Summer Project Report - 2015

Binary logic using spatially patterned deaths in chemical oscillators

Submitted by

Kaushal Gianchandani
School of Physical Sciences
National Institute of Science Education and Research (NISER), Bhubaneswar
kaushal.g@niser.ac.in

Guided by

Dr. Shakti N. Menon
Prof. Sitabhra Sinha

Theoretical Physics
Institute of Mathematical Sciences (IMSc.)
Chennai

Abstract

A wide range of interesting spatiotemporal patterns have been obtained using the well studied micro-fluidic system of chemical oscillator. The transformation of Spatially Patterned Oscillator Death(SPOD) state is the most intriguing possibility of finding the so called 'logic-gates' in such assemblies. We show that such deterministic gates indeed exist. A simple 'parity checker', a NOT gate, an OR gate and a NOR gate have been easily obtained. The presence of the NOR gate essentially ensures the existence of universal computation. But the search for the NAND gate to make the circuits less cumbersome still continues.

Contents

List of Figures

Chapter 1

Introduction

Synchronisation amongst entities is a very commonly occurring phenomenon in nature, be it fireflies flashing in sync or the heart muscles contracting and relaxing. It has amused scientists and researchers for a very long time, and a lot of attempts have been made to understand the science behind synchronisation. One such attempt was to understand it using chemical reaction-diffusion systems or 'chemical-oscillators'[3]. Efforts have been made to draw an analogy between such systems of coupled chemical oscillators and living systems.

The **Belousov-Zhabotinsky** reaction or the **BZ** reaction is a reaction which has been studied extensively in context of chemical oscillators, both experimentally and theoretically. When the evolution of the reaction solution is studied in a micro-fluidic assembly[4], interesting spatial patterns are observed as the system evolves over time. One can actually observe, in-phase oscillations, anti-phase oscillations and the so called 'Turing-Patterns' as well. One of the simpler theoretical models which can be used to mimic these experimental results is the **FitzHugh-Nagumo** model.

Such diffusively coupled nano-oscillators can serve as a good analog for various biological processes. There is also a possibility of developing computational devices using such nano-oscillators. Some work has already been carried out. We try to explore such possibilities through the course of this report using the FitzHugh-Nagumo model.

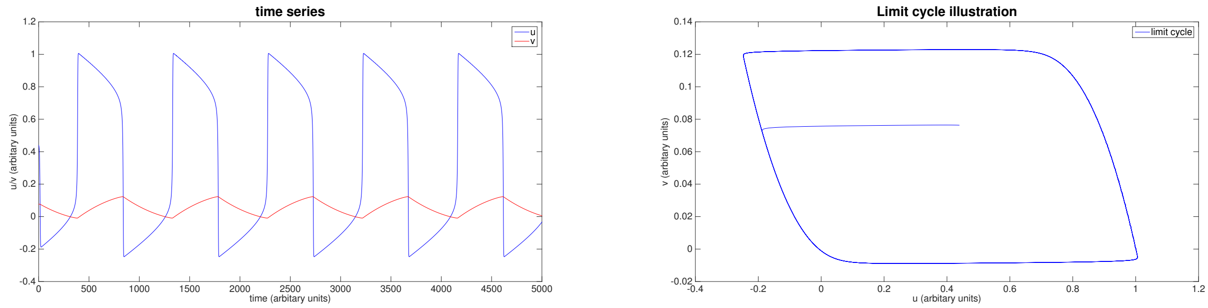
1.1 The FitzHugh-Nagumo(FHN) model

The FitzHugh-Nagumo(FHN) model was aimed at excitation and propagation from electrochemical properties of Na and K ion flow. The model can be used to study the signal transmission in firing neurons. The model consists of two variables viz. fast activation variable, u and slow excitation variable v. They are also known as the excitatory and inhibitory variables respectively. The excitatory variable has a cubic non linearity and the inhibitory variable has a linear behaviour. The set of differential equations describing the model is:

$$\dot{u} = f(u,v) = u(1-u)(u-\alpha) - v \tag{1.1a}$$

$$\dot{v} = g(u,v) = \epsilon(ku - v - b) \tag{1.1b}$$

where the parameters α and k describe the kinetics, ϵ characterises the recovery rate and b is a measure of the asymmetry of an entity in the system.



(a) Time series of a single oscillator

(b) Limit cycle for a single oscillator

Figure 1.1: Time series and Limit Cycle illustrations for a single oscillator described by FHN model

1.2 Modelling of Coupled Oscillators

The model taken into consideration is a system of N relaxation oscillators interacting with each other in some specific topological configuration. The dynamics of these individual oscillators are dictated by the FHN equations (refer to equation 1.1). The values of the parameters use are: $\alpha = 0.139$, $k = 0.6$ and $\epsilon = 0.001$. In the model the oscillators in the nearest neighbourhoods are diffusively coupled via the inactivation variable v. Dynamics of the resulting system are described by the following equations:

$$\dot{u}_i = f(u_i, v_i) \tag{1.2a}$$

$$\dot{v}_i = g(u_i, v_i) + D_v(v_{i-1} + v_{i+1} - 2v_i) \tag{1.2b}$$

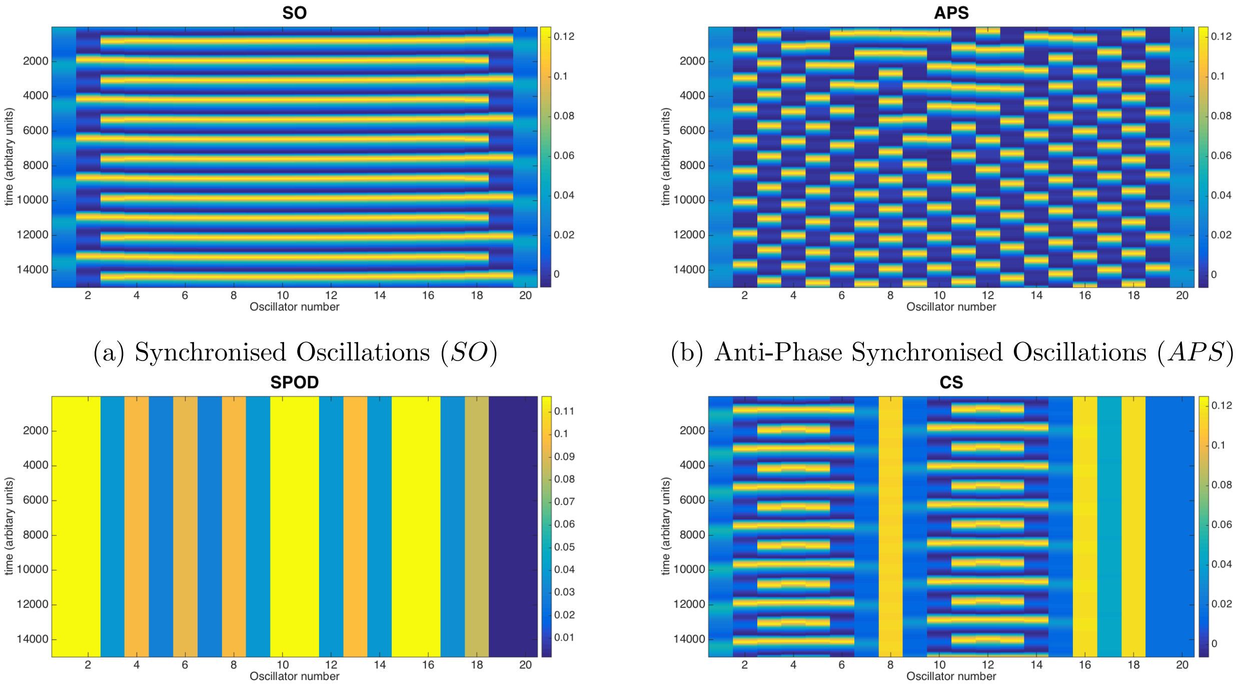
$$\dot{v}_0 = D_v(v_1 - v_0) \tag{1.2c}$$

$$\dot{v}_{N+1} = D_v(v_N - v_{N+1}) \tag{1.2d}$$

Here $i = 1, 2, ..., N$, D_v represents the strength of the coupling between neighbouring relaxation oscillators through the inhibitory variables. For various regions in the $b - D_v$ domain, certain interesting spatiotemporal patterns are obtained. They can be broadly classified into 4 categories.

- Synchronised Oscillations (SO): The oscillators, except for the ones at the boundaries, are in the same phase.
- Anti-phase Synchronisation (APS): The adjoining oscillators are in the opposite phase.
- Spatially Patterned Oscillator death (SPOD): Oscillators stay arrested in various stationary states or modes.
- Chimera states (CS): Oscillators can exhibit a combination of two or more of the states discussed.

Figure 1.2 illustrates the patterns mentioned above.



(a) Synchronised Oscillations (SO)

(b) Anti-Phase Synchronised Oscillations (APS)

(c) Spatially Patterned Oscillator Death (SPOD)

(d) Chimera State Oscillations (CS)

Figure 1.2: Spatiotemporal evolution of a system of 20 oscillators

Different dynamical regimes[2] of such an array of oscillators has been explored in detail in the past. Most of the initial conditions result in some particular states for specific regions in the $b - D_v$ domain.

For the purpose of computation, SPOD state is the most apt state for some obvious reasons.

- Information can easily encoded into the SPOD states in the form of binaries. One can assign the value 1 to the oscillators which are in high mode and 0 to the oscillators in low mode. The vice-versa is certainly valid too.

- The SPOD state remains steady with evolution of time unless there is a change induced/triggered externally. This is yet another desirable property which one would require for computation, i.e. if no external stimulus is provided, the information stored shouldn't change with time.

Such properties of the SPOD state tempt one to consider that computation is possible using such states. However, the basic essence of computations lies in predictability, given sufficient information about the system, the input(s) and the procedure of computations, one should be able to predict the output without actually implementing the procedure via some experiments/simulations. Also, the output should be consistent, i.e. it should be independent of the surroundings and should depend only on the input(s) and the procedure to evaluate it. Such a discussion might seem quiet ambiguous at this point of time, but further in the course of this report, these ambiguities will be resolved.

Chapter 2

Implementation of SPOD logic

A logic gate refers to a physical device which can implement a boolean function, i.e. it takes in two or more logic inputs and gives a logical output. In other words, it takes an input state and transforms it to some other state. A few studies[1] carried out in the past have indicated that there is a possibility of finding such logic gates in these chemical oscillators. The idea is to transform a particular system state in the SPOD regime via some local stimulations/disruptions in the system added externally. It wouldn't be overstating to say that there might be an analogy between such stimulation and the different logic gates.

2.1 Finding appropriate values for various parameters

It is most amusing to note that the set of equations 1.1 are the source of obtaining such interesting spatiotemporal patterns. To implement any sorts of computation logic using a SPOD state, one essentially needs to find appropriate values of control parameters which govern the dynamics of the system of chemical oscillators.

2.1.1 The $b - D_v$ domain

Sufficient information is available about the parameters α, k and ϵ associated with the set of differential equations 1.1. Since the SPOD state can be obtained for only certain regions in the $b - D_v$ space, we can take the values of b and D_v to be in the vicinity 0.16 and 0.002 It is important to know that the choice of these values is not unique. Any other values are also appropriate as long as it is ensured that the system remains in the SPOD state regime upon application of the 'stimulation'.

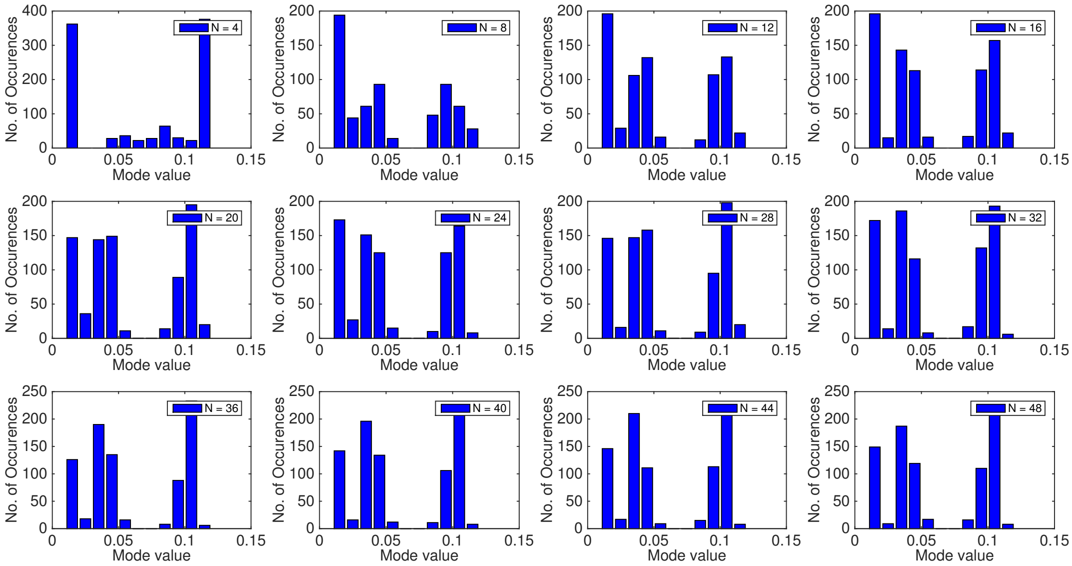
2.1.2 No. of Oscillators and the threshold between 0 and 1

To make the choice of the no. of oscillators (N) required to fulfil the purpose of computations is a tricky task for two reasons.

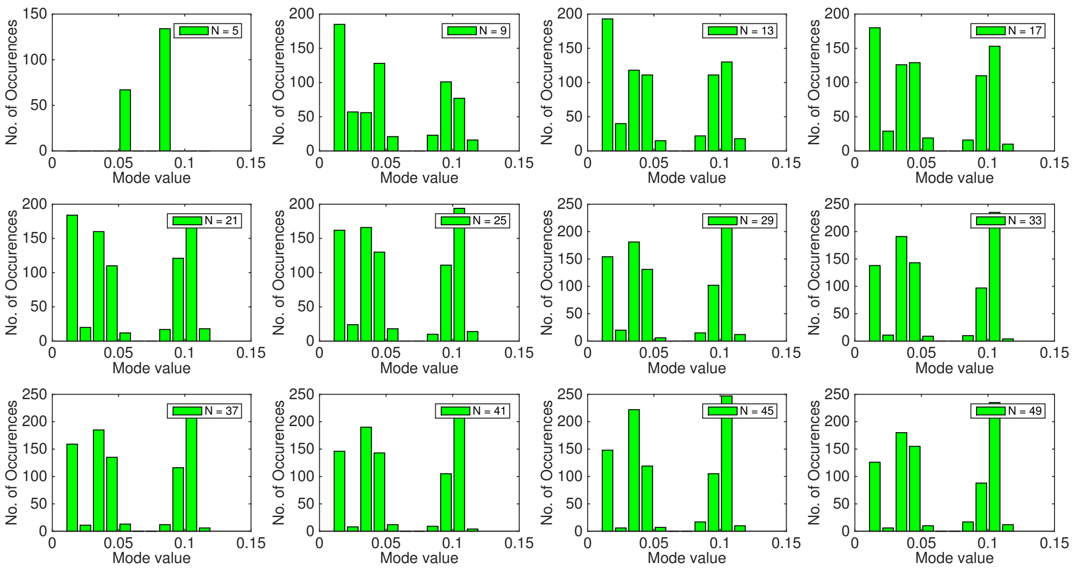
1. It is necessary to make sure that the choice of N is such that the system becomes independent of its orientation i.e. an array of say n oscillators arranged in a linear fashion should behave similar to a ring comprising of n oscillators.

2. Since, the information is to be encoded in the form of binaries and the values which modes of the SPOD state attain can be more than 2, one needs to assign a **threshold** below which the mode would be considered to be in mode 0 and above which it would be considered to be in mode 1.

The two problems mentioned above can be tackled easily by finding out what steady values of modes do the oscillators obtain for different N and randomly chosen initial conditions. One can expect a clear inclination towards 2 values or rather 2 ranges of values, the mean of these 2 ranges of values can be considered the *threshold* between 0 and 1. Also if this ‘threshold’ doesn’t vary much with change in N, it can be assumed safely that the system has become independent of its orientation.



(a) Occurrence histograms for even no. of oscillators



(b) Occurrence histograms for odd no. of oscillators

Figure 2.1: Histograms for Occurrence of various mode values for different N

From the histograms illustrated above, it is very evident that there is a distinct difference between the ‘high’ and the ‘low’ values which the modes attain. An appropriate threshold was taken to be 0.07. Thus any mode with value above 0.07 was considered 1 and and mode with value less than 0.07 was considered 0. Also it was found that any no. of oscillators more than $N = 8$ can be considered as independent of their orientation, moreover the threshold is also independent of the orientation. Keeping this in mind, we have used the values $10, 11, 20$ and 21

for N, through the course of this work.

2.1.3 Characteristics of the stimulation

It is of utmost necessity that various characteristics associated with the stimulation such as amplitude, duration and channels stimulated are declared in advance. Also they should remain the same for a particular type of 'procedure' which is being implemented. In other words, these shouldn't vary for any of the inputs viz. 00, 01, 10 and 11. This is necessary to ascertain that the so called "logic gates" which one is attempting to construct are consistent.

It is very evident that there are a large number of combinations in which the channels to be stimulated can be chosen. There aren't any suitable parameters to evaluate or quantify the robustness of a particular choice. The closest one can come to justifying a particular choice would be to argue that the output for all the inputs (00, 01, 10 and 11) are consistent i.e. the same output should be obtained in every interaction or at least in a significantly large number of iterations, regardless of the neighbourhood of the input channels.

Although it might seem that it is impossible to find an appropriate configuration for the stimulation, it isn't completely true. It is, without any doubt, equivalent to looking for a needle in a haystack, but one could essentially build a magnet to do the job.

In order to understand the working of a stimulation scheme, we started with the very basic types of stimulations. We considered stimulating only the input channels. The next step was to fix the other characteristics associated with the stimulation i.e. the amplitude(A_s) and duration(Δt_s) for which it is provided.

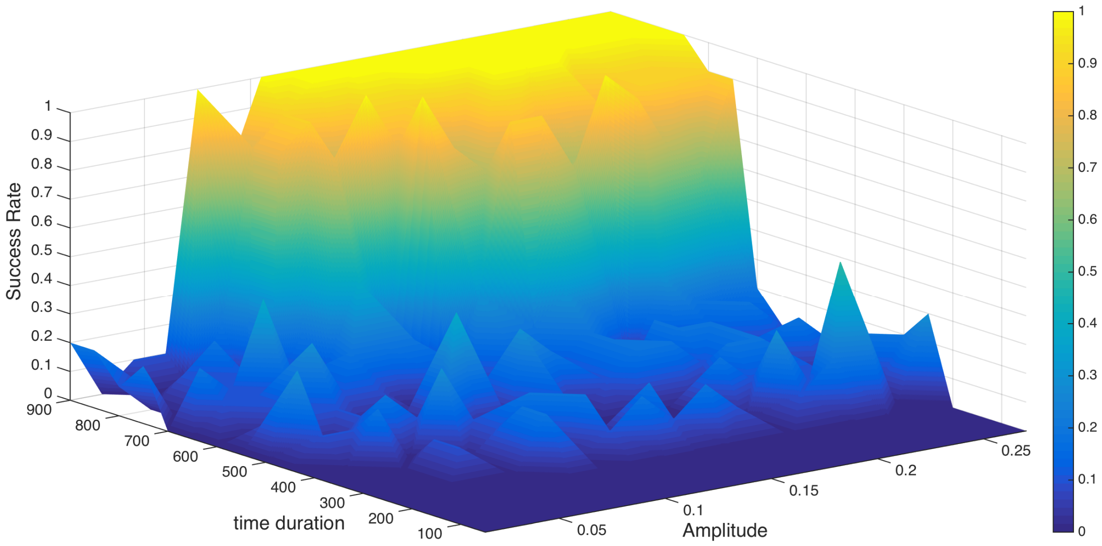
Let us define two sets, N_{inp} and N_{sim}, such that the set N_{inp} is the set of channels which act as our input channels and the set N_{sim} is the set of all channel(s) which is(are) stimulated. The differential equations defining the system get modified as:

$$\dot{v_0} = D_v(v_1 - v_0) \tag{2.1a}$$
$$\dot{v}_{N+1} = D_v(v_N - v_{N+1}) \tag{2.1b}$$
$$\dot{u_i} = f(u_i, v_i) \tag{2.1c}$$
$$\dot{v_i} = \begin{cases} g(u_i, v_i) + D_v(v_{i-1} + v_{i+1} - 2v_i) + A_s, & \text{if } i \in N_{sim} \text{ and } t_s \in \Delta t \\ g(u_i, v_i) + D_v(v_{i-1} + v_{i+1} - 2v_i), & \text{otherwise} \end{cases} \tag{2.1d}$$

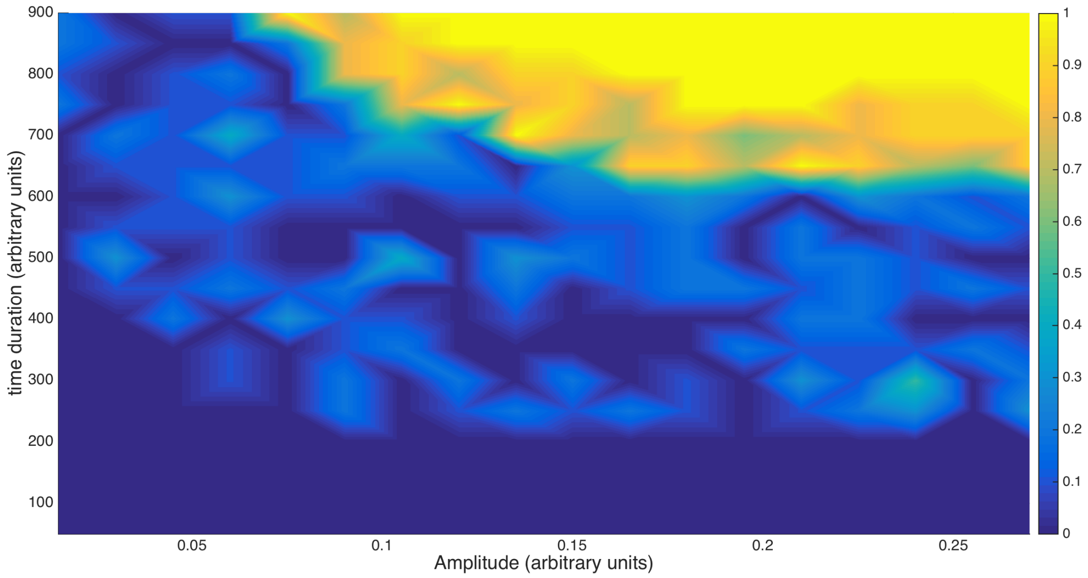
It is necessary to note that the choice of N_{sim} is independent of the elements in N_{inp}.

For a set of trials the following set of parameters were chosen: $N = 20$, $N_{inp} = \{10, 11\}$. A large number of initial conditions were generated and stored for the 4 inputs $00, 01, 10$ and 11. These conditions were generated using a random number generator. In a large number of iterations, it was found that the probability of occurrence of 01 and 10 was much greater than that of 00 and 11. It is a very robust indication of the fact that the system prefers 01 or 10 over 00 or 11. It was found that the channel 00 transformed to 10 for $N_{sim} = \{09, 10\}$.

For a particular N_{sim}, success rate is defined as the ratio of the number of times an input state transformed to the output state and the total number of iterations, for a particular A_s and Δt_s. The success rate for different combinations of amplitude(A_s) and time durations(Δt_s) was calculated and the results obtained have been illustrated in the figure 2.2.



(a) Surface plot for Success Rate for input 00 → 10



(b) Success Rate for different combination of A_s and Δt_s for input 00 → 10

Figure 2.2: Success rate as a function of A_s and Δt_s

2.2 What does a "logic gate" mean in terms of a SPOD state?

It can be safely said that appropriate values for the set of control parameters can be determined using simple arguments. We now need to address the "elephant in the room". What does a "logic gate" mean in the context of a SPOD state. The following list summarises everything which has been stated till now about this so called "SPOD logic".

- Any SPOD state can be expressed in terms of binaries, 0 and 1.
- For an appropriate set of parameters, such stimulation(s) exist which can transform one SPOD state to another.

- Such stimulations are consistent, i.e. an input get transformed to the same output in a significantly large number of iterations, if the procedure is repeated. Also the output is independent of the neighbourhood of the input.

All these characteristics imply that such stimulations can be regarded as "logic gates" in the SPOD regime.

Chapter 3

Analysis

3.1 The Parity Checker

A set of values of different driving parameters associated with the system was determined. These values have been recorded in table 3.1. This section gives a detailed analysis of the sort of analysis which was carried out using these values.

S.No.	Parameter	Values(in arbitrary units)
1.	N	20
2.	(b,D_v)	(0.16,0.002)
3.	N_{inp}	$\{10, 11\}$
4.	A_s	0.2
5.	Δt_s	600

Table 3.1: A set of values for various control parameters

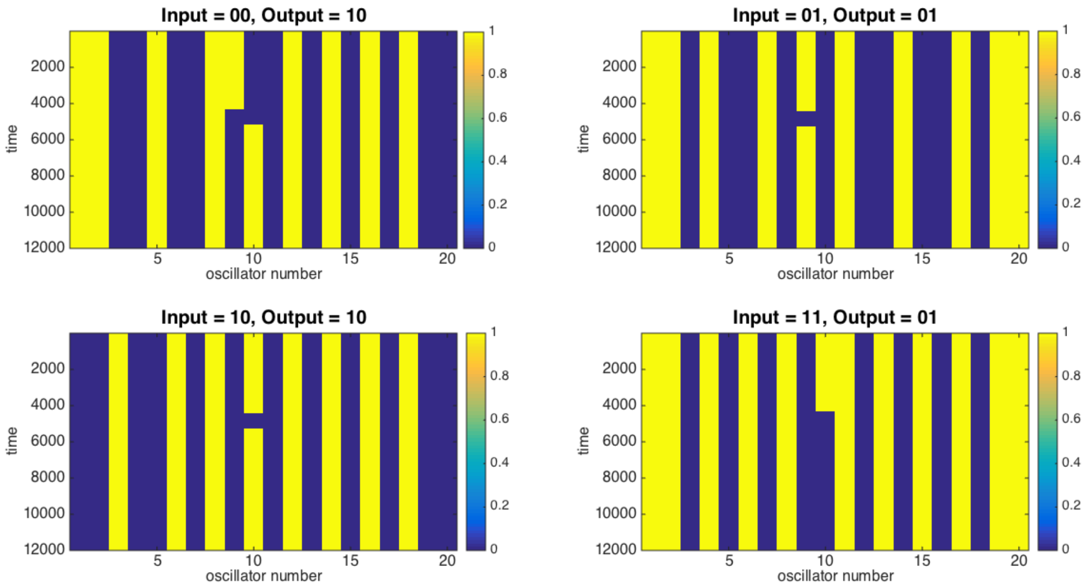
N_{inp} has been chosen to be closest at the centre so as to avoid any effects which might be there due to avoid any bias which might occur if the input channels are too close to the edges. However it is noteworthy that the analysis gave extremely similar results even when the input channels were chosen quiet close to the edges ($N_{inp} = \{04, 05\}$). Thus one can safely assume that the bias due to the edges either isn't very severe or it isn't very evident after a small displacement away from the edges. This can probably be attributed the fact that the the system is build on a nearest neighbour coupling condition. Figure 3.1 illustrate the transformations which occur in the input state for $N_{sim} = \{09, 10\}$ and $\{11, 12\}$ respectively. The results have been tabulated in the form of truth table 3.2 and 3.3

Table 3.2: Truth table for $N_{sim} = \{09, 10\}$

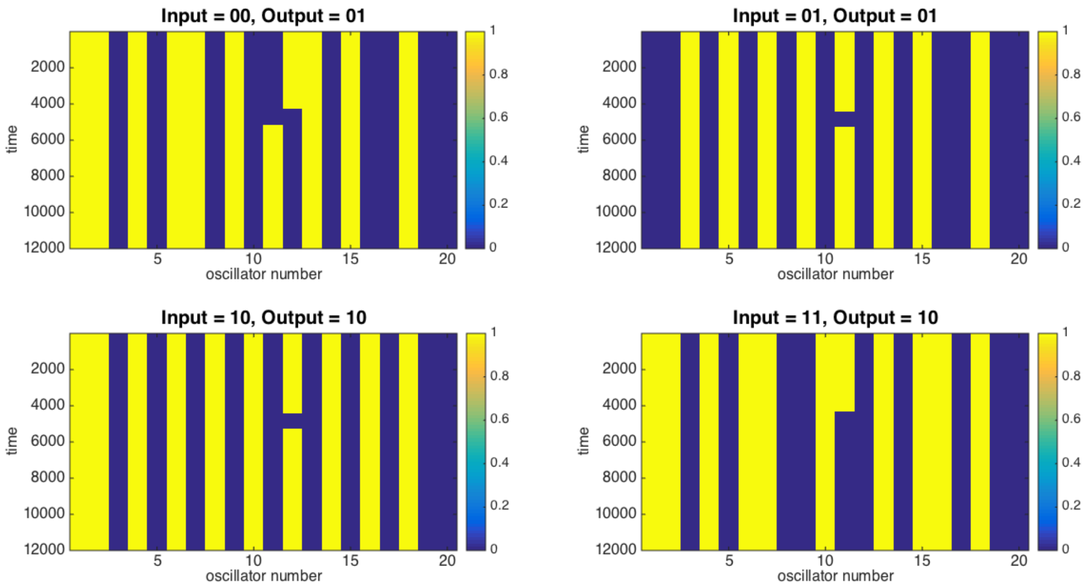
Input	Output
00	10
01	01
10	10
11	01

Table 3.3: Truth table for $N_{sim} = \{11, 12\}$

Input	Output
00	01
01	01
10	10
11	10



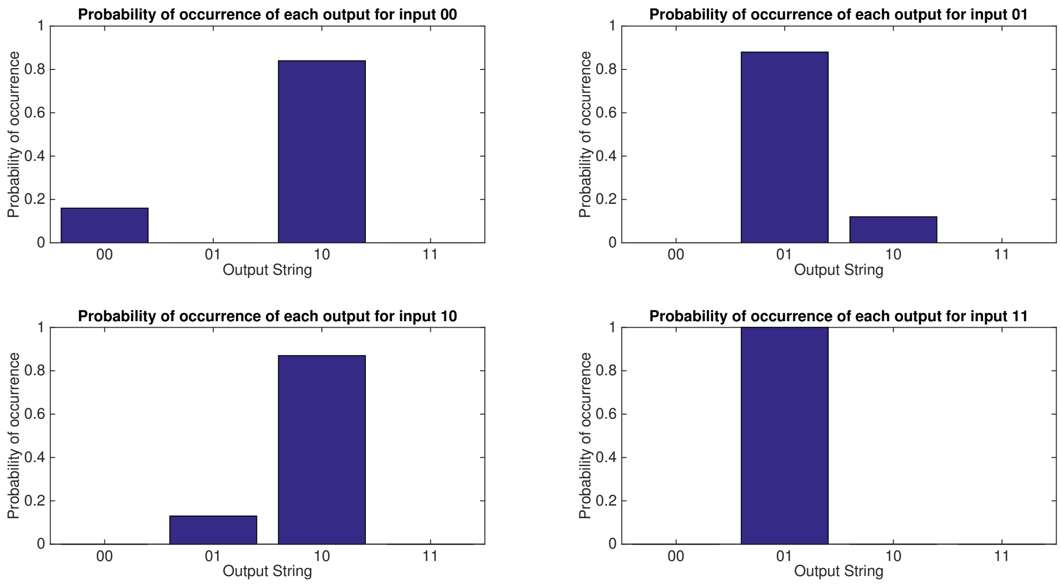
(a) Transformation of input states to output states for $N_{sim} = \{09, 10\}$



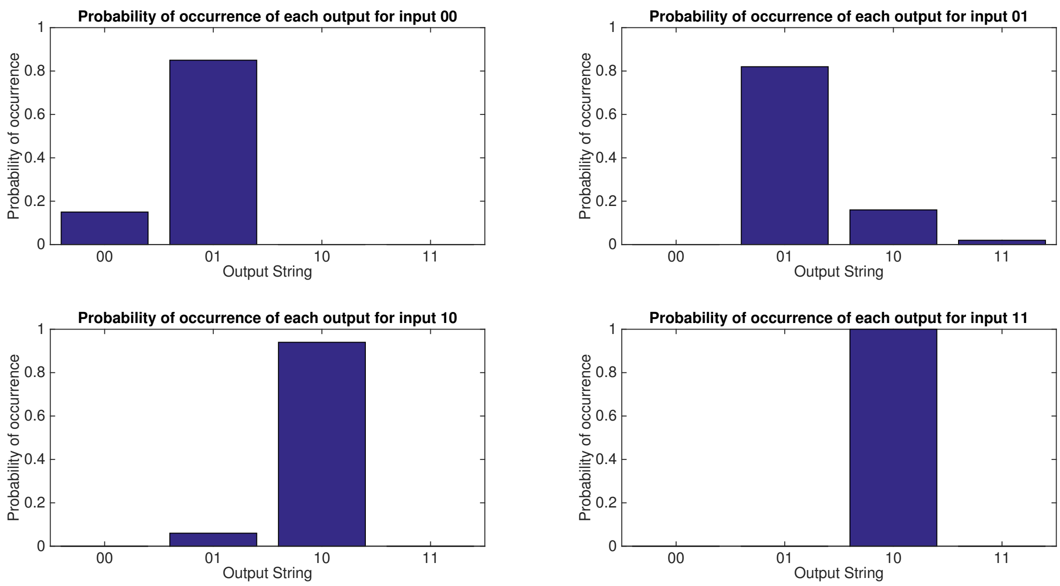
(b) Transformation of input states to output states for $N_{sim} = \{11, 12\}$

Figure 3.1: Transformation of input states to output states for two perturbation schemes

Figure 3.2a depicts the probabilities of occurrence of all the outputs for each given input.



(a) Probability of Occurrence of different outputs for various inputs for $N_{sim} = \{09, 10\}$



(b) Probability of Occurrence of different outputs for various inputs for $N_{sim} = \{11, 12\}$

Figure 3.2: Probability of occurrences of different output states for two perturbation schemes

The truth table 3.2 and 3.3 clearly show that this particular stimulation scheme can be used as a **parity checker**. One can determine the direction from which the stimulation is being provided by using the inputs 00 or 11.

Many different stimulation schemes were put to trial and their outputs have been tabulated.

Table 3.4: for $N_{sim} = \{09\}$

Input	Output
00	00
01	01
10	10
11	01

Table 3.5: for $N_{sim} = \{10\}$

Input	Output
00	00
01	01
10	10
11	01

Table 3.6: for $N_{sim} = \{11\}$

Input	Output
00	00
01	01
10	10
11	10

Table 3.7: for $N_{sim} = \{12\}$

Input	Output
00	00
01	01
10	10
11	10

Table 3.8: for $N_{sim} = \{08, 09\}$

Input	Output
00	10
01	01
10	10
11	01

Table 3.9: for $N_{sim} = \{12, 13\}$

Input	Output
00	01
01	01
10	10
11	10

Table 3.10: for $N_{sim} = \{08, 10\}$

Input	Output
00	00
01	01
10	10
11	01

Table 3.11: for $N_{sim} = \{11, 13\}$

Input	Output
00	00
01	01
10	10
11	10

Table 3.12: for $N_{sim} = \{09, 11\}$

Input	Output
00	00
01	01
10	10
11	10

Table 3.13: for $N_{sim} = \{10, 12\}$

Input	Output
00	00
01	01
10	10
11	01

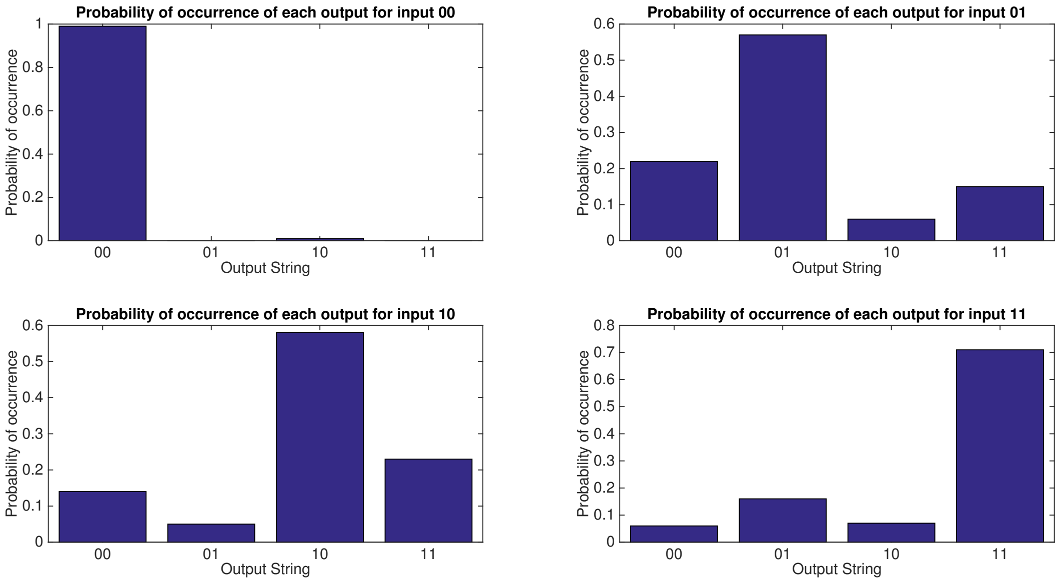
Table 3.14: for $N_{sim} = \{10, 11\}$

Input	Output
00	00
01	01
10	10
11	11

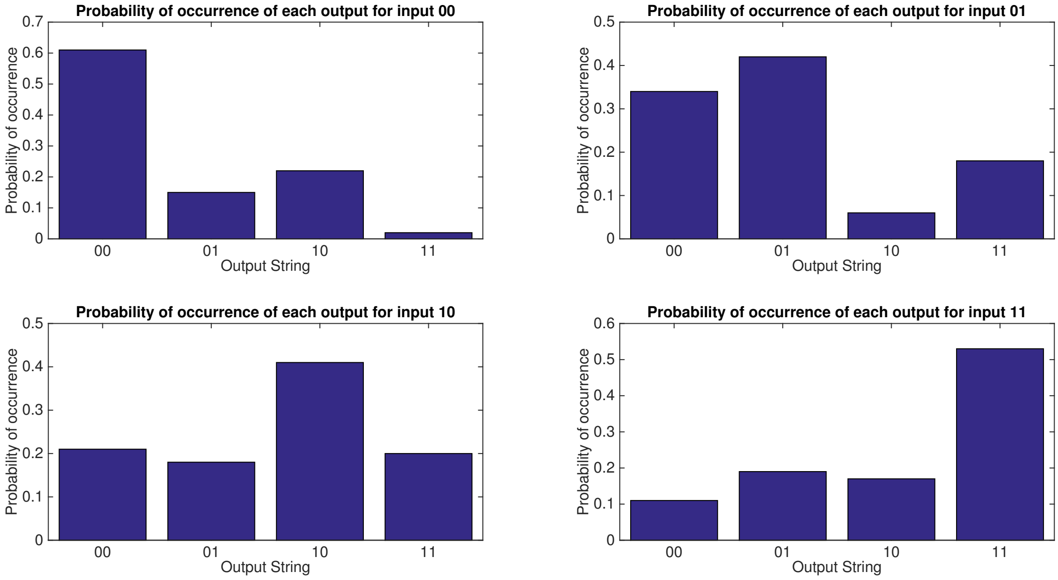
Table 3.15: for $N_{sim} = \{09, 10, 11, 12\}$

Input	Output
00	00
01	01
10	10
11	11

The probability distribution for $N_{sim} = \{10, 11\}$ and $N_{sim} = \{09, 10, 11, 12\}$ are depicted by the figure 3.3.



(a) Probability of Occurrence of different outputs for various inputs for $N_{sim} = \{10, 11\}$



(b) Probability of Occurrence of different outputs for various inputs for $N_{sim} = \{09, 10, 11, 12\}$

Figure 3.3: Transformation of input states into outputs states with a large basin for all the output states

Although figure 3.3 might not seem very enlightening, one can still take away something positive. If we consider our output channel to be only channel 11, we see that 01, 11 and 00, 10 are equivalent. Thus the modified truth tables can be written as:

Table 3.16: for $N_{sim} = \{10, 11\}$

Input	Output
00	0
01	1
10	0
11	1

Table 3.17: for $N_{sim} = \{09, 10, 11, 12\}$

Input	Output
00	0
01	1
10	0
11	1

If we are somehow able to modify the stimulation scheme such that the state 10 transforms to 01 or 11 majority of the times, we would have essentially obtained an OR gate.

3.2 The OR Gate

The multiple attractor nature of certain configuration was further investigate by concentrating selectively on one channel as output. Also, A_s and Δt_s was also increased a bit to check if that forces the system to move towards a less number of attractors. The parameters used for further simulations are tabulated in table 3.18.

S.No.	Parameter	Values(in arbitrary units)
1.	N	20
2.	(b,D_v)	(0.16,0.002)
3.	N_{inp}	$\{10, 11\}$
4.	A_s	0.225
5.	Δt_s	1000

Table 3.18: A set of values for various control parameters for an OR gate

Table 3.19 is the truth table obtained for $N_{sim} = \{1, 3, 5..., 19\}$.

Table 3.19: for $N_{sim} = \{1, 3, 5..., 19\}$

Input	Output
00	0
01	1
10	1
11	1

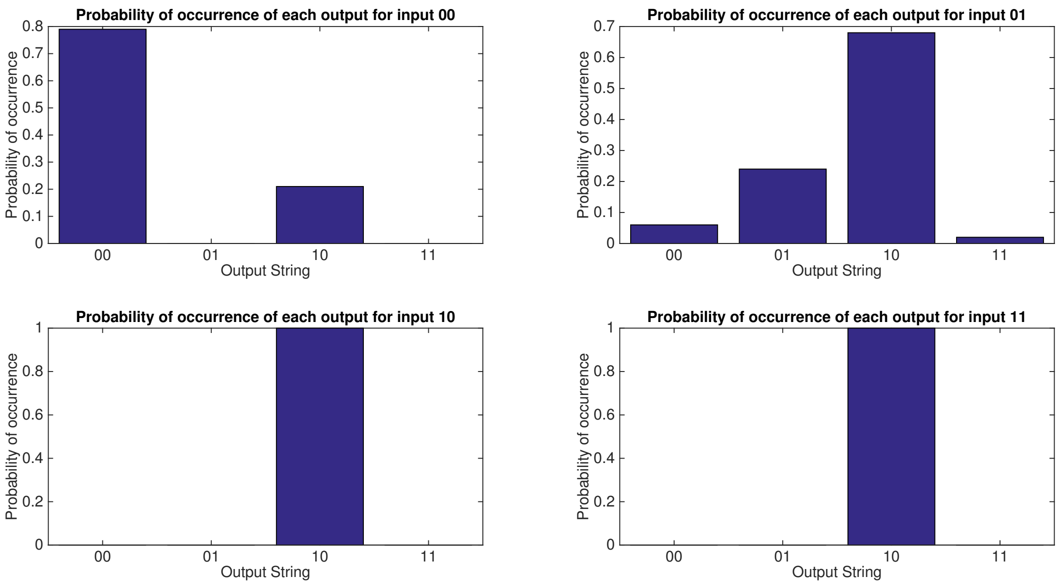


Figure 3.4: Probability of Occurrence of different outputs for various inputs for $N_{sim} = \{1, 3, 5, ..., 19\}$

The figure 3.4 clearly indicates that the system is more inclined towards a particular basin and an OR gate is obtained if we concentrate only on the left channel as output. Many other configurations were tried and the results have been documented as truth tables. Let $n_o = 1, 3, 5, 7$ and $n_e = 14, 16, 18, 20$.

Table 3.20: for $N_{sim} = \{n_o, 9, n_e\}$

Input	Output
00	00
01	01
10	10
11	01

Table 3.21: for $N_{sim} = \{n_o, 12, n_e\}$

Input	Output
00	00
01	01
10	10
11	10

Table 3.22: for $N_{sim} = \{n_o, 10, n_e\}$

Input	Output
00	00
01	01
10	10
11	01

Table 3.23: for $N_{sim} = \{n_o, 11, n_e\}$

Input	Output
00	00
01	01
10	10
11	10

Table 3.24: for $N_{sim} = \{n_o, 9, 10, n_e\}$

Input	Output
00	10
01	01
10	10
11	01

Table 3.25: for $N_{sim} = \{n_o, 11, 12, n_e\}$

Input	Output
00	01
01	01
10	10
11	10

Table 3.26: for $N_{sim} = \{n_o, 9, 11, n_e\}$

Input	Output
00	00
01	10
10	10
11	10

Table 3.27: for $N_{sim} = \{n_o, 10, 12, n_e\}$

Input	Output
00	00
01	01
10	01
11	01

Table 3.28: for $N_{sim} = \{n_o, 9, 12, n_e\}$

Input	Output
00	00
01	01
10	10
11	11

Table 3.29: for $N_{sim} = \{n_o, 10, 11, n_e\}$

Input	Output
00	00
01	01
10	10
11	11

Table 3.30: for $N_{sim} = \{n_o, 9, 10, 11, n_e\}$

Input	Output
00	10
01	11
10	01
11	11

Table 3.31: for $N_{sim} = \{n_o, 9, 10, 12, n_e\}$

Input	Output
00	10
01	01
10	01
11	01

Table 3.32: for $N_{sim} = \{n_o, 9, 11, 12, n_e\}$

Input	Output
00	01
01	10
10	10
11	10

Table 3.33: for $N_{sim} = \{n_o, 10, 11, 12, n_e\}$

Input	Output
00	01
01	01
10	11
11	11

Table 3.34: for $N_{sim} = \{n_o, 9, 10, 11, 12, n_e\}$

Input	Output
00	11
01	11
10	11
11	11

Upon considering only the left channel as the output, it was found that table3.26 are the same as that of an OR gate. It is also noteworthy that table 3.31 is the table for an OR gate on the left channel and that of a NOR gate on the right channel. Table 3.32 is the table for a NOR gate when the left channel is observed as the output and a table for an OR gate when the right channel is considered as the output. This behaviour is quiet expected because of the symmetry of the channel choices, $\{n_o, 9, 10, 12, n_e\}$ and $\{n_o, 9, 11, 12, n_e\}$

3.3 The NOT gate

While looking for the universal gate, some similar gates were found without much efforts. One of those gates is the NOT gate. If the truth table 3.8 is revisited, a NOT gate can easily be spotted. On considering the right channel as the input and the left channel as the input, the NOT gate is quiet evident

Table 3.35: for $N_{sim} = \{8, 9\}$

Input	Output
00	10
01	01
10	10
11	01

Table 3.36: for $N_{sim} = \{8, 9\}$

Input	Output
0	1
1	0
0	1
1	0

It won't be an over statement to say that the gates, the parity checker, the NOT gate and the OR gate have been successfully constructed. Such results definitely are a positive indicators of the fact that construction of logic gates using SPOD state is not a far fetched possibility.

3.4 Strength Parameters

It was found that more than one configurations of stimulation(N_{sim}) could be used to construct the same gate, in this case one would like to select the most consistent choice of N_{sim}. In order to quantify the consistency of a gate, two strength parameters have been introduced, viz. ζ_m and ζ_t.

In a random process it is most likely that a user, wishing to do some computation, will provide any of the possible inputs with equal probabilities, i.e. the chance of 01 being an input is the same as 00 being an input. Thus we can say that all the inputs are equally likely and the probability p_i with which they can occur is $\frac{1}{4}$. It is well established that the outputs corresponding to each inputs have some definite probability of occurrence for each of these inputs.

The strength parameter ζ_m is defined as follows:

$$\zeta_m = p_i.p_s^{00} + p_i.p_s^{01} + p_i.p_s^{10} + p_i.p_s^{11} \tag{3.1}$$

$$\implies \zeta_m = p_i(p_s^{00} + p_s^{01} + p_s^{10} + p_s^{11}) \tag{3.2}$$

Since $p_i = \frac{1}{4}$ equation 3.2 can be rewritten as:

$$\zeta_m = \frac{1}{4}(p_s^{00} + p_s^{01} + p_s^{10} + p_s^{11}) \tag{3.3}$$

Where p_s^{xx} is probability that the desired output corresponding to each of the inputs is obtained, i.e. if an AND gate is to be constructed $p_s^{00}, p_s^{01}, p_s^{10}$ would take value 1 if the output is 0, and p_s^{11} would take the value 1 if the output is 1.

Figures 3.5, 3.6 and 3.7 illustrate the different probabilities of occurrences corresponding to different outputs for each input.

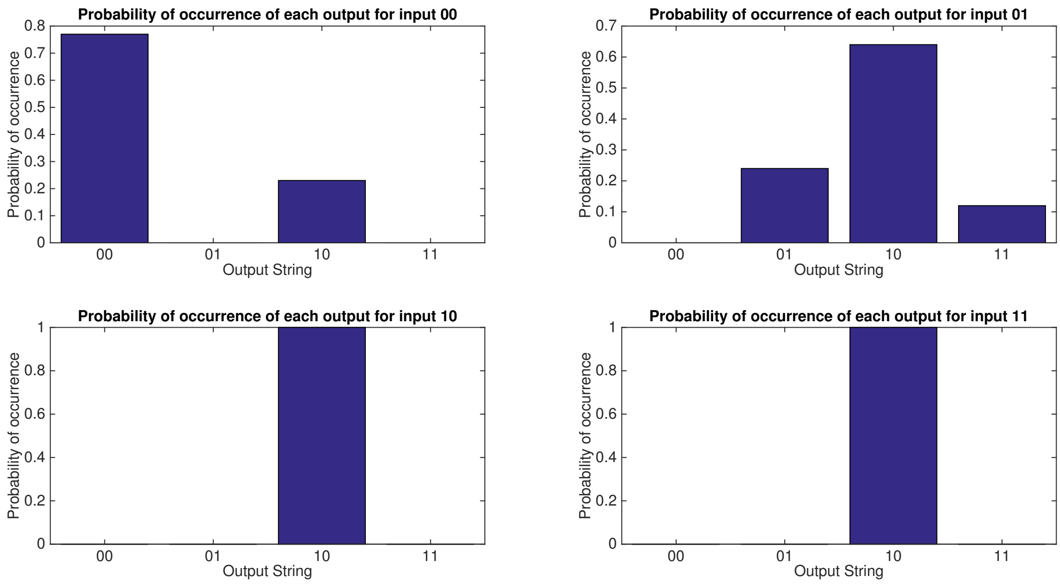


Figure 3.5: Probability of Occurrence of different outputs for various inputs for $N_{sim} = \{n_o, 9, 11, n_e\}$

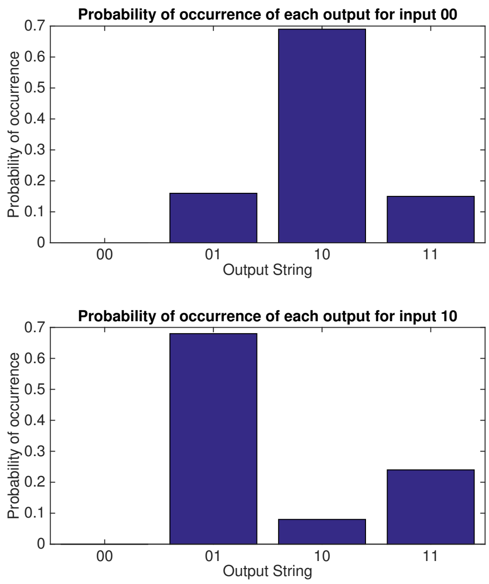


Figure 3.6: Probability of Occurrence of different outputs for various inputs for $N_{sim} = \{n_o, 9, 10, 12, n_e\}$

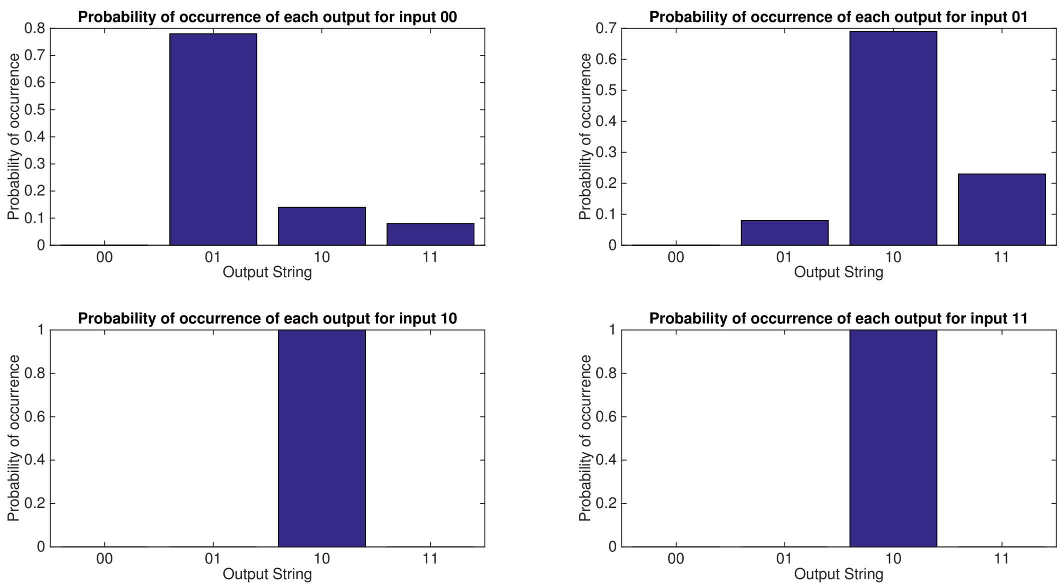


Figure 3.7: Probability of Occurrence of different outputs for various inputs for $N_{sim} = \{n_o, 9, 11, 12, n_e\}$

It should be noted that since only one channel is being considered as the output channel, all the outputs with the desired value are equivalent and their probabilities can be added. For the purpose of illustration, the calculation for the value of ζ_m has been done for $N_{sim} = \{n_o, 9, 11, n_e\}$, which is one of the OR gates, has been carried out.

For an OR gate the acceptable outputs corresponding to each input have been tabulated in table 3.37 and table 3.38 denotes the probabilities p_s^{xx}. This is valid when the left channel is the output channel.

Table 3.37: Acceptable outputs for each input for an OR gate

Input	Output	Acceptable Outputs
00	0	00/01
01	1	10/11
10	1	10/11
11	1	10/11

Table 3.38: Probability values for obtaining desired outputs for different inputs for $N_{sim} = \{n_o, 9, 11, n_e\}$

p_s^{xx}	Value
p_s^{00}	0.77
p_s^{01}	0.76
p_s^{10}	1.00
p_s^{11}	1.00

$$\begin{aligned}\zeta_m &= \frac{1}{4}(p_s^{00} + p_s^{01} + p_s^{10} + p_s^{11}) \\ &= \frac{1}{4}(0.77 + 0.76 + 1.00 + 1.00) \\ &= \frac{1}{4}(3.53) \\ \zeta_m &= 0.8825\end{aligned}$$

The strength parameter ζ_t for a gate is defined as the minimum probability with which the correct/desired output would be obtained for any of the input values.

$$\zeta_t = min\{p_s^{00}, p_s^{01}, p_s^{10}, p_s^{11}\} \tag{3.4}$$

The values of both the strength parameters have been recored in table 3.39 for different OR gates found so far.

N_{sim}	ζ_m	ζ_t
$\{n_o, 9, 11, n_e\}$	0.8825	0.76
$\{n_o, 9, 10, 12, n_e\}$	0.9025	0.69
$\{n_o, 9, 11, 12, n_e\}$	0.9225	0.78

Table 3.39: Strength parameters for different OR gate conigurations

It is quiet evident that the configuration $N_{sim} = \{1, 3, 5, 7, 9, 11, 12, 14, 16, 18, 20\}$ is clearly the most apt choice for an OR gate because the values of both ζ_m and ζ_t are high for it.

Chapter 4

Conclusion

This report barely begins to scratch the surface of the immense possibilities which lie in the field of unconventional computing using the SPOD logic. It was clearly established that the SPOD state can be encoded in terms of binaries 0 and 1 and the binary strings generated are robust. A large number stimulations exists which can transform one SPOD state to another for appropriate values of parameters. The outputs obtained after via such stimulations are fairly consistent.

A simple "parity checker" and a NOT gate were readily constructed without putting in any significant amount of efforts. The presence of the NOR gate essentially ensures universal computation. The exploration for a NAND gate still continues in order to make the circuits less cumbersome.

We strongly believe that much stricter definitions can be attributed to the characteristics of the stimulation. For instance, A_s can be expressed as a multiple of max amplitude attained by v and Δt_s can be expressed in multiples of oscillation periods for a single FHN oscillator. Certain interesting results can be obtained on exploring various other stimulation schemes, changing b instead stimulating the system seems to be very promising area to look into.

Another challenge which lies ahead is to determine if staggering of gates works. If so it would be a great leap towards a practical implementation of such gates.

Either way it is a really good start to a really promising area of interdisciplinary research for physicists, computer scientists and chemists. If chemical oscillators are turned into actual computing machines, the model can be translated to bio-chemical reactions thus enabling living systems to be treated as computers. The possibilities which lie ahead of that are truly unbounded.

Bibliography

[1] Shakti N Menon and Sitabhra Sinha. "defective" logic: Using spatiotemporal patterns in coupled relaxation oscillator arrays for computation. In *Signal Processing and Communications (SPCOM), 2014 International Conference on*, pages 1–6. IEEE, 2014.

[2] Rajeev Singh and Sitabhra Sinha. Spatiotemporal order, disorder, and propagating defects in homogeneous system of relaxation oscillators. *Physical Review E*, 87(1):012907, 2013.

[3] Masahiro Toiya, Hector O González-Ochoa, Vladimir K Vanag, Seth Fraden, and Irving R Epstein. Synchronization of chemical micro-oscillators. *The Journal of Physical Chemistry Letters*, 1(8):1241–1246, 2010.

[4] Masahiro Toiya, Vladimir K Vanag, and Irving R Epstein. Diffusively coupled chemical oscillators in a microfluidic assembly. *Angewandte Chemie*, 120(40):7867–7869, 2008.