Hindawi Publishing Corporation
Journal of Nanomaterials
Volume 2015, Article ID 120365, 11 pages
http://dx.doi.org/10.1155/2015/120365

Research Article

Molecular Logic Computation with Debugging Method

Xiangrong Liu,[1] Juan Suo,[1] Juan Liu,[2] Yan Gao,[1] and Xiangxiang Zeng[1]

[1]*Department of Computer Science, Xiamen University, Xiamen 361005, China*
[2]*Department of Electric and Computer Engineering, Institute of Physical and Mechanical and Electrical Engineering, Xiamen University, Xiamen 361005, China*

Correspondence should be addressed to Xiangxiang Zeng; xzeng@xmu.edu.cn

Received 10 December 2014; Accepted 19 January 2015

Academic Editor: Zhida Xu

Seesaw gate concept, which is based on a reversible DNA strand branch process, has been found to have the potential to be used in the construction of various computing devices. In this study, we consider constructing full adder and serial binary adder, using the new concept of seesaw gate. Our simulation of the full adder preformed properly as designed; however unexpected exception is noted in the simulation of the serial binary adder. To identify and address the exception, we propose a new method for debugging the molecular circuit. The main idea for this method is to add fan-outs to monitor the circuit in a reverse stepwise manner. These fan-outs are fluorescent signals that can obtain the real-time concentration of the target molecule. By analyzing the monitoring result, the exception can be identified and located. In this paper, examples of XOR and serial binary adder circuits are described to prove the practicability and validity of the molecular circuit debugging method.

1. Introduction

In recent years, the gap between the finite capability of silicon and the infinite demand of consumers has widened gradually. This condition has given rise to molecular computing as an alternative approach to silicon-based computing. Molecular computing is aimed at building computational devices using various kinds of molecules, including organic molecules, proteins, nucleic acids, enzymes, and even supramolecular hydrogels [1–5]. Various computing devices have been constructed with biological molecules [6–12]. Given its parallel and microscopic nature, DNA has great data storage capacity and flexible design. It thus shows great potential as a material for large-scale information processing.

Several computing devices, such as logic gates, circuits, and tiny circuit boards, have been developed based on DNA molecules. In 2004, Stojanovic and Stefanovic proposed a series of molecular logic gate circuits based on nucleic acid catalysts [13]. In 2006, Seelig et al. reported the design and experimental implementation of DNA-based digital logic circuits using single-stranded nucleic acids [4]. In 2013, Zhang et al. established AND and OR gates based on DNA self-assembly and strand branch migration methods [12]. In 2014, a microfluidic half adder chip is achieved by controlling the annealing and denaturation of double-stranded DNA [14].

With complex and uncertain parameters (e.g., concentration and temperature) to balance, molecular circuits always function abnormally, just not as we expected. In 2009, Qian and Winfree proposed a simple DNA gate motif, a seesaw gate, that used a reversible strand displacement reaction based on the toehold exchange principle [15]. Zou et al. proposed a dynamic selection strategy which can be DNA coding [16]. Unlike circuits with other techniques that can only involve at most tens of gates, the seesaw catalytic gate architecture appeared suitable for practical synthesis of large-scale circuits involving possibly thousands of gates. Furthermore, it was argued that synthesis and preparation of circuit components can be parallel and scalable. To illustrate the potential of this approach, some logic circuits even a four-bit square-root circuit that comprised 130 DNA strands were shown in [15, 17], which all enabled fast and reliable function.

In this study, we aim to employ seesaw gates to construct an important computational component, adder. As we know, adders are the basic components of silicon-based computers and an indispensable component of all computer systems.

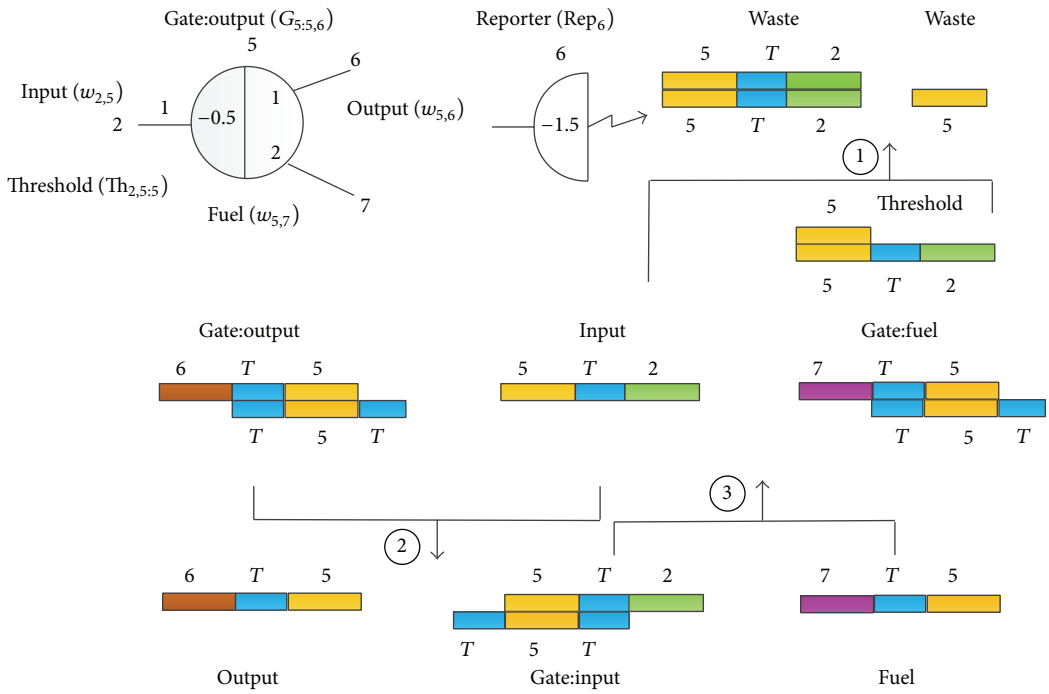


Figure 1: A simple DNA gate motif of seesaw gate.

Based on the seesaw gate concept proposed by Qian et al., we construct full adder and serial binary adder. As indicated by the term seesaw, the computing process is not static but a complex process of dynamic interaction, which can lead to a cyclic response based on DNA strand migration. With the recycling mechanism, the electrical adder can be transformed into some cyclic chemical reactions that can ensure a beneficial cycle in a long time. To test and verify the function of the molecular adder models, some simulations based on kinetic equations are performed. The simulation results show that the full adder yields expected results. However, the same is not achieved in the simulation of the serial binary adder, in which unexpected exception occurs. To identify and address this exception, we propose a method for debugging the molecular circuit by adding fluorescent signals to monitor the target molecule. As the fluorescent signals can detect the location of the exception, the abnormal circuit can be corrected using the error message.

The remainder of this paper is arranged as follows: the next section introduces the concept of seesaw gate; Section 3 describes the design and the simulation of full and serial binary adders based on seesaw gates; Section 4 proposes a debugging method for molecular circuits, and examples of XOR and serial binary adders are described to prove the practicability and validity of the debugging method; and conclusions and some open problems for future work are presented in Section 5.

2. Simple DNA Gate Motif, Seesaw Gate

In this section, we introduce a simple DNA gate motif, "seesaw" gate, which is an enzyme-free DNA machinery mainly based on DNA strand migration [17, 18]. The entire reaction is a cyclic response, which means the reaction can work permanently in a self-motivated and self-directed way in a given condition. Typically, a threshold must be exceeded before catalysis occurs. When connected into circuits involving many interacting catalytic gates, complex circuit behavior can be obtained.

The reaction mechanism is a simplified version of the entropy-driven catalytic gate introduced in [15, 19]. The fundamental operation is toehold exchange, which is a toehold-mediated strand displacement reaction that results in a free right-side signal strand replacing a bound left-side signal strand. Figure 1 shows a simple DNA gate motif, seesaw gate. In the top left corner, it is abstract diagram for a seesaw gate and a reporter. Numbers ahead of or at the top of nodes indicate identities of nodes (or interfaces to those nodes in a network) and numbers within the nodes or on the wires indicate relative concentration of different initial DNA species. Each species plays a specific role (e.g., input is a single molecular; gate:output is a compound molecular) and has a unique name that is useful while translating the reactions into some equivalent kinetic equations (e.g., $w_{2,5}$ and $G_{5:5,6}$). The other part is about basic reaction mechanisms involved in a seesaw network, which consists of three parts: thresholding, seesawing, and reporting. First the threshold gate absorbs the input strand. When the threshold is exceeded, the input strand begins to displace the output strand from the gate:output complex. An analogous process then allows the fuel strand to similarly displace the input strand from the new gate:input complex, completing a catalytic cycle that has the net effect of exchanging one left-side signal strand in solution (the fuel) for another left-side signal strand (the output).

Finally, the output production can be acquired through the concentration of the reporter molecular. In principle, enough output can be obtained in a given condition with enough fuel and gate:output complex.

The chemical reactions described in Figure 1 can be equivalently translated into some kinetic equations. The main equations are shown below from (1) to (3). In these equations, $w_{2,5}$ and others indicate reacting molecular species, k_s indicates the slow strand displacement rate of seesawing and reporting reactions, and k_f indicates the fast strand displacement rate of thresholding reactions. Other side-reactions are ignored, which are also discussed in detail in [17].

Seesawing reactions:

$$\begin{aligned} w_{2,5} + G_{5:5,6} &\underset{k_s}{\overset{k_s}{\Longleftrightarrow}} G_{2:5,5} + w_{5,6}, \\ w_{2,5} + G_{5:5,7} &\underset{k_s}{\overset{k_s}{\Longleftrightarrow}} G_{2:5,5} + w_{5,7}. \end{aligned} \tag{1}$$

Thresholding reactions:

$$w_{2,5} + \mathrm{Th}_{2,5:5} \xrightarrow{k_f} \text{waste}. \tag{2}$$

Reporting reactions:

$$w_{5,6} + \mathrm{Rep}_6 \xrightarrow{k_s} \mathrm{Flour}_6. \tag{3}$$

These chemical reactions modelling the toehold exchange steps and threshold absorption steps can be written uniformly. For all $i, j, k \in \{1, 2, \ldots, N\}$, where the variables refer to the molecular species,

$$\begin{aligned} w_{ji} + g_{i:ik} &\underset{k_0}{\overset{k_0}{\Longleftrightarrow}} g_{ji:i} + w_{ik}, \\ w_{ij} + \mathrm{th}_{ij:j} &\xrightarrow{k_1} \text{waste} \xleftarrow{k_1} \mathrm{th}_{i:ij} + w_{ij}. \end{aligned} \tag{4}$$

Using standard mass action chemical kinetics, it gives rise to a system of ordinary differential equations (ODEs) for the dynamics. In the following, w_{ij} and similar terms refer to the concentration of the respective species, rather than to the species themselves:

$$\begin{aligned} \frac{d_{w_{ij}}}{d_t} &= k_0 \left(\sum_{n=1}^{N} w_{ni} \cdot g_{i:ij} + w_{jn} \cdot g_{ij:j} - w_{ij} \cdot g_{ni:i} - w_{ij} \cdot g_{j:jn} \right), \\ \frac{d_{g_{i:ij}}}{d_t} &= k_0 \left(\sum_{n=1}^{N} w_{ij} \cdot g_{ni:i} - w_{ni} \cdot g_{i:ij} \right), \\ \frac{d_{g_{ij:j}}}{d_t} &= k_0 \left(\sum_{n=1}^{N} w_{ij} \cdot g_{j:jn} - w_{jn} \cdot g_{ij:j} \right), \\ \frac{d_{\mathrm{th}_{i:ij}}}{d_t} &= -k_1 \cdot w_{ij} \cdot \mathrm{th}_{i:ij}, \\ \frac{d_{\mathrm{th}_{ij:j}}}{d_t} &= -k_1 \cdot w_{ij} \cdot \mathrm{th}_{ij:j}. \end{aligned} \tag{5}$$

These dynamics have conserved quantities for each gate node i and for each signal wire ij:

$$\begin{aligned} \sum_{n=1}^{N} g_{ni:i} + g_{i:in} &\equiv c_i, \\ g_{i:ij} - \mathrm{th}_{i:ij} + w_{ij} + g_{ij:j} - \mathrm{th}_{ij:j} &\equiv c_{ij}, \\ \frac{d_{c_i}}{d_t} = \frac{d_{c_{ij}}}{d_t} &= 0. \end{aligned} \tag{6}$$

With these dynamic equations above, simulations have been performed masterly by Mathematic in a PC platform (Windows OS, I3 processor, 2 G RAM).

3. Molecular Adders Based on the Concept of Seesaw Gate

In electronics, an adder is a digital circuit that performs addition of numbers. In many computers and other types of processors, adders are used not only in arithmetic logic units, but also in other parts of the processor, where they are used to calculate addresses, table indices, and similar operations. In this section, we propose full adder and serial binary adder, with the concept described above, which is based on toehold-mediated DNA strand displacement.

3.1. Molecular Full Adder. A full adder adds binary numbers and accounts for values carried in and out. A single-bit full adder adds three single-bit numbers, often written as A, B, and C_0; A and B are the operands, and C_0 is a bit carried in from the next less significant stage. The full adder is usually a component in a cascade of adders which add 8-, 16-, and 32-bit binary numbers, and so on. The circuit produces a two-bit output, namely, output carry and sum, which are typically represented by the signals C_1 and S. The logic expression of the full adder is shown below:

$$\begin{aligned} S &= A \bigoplus B \bigoplus C_0, \\ C_1 &= A \cdot B + C_0 \cdot \left(A \bigoplus B \right). \end{aligned} \tag{7}$$

Add a file containing a digital circuit netlist, which can translate into an equivalent dual-rail circuit, in which each input is replaced by a pair of inputs, representing logic ON and OFF separately. The equivalent seesaw circuit of the full adder is shown in Figure 2, in which numbers ahead of or at the top of nodes indicate identities of nodes (or interfaces to

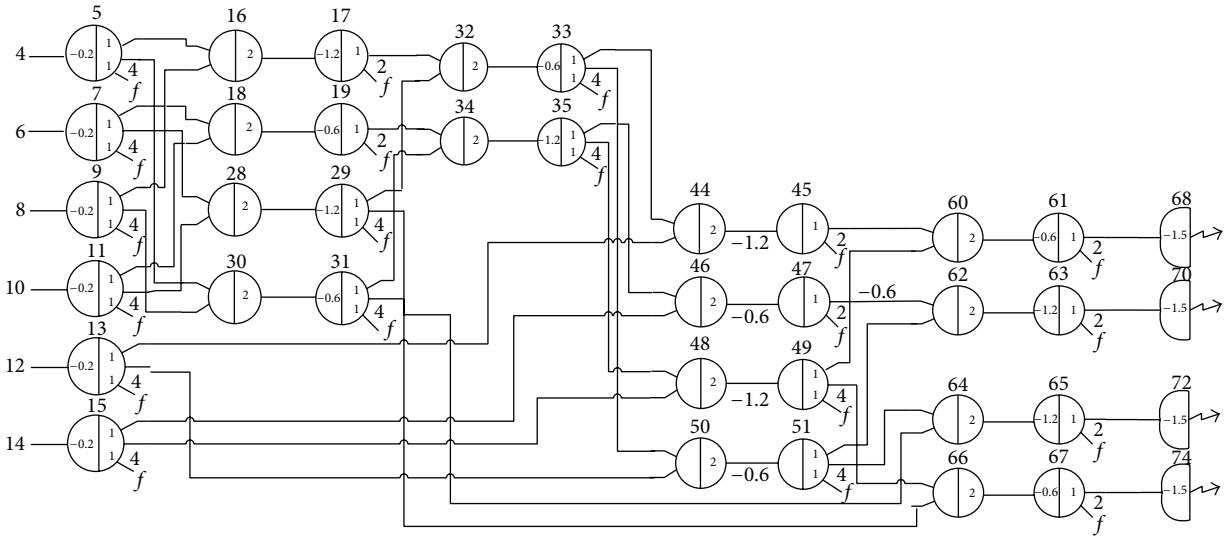


Figure 2: The seesaw circuit of the full adder.

those nodes in a network) and numbers within the nodes or on the wires indicate relative concentration of different initial DNA species. The seesaw circuit consists of 38 seesaw gates, any one of which is equivalent to the seesaw gate described in Figure 1. For example, the top left gate (gate:output $G_{5:5,16,30}$) is a general seesaw gate, whose input is $w_{4,5}$, threshold is $\mathrm{Th}_{4,5:5}$, outputs are $w_{5,16}$ and $w_{5,30}$, and fuel is $w_{5,f}$. The seesaw reactions of gate 5 (namely, $G_{5:5,16,30}$) are described after Figure 2.

Seesawing reactions of gate 5:

$$\begin{aligned} w_{4,5} + G_{5:5,16,30} &\underset{k_s}{\overset{k_s}{\Longleftrightarrow}} G_{4:5,5} + w_{5,16} + w_{5,30}, \\ w_{4,5} + G_{5:5,f} &\underset{k_s}{\overset{k_s}{\Longleftrightarrow}} G_{4:5,5} + w_{5,f}. \end{aligned} \tag{8}$$

Thresholding reactions of gate 5:

$$w_{4,5} + \mathrm{Th}_{4,5:5} \xrightarrow{k_f} \text{waste}. \tag{9}$$

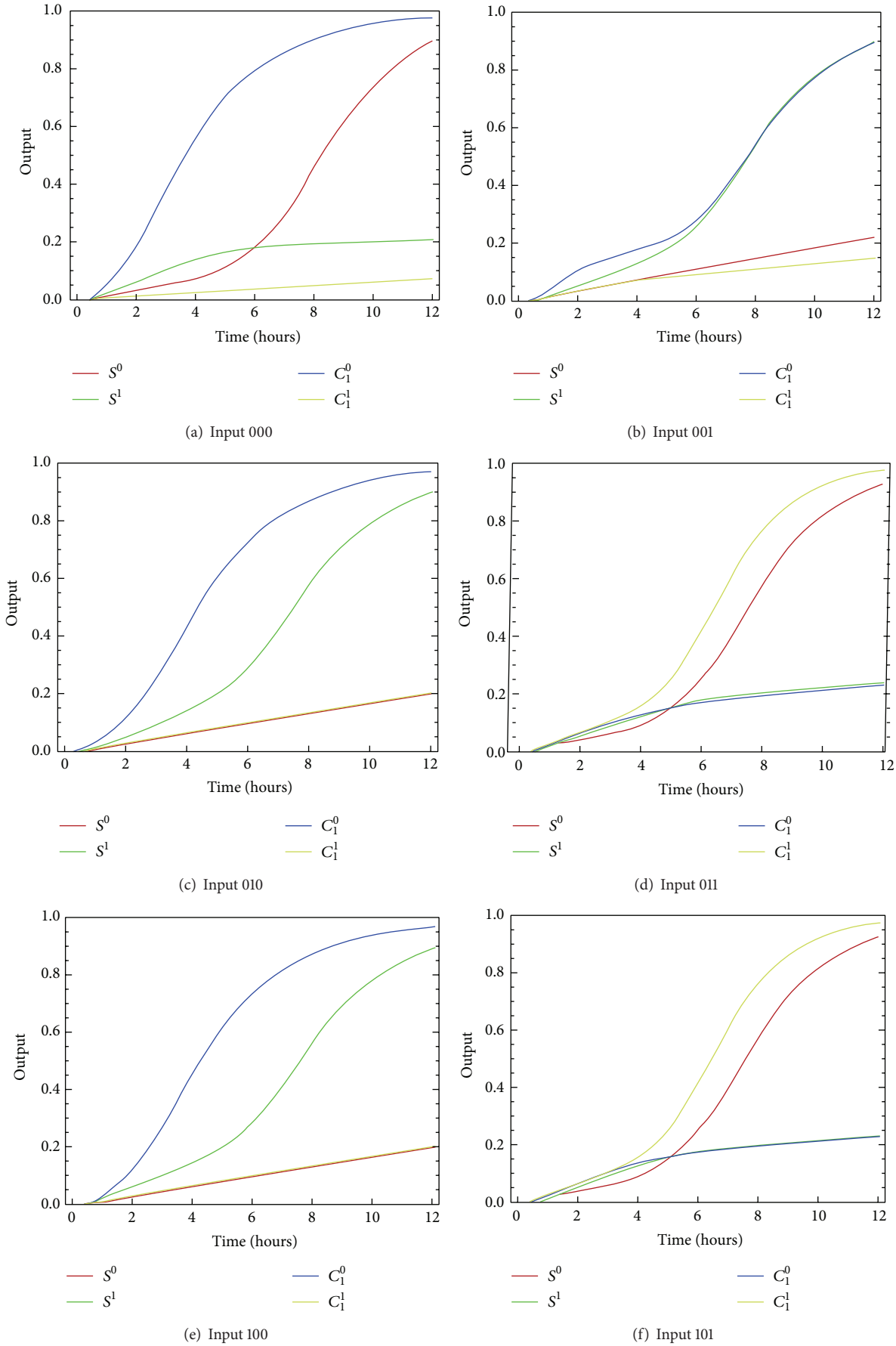
All of the other 37 gates are just like gate 5, are general seesaw gates, and can be similarly translated into corresponding seesaw reactions. Introduced in Section 2, all these seesaw reactions can be translated into equivalent dynamic equations, which can be simulated by mathematic in a PC platform.

Simulations are performed with the reference concentration 1x = 30 nM. The simulation result is shown in Figure 3, the horizontal axis represents the reaction time (from 0 h to 10 h), and the vertical axis represents the relative output concentration (from 0x to 10x). Moreover, the red curve describes the slope of the sum ($S = 0$) over time, the green curve describes the slope of the sum ($S = 1$) over time, the blue curve describes the slope of carry-over ($C = 0$) over time, and the yellow curve describes the slope of carry-over ($C = 1$) over time. When the reaction is almost completed, only one of the red and green curves (similarly blue and yellow) must stay at the high concentration (≥0.8x), and the other must stay at the low concentration (≤0.2x). This guarantees that only one result of $S = 1$ and $S = 0$ (similarly $C = 1$ and $C = 0$) can stay true, which is also reasonable in the real world. Any other condition is regarded as an exception. Finally, the high concentration curves represent the final true result. For example, if input signals A, B, and C_0 all stay OFF (i.e., between 0x and 0.1x), as shown in Figure 3(a), the final high concentration curves are $S = 0$ and $C_1 = 0$, which is the true result, which means that the computing result is (sum = 0, carry = 0). The whole value table of full adder is shown in Table 1. It can be found that all the conditions go cohere with the theoretical value.

3.2. Molecular Serial Binary Adder. The serial binary adder is a digital circuit that performs binary addition bit by bit. It has two two-bit inputs for the numbers to be added (A_2A_1, B_2B_1) and a single-bit input for the carry-in (C_0). The outputs are a two-bit output for the sum (S_2S_1) and single-bit output for carry-out (C_2). First A_1, B_1, and C_0 are added together to produce the low-bit sum (S_1) and low-bit carry-out (C_1). The high-bit carry-in signal is the previously calculated low-bit carry-out signal, so A_2, B_2, and C_1 are added together to produce the high-bit sum (S_2) and high-bit carry-out (C_2). Addition is performed by adding each bit from lowest to highest, one per clock cycle. The logic expression of the serial binary adder is shown below:

$$\begin{aligned} S_1 &= A_1 \bigoplus B_1 \bigoplus C_0, \\ C_1 &= A_1 * B_1 + (A_1 + B_1) * C_0, \\ S_2 &= A_2 \bigoplus B_2 \bigoplus C_1, \\ C_2 &= A_2 * B_2 + (A_2 + B_2) * C_1. \end{aligned} \tag{10}$$

In this paper, we use two full adders to achieve the serial binary adder, in which the carry-over of the first full adder is regarded as the input of the second full adder. Unfortunately, an exception occurs unexpectedly during simulation. As shown in Figure 4, when the input is (0, 0, 0, 0, 0),



(a) Input 000

(b) Input 001

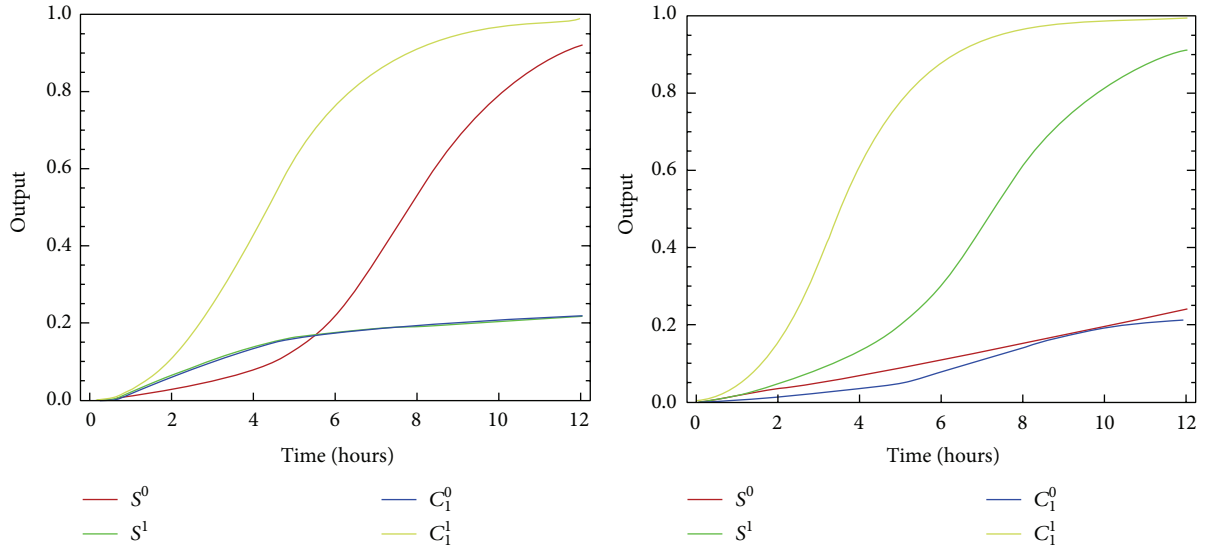
(c) Input 010

(d) Input 011

(e) Input 100

(f) Input 101

Figure 3: Continued.



(g) Input 110 (h) Input 111

Figure 3: The simulation result of the full adder.

Table 1: The value table of the half adder seesaw circuit.

Inputs	$S = 0$	$S = 1$	$C_1 = 0$	$C_1 = 1$	Computing results
(A = OFF, B = OFF, C_0 = OFF)	1	0	0	1	$S = 0, C_1 = 1$
(A = OFF, B = OFF, C_0 = ON)	0	1	1	0	$S = 1, C_1 = 0$
(A = OFF, B = ON, C_0 = OFF)	0	1	1	0	$S = 1, C_1 = 0$
(A = OFF, B = ON, C_0 = ON)	1	0	0	1	$S = 0, C_1 = 1$
(A = ON, B = OFF, C_0 = OFF)	0	1	1	0	$S = 1, C_1 = 0$
(A = ON, B = OFF, C_0 = ON)	1	0	0	1	$S = 0, C_1 = 1$
(A = ON, B = ON, C_0 = OFF)	1	0	0	1	$S = 0, C_1 = 1$
(A = ON, B = ON, C_0 = ON)	0	1	0	1	$S = 1, C_1 = 1$

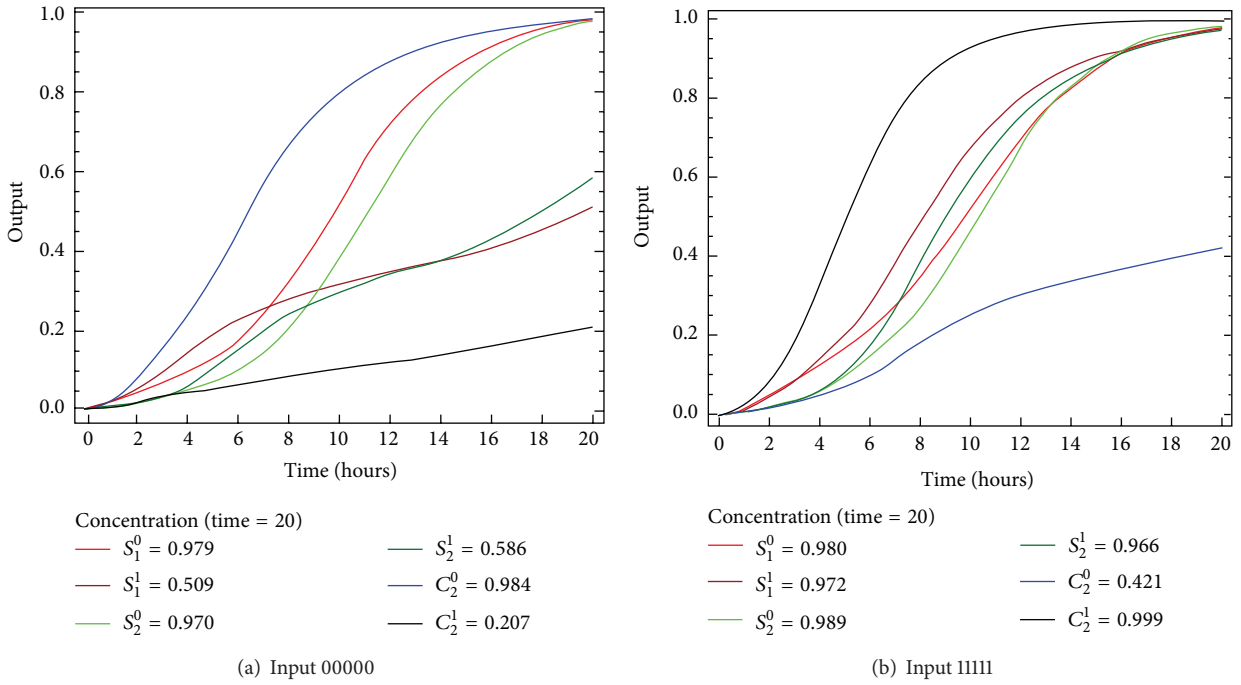
the concentrations of $S_1 = 1$ and $S_1 = 0$ (similarly $S_2 = 1$ and $S_2 = 0$) are both over 0.5, indicating that both $S_1 = 1$ and $S_1 = 0$ (similarly $S_2 = 1$ and $S_2 = 0$) are true results, which are contrary to the reality. However, at this point, the final values of $S_1 = 1$ and $S_1 = 0$ (similarly $S_2 = 1$ and $S_2 = 0$) still can be distinguished, which have not been the worst offenders. However, when the input is (1, 1, 1, 1), the concentrations of $S_1 = 1$ and $S_1 = 0$ (similarly $S_2 = 1$ and $S_2 = 0$) ascend nearly at the same height (which are both over 0.9), such that we are unable to distinguish the "low concentration" and "high concentration," which is a very serious exception. In the next part, we will analyze the reason for this situation and propose a method to solve it.

4. A Debugging Method for Molecular Circuit

As of this writing, many well-worked molecular computing devices, including logic gates, circuits, and tiny circuit boards, exist. However, an effective debugging method for molecular circuit is still needed. In the molecular circuit, exceptions always occur unexpectedly because of the tiny differences in temperature, concentration, reaction duration, and other factors. Thus, proposing a method to debug the molecular circuits is of great importance to the development of molecular computing. In this paper, we propose a method to address this issue.

The main idea is to add fan-outs to the seesaw circuit. Fan-outs are molecular fluorescent signals that can monitor the target molecule (any possible abnormal molecule). The real-time concentration of the target molecule can be obtained, which can help us analyze the circuit exception. If the site of the exception is unknown, debugging can be implemented from the output end step by step, until the exception molecular is discovered. The results in Figure 6 indicate that the new fluorescent signal nearly does not influence the initial molecular circuit, which means that our debugging does not change the original circuit. In the following section, some examples are shown to explain the process and manipulation of this method, as well as to prove the practicability and validity of the method.

4.1. Debugging of the XOR Circuit. Exclusive or XOR is a logical operation that outputs truly whenever both inputs differ (one is true, and the other is false). The opposite of XOR is logical biconditional, which outputs truly whenever both



(a) Input 00000 (b) Input 11111

Figure 4: The simulation result of the serial binary adder.

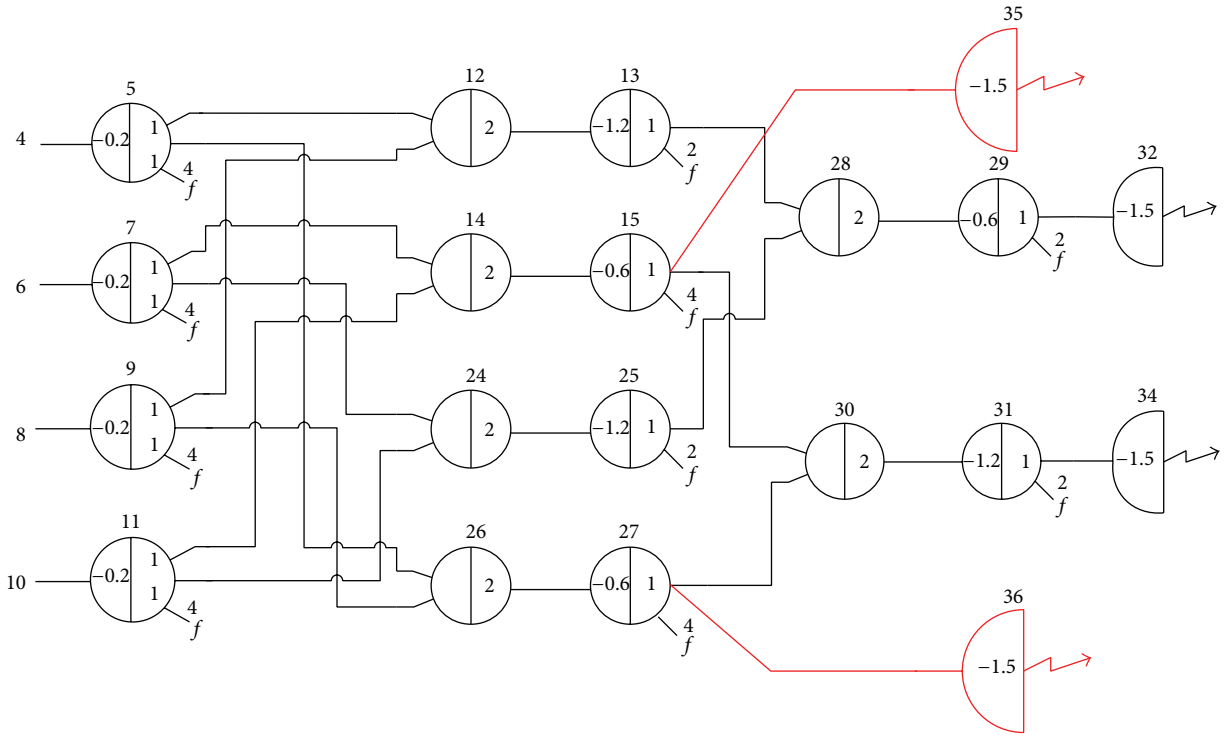
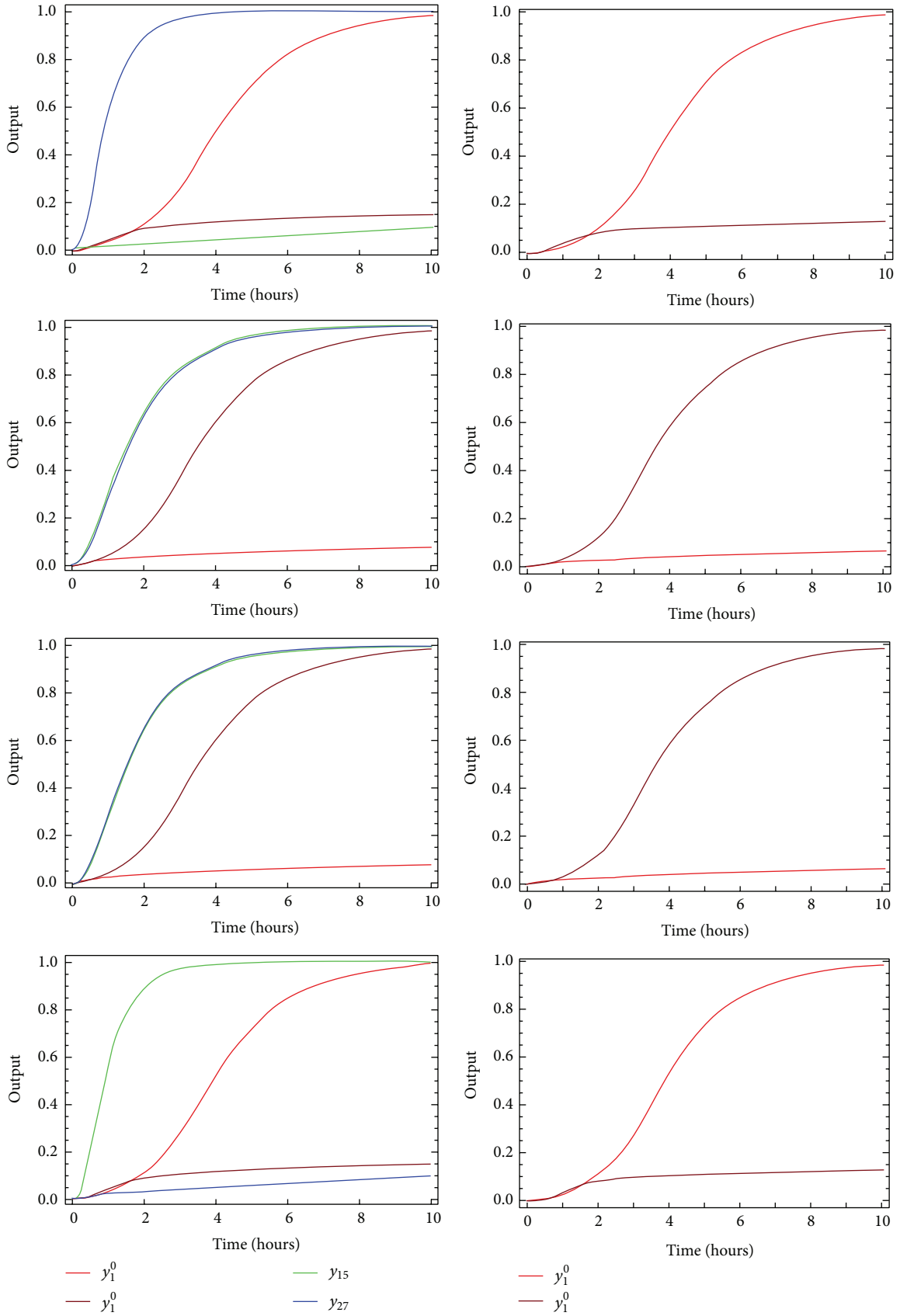


Figure 5: The seesaw circuit of the XOR circuit with monitoring molecular.

inputs are the same. It gains the name "exclusive or" because the meaning of "or" is ambiguous when both operands are true; "exclusive or" excludes that case. XOR is sometimes thought of as one or the other but not both. The XOR circuit is a simple circuit that is used to obtain the monitoring results, so we first choose the XOR circuit for analysis. Two inputs (A and B) and an output (S) exist in the XOR circuit. The logic expression of the XOR circuit is shown below:

$$S = A \bigoplus B = A \cdot B' + A' \cdot B. \tag{11}$$

To debug the XOR circuit, as shown in Figure 5, we add two fluorescent signals (35 and 36) for monitoring the target



(a) With monitoring molecular (35, 36)

(b) Without monitoring molecular

Figure 6: The simulation result of the XOR circuit.

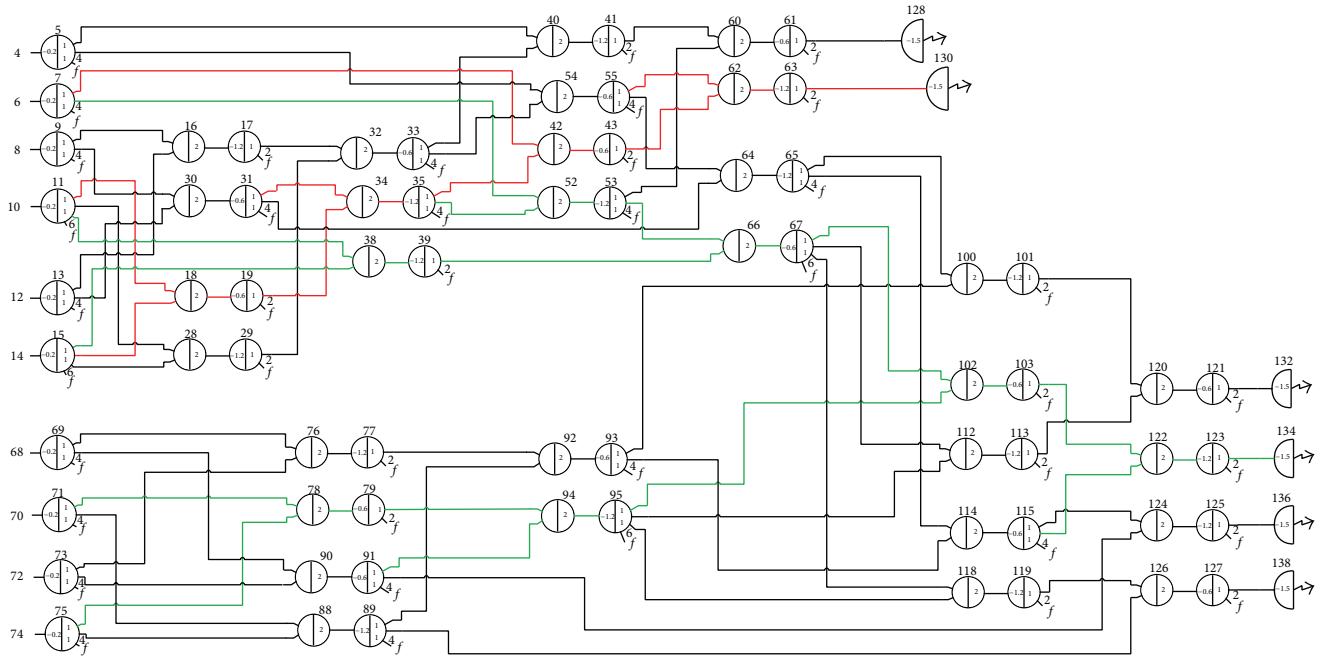


Figure 7: The error line of the serial binary adder circuit.

Table 2: The contrast between the simulation result of with monitoring and without monitoring.

Input	Monitor	$T = 2$	$T = 4$	$T = 6$	$T = 8$
(0, 0)	yes/no	0.033/0.029	0.046/0.041	0.055/0.050	0.064/0.059
(0, 1)	yes/no	0.106/0.102	0.497/0.506	0.825/0.833	0.943/0.947
(1, 0)	yes/no	0.033/0.029	0.046/0.041	0.055/0.050	0.064/0.059
(1, 1)	yes/no	0.112/0.107	0.525/0.534	0.853/0.853	0.952/0.956

molecule directly before the output signal, which is the direct input of the final output, which will directly influence the final result. Four fluorescent signals exist in the circuit. The original fluorescent signals (32, 34) monitor the real output. The added fluorescent signals (35, 36) monitor the temporary output, wherein an exception may occur. Other parts of the circuit remain unchanged.

The simulation result is shown in Figure 6. The changed circuit results with monitoring molecule are shown in Figure 6(a), whereas the original circuit results without monitoring molecule are shown in Figure 6(b). Table 2 shows the actual simulation results of the two cases, from which we can find the contrast between the simulation result of with monitoring molecule and without monitoring molecule and that the average of the difference between the two cases equals 0.004, which proves that the output almost remains unchanged and the debugging method does not influence the initial molecular circuit. Furthermore, the output of the monitoring molecule is consistent with the theoretical value and proves that circuit functions properly at the monitoring point.

4.2. Debugging of the Serial Binary Adder Circuit. In the aforementioned section, we pointed out that the serial binary adder does not function properly. An exception occurs when the input is (1, 1, 1, 1, 1) and concentrations of $S_1 = 1$ and $S_1 = 0$ (similarly $S_2 = 1$ and $S_2 = 0$) are both over 0.9. Thus, both $S_1 = 1$ and $S_1 = 0$ (similarly $S_2 = 1$ and $S_2 = 0$) are true results. Actually, the concentration of $S_1 = 1$ and $S_2 = 1$ should be low, with a final value that does not exceed 0.2. Therefore, an exception must occur somewhere. To determine where the exception occurs, we debug the circuit with the above method in a reverse stepwise manner.

We analyze the exception of $S_1 = 1$. The concentration of $S_1 = 1$ is obtained from fluorescent signal 130 (Figure 7), which represents the output of AND (62, 63). The direct input of AND (62, 63) is the first target molecule to monitor, which are molecules 43 and 55, so we add two fluorescent signals to monitor molecules 43 and 55. The monitoring result is shown in Figure 8(a), showing that molecule 55 functions well, but molecule 43 indicates an exception, which should be a low concentration in theory. We then monitor the direct input of molecule 43, which are molecules 7 and 35. The monitoring results are shown in Figure 8(b), in which both molecules indicate an emerging exception. Subsequently, we monitor the direct input of molecule 35, which are molecules 19 and 31, and the monitoring results are shown in Figure 8(c), which shows that molecule 19 indicates an exception. We then

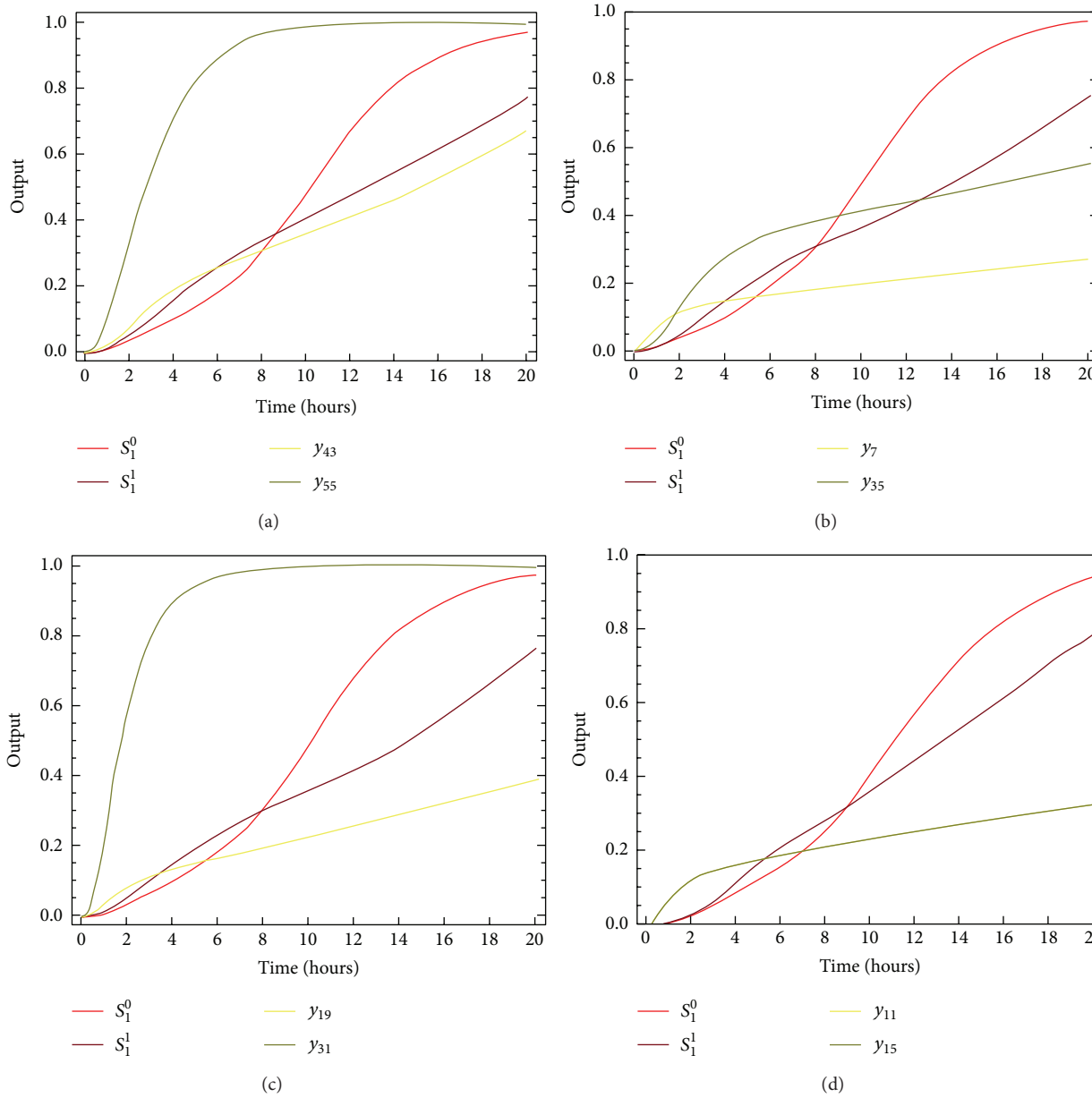


Figure 8: The simulation result of the serial binary adder circuit.

monitor the direct input of molecule 19, which are molecules 11 and 15. The monitoring results are shown in Figure 8(d); both molecules demonstrate an exception, which is also the point where exception occurs first. Thus, the exception point is found. The error transmission line described above is shown in Figure 7 (red line). The other error line of $S_2 = 1$ can be obtained in the same way, which is shown in Figure 7 (green line).

5. Conclusions and Future Work

In this paper, we construct several adders with a simple DNA gate motif, namely, the seesaw gate. The seesaw gate is a single DNA strand that can bind to signal strands via toehold domain. In the initial state, the input signals with a high concentration will react with the compound molecule in which the output signals are bound. The fundamental operation is toehold exchange, which is a toehold-mediated strand displacement reaction that results in a free right-side signal strand replacing a bound left-side signal strand. Typically, circuits with any arbitrarily complex can be contracted via cascading the simple gates. We build some adders with the above concept, including half adder, full adder, and serial binary adder. The validity and correctness of the half adder and full adder are demonstrated by simulation experiment.

During the simulation of the serial binary adder, an exception occurs, which leads to the incorrectness of the output. We find that the fan-out of the seesaw gate can

influence the output, which is bordered by the threshold. When the output is with high concentration, more fan-outs lead to lower concentration of the output; when the output is with low concentration, more fan-outs lead to higher concentration of the output. In the case of the serial binary adder, the fan-outs number has reached upper limit, and the output with low concentration is higher than that under normal conditions, thereby leading to the exception.

To solve this problem, we propose a method to debug the molecular circuit. The main idea is to add fan-outs to the seesaw circuit, which is a molecular fluorescent signal. With the fluorescent molecule, we can obtain the output of any gate. Thus, the debugging process can be implemented in a stepwise manner, similar to the occasion of the serial binary adder circuit above.

For future studies, two directions can be considered. First, we should design an adaptive debugging method, which is a self-correcting mechanism that can adjust the parameters (e.g., concentration and threshold) aimed at different circuits automatically. Second, to solve the abnormal occasion, we should study the signal reconstruction mechanism for recovery of the distorted signal, which can help deal with the output exception of the serial binary adder circuit. Moreover, with the signal reconstruction mechanism, more large-scale circuits are more likely to be implemented.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The work is supported by National Natural Science Foundation of China (Grants nos. 61472333, 61370010, 51405408, and 71103154), Natural Science Foundation of Fujian Province of China (nos. 2011J01334, 2014J01253), Ph.D. Programs Foundation of Ministry of Education of China (20120121120039), and China Scholarship Council (201308350065).

References

[1] S. Coe, W.-K. Woo, M. Bawendi, and V. Bulović, "Electroluminescence from single monolayers of nanocrystals in molecular organic devices," *Nature*, vol. 420, no. 6917, pp. 800–803, 2002.

[2] S. Dydel and P. Bała, "Large scale protein sequence alignment using FPGA reprogrammable logic devices," in *Field Programmable Logic and Application*, vol. 3203 of *Lecture Notes in Computer Science*, pp. 23–32, Springer, Berlin, Germany, 2004.

[3] H. Komatsu, S. Matsumoto, S.-I. Tamaru, K. Kaneko, M. Ikeda, and I. Hamachi, "Supramolecular hydrogel exhibiting four basic logic gate functions to fine-tune substance release," *Journal of the American Chemical Society*, vol. 131, no. 15, pp. 5580–5585, 2009.

[4] G. Seelig, D. Soloveichik, D. Y. Zhang, and E. Winfree, "Enzyme-free nucleic acid logic circuits," *Science*, vol. 314, no. 5805, pp. 1585–1588, 2006.

[5] D. R. Stewart, D. A. A. Ohlberg, P. A. Beck et al., "Molecule-independent electrical switching in Pt/organic monolayer/Ti devices," *Nano Letters*, vol. 4, no. 1, pp. 133–136, 2004.

[6] Y. Benenson, R. Adart, T. Paz-Elizur, Z. Livneh, and E. Shapiro, "DNA molecule provides a computing machine with both data and fuel," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 100, no. 5, pp. 2191–2196, 2003.

[7] Y. Benenson, T. Paz-Elizur, R. Adar, E. Keinan, Z. Llvneh, and E. Shapiro, "Programmable and autonomous computing machine made of biomolecules," *Nature*, vol. 414, no. 6862, pp. 430–434, 2001.

[8] W. He, X. Ma, X. Yang, Y. Zhao, J. Qiu, and H. Hang, "A role for the arginine methylation of Rad9 in checkpoint control and cellular sensitivity to DNA damage," *Nucleic Acids Research*, vol. 39, no. 11, pp. 4719–4727, 2011.

[9] C. Mao, T. H. LaBean, J. H. Reif, and N. C. Seeman, "Logical computation using algorithmic self-assembly of DNA triple-crossover molecules," *Nature*, vol. 407, no. 6803, pp. 493–496, 2000.

[10] J. Yang, C. Dong, Y. Dong, S. Liu, L. Pan, and C. Zhang, "Logic nanoparticle beacon triggered by the binding-induced effect of multiple inputs," *ACS Applied Materials & Interfaces*, vol. 6, no. 16, pp. 14486–14492, 2014.

[11] C. Zhang, J. Ma, J. Yang, H. Inaki Schlaberg, S. Liu, and J. Xu, "Nanoparticle aggregation logic computing controlled by DNA branch migration," *Applied Physics Letters*, vol. 103, no. 9, Article ID 093106, 2013.

[12] C. Zhang, L. Ma, Y. Dong, J. Yang, and J. Xu, "Molecular logic computing model based on DNA self-assembly strand branch migration," *Chinese Science Bulletin*, vol. 58, no. 1, pp. 32–38, 2013.

[13] M. N. Stojanovic and D. Stefanovic, "A deoxyribozyme-based molecular automaton," *Nature Biotechnology*, vol. 21, no. 9, pp. 1069–1074, 2003.

[14] J. Wang and R. Huang, "Design and implementation of a microfluidic half adder chip based on double-stranded DNA," *IEEE Transactions on Nanobioscience*, vol. 13, no. 2, pp. 146–151, 2014.

[15] L. Qian and E. Winfree, "A simple DNA gate motif for synthesizing large-scale circuits," in *DNA Computing*, vol. 5347 of *Lecture Notes in Computer Science*, pp. 70–89, Springer, Berlin, Germany, 2009.

[16] Q. Zou, Y. Mao, L. Hu, Y. Wu, and Z. Ji, "miRClassify: an advanced web server for miRNA family classification and annotation," *Computers in Biology and Medicine*, vol. 45, no. 1, pp. 157–160, 2014.

[17] L. Qian and E. Winfree, "Scaling up digital circuit computation with DNA strand displacement cascades," *Science*, vol. 332, no. 6034, pp. 1196–1201, 2011.

[18] L. Qian, E. Winfree, and J. Bruck, "Neural network computation with DNA strand displacement cascades," *Nature*, vol. 475, no. 7356, pp. 368–372, 2011.

[19] D. Y. Zhang, A. J. Turberfield, B. Yurke, and E. Winfree, "Engineering entropy-driven reactions and networks catalyzed by DNA," *Science*, vol. 318, no. 5853, pp. 1121–1125, 2007.