

Implementation of Fundamental Logical Gates by 1-D Chaotic Elements

Toshinori Munakata* and Sudeshna Sinha**

* Computer and Information Science Department, Cleveland State University, Cleveland, OH 44115, U.S.A. munakata@cis.csuohio.edu ** The Institute of Mathematical Sciences, C.I.T. Campus, Madras 600 113, India. sudeshna@imsc.ernet.in

Recently ideas of computer architecture based on totally new principles other than traditional silicon chips have been given much attention. While practical, everyday-use of these proposals as computing devices is yet to be seen, these ideas have stimulated the scientific community due to their fundamental nature, novelty, and potentials for new forms of information processing and applications. In the case of chaos computing, the current stage is primarily on developing theory along with digital simulations by computers and analog simulations by electric circuits. There are many potential advantages which include the following [1 and references herein]: Low power and high speed. Reconfigurable or dynamic logical architecture. General-purpose computing. Robustness against noise. Simple yet cost effective. Abundant candidates in nature and engineering. It can overlap and interact with other areas such as nanotechnology. Furthermore, by combining these logical gates, we can build adders and memory.

In this paper, we present an analysis for implementing a *complete* set of logical gates with two inputs and one output, realized by 1-D (one-dimensional) chaotic (or very nonlinear system) element. In the following, I_1 and I_2 are the two "interpreted" inputs and O is the "interpreted" output. Each of these variables can assume either 0 or 1. X_1 and X_2 are the two "actual" inputs and Z is the "actual" output. Each of these variables can assume either 0 or δ , where δ is a positive constant. The actual values of 0 and δ correspond to 0 and 1 of interpreted values, respectively.

Step 1. Initialization of the state x of the system to x_0 and addition of external inputs: $x = x_0 + X_1 + X_2$.

Step 2. Chaotic update: $x \rightarrow f(x)$, where $f(x)$ is a chaotic function.

Step 3. Threshold mechanism to obtain output, i.e. the output Z is: $Z = 0$ if $f(x) \leq x^*$; $Z = \delta = f(x) - x^*$ if $f(x) > x^*$, where x^* is called the threshold value of the system.

Table I. All possible gates with two inputs and one output, where table entries represent output.

I_1, I_2	AND	OR	NAND	NOR	XOR	XNOR	Tautology	Contradiction
0, 0	0	0	1	1	0	1	1	0
0, 1 or 1, 0	0	1	1	0	1	0	1	0
1, 1	1	1	0	0	0	1	1	0

Among these gates, the last two are trivial (constant output can be emitted regardless the inputs). Hence, when we realize the first six gates, we cover all the possible cases. Table II shows conditions for these six gates in general.

Table II. Necessary and sufficient conditions to be satisfied by a chaotic element to implement each of the logical gates. Row 1 corresponds to input $(X_1, X_2) = (0, 0)$, Row 2 $(0, \delta)$ or $(\delta, 0)$, and Row 3 (δ, δ) . δ is typically a common positive constant for all the operations, whereas, x_0 and x^* differ among the operations.

Gate	AND	OR	NAND	NOR	XOR	XNOR
Cond	$f(x_0) \leq x^*$	$f(x_0) \leq x^*$	$f(x_0) - x^* = \delta$	$f(x_0) - x^* = \delta$	$f(x_0) \leq x^*$	$f(x_0) - x^* = \delta$
	$f(x_0 + \delta) \leq x^*$	$f(x_0 + \delta) - x^* = \delta$	$f(x_0 + \delta) - x^* = \delta$	$f(x_0 + \delta) \leq x^*$	$f(x_0 + \delta) - x^* = \delta$	$f(x_0 + \delta) \leq x^*$
	$f(x_0 + 2\delta) - x^* = \delta$	$f(x_0 + 2\delta) - x^* = \delta$	$f(x_0 + 2\delta) \leq x^*$	$f(x_0 + 2\delta) \leq x^*$	$f(x_0 + 2\delta) \leq x^*$	$f(x_0 + 2\delta) - x^* = \delta$

Table III. Numeric examples for x_0 and x^* for $f(x) = 4x(1 - x)$ and $\delta = 1/4$. XNOR is not possible for $4x(1 - x)$.

Gate	AND	OR	NAND	NOR	XOR	XNOR
x_0	0	1/8	3/8	1/2	1/4	-
x^*	3/4	11/16	11/16	3/4	3/4	-

[1] T. Munakata, S. Sinha and W. L. Ditto, "Chaos computing: Implementation of fundamental logical gates by chaotic elements," *IEEE Trans. Circuits and Systems - I*, 49, 11, Nov. 2002, pp. 1629-1633.