PHYSICAL REVIEW E **75**, 025201(R) (2007)

Using synchronization to obtain dynamic logic gates

K. Murali*
Department of Physics, Anna University, Chennai 600 025, India

Sudeshna Sinha†
The Institute of Mathematical Sciences, Taramani, Chennai 600 113, India

(Received 5 August 2006; published 7 February 2007)

We introduce a scheme to obtain key logic-gate structures, using synchronization of nonlinear systems. We demonstrate the idea explicitly by numerics and experiments on nonlinear circuits. A significant feature of this scheme is that a single nonlinear drive-response circuit can be used to flexibly yield the different logic gates, and switch logic behavior by small changes in the parameter of the response system; so the response system can act as a "logic output controller." Thus this scheme may help to construct dynamic general-purpose computational hardware with reconfigurable abilities.

DOI: 10.1103/PhysRevE.75.025201 PACS number(s): 05.45.Xt

In recent years there has been a new direction in harnessing the richness of chaos, namely, the exploitation of chaos to do flexible computations, the so-called "chaos computing" paradigm [1,2]. The aim is to use a *single chaotic unit to emulate different logic gates and perform different arithmetic tasks*, and further have the *ability to switch easily between the different operational roles*. Such a computing unit may then allow a more dynamic computer architecture and serve as ingredients of a general-purpose device more flexible than statically wired hardware [3].

The explicit chaos-computing schemes that have been designed and implemented so far have all been based on the thresholding (clipping/limiter) method to achieve controlled responses from a chaotic system. In this work we propose a very different scheme to implement dynamic logic gates: We will use the *synchronization* of a driver-and-response nonlinear system, to achieve logic operations [4]. We will also explicitly demonstrate the success of the scheme in a circuit implementation.

Now cooperative behavior of nonlinear systems, as exemplified by synchronization phenomena, has received much attention [4]. Generally, synchronization can be considered as the appearance of some relations between functionals of two processes due to interactions. Complete synchronization, namely, the exact coincidence of the states of systems, is the strongest form of synchronization, and here we will base our scheme on this phenomenon. Note that synchronization has been widely used as a basis for communication schemes [5]. Now, in this Rapid Communication, we will demonstrate the use of synchronization to obtain dynamic logic gates.

SCHEME

First we outline the scheme for dynamically and flexibly implementing the fundamental logic gate NOR (from which all gates may be constructed), and the logic gates AND and XOR (which yield the building block of arithmetic processing: the bit-by-bit addition operation). The last column of Table I gives the output corresponding to the two inputs I_1 and I_2, for NOR, AND, and XOR logic. Note that the four distinct possible input sets (0,0), (0,1), (1,0), and (1,1) reduce to three conditions as (0,1) and (1,0) are symmetric.

The logic cell here is comprised of one-way coupled nonlinear systems. The logic output will be given by the *synchronization error* between the two systems comprising the cell (see the schematic in Fig. 1). In our scheme the parameter of the drive system α_d is determined by the inputs and the parameter of the response system α_r acts as a "logic gate controller."

Setting inputs. The input set (I_1, I_2) determines the parameter of the drive system α_d: for example, input set (0,0) can correspond to α_1, $(0,1)/(1,0)$ to α_2, and (1,1) to α_3, where $\alpha_1, \alpha_2, \alpha_3$ are three reasonably separated parameter settings of the drive system (see Table I).

Obtaining output. The output is simply the error between the drive-and-response systems: large error is output 0, and output 1 corresponds to very small error. That is, *when the drive and response are quite synchronized, one obtains output* 1, *and* 0 *otherwise*.

Logic control. The logic output is determined by the parameter setting of the response system: for example, if the parameter of the response system α_r is set close to α_1, one obtains NOR; $\alpha_r \sim \alpha_2$ gives XOR, and $\alpha_r \sim \alpha_3$ gives AND (see Table I).

Note that the NOR gate is fundamental and in fact can be used to build any circuit by concatenation. All gates can be constructed by combining the NOR operation proposed above.

Further, bit-by-bit arithmetic addition, the most fundamental form of arithmetic operation is constructed from XOR and AND gates. Other types of arithmetic operations can then easily be performed using this basic addition or similar operation. For example, the addition of larger numbers (e.g., addition of two 32-bit numbers) can be carried out by extending the bit-by-bit operation to a higher number of bits. Subtraction can be done as addition of the complement numbers. Multiplication can be achieved as a repeated addition or

*Electronic address: kmurali@annauniv.edu
†Electronic address: sudeshna@imsc.res.in

TABLE I. Parameter setting of the drive system determined by the input set I_1, I_2, parameter setting of the response system for logic response control, and synchronization error yielding the output of operation for the NOR, XOR, and AND logic gates.

Logic gate	Input set (I_1, I_2)	Drive parameter (Input set)	Response parameter (Logic control)	Synchronization error	Output
NOR	(0,0)	α_1	$\sigma_{\mathrm{NOR}} \sim \alpha_1$	small	1
	(0,1)/(1,0)	α_2		large	0
	(1,1)	α_3		large	0
XOR	(0,0)	α_1	$\sigma_{\mathrm{XOR}} \sim \alpha_2$	large	0
	(0,1)/(1,0)	α_2		small	1
	(1,1)	α_3		large	0
AND	(0,0)	α_1	$\sigma_{\mathrm{AND}} \sim \alpha_3$	large	0
	(0,1)/(1,0)	α_2		large	0
	(1,1)	α_3		small	1

its variations; similarly, division can be done as repeated subtraction. Further, using logical operations such as AND, computer memory based on integrated circuits can be constructed. Such memory architecture is based on flip-flops, which in turn are built by combining logical gates [6]; so the scheme above yields the basic ingredients of a computing machine, flexibly from a single dynamic logic cell [7].

Further, other logic responses can also be obtained by using appropriate thresholds on the synchronization error to define output. The idea is to use partial synchronization in order to generate a larger set of synchronization response patterns. For instance, to get NAND and OR logic one can use the logic control parameters in the response system α_{NAND} to lie between parameters α_1 and α_2, for the NAND gate characteristics; and a parameter setting α_{OR} lying in between α_2 and α_3, for OR gate characteristics. Defining a suitable degree of synchronization, i.e., some small prescribed error as output 1, and large synchronization error as output 0, we can get NAND and OR output responses as well. For instance, for NAND, if we design the parameters appropriately, partial synchronization will result in the first two input settings (as α_1 and α_2 are close to α_{NAND}) but not in inputs $I_1=1$, $I_2=1$ (whose drive parameter setting α_3 is far from α_{NAND}). Similarly for OR, only input set $I_1=0$, $I_2=0$ with parameter setting α_1 will give no synchronization (namely, large error) and output 0, while the other two input sets (settings α_2 and α_3) will synchronize reasonably and give output 1 (as α_{OR} is in between, and close to, both α_2 and α_3).

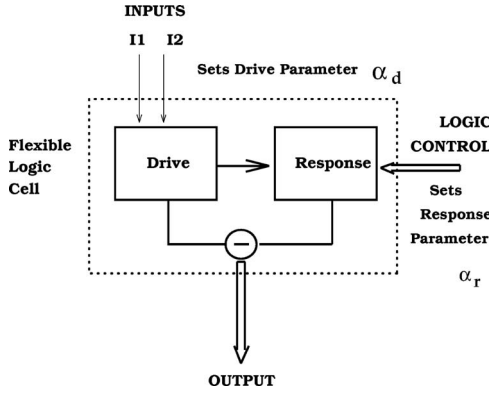

FIG. 1. Schematic diagram of the flexible dynamic logic cell comprised of one-way coupled nonlinear systems: the drive and the response, with the difference between the states of these determining the output.

IMPLEMENTATION

Specifically, in our circuit implementation we consider one-way coupled Chua circuits as the computing element. This is given by the following set of (rescaled) coupled ODEs [8,9] for the drive:

$$\dot{x}_1 = \alpha_d[x_2 - x_1 - g(x_1)], \tag{1}$$

$$\dot{x}_2 = x_1 - x_2 + x_3, \tag{2}$$

$$\dot{x}_3 = -\beta x_2. \tag{3}$$

And for the response,

$$\dot{x}_1' = \alpha_r\{x_2' - x_1' - g(x_1') + \epsilon(x_1' - x_1)\}, \tag{4}$$

$$\dot{x}_2' = x_1' - x_2' + x_3', \tag{5}$$

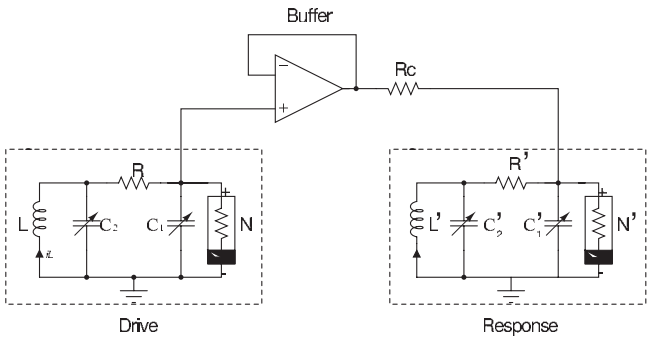

FIG. 2. Circuit diagram for one-way coupled Chua's circuit. Here C_1 and C_1' are the voltage-controlled capacitors, determining variable drive and response parameters α_d and α_r, respectively.

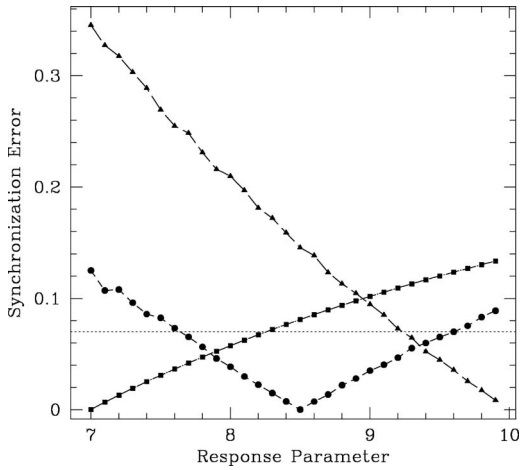

FIG. 3. Synchronization error envelope (namely, the envelope of maximum error) for drive parameters: $\alpha_1=7$ corresponding to input set $(I_1=0, I_2=0)$ (solid square); $\alpha_2=8.5$ corresponding to input set $(I_1=0, I_2=1)/(I_1=1, I_2=0)$ (solid circle); $\alpha_3=10$ corresponding to input set $(I_1=1, I_2=1)$ (solid triangle). The horizontal line shows the synchronization error=0.07 line.

$$\dot{x}_3' = -\beta x_2', \tag{6}$$

where coupling $\epsilon=1$, $\beta=14.87$, and the piecewise linear function $g(x)=bx+\frac{1}{2}(a-b)(|x+1|-|x-1|)$ with $a=-1.27$ and $b=-0.68$. The corresponding circuit component values are $L=18$ mH, $R=1710\ \Omega$, $C_1=10$ nF, $C_2=100$ nF. Chua's diode parameters are $R_1=220\ \Omega$, $R_2=220\ \Omega$, $R_3=2.2\ \text{k}\Omega$, $R_4=22\ \text{k}\Omega$, $R_5=22\ \text{k}\Omega$, $R_3=3.3\ \text{k}\Omega$, and buffer=opamp AD712. Note that the circuit is the one-way coupling configuration of the classic Chua's circuits [9] (see Fig. 2 for the circuit diagram).

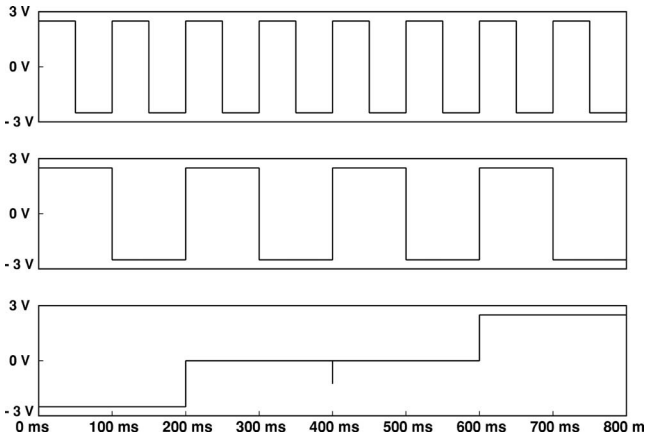

FIG. 4. From top to bottom: panels 1 and 2 show a stream of input signals I_1 and I_2, determining the input set (I_1, I_2). So the input sets (1,1), (0,1), (1,0), and (0,0) are repeated every 200 ms, in that order. Panel 3 shows the transmitted signal from drive to response.

The parameters varied in the drive and response systems are α_d and α_r. The detailed circuit implementation of the voltage-controlled capacitor, which enables easy control of the parameters, is discussed in Ref. [10]. The value of $\alpha_{r/d}$ is varied around ~ 9.

Numerical investigation of this system with respect to the different sets of drive and response parameters yields the synchronization error displayed in Fig. 3. It is clear that different logic behavior can be obtained by setting different response parameters and using appropriate synchronization error thresholds.

For instance, we can use a scheme where the output is 1 if the maximum of the synchronization error is below 0.07, and 0 otherwise, i.e., we get output 1 if the synchronization error

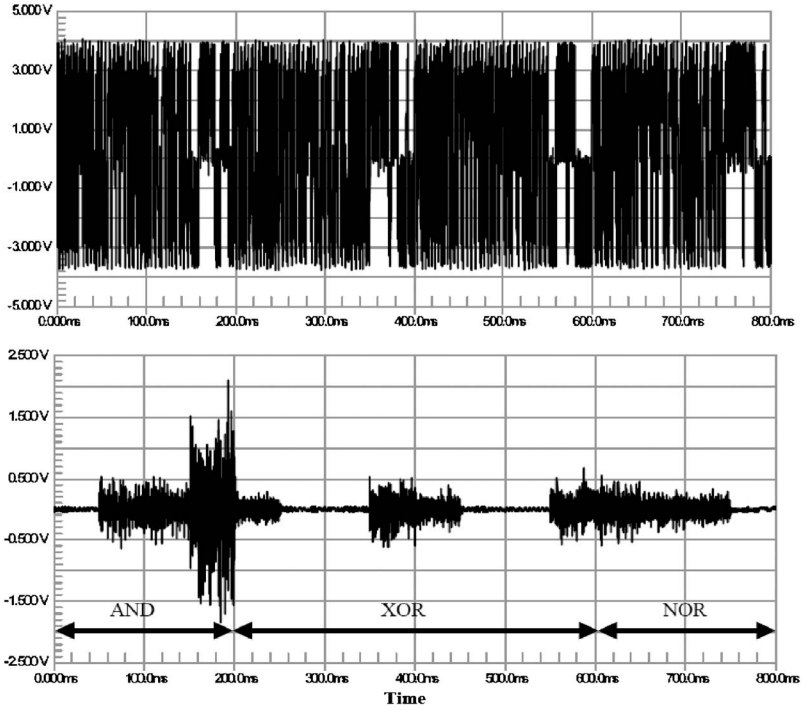

FIG. 5. From top to bottom: panel 1 shows the dynamic logic configuration signal; panel 2 shows the synchronization error signal giving a dynamic logic response: [0–200 ms] is the AND response, [200–600 ms] is the XOR response, and [600–800 ms] is the NOR response. The signal transmitted from drive to response can be periodic or chaotic, depending upon the value of the tuning parameter in the drive system.

lies below the 0.07 line displayed in Fig. 3, and output 0 otherwise. One can then clearly see from Fig. 3 that the AND, XOR, and NOR gates can be realized using response parameters of $\alpha_{\mathrm{NOB}}=7$, $\alpha_{\mathrm{XOR}}=8.95$, and $\alpha_{\mathrm{AND}}=10$. In addition to these key gates one can also obtain the fundamental NAND gate with $\alpha_{\mathrm{NAND}}=7.8$, as well as the OR gate with $\alpha_{\mathrm{OR}}=9.3$. All these response parameter values controlling the logic output can be inferred graphically from Fig. 3.

Results obtained by the simulation program PSPICE: Figures 4 and 5 show the timing pulses of the dynamic NOR, AND, and XOR gates in a PSPICE simulation. The values of α used here are $\alpha_1 \sim 8$, $\alpha_2 \sim 9$, and $\alpha_3 \sim 10$. The PSPICE results agree completely with numerical simulations. Note that the latencies involved in switching logic responses is small compared to the natural time scales of the system, as is evident from Fig. 5.

In this explicit example we have tuned parameter α. However, it is possible to *tune different parameters in parallel* in the drive for *multibit logic synthesis*. Such a realization will achieve parallel logic operations using the same logic cell.

The interesting inference one can draw at this point is the feasibility of synchronization-based computing. However, it is not appropriate at this incipient stage to debate the optimality of this scheme. Clearly, the operational speeds attainable by such a dynamic logic cell will be determined by the natural time scales of the dynamical system. So it is evident that the scheme could potentially work at considerable speeds if one can implement it on fast dynamical systems, such as electronic circuits operating in the GHz regime [11] and the semiconductor or fiber lasers yielding chaotic subnanosecond or picosecond pulses [12].

CONCLUSIONS

In summary, we have introduced a scheme to obtain key logic-gate structures using synchronization of nonlinear systems. We have demonstrated the idea explicitly by numerics and experiments on nonlinear circuits. In particular, we have shown the direct and flexible implementation of the basic logic gates NOR, AND, and XOR, using a single drive-response unit. Arrays of such logic cells can be reconfigured easily by a stream of logic-control parameter values to the response system, as in our circuit implementation. So such systems can conceivably be programmed on the fly, and be optimized for the task at hand. Thus architectures based on such logic implementations may serve as ingredients of a general-purpose computing device more flexible than statically wired hardware.

[1] S. Sinha and W. Ditto, Phys. Rev. Lett. **81**, 2156 (1998); S. Sinha and W. Ditto, Phys. Rev. E **60**, 363 (1999).

[2] S. Sinha *et al.*, Phys. Rev. E **65**, 036214 (2002); **65**, 036216 (2002); T. Munakata *et al.*, IEEE Trans. Circuits Syst. **49**, 1629 (2002); K. Murali *et al.*, Int. J. Bifurcation Chaos Appl. Sci. Eng. **13**, 2669 (2003).

[3] See G. Taubes, Science **277**, 1935 (1997), for a general discussion of the advantages of reconfigurable computer architectures in the context of field programmable gate arrays (FPGAs).

[4] A. Pikovsky, M. Rosenblum, and J. Kurths, *Synchronization: A Universal Concept in Nonlinear Sciences* (Cambridge University Press, Cambridge, 2001).

[5] For instance, G. D. Van Wiggeren, and R. Roy, Science **279**, 1198 (1997); A. Argyris *et al.*, Nature (London) **438**, 343 (2005) investigate communication schemes using synchronization of chaotic lasers in the laboratory and with commercial fiber-optic links, respectively. Further, control schemes such as in S. Hayes *et al.*, Phys. Rev. Lett. **70**, 3031 (1993), use chaos to transmit information and encode messages by guiding a chaotic trajectory onto a predetermined binary sequence.

[6] M. M. Mano, *Computer System Architecture*, 3rd ed. (Prentice Hall, Englewood Cliffs, 1993); T. C. Bartee, *Computer Architecture and Logic Design* (Mc-Graw Hill, New York, 1991).

[7] The theoretical optoelectronic realization of the NOR logic gate using chaotic two-section lasers was proposed by K. E. Chlouverakis and M. J. Adams, in Electron. Lett. **41**, 359 (2005). Here two self-pulsating laser diodes were commonly driven by a monochromatic light beam resulting in chaos and the logic gate was finally implemented by appropriately synchronizing the two chaotic attractors. Our scheme here is significantly different however, as it flexibly yields three logic gates from the *same* "logic cell." Further, our scheme is also verified in experiments.

[8] R. J. Maddock and D. M. Calcutt, *Electronics: A Course for Engineers* (Addison-Wesley Longman Ltd., 1997), p. 542.

[9] M. Lakshmanan and K. Murali, *Chaos in Nonlinear Oscillators: Controlling and Synchronization* (World Scientific, Singapore, 1996).

[10] G. O. Zhong, R. Bargar, and K. S. Halle, J. Franklin Inst. **331B**, 743 (1994).

[11] K. Myneni *et al.*, Phys. Rev. Lett. **83**, 2175 (1999); D. W. Sukow *et al.*, Chaos **7**, 560 (1997); J. N. Blakely *et al.*, Phys. Rev. Lett. **92**, 193901 (2004); J. N. Blakely *et al.*, IEEE J. Quantum Electron. **40**, 299 (2004).

[12] I. Fischer *et al.*, Phys. Rev. Lett. **76**, 220 (1996); Q. L. Williams, J. Garcia-Ojalvo, and R. Roy, Phys. Rev. A **55**, 2376 (1997).