CCIT Report #795 August 2011

Memristor-based IMPLY Logic Design Procedure

Shahar Kvatinsky, Avinoam Kolodny,
and Uri C. Weiser

Department of Electrical Engineering
Technion – Israel Institute of Technology
Haifa 32000 ISRAEL
{skva@tx, kolodny@ee, uri.weiser@ee}.technion.ac.il

Eby G. Friedman

Department of Electrical and Computer Engineering
University of Rochester
Rochester, NY 14627 USA
friedman@ece.rochester.edu

***Abstract* — Memristors can be used as logic gates. No design methodology exists, however, for memristor-based combinatorial logic. In this paper, the design and behavior of a memristive-based logic gate – an IMPLY gate - are presented and design issues such as the tradeoff between speed (fast write times) and correct logic behavior are described, as part of an overall design methodology. A memristor model is described for determining the write time and state drift. It is shown that the widely used memristor model - a linear ion drift memristor - is impractical for characterizing an IMPLY logic gate, and a different memristor model is necessary such as a memristor with a current threshold.**

Keywords- memristor; memristive systems; IMPLY; design methodology; logic

I. INTRODUCTION

Memristors are passive elements with varying resistance (also known as a memristance), conceived theoretically in [1]. Changes in the memristance depend upon the history of the device, the total charge which passes through it, or, alternatively, the total flux in the device (the integral over time of the applied voltage at the ports of the device).

In 2008, Hewlett-Packard announced the fabrication of a working memristor [2]. A linear ion drift model was proposed for describing the behavior of this memristor. The memristance of a linear ion drift memristor is

$$M(q) = R_{OFF}\left(1 - \frac{\mu_v R_{ON}}{D^2} q(t)\right), \quad (1)$$

where R_{OFF} and R_{ON} are, respectively, the maximum and minimum resistance of the memristor, μ_v is the average ion mobility, D is the memristor physical thickness, and $q(t)$ is the total charge passing through the memristor. The linear ion drift model is the most commonly used memristor model, although practical memristors exhibit highly non-linear behavior.

Memristors can be used for numerous applications, such as memory [3], neuromorphic systems [4], and analog circuits (e.g., see [5]). One interesting application of memristors is logic, using memristors as building blocks of logic gates. To use memristors in a digital manner, a high memristance is considered as logic *0* and a low memristance is considered as logic *1*. Several approaches for memristor-based logic have been proposed, e.g., [6] and [7], which suggest using memristors as configurable switches as in an FPGA. The logic gates are designed as CMOS gates or as programmable majority logic array (PMLA) based on Goto pairs as logic gates [8].

Another approach is to use memristors as the primary building blocks of a logic gate. Each memristor acts as an input, output, computational logic element, and a latch in different stages of the computing process [9]. In [10], a memristor-based logic gate - the IMPLY gate, is presented. Since this logic function together with FALSE (a function that always yields the value *0* as an output) comprise a computationally complete logic structure, it may potentially provide a basic logic element for a memristor-based circuit. The truth table for *p IMPLY q* is listed in Table 1. Unlike CMOS logic [11], no design methodology exists for memristor-based logic circuits.

In this paper, a design methodology is suggested for memristor-based IMPLY logic gates. A memristor-based IMPLY gate and related limitations are also presented here. The tradeoff between performance and robustness is described as well as the necessity to refresh the logic gate.

This paper is organized as follows. In Section II, the operation of a memristor-based IMPLY gate is described. In section III, the performance and limitations of this logic gate are presented. In section IV, a design example is described, and simulation results of the IMPLY gate are shown. The paper is summarized in section V.

II. MEMRISTOR-BASED IMPLY GATE

The logic function $p \rightarrow q$ (also known as "*p IMPLIES q*," "*material implication*," and "*if p then q*") is described in [10]. The proposed memristor logic is based upon a resistor R_G ($R_{ON} < R_G < R_{OFF}$) connected to two memristors, named P and Q, acting as digital switches. The corresponding initial memristances p and q are the inputs of the gate; while the output of the gate is the final memristance of Q (the result is written into the logic state q). A schematic of an IMPLY gate is shown in Figure 1.

The basic concept is to apply different negative voltages to P and Q, where V_{SET}, the applied voltage on Q, has a higher magnitude than V_{COND}, the applied magnitude on P ($|V_{COND}| < |V_{SET}|$). If $p = 1$ (low resistance), the voltage on the common terminal is approximately V_{COND} and the voltage on

This work was partially supported by Hasso Plattner Institute and by Intel grant "Heterogeneous Computing, the Inevitable Solution: Power Management, Scheduling and ISA" grant no. 864-737-13.