed and shown that these state and
voltages variabilities cause a drop in the yield and behavior of
the IMPLY circuit in [8].

III. AN ELECTROMECHANICAL MEMRISTOR EMULATOR CIRCUIT

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:
• 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 $V_{reset}$) has to be clearly defined exhibiting a neglectible cycle to cycle variability.

• It has to exhibit an inherent nonvolatility property.

• It is desiderable that the device be passive.

Fig. 4 shows the proposed emulator circuit:

![Proposed emulator circuitry](image)

In Fig. 4 a and b represent the two terminals of the emulator circuit, so 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 $R_{on}$ 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 set and rest voltages. When $V_{ab}$ is in-between the set and reset voltages (zeners are off) the branch does not conduct current. The values of the $R_{on}$ and $R_{off}$ of the emulated memristor are given by the two resistors and the transition voltages by ($V_a$-$V_b$ criteria):

$$V_{set} = V_{z2} + V_i$$

$$V_{reset} = -V_{z1} - V_i$$

where $V_i$ is the forward voltage of diodes. It is possible to adapt the design to any set and reset voltages selecting the appropriated 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:

$$V_{set} = V_{z2}$$

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 and a push button and an external power supply the functionality to pre-set the emulator of any of the two resistance states (on, off). 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.

![Complete emulator circuitry including display and pre-setting facilities](image)

IV. PHYSICAL CONSTRUCTION OF THE EMULATOR AND DESCRIPTION OF A 7-MEMRISTORS IMPLY CIRCUIT SET-UP

Fig. 6 shows the physical aspect of the emulator implementation (25×25 mm) 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/off selector switch, 5 the setting button and 6 the state led. Fig. 7 shows I-V characteristic of the emulator.

![Memristor emulator implementation](image)

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 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 meristors (2 per register, RA1/RA0 for register A, RB1/RB0 for register B and 3 auxiliary (A0, A1, A2), enough for the set of instruction of section V) for storage of intermediate results (see Fig. 8).

![Image](vertical-axis: 25 mA/div., horizontal axis: 2 volts/div.).

The seven pulsed signals are independent and are generated with a sequencer implemented with a FPGA board with the appropriated level shifter buffers. Fig. 9 shows the complete setup implementation.

In the experiment the following settings have been used: 
- \( R_{off} = 1k\Omega \), \( R_{on} = 100\Omega \), \( R_G = 220\Omega \), \( V_{set} = 7V \), \( V_{cond1} = 9V \), \( V_{cond2} = 6.5V \), \( V_{clear} \) (to reset the meristors) = -2V. The FPGA generates for each of the 7 pulsed voltages, a 2-bit code of the applied voltage \((V_{cond1}, V_{cond2}, V_{clear}, \text{high impedance})\), decoded by the level shifter buffer board.

![Image](Universal data path implementation for two registers data path by using the Imply principle and the scheme of Fig. 1.)

### V. APPLICATION OF THE UNIVERSAL DATA PATH CIRCUITRY TO AN I-4004-LIKE INSTRUCTION SET

We will 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), 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 destiny of the instructions, R corresponds to rotate, so we consider 10 instructions.

![Image](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 ALTERA D-2 FPGA board as sequence generator.)

#### A. Application of sequences and performances

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

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

#### 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 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 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

Fig. 10 shows the 20 operations expressed as pulsed voltages on the respective memristors. The different operations are performed just 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. IMPLY can be considered an interesting approach to unconventional processors.

VI. CONCLUSION

This paper has 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 function. The result is that it is a satisfactory and universal way to implement a set of instructions that in the case of the present article have been inspired from the basic set of instructions of the i-4004 microprocessor. The ten basic considered instructions can be implemented just modifying the sequence of pulsed voltages sequence. Evidently the sequence is independent of the data and also the architecture is simple and unique for the ten instructions and for any other that could be selected. The memristor used in the experience is an electromechanical emulator, a quasi-ideal easily customizable device based on a bistable relay. The use of such quasi-ideal memristor emulator allows to experiment without taking into account process and cycle to cycle variability limitations present in modern nanodevices. This variability introduces yield limitations and requiring in many cases memristor state refreshing. The experimentation shows the benefits of the IMPLY as a unconventional universal processing unit.

ACKNOWLEDGMENT

This work has been funded by the Spanish MINECO and ERDF through project Maragda (TEC2013-45638-C3-2-R).

REFERENCES