CHAPTER 6

REALIZATION OF CHAOS BASED NOR GATE AND SET-RESET (SR) FLIP-FLOP

6.1 INTRODUCTION

Recently, ideas of computer architecture based on totally new principles other than 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. Some of the new technologies are hoped to complement the current majoritily static computer architecture based on silicon chips. Desirable input-output mapping can be achieved by utilizing the complex dynamics of the individual elements, as well as their interactive couplings and adaptive processes, implemented in particular as a threshold mechanism. Based on this, a new theoretical direction in harnessing the richness of chaos, namely, the exploitation of chaos to do flexible computations (Sinha and Ditto 1998 and 1999 Sinha et al 2002) has been considered. The aim is to use a single chaotic element 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 more dynamic computer architecture and serve as ingredients of a general-purpose device more flexible than statically wired hardware.
In this chapter, a simple experimental realization of a threshold controller that clips the chaos into a simple ordered phenomenon has been discussed first. Then using this control method, a new and simpler version of the chaos computing scheme has been demonstrated experimentally (Murali et al 2003) by directly implementing the fundamental NOR gate with a continuous-time simple analog simulation type chaotic circuit (Elwakil and Kennedy 2001, Sprott 2000). The advantage of this circuit configuration is that, it can be easily implemented with monolithic integrated circuits for low-voltage or low-power applications as it uses Current-Feedback Operational Amplifiers (CFOAs). Then this technique has been extended to the familiar Lorenz chaotic system by applying threshold mechanism on two state variables separately. Finally the SR flip-flop operation has been achieved from two cross-coupled NOR gates obtained from controlled Lorenz systems.

6.2 CONTROLLING OF CHAOS USING A THRESHOLD CONTROL

Consider a single chaotic element, described by the evolution equation

$$\frac{dX}{dt} = F(X; t),$$

(6.1)

where $X = (X_1, X_2, \ldots, X_N)$ are the state variables, and $F$ is a strongly nonlinear function. In this system, a variable $X_i$ is chosen as state variable to be thresholded, i.e., whenever the value of this variable exceeds a critical threshold $X^*$ (i.e., when $X_i > X^*$), it re-sets to $X^*$. The dynamics continues till the next occurrence of $X_i$ exceeding the threshold value, when control resets its value to $X^*$ again. So, this controller does not alter the original system in any way, as there is no perturbation on the parameters. Further, no run-time computation is needed to obtain the necessary control. The
theoretical basis of this method lies in clipping desired time sequences and enforcing a periodicity on the sequence through the threshold action, which acts as a partial resetting of initial conditions (Sinha 1994, Sinha et al 2002 and 2002a). The effect of this scheme is to limit the dynamic range slightly, i.e., “snip” off small portions of the available phase-space, and this small controlling action is effective in yielding a range of stable behaviors. Chaos is quite advantageous here, as it possesses a rich range of temporal patterns that can be clipped to different behaviors. This immense variety is not available from thresholding regular systems.

It can be shown analytically for one-dimensional maps and numerically for multidimensional systems that the threshold mechanism yields stable orbits of all orders by simply varying threshold level (Murali et al 2003, Sinha 1994). But so far limited experimental work has been carried out for the direct verification of this control scheme and its applications for chaos computing concepts. Here, the method on a very simple and easily reproducible analog simulation type chaotic circuit has been implemented. The experimental set-up is the realization of a simple nonlinear third-order Ordinary Differential Equation (ODE), a form that can capture the essential dynamics of double-scroll-like chaotic attractor. The model is given by

\[
\frac{d^3 x}{dt^3} = -a \left( \frac{d^2 x}{dt^2} + \frac{dx}{dt} + x - f(x) \right), \tag{6.2a}
\]

and

\[
f(x) = \text{sgn}(x). \tag{6.2b}
\]

The \text{sgn} (x) nonlinearity is odd-symmetrical and the system has a single parameter \textquoteleft a\textquoteright, through which its dynamical behavior can be changed (Elwakil and Kennedy 2001, Sprott 2000).
Figure 6.1  Circuit implementation of equation (6.2), with the precision clipping control circuit depicted in the dotted box. $V_T$ is the threshold controlled signal.

An experimental setup of this circuit as depicted in Figure 6.1 was constructed with $R_1 = 1 \text{ K}\Omega$, $C = 1 \text{nF}$, $k = 9$ and $a = 0.8$. AD844 CFOAs were used for $U_1$, $U_2$, $U_3$ and $U_4$. AD712 VOAs were used for $U_T$ and $U_C$. All the operational amplifiers were biased with $\pm 9\text{V}$ supplies. The implementation involves three successive active integrators to generate $d^3x/dt^3$, $dx/dt$ and $x$ from $d^3x/dt^3$, coupled with a nonlinear element (Comparator circuit) that generates $f(x)$ and feed it back to $d^3x/dt^3$ through a summing amplifier.
Now the threshold mechanism has been implemented on the dynamical variable $\frac{dx}{dt}$, namely, whenever $\frac{dx}{dt} > \frac{dx^*}{dt}$, $\frac{dx}{dt}$ is clipped to $\frac{dx^*}{dt}$, i.e., $\frac{dx}{dt}$ is replaced by $\frac{dx^*}{dt}$ in equation (6.2a) when $\frac{dx}{dt} > \frac{dx^*}{dt}$. There is no controlling action at all when $\frac{dx}{dt} \leq \frac{dx^*}{dt}$. Note that the variable $\frac{dx}{dt}$ corresponds to the output of active integrator $U_2$. A precision clipping circuit (Lakshmanan and Murali 1996, Maddock and Calcutt 1997) as depicted in the dotted box in Figure 6.1 is employed for threshold control mechanism. The component values for the control circuit are chosen to be: opamp = AD712, diode = IN4148, load resistor = 1 KΩ. The threshold reference voltage $E$ in the Figure 6.1 is $\frac{dx^*}{dt}$ and the threshold controlled signal is $V_T$. It corresponds to $\frac{dx}{dt}$ when $\frac{dx}{dt} \leq \frac{dx^*}{dt}$ and is $\frac{dx^*}{dt}$ when $\frac{dx}{dt} > \frac{dx^*}{dt}$. Note that, $\frac{dx}{dt}$ has been used for control here, the other variables are suitable for threshold control as well. But since $\frac{dx}{dt}$ is symmetric about 0, it is more useful for designing the computing module application.

Both experiments and numerical simulations of equation (6.2), which are in complete agreement, show that this scheme successfully yields regular stable cycles under a very wide range of thresholds. A representative example with threshold set at 0.1 V is displayed in Figure 6.2, which depicts the controlled cycle in the ($x$- $\frac{dx}{dt}$) plane of the circuit shown in Figure. 6.1 corresponding to equation (6.2).
Figure 6.2  (a) Uncontrolled chaotic attractor for $E (= \frac{dx}{dt}) = 5V$ is set beyond the limits of the attractor; thus no control occurs. Here ($x_dot = \frac{dx}{dt}$), (b) Controlled attractor for threshold=$E= 0.1V$, in the $(x-\frac{dx}{dt})$ plane corresponding to equation (6.2)
This control method is especially useful to the situation where one wishes to design controllable components that can switch flexibly between different regular behaviors. Also, the simplicity of the controller implies low complexity costs, which is important in technical applications seeking to exploit the richness of chaos in a direct and efficient way. Now, in the following section, one particular application, namely, the realization of the fundamental NOR gate has been shown. A system is capable of universal general purpose computing if it can emulate a NOR or NAND gate since any logic gate can be obtained by an adequate connection of NOR/NAND gate (i.e., any Boolean circuit can be built using a NOR (NAND) gate) (Mano 1993). So this experiment constitutes a proof-of-principle demonstration of the universal computing capability of chaotic circuits. Further, since this circuit is suitable for monolithic integrated circuit implementation, it is of considerable practical significance.

### 6.3 SCHEME FOR OBTAINING THE FUNDAMENTAL NOR GATE WITH A CHAOTIC CIRCUIT

The basic NOR logic operation on a pair of inputs simply involves the setting of an inputs-dependent threshold, namely the threshold is \( dx/dt + I_1 + I_2 \). In this experiment \( I_{1,2} = 0 \) V if logic input is 0 and \( I_{1,2} = 5 \) V if logic input is 1. So the reference threshold voltage \( E \), which is equal to \( dx/dt + I_1 + I_2 \), is

1. \( E = dx/dt + 0 \) V, for logic input set (0,0);
2. \( E = dx/dt + 5 \) V, for logic input set (0,1) / (1,0);
3. \( E = dx/dt + 10 \) V, for logic input set (1,1).

The logic output is interpreted from an actual output voltage \( V_0 \), obtained from the same threshold mechanism as follows: if \( dx/dt \leq E \), \( V_0 = 0 \) and logic
output is 0; when $dx/dt > E$, $V_0$ is $(dx/dt - E) = \delta$ and logic output is 1. The schematic diagram for the NOR gate implementation is depicted in Figure 6.3.

**Figure 6.3** Schematic diagram for implementing the NOR gate. Here $\dot{x} = dx/dt$

In this implementation the relationship between interpreted and actual values is the same for both input and output. Thus in this design the actual voltage associated with logic output 1, namely, $V_0 = (dx/dt - E) = \delta$, is 5 V, just as 5 V is associated with logic input 1. This will allow the output of one gate element to easily couple to another gate element as input, so that the gates can be wired directly into gate arrays implementing compounded logic operations.

Now for NOR gate implementation the following must hold true: (i) when both logic inputs are 0, the logic output is 1; (ii) when either one logic input is 1 and the other 0, the logic output is 0; (iii) when both logic inputs are 1, the logic output is 0. This is illustrated in the following truth table (Table 6.1).
Table 6.1 Truth Table for NOR gate

<table>
<thead>
<tr>
<th>Input 1</th>
<th>Input 2</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

In order to design the NOR gate, one has to use the knowledge of the dynamical evolution of the nonlinear system and reverse engineer and find the values of threshold that will satisfy and yield the above input-output association in a consistent and robust manner.

Figure 6.4 Circuit module for implementing the NOR gate. This circuit contains the double-scroll type chaotic circuit with a threshold control circuit. Here E corresponds to $\frac{dx}{dt} + I_1 + I_2$. AD712 VOA is used for $U_D$. 
In this implementation, the computing element (the “chaotic chip” so to speak) is the double-scroll type chaotic circuit. Figure 6.4 displays the actual circuit implementation of the scheme shown in Figure 6.3. The NOR gate is realized around \( \frac{dx^*/dt}{dt} \sim 0 \text{V} \). At this value \( \frac{dx^*/dt}{dt} \), the following responses are observed: (i) logic input set \((0,0)\), i.e., threshold level \( E = 0 \), yields \( V_0 = 5 \text{V} \), which is interpreted as logic output 1; (ii) logic input set \((0,1) / (1,0)\), i.e., threshold level \( E = 5 \text{V} \), yields \( V_0 = 0 \text{V} \), which is interpreted as logic output 0; (iii) logic input set \((1,1)\), i.e., threshold level \( E=10 \text{V} \), yields \( V_0 = 0 \text{V} \), which is interpreted as logic output 0. In the latter two cases the threshold is beyond the bounds of the chaotic attractor and so \( dx/dt \) is always less than the threshold, hence output is 0V. Figure 6.5 depicts representative timing sequences illustrating the NOR response.

Figure 6.5 Voltage timing sequences from top to bottom: (a) First input \( I_1 \), (b) Second input \( I_2 \), (c) Output \( V_T \), (d) Output \( V_0 \) and (e) Desired output \( R_T \) from \( V_0 \) corresponds to NOR \( (I_1, I_2) \) obtained by thresholding.
It is evident then that the knowledge of the dynamics has allowed to reverse engineer and obtain what threshold must be set in order to select out the temporal patterns emulating NOR gate.

### 6.4 NOR GATE IMPLEMENTATION WITH LORENZ SYSTEM

The success of this scheme has been further demonstrated by considering another chaotic system, namely, the Lorenz system. Lorenz (1963) derived this three dimensional system from a drastically simplified model of convection rolls in the atmosphere. It is given by the following set of three-coupled ordinary differential equations

\[
\frac{dx}{dt} = \sigma (y-x), \quad (6.3a)
\]

\[
\frac{dy}{dt} = rx - y - xz, \quad (6.3b)
\]

\[
\frac{dz}{dt} = xy - bz, \quad (6.3c)
\]

with parameters \(\sigma = 10, r = 28, b = 8 / 3\). The threshold action can be implemented on any one of the three variables of Lorenz system (x, y and z). Presently, x and y are considered separately for controlling the dynamics of the system. Therefore the variable x or y should not exceed the prescribed threshold values \(x^*\) or \(y^*\) respectively.

#### 6.4.1 State variable x of the system as the threshold signal

Consider the case of the threshold mechanism imposed on the x variable. Then equation (6.3) takes the form, when \(x > x^*\)

\[
\frac{dx}{dt} = \sigma (y - x^*), \quad (6.4a)
\]
\[ \frac{dy}{dt} = r x^* - y - x^* z, \]  
(6.4b)

\[ \frac{dz}{dt} = x^* y - b z. \]  
(6.4c)

and there is no controlling action when \( x \leq x^* \). As mentioned earlier in this chapter, this scheme successfully yields regular stable cycles under a very wide range of thresholds. Figure 6.6 shows MATLAB simulink model, i.e., a numerical scheme to implement a NOR gate operation. The NOR gate is realized around \( x^* \sim 0 \) and \( \delta = 25 \). For this value of \( x^* \), the following cases are observed. For inputs (0,0) the threshold level is at 0, which yields \( Z \sim 25 \); for inputs (1,0) or (0,1), the threshold level is at 25, which yields \( Z \sim 0 \); and for inputs (1,1), the threshold level is at 50, which yields \( Z=0 \) as the threshold is beyond the bounds of the chaotic attractor. This is illustrated in the timing sequences diagram as depicted in Figure 6.7.
Figure 6.6  MATLAB simulink model for implementing NOR gate by thresholding x variable
6.4.2 State variable $y$ as the threshold signal

When the threshold mechanism is applied to the $y$ variable, then equation (6.3) takes the form, when $y > y^*$

$$\frac{dx}{dt} = \sigma ( y^* - x ), \quad (6.5a)$$
$$\frac{dy}{dt} = r x - y^* - x z, \quad (6.5b)$$
$$\frac{dz}{dt} = x y^* - b z. \quad (6.5c)$$

There is no controlling action when $y \leq y^*$. In this case $y^* \sim 0$ and $\delta = 30$. The MATLAB simulink model and its timing sequence diagram are shown in Figures 6.8 and 6.9 respectively.
Figure 6.8 MATLAB simulink model for implementing NOR gate by thresholding y variable
Figure 6.9 Timing sequences from top to bottom: (a) First input $I_1$, (b) Second input $I_2$, (c) Output $V_T$, (d) Output $V_O$ and (e) Desired output $R_T$ from $V_O$ corresponds to NOR ($I_1, I_2$) obtained by thresholding $y$

6.5 SEQUENTIAL GATE: SR FLIP-FLOP

A sequential circuit can be defined as a circuit whose output depends not only on the present inputs, but also on the past outputs, i.e., a circuit that contains at least one memory element. A device that exhibits two stable states is extremely useful as a memory element in a binary system. Any electrical circuit that has this characteristic falls into the category of the devices, known as flip-flop. The most basic type of flip-flop is the set / reset (SR) flip-flop. This can be built by using either two NOR gates or two NAND gates. Each flip-flop has two outputs $Q$ and $Q'$. When $Q = 1$ and $Q' = 0$, the flip flop is said to be set. When $Q = 0$ and $Q' = 1$, the flip-flop is said to be reset. The schematic diagram of SR flip-flop using two NOR gate is given in Figure 6.10.
The binary truth table for the SR flip-flop is shown in Table 6.2.

<table>
<thead>
<tr>
<th>Set</th>
<th>Reset</th>
<th>Output Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Previous value</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Illegal (not allowed)</td>
</tr>
</tbody>
</table>

### 6.5.1 Chaos based Set-Reset (SR) flip-flop

Apart from realizing chaos-based combinational gates, it is interesting to construct chaos based sequential gate (Cafagna and Grassi 2006). Now by considering chaotic Lorenz system based NOR gates, Set-Reset flip-flop has been constructed. The two Lorenz systems with x threshold and with NOR logic operation are cross coupled, i.e., the output of first NOR gate is connected to one of the inputs for second NOR gate and the output of second NOR gate is connected to the first NOR gate as shown in the
schematic diagram (Figure 6.10). The corresponding MATLAB simulink model is shown in Figure 6.11.

Figure 6.11  MATLAB simulink model for implementing Set-Reset (SR) flip-flop in two cross-coupled NOR gates via Lorenz systems. Here x-variable is used for thresholding.

The following results are elucidating the phenomenon of the cross-coupling between the implemented NOR gates which generate a chaos based SR flip-flop. By referring Figures 6.10 and 6.11, the rescaled outputs of individual NOR gate are fed back to the input stage and therefore generating Q’ and Q, whereas two separate logic inputs act as Set and Reset inputs.
The timing behaviors of the implemented SR flip-flop are shown in Figure 6.12. The results clearly show the effectiveness of the proposed implementation. The behavior related to the cases (set = 1, reset = 0) and (set = 0, reset = 1) can be easily derived from Figure 6.12. The case (set =1, reset = 1) is not allowed. The remaining combination of set = 0 and reset = 0 retains the previous output state. This approach has been extended to variable y thresholded NOR Lorenz systems and similar results have been reproduced.

![Figure 6.12 Timing sequences from top to bottom](image)

**Figure 6.12** Timing sequences from top to bottom: (a) Set input, (b) Reset input, (c) Output of NOR gate-1 VT₁, (d) Output of NOR gate-2 VT₂, (e) Output VO₁, (f) Output VO₂ and (g) Desired output Q from VO₁ corresponds to Set input.
6.6 CONCLUSION

In conclusion, the basic principles of threshold control, as well as the direct implementation of the fundamental NOR logic gate utilizing thresholded chaotic dynamics have been demonstrated for third order autonomous system and for Lorenz system. Also the construction of SR flip-flop (sequential gate) from two cross-coupled Lorenz system has been demonstrated. Thus a proof-of-principle experiment of the capability of chaotic systems for universal computing has been provided. Most importantly, due to the simple circuit implementation, by choosing suitable threshold and input levels, one can also easily emulate other logic gates, such as AND, OR, NOT, XOR and NAND from the same physical element, i.e., using the same double-scroll type chaotic circuit. Interestingly, this double scroll type chaotic circuit is suitable for monolithic integrated circuit implementation and also it can be used as a separate module, which can be easily hooked along with the conventional logic gates due to compatible logic input and out levels. Further optimization of the circuit for wide-linear-range, low-voltage or low-power applications can be done by suitable modifications in the basic transconductor cell. Also by extending this approach, one is able to construct suitable schemes with chaotic circuits for bit-by-bit arithmetic operations and implementation of computer memory design. So the methodology described here is universal and robust in that it allows realization of a general purpose computer architecture based on the functionality and configuration of chaotic circuits.