Chapter 5
Combinational Logic Circuit Based on BZ Reaction

Mingzhu Sun and Xin Zhao

Abstract As a basic unit of large scale integration, combinational logic circuit is very important in the development of digital computer. Chemical computation should have the ability to replicate the basic function of combinational logic circuit in BZ medium, in order to realize chemical computer. In this chapter, we design and implement different types of combinational logic circuits from two perspectives. On one hand, based on the basic chemical processors and logic gates, the cascade method is applied to achieve the functions of multi-bit combinational logic by using low-bit logic circuits. On the other hand, a universal method is put forward to construct combinational logic circuits according to their sum-of-products expressions. Simulation results demonstrate the effectiveness of the two construction methods, as well as all the combinational logical circuits designed in this chapter. We believe that the realization of combinational logical circuits will be helpful to fulfil other logic and arithmetic functions, and ultimately can bring great potential applications for the implementation of chemical computer and other intelligent systems.

5.1 Introduction

Belousov–Zhabotinsky (BZ) reaction is an important chemical oscillating reaction discovered by Boris Belousov and further developed by Zhabotinsky [17]. It has been proposed that some types of photosensitive BZ reactions can implement various computational operations [6, 16, 27]. As an unconventional strategy of information processing, this kind of computation, which is so called chemical computation,

M. Sun · X. Zhao (✉)
Nankai University, No. 94 Weijin Road, Tianjin, China
e-mail: zhaoxin@nankai.edu.cn

M. Sun
e-mail: sunmz@nankai.edu.cn

A. Adamatzky (ed.), *Advances in Unconventional Computing*,
Emergence, Complexity and Computation 23,
DOI 10.1007/978-3-319-33921-4_5

relies on geometrically constrained excitable chemical medium, and uses the change of concentrations of the BZ reagents to transmit information and realize computation [4, 8].

The first successful application of chemical computation was image processing in 1989 [13]. In 1995, the properties of BZ excitable medium were exploited to find minimum-length paths in complex labyrinths [24]. Since then, many different prototypes of chemical computation have been designed and tested on a variety of tasks, for example, optimal path search in a labyrinth [20, 21], image processing [19], robot navigation [1], direction detection [18], Voronoi diagram solving [2], and so on, which indicates that chemical computation has the ability and high efficiency for some kinds of complex computations.

Beyond that, one benchmark of chemical computation is capable of replicating components used in conventional computation, such as logic circuit [10]. According to the construction of digital computer, logic circuit based on BZ reaction can be connected to create chemical computer, which is a type of non-classical computers and can perform universal computation. In the past 20 years, the chemical implementations of logic circuit have attracted the attention of many researchers. Since the Showalter Laboratory realized the first logic gates in BZ medium in 1994 [28, 29], researchers have built several logic circuits, such as chemical diode [3, 7, 12], Boolean logic gates [14, 15, 23, 25], adders [5, 11, 30], decoders [26], counters [9], memory cells [14], and so on. Many of them have already been verified in simulations or in chemical experiments.

As a basic unit of digital circuit system, combinational logic circuit is the precondition for the development of digital computer. Though adders and decoders belong to combinational logic circuit, no one has discussed completely how to construct different types of combinational logic circuits based on BZ reaction until now. In this chapter, we focus on the design and simulation of combinational logic circuits in BZ medium. Based on the basic chemical signal processors implemented before, we systematically analysis the design of combinational logic circuits, and use the conventional method and universal method to fulfill construction respectively. On one hand, logic gates are designed based on existing structures and applied to construct low-bit logic circuits according to their logic expressions; then multi-bit combinational logic circuits, including binary adder, binary comparator, binary encoder and binary decoder, are designed and implemented in simulation by the cascade method. On the other hand, a universal method is put forward to design logic circuits, since the logical function of any combinational logic circuit can be expressed by sum-of-products expressions. Universal structure of sum-of-products expressions and multi-input multi-output structure are formed and tested respectively for logical arithmetic and signal transmission, then universal structure of combinational logic circuits is designed and verified.

The rest of the chapter is organized as follows: we present the details of the two-variable Rovinsky–Zhabotinsky (RZ) model of BZ reaction for simulation in Sect. 5.2; Sect. 5.3 outlines some basic chemical signal processors, which have been

verified before and are used as components of combinational logic circuits. Sections 5.4 and 5.5 show the construction and simulation results of two types of combinational logic circuits. At last, the chapter is concluded in Sect. 5.6.

5.2 The RZ Model of BZ Reaction

We employ the RZ model of the BZ reaction to calculate the propagation of the pulses [9, 26, 27, 30]. The RZ model, which is derived from the Field–Koros–Noyes (FKN) reaction mechanism [17], has two variables, x and z, corresponding to dimensionless concentrations of activator $HBrO_2$ and of catalyst $Fe(phen)_3^{3+}$. In the active regions, which contain the catalyst, the time evolution of the concentrations of x and z is described by Eq. (5.1).

$$\begin{aligned} \frac{\partial x}{\partial \tau} &= \frac{1}{\varepsilon}\left[x(1-x) - (2q\,\alpha\,\frac{z}{1-z} + \beta)\frac{x-\mu}{x+\mu}\right] + \nabla^2 x, \\ \frac{\partial z}{\partial \tau} &= x - \alpha\,\frac{z}{1-z}. \end{aligned} \tag{5.1}$$

In the passive regions, where catalyst is absent, the concentrations of x and z evolve according to Eq. (5.2).

$$\begin{aligned} \frac{\partial x}{\partial \tau} &= -\frac{1}{\varepsilon}\left[x^2 + \beta\frac{x-\mu}{x+\mu}\right] + \nabla^2 x, \\ z &= 0. \end{aligned} \tag{5.2}$$

In numerical calculations, Eqs. (5.1) and (5.2) are solved numerically using Euler method with a five-node Laplace operator for the diffusion term. The time step $\Delta\tau$ is 0.0001, and the distance between each grid point $\Delta\rho$ is 0.3301 [26, 30]. The other parameters are the same values as considered in Refs. [9, 26, 30]: $\varepsilon = 0.1176$, $q = 0.5$, $\alpha = 0.068$, $\beta = 0.0034$, and $\mu = 0.00051$. For these values of parameters, the stationary concentrations of x and z in the active region, which are the stationary solution of Eq. (5.1), are: $x = 7.27 \times 10^{-4}$, $z = 1.06 \times 10^{-2}$; and the stationary concentrations in the passive region, which are the stationary solution of Eq. (5.2), are: $x = 5.12 \times 10^{-4}$, $z = 0$.

Based on the geometrical configuration of the channels, we can implement many kinds of computational devices by the model. In simulations, the pulses in the active regions are initiated by increasing the value of x to 0.1 at the end of signal channels, and then the excitation waves will propagate inside the channels. In simulation results, the black color and the white color show the distribution of the active and passive regions respectively. A high concentration of activator x is marked as a gray wave in the active regions.

5.3 Basic Chemical Signal Processors Based on BZ Reaction

The combinational logic circuits designed here are constructed by using some kinds of basic chemical signal processors, such as, T-shaped coincidence detector, chemical diode, time delay unit and crossover structure, which are constituted with simple straight line boundaries. Since the basic structures of these processors have already been verified in numerical simulations and chemical experiments [3, 8, 9, 14], the combinational logic circuits will work well in BZ medium, if they can be implemented and verified in simulations. In order to keep the structures of combinational logic circuits uniform and compact, we set the width of the channels to 20 grid points in all simulations.

5.3.1 T-Shaped Coincidence Detector

A T-shaped coincidence detector is used to detect whether two pulses meet in the proper area of the channel. As shown in Fig. 5.1, the device has two parts: a horizontal bar above as the signal channel, and a T-shaped structure below to detect coincidence [9]. The distance between the above bar and the horizontal bar on the T (detector bar) is very important. If two pulses propagate simultaneously from two sides of the above bar, they will meet in the middle and annihilate. With proper distance between the two bars, a pulse appears in the detector bar, and an output pulse is sent through the vertical part of the T. However, when a single pulse propagates inside the bar, it dies at the other end and does not excite the detector bar.

In Fig. 5.1, the distance between the above bar and the horizontal bar on "T" is set to 10 grid points for a 20-grid point wide channel. This structure can detect the meeting areas of the pulses and will be employed to construct logic gates and combinational logic circuits in Sect. 5.4.

5.3.2 Chemical Diode

Chemical diode is used for unidirectional signal transmission. The structure shown in Fig. 5.2 can achieve this purpose. The structure is formed by a straight channel on

Fig. 5.1 A classical T-shaped coincidence detector [9]. Grid size is 100×90, the channels are 20 grid points wide and the gap is 10 grid points wide

Fig. 5.2 A classical chemical signal diode [3]. Grid size is 120 × 60, the channels are 20 grid points wide and the gap is 8 grid points wide

the left and a triangular-tipped channel on the right [3]. The asymmetric geometry in BZ medium results in an asymmetric pulse propagation, such as unidirectional propagation. By selecting the proper width of the passive gap between two channels, pulse propagates through the gap from left to right, but not vice versa.

In Fig. 5.2, the distance between the straight channel and the triangular-tipped channel is set to 8 grid points for a 20-grid point wide channel. This structure is used to implement binary encoder in Sect. 5.4.

5.3.3 Time Delay Unit

In numerical simulations, the propagation speed of the pulse is a constant, if the parameters of simulation model are fixed. The time delay unit can reduce the propagation speed due to the penetration property of the BZ reaction, so that the synchronized input pulses can be generated. In BZ reaction, penetration means a pulse propagating in the active region can penetrate into a passive part and disappears after some distance [8]. The pulse can excite the active region behind the passive stripe, if the stripe is narrow enough [22]. The pulse propagates more quickly in active region than passive region, so penetration can be used for pulse delay.

The time delay unit is shown in Fig. 5.3. With a 20-grid point wide channel and an 8-grid point wide passive stripe, the time delay unit leads to 13-grid point delay under the simulation condition. This property will be applied to implement binary adder and binary decoder in Sect. 5.4.

Fig. 5.3 Pulse delay by penetration property of the BZ reaction. Grid size is 120 × 80, the channels are 20 grid points wide and the gap is 8 grid points wide. It leads to 13-grid point delay after penetration. **a** Pulses propagation before penetration. **b** Pulses propagation after penetration

5.3.4 Crossover Structure

Crossover structure is used for cross propagation of the pulses. This structure utilizes another property of the BZ reaction: angle dependent penetration of passive region [8]. When a pulse passes through the passive region, it is said that the maximum width of the passive stripe depends on the angle between the wave vector of the pulse and the normal to the stripe. A pulse with the wave vector perpendicular to the stripe can pass a wider stripe than a pulse propagates along the strip. Thus, in a junction of two channels, the interactions can be excluded by adding passive stripes with proper width.

When channel is 20 grid points wide, we cannot find a proper width of the passive stripe under the simulation condition. While for 40-grid point wide channel, the width of the stripe can be set to 10 grid points. Figure 5.4a illustrates the crossover structure, in which, a junction of two 40-grid point wide channels are separated by four 10-grid point wide passive stripes, and connected to 20-grid point wide channels. This device permits the pulse to propagate through channel AA' or BB' without outputs in channel BB' or AA'. For unidirectional propagation, we reduce the length of 40-grid point wide channels to 10 grid points, as shown in Fig. 5.4b. The pulse from O_1 or O_2 cannot pass the passive stripe as it cannot penetrate so long in an almost 20-grid point wide channel.

This crossover structure, which realizes single input single output cross propagation, will be employed to construct binary adder and binary encoder in Sect. 5.4. When both input channels of the crossover structure are excited simultaneously, there's no output pulse in any output channels, since the pulses meet in the center and annihilate. This property is used to achieve NOT operation in Sect. 5.5.

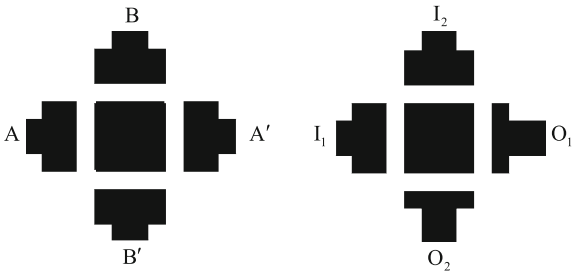

Fig. 5.4 Crossover structures for single input and single output. Grid size is 120 × 120, the cross channels and the propagation channels are 40 and 20 grid points wide respectively; the width of the passive stripes is 10 grid points. **a** A bidirectional crossover structure. **b** A unidirectional crossover structure. The length of the 40-grid point wide cross channels is 10 grid points. The pulse can only propagate from I_1 to O_1 or from I_2 to O_2

5.4 Classic Combinational Logic Circuits

In digital electronics, combinational logic circuits are constructed by combining logic gates directly, or by extending low-bit logic circuits by cascade method. Similarly, in this section, we design combinational logic circuits in BZ medium in these two ways. For one type of combinational logic circuit, we first build 1-bit or 2-bit logic circuit based on logic gates; then multi-bit combinational logic circuit is constructed by cascade method. Generally, An n-bit ($n \geqslant 2$) combinational logic circuit can be built by using one or two $(n-1)$-bit logic circuits.

In this section, we build logic gates based on basic chemical signal processors firstly, then discuss the construction of binary adder, binary comparator, binary encoder, binary decoder, which are four types of classic combinational logic circuits.

5.4.1 Logic Gates

The logic gates used here are described as follows:

1. AND gate: In AND gate, a HIGH output (1) results only if both inputs are 1. If neither or only one input is 1, a LOW output (0) results. A T-shaped coincidence detector is regarded as chemical realization of AND gate, as shown in Fig. 5.5a, c, d. With the presence of two input pulses, the detector becomes excited and the output is 1; in other cases, the detector is not excited and the output is 0.
2. OR gate: In OR gate, if one or both inputs are 1, the output is 1. If neither input is 1, the output is 0. In BZ reaction, OR gate is realized by linking channels directly

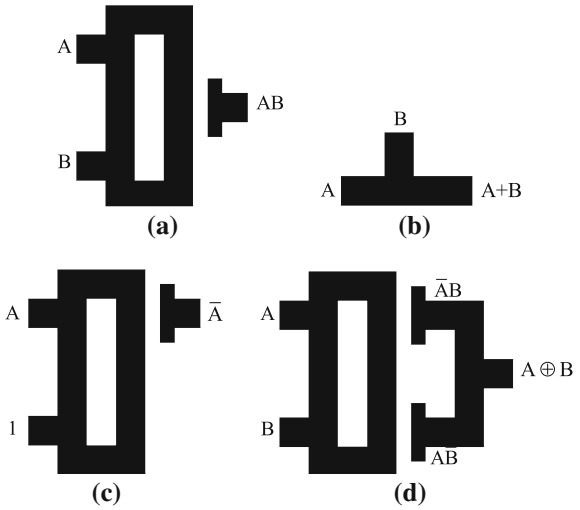

Fig. 5.5 The structures of logic gates based on the BZ reaction. **a** AND gate. **b** OR gate. **c** NOT gate. **d** XOR gate

(Fig. 5.5b). The output is 0 only if no pulse appears at channel A or B; otherwise, an output pulse is sent through the channel, which gives 1 as the output.
3. NOT gate: NOT gate outputs the opposite logic level to its input. As shown in Fig. 5.5c, NOT gate is replaced by AND-NOT gate with a constant auxiliary 1 input. With the absence of pulse at channel A, the auxiliary 1 input propagates through the channel, and the output is 1; while if there is a pulse at A, the two pulses will meet and annihilate, as a result, the output is 0.
4. XOR gate: In XOR gate, if one and only one input is 1, the output is 1. If both inputs are both 0 or 1, the output is 0. Figure 5.5d shows the structure of XOR gate, which combines AND, OR and NOT gates.

Note that the pulse from one input will propagate into the other in the structures in Fig. 5.5, but it has no influence on the outputs. In the rest of this chapter, we do not add additional chemical diodes in the input channels for simplicity, if the outputs do not interfere with the inputs.

5.4.2 Binary Adder

5.4.2.1 One-Bit Binary Adder

Binary adder performs addition of binary numbers. A one-bit half-adder adds two single binary digits A_i and B_i, with two outputs sum S_i and carry C_i. The outputs are expressed as:

$$\begin{aligned} S_i &= A_i \oplus B_i = \bar{A}_i B_i + A_i \bar{B}_i, \\ C_i &= A_i B_i. \end{aligned} \tag{5.3}$$

From Eq. 5.3, one-bit half-adder is constructed by combining an AND gate and an XOR gate directly. As shown in Fig. 5.6, we put carry channel C_i on the top of the structure, so the output channel of AND gate $A_i B_i$ should cross the channel $\bar{A}_i B_i$.

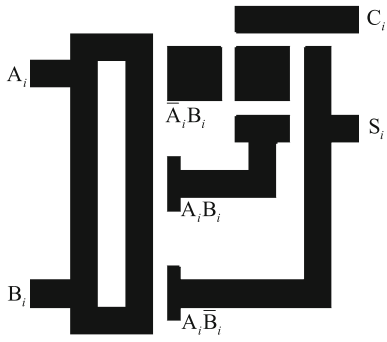

Fig. 5.6 The structure of one-bit half-adder. Grid size is 240 × 260. The devise consists of an AND gate, an XOR gate and a crossover structure, which is used to exchange the order of $A_i B_i$ and $\bar{A}_i B_i$

Since A_iB_i and $\bar{A}_iB_i$ cannot be 1 at the same time, a modified crossover structure is applied to achieve channel crossing.

For one-bit half-adder, we should consider four cases of the two inputs:

1. $A_i = 0, B_i = 0$: With the absence of the two pulses, the BZ medium stays at the stable stationary state. There is no pulse in the output channels, so the outputs C_i and S_i are 0 and 0.
2. $A_i = 1, B_i = 0$: Pulse from channel A_i propagates through the channel, generating a new pulse in channel $A_i\bar{B}_i$. Finally, the pulse is sent through channel S_i, so the outputs are 0 and 1.
3. $A_i = 0, B_i = 1$: Similarly, pulse from channel B_i generates a new pulses in channel $\bar{A}_iB_i$. The pulse passes through the crossover structure, giving 0 and 1 as the outputs.
4. $A_i = 1, B_i = 1$: Two pulses from channel A_i and B_i meet in the middle, and generates a pulse in channel A_iB_i. Through the crossover structure, this pulse is sent through channel C_i. As a result, the final outputs are 1 and 0.

Figure 5.7 shows the simulation results of one-bit half-adder in BZ reaction. Table 5.1 lists all possible inputs and the corresponding outputs of half-adder. The results indicates that this device realizes the addition of two one-bit binary numbers.

A full-adder adds binary numbers and accounts for values carried in as well as carried out. A one-bit full-adder adds three one-bit numbers, A_i, B_i and C_{i-1}, A_i and B_i are the operands, and C_{i-1} is the carry from the previous significant bit position. We can build a one-bit full-adder by using two half-adders: the first half-adder adds

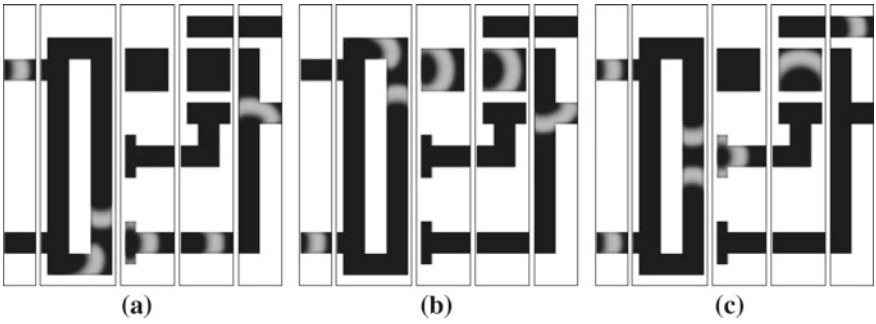

Fig. 5.7 Simulation results of one-bit half-adder when inputs (A_i and B_i) are **a** 1, 0. **b** 0, 1. **c** 1, 1

Table 5.1 Inputs and corresponding outputs of the structure of half-adder

Input		Output	
A_i	B_i	S_i	C_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

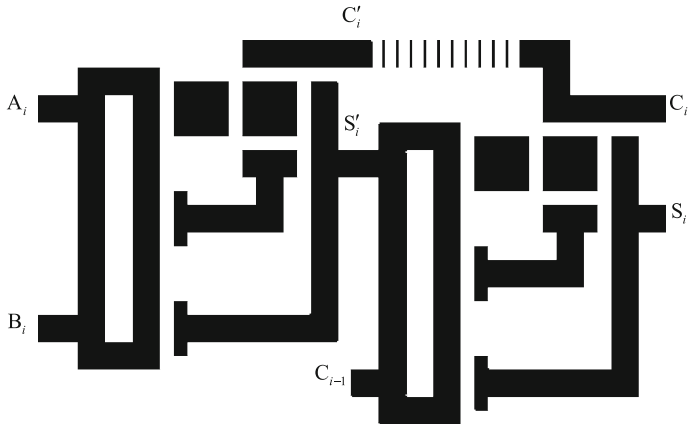

Fig. 5.8 The structure of one-bit full-adder. Grid size is 460 × 300. One-bit full-adder uses two one-bit half-adders to add three one-bit numbers

A_i and B_i, with outputs S_i' and C_i'; the second half-adder adds S_i' and C_{i-1} to generate final sum output S_i, and the final carry output C_i is produced by ORing carries of two half-adders. The outputs can be expressed as Eq. (5.4). The structure of one-bit full-adder is shown in Fig. 5.8, some time delay units are involved to keep the outputs synchronized with different inputs.

$$\begin{aligned} S_i &= S_i' \oplus C_{i-1} = A_i \oplus B_i \oplus C_{i-1}, \\ C_i &= C_i' + S_i' C_{i-1} = A_i B_i + (A_i \oplus B_i) C_{i-1}. \end{aligned} \tag{5.4}$$

Figure 5.9 shows the simulation results when previous carry input C_{i-1} is 1. In Fig. 5.9a, A_i and B_i are both 1, the outputs C_i' and S_i' of first half-adder are 1 and 0. Consider the previous carry input C_{i-1}, the final outputs C_i and S_i are 1 and 1. In Fig. 5.9b, A_i is 1 and B_i is 0, the first half-adder outputs 0 and 1 for C_i' and S_i'. In the second half-adder, two 1s add together, the final outputs C_i and S_i are 1 and 0.

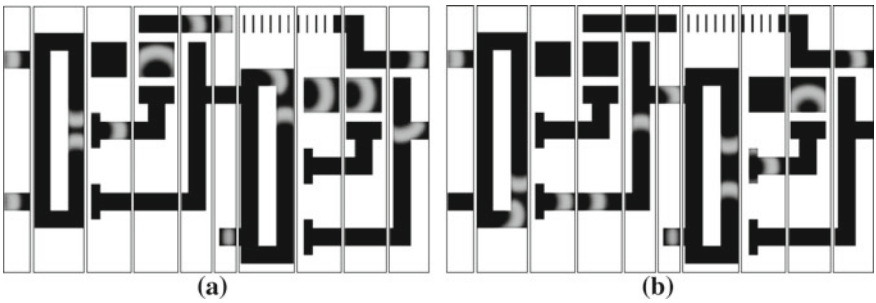

Fig. 5.9 Simulation results of one-bit full-adder when inputs (A_i, B_i, C_{i-1}) are **a** 1, 1, 1. **b** 1, 0, 1

The simulation results demonstrate that this structure fulfils the function of one-bit full-adder.

5.4.2.2 Two-Bit Binary Adder

Two-bit binary adder adds two binary numbers A_1A_0 and B_1B_0, giving S_1, S_0 as sum outputs and C_1 as carry output. We can extend the structure of one-bit binary adder to achieve two-bit binary adder, which combines a one-bit half-adder and a one-bit full-adder. The half-adder achieves the addition of two lower bits A_0 and B_0, the sum outputs as the lower sum S_0. The full-adder achieves the addition of two higher bits A_1 and B_1, as well as the carry output from the half-adder, the outputs of the full-adder are set as higher sum S_1 and carry C_1 of the two-bit adder. Figure 5.10 shows the structure of two-bit adder, in which, time delay units are used in the structure for pulse delay.

There are sixteen kinds of inputs for two-bit binary adder. Figure 5.11 shows two typical simulation results. In Fig. 5.11a, the inputs A_1A_0 and B_1B_0 are both 11, so the carry and sum of the half-adder are 1 and 0; three 1s add together in the full-adder, suggesting that the outputs $C_1S_1S_0$ of two-bit adder are 110. In Fig. 5.11b, the inputs A_1A_0 and B_1B_0 are 11 and 10, so the carry and sum of the half-adder are 0 and 1; the full-adder adds two 1s, the final outputs are 101. We have performed simulations for all 16 input combinations. The simulation results indicate that this structure realizes the addition function of two-bit binary numbers.

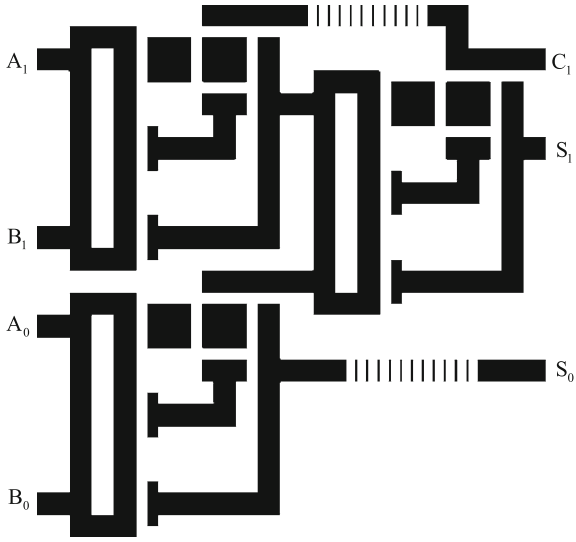

Fig. 5.10 The structure of two-bit binary adder. Grid size is 460 × 500. The structure consists of a one-bit full-adder and a one-bit half-adder

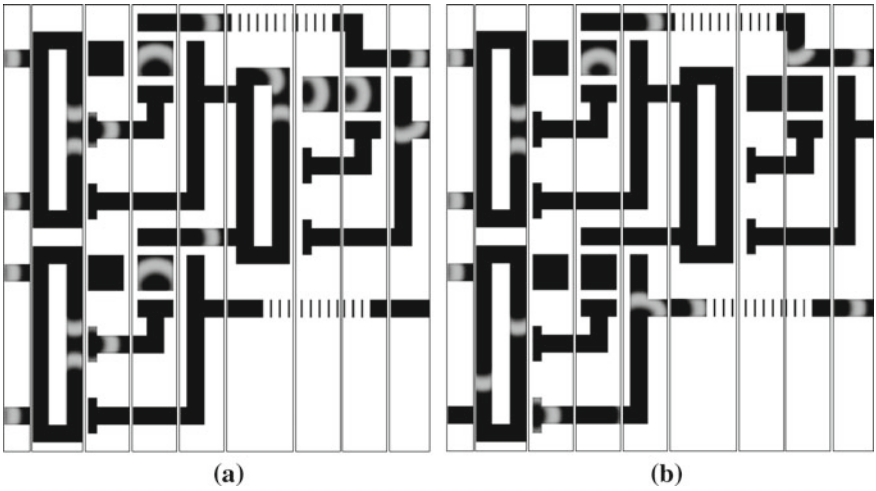

Fig. 5.11 Simulation results of two-bit binary adder when inputs (A_1A_0 and B_1B_0) are **a** 11, 11. **b** 11, 10

5.4.2.3 Three-Bit Binary Adder

It is easy to design three-bit binary adder by the similar cascade method. Three-bit binary adder consists of a two-bit binary adder at the bottom and a one-bit full-adder on the top. As shown in Fig. 5.12, A_2, A_1, A_0 and B_2, B_1, B_0 are the input channels for the two three-bit binary numbers; S_2, S_1, S_0 are the sum output channels, and C_2

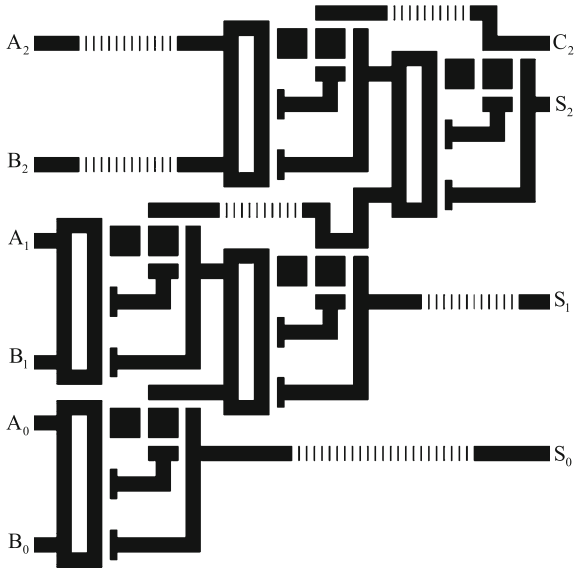

Fig. 5.12 The structure of three-bit binary adder. Grid size is 680 × 760. The structure consists of a two-bit binary adder and a one-bit full-adder

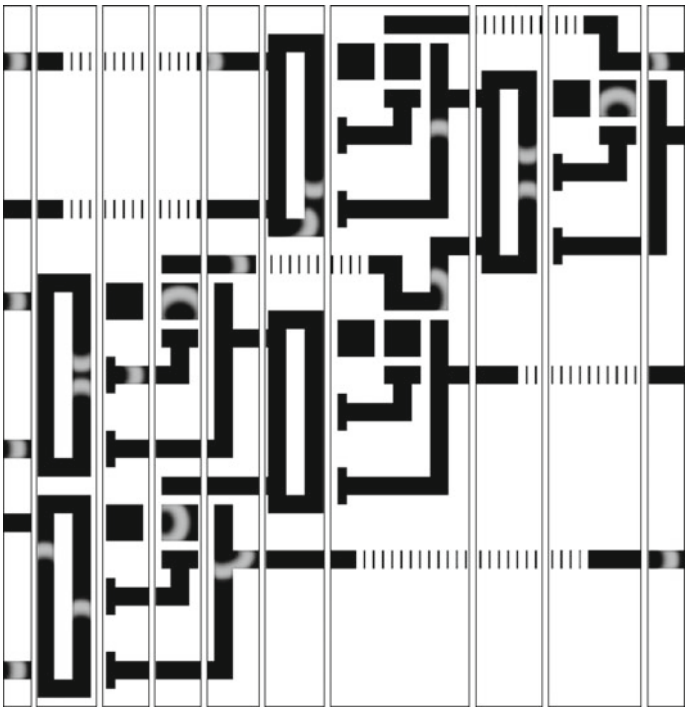

Fig. 5.13 Simulation result of three-bit binary adder when inputs ($A_2A_1A_0$ and $B_2B_1B_0$) are 110 and 011

is the carry output channel. We get two lower sums S_1 and S_0 from the two-bit adder, and add two highest bits of the inputs and the carry of the two-bit adder to generate the highest sum S_2 and the carry C_2 of the three-bit adder. Figure 5.13 shows one of the simulation results. When inputs ($A_2A_1A_0$ and $B_2B_1B_0$) are 110 and 011, the corresponding outputs ($C_2S_2S_1S_0$) are 1001. The simulations of all 64 cases of input combinations have been done to indicate that this structure can achieve the function of three-bit binary adder.

Using this cascade method and the same basic structure, higher bit binary adders can be implemented easily. An n-bit ($n \geqslant 2$) binary adder can be constructed by combining a $(n-1)$-bit adder and a one-bit full-adder.

5.4.3 Binary Comparator

5.4.3.1 One-Bit Binary Comparator

Binary comparator is used to compare binary numbers. It takes two binary numbers as input and determines whether one number is greater than, less than or equal to the

other number. A one-bit binary comparator compares two single bits A_i and B_i, and outputs the comparison result, which can be expressed as Eq. (5.5).

$$\begin{aligned}(A_i > B_i) &= A_i\bar{B}_i,\\ (A_i = B_i) &= \bar{A}_i\bar{B}_i + A_iB_i,\\ (A_i < B_i) &= \bar{A}_iB_i.\end{aligned} \tag{5.5}$$

In BZ medium, it is easy to realize $A_i > B_i$ and $A_i < B_i$ by using basic logic gates. In our design, the binary comparator only outputs the comparison result of "greater than" and "less than". If there is no pulse in these two output channels, it means that the two numbers are equal ($A_i = B_i$). Figure 5.14 shows the structure of one-bit binary comparator, which uses two T-shaped coincidence detectors to realize the logical functions of $A_i\bar{B}_i$ and $\bar{A}_iB_i$. This structure is vertically symmetrical, so that the pulses from different input channels propagate through the output channels at the same time.

For one-bit binary comparator, we consider all four combinations of inputs listed below. The simulation results are shown in Fig. 5.15.

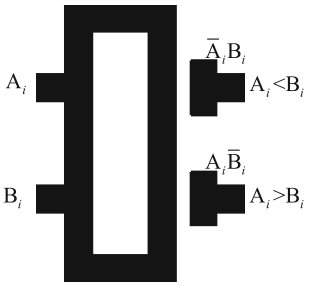

Fig. 5.14 The structure of one-bit binary comparator. Grid size is 150 × 220. Two T-shaped coincidence detectors are used in the structure to realize the logical functions of $A_i\bar{B}_i$ and $\bar{A}_iB_i$

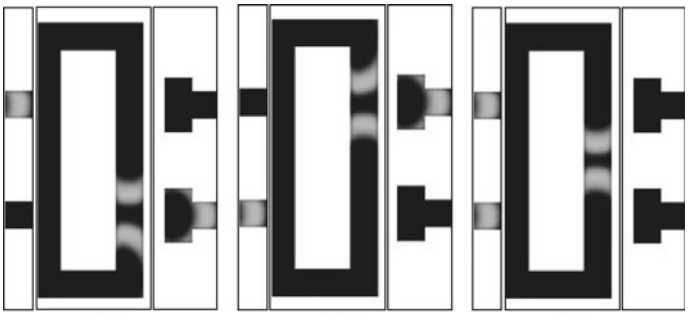

Fig. 5.15 Simulation results of one-bit binary comparator when inputs (A_i and B_i) are **a** 1, 0. **b** 0, 1. **c** 1, 1

Table 5.2 Inputs and corresponding outputs of the structure of one-bit binary comparator

Input		Output		
A_i	B_i	$A_i > B_i$	$A_i < B_i$	$A_i = B_i$
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

1. $A_i = 0, B_i = 0$: Similar to one-bit half-adder, the BZ medium stays at the stable stationary state without stimulation. There is no pulse in the output channels, so the outputs ($A_i > B_i$ and $A_i < B_i$) are 0 and 0, which means $A_i = B_i$.
2. $A_i = 1, B_i = 0$: As shown in Fig. 5.15a, pulse from channel A_i propagates through the channel, generating a new pulse in channel $A_i\bar{B}_i$. So the outputs are 1 and 0, which means the comparison result is $A_i > B_i$.
3. $A_i = 0, B_i = 1$: Opposite to 2, pulse from channel B_i generates a new pulses in channel $\bar{A}_iB_i$ in Fig. 5.15b. So the outputs are 0 and 1, which means the comparison result is $A_i < B_i$.
4. $A_i = 1, B_i = 1$: In Fig. 5.15c, two pulses from channel A_i and B_i propagate through the channels, meet in the middle and annihilate. There is no new pulse generated in output channels, so the outputs are 0, 0, which means the comparison result is $A_i = B_i$.

Table 5.2 lists all the outputs of one-bit binary comparator with different input combinations. Though the comparison result $A_i = B_i$ can not be obtained in the structure directly, it can be expressed by $A_i > B_i$ and $A_i < B_i$ indirectly. When $A_i > B_i$ and $A_i < B_i$ are both 0, it means $A_i = B_i$. Therefore, we conclude that the device shown in Fig. 5.14 realizes the comparison operation of two one-bit numbers.

5.4.3.2 Two-Bit Binary Comparator

In two-bit binary comparator, the two binary numbers to be compared are $A = A_1A_0$, $B = B_1B_0$, and the comparison result is represented by $A > B$ or $A < B$. Same as one-bit comparator, when two outputs are both 0, it means $A = B$. We implement two-bit binary comparator based on the structure of one-bit binary comparator. The structure of two-bit comparator includes two parts: the left part, which consists of two one-bit comparators, achieves comparison bit by bit; the right part synthesizes the by-bit comparison results to get the final result. Since the final result of binary comparator is mainly decided by the comparison result of the higher bits, the outer one-bit comparator compares the higher bit of the input data on the left part of the two-bit comparator. The structure of two-bit binary comparator is shown in Fig. 5.16, in which, A_1, A_0, B_1, B_0 are four input channels for two two-bit numbers, and $A > B, A < B$ are two output channels.

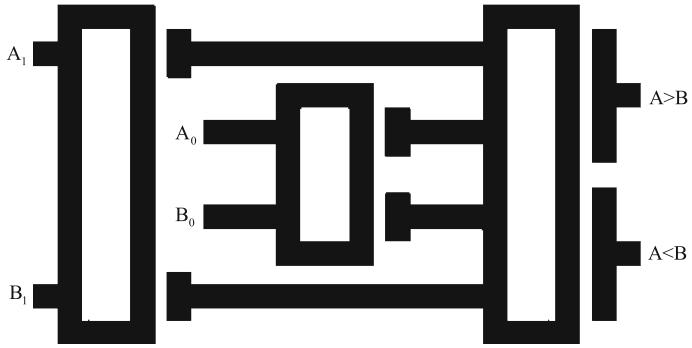

Fig. 5.16 The structure of two-bit binary comparator. Grid size is 500 × 300. The structure includes two parts: the left part achieves comparison bit by bit, the right part synthesizes the by-bit comparison results to get the final result

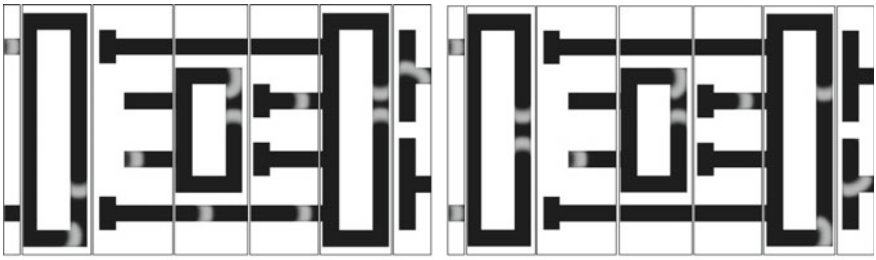

Fig. 5.17 Simulation results of two-bit binary adder when inputs (A_1A_0 and B_1B_0) are **a** 10, 01. **b** 10, 11

For this structure, we performed simulations for all sixteen input combinations to indicate that the structure can realize the comparison of two-bit binary numbers. Figure 5.17 shows two typical simulation results. In Fig. 5.17a, when inputs A_1A_0 and B_1B_0 are 10, 01, the final result is decided by the comparison result of the higher bits. So the outputs are 10, which means $A > B$. In Fig. 5.17b, when inputs A_1A_0 and B_1B_0 are 11 and 10, two higher bits are equal, the final result is decided by the comparison result of the lower bits, which means that the outputs are 01 and $A < B$.

5.4.3.3 Three-Bit Binary Comparator

Three-bit binary comparator compares two three-bit binary numbers, which are $A = A_2A_1A_0$ and $B = B_2B_1B_0$, and gives information of $A > B$ and $A < B$ as the result. Similar to two-bit comparator, three-bit comparator consists of the by-bit comparison unit and the comprehensive comparison unit. In by-bit comparison unit, three one-bit comparators on the left compare each bit of the two numbers; while

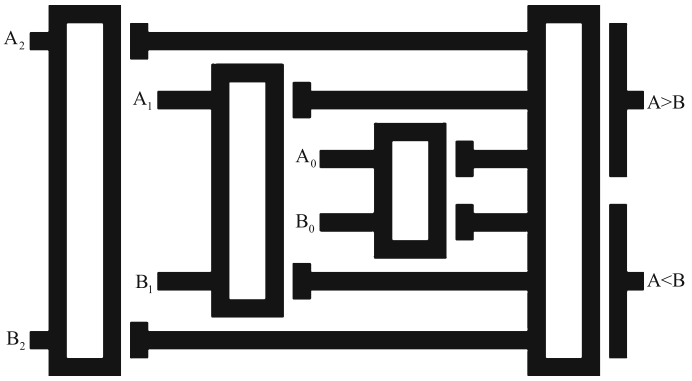

Fig. 5.18 The structure of three-bit binary comparator. Grid size is 680 × 430. There are three one-bit comparators on the left and a comprehensive comparison unit on the right

the comprehensive comparison unit on the right with six inputs synthesizes the comparison results. As shown in Fig. 5.18, the structure of three-bit comparator has six input channels $A_2, A_1, A_0, B_2, B_1, B_0$ and two output channels $A > B, A < B$. We have performed all the simulations to ensure that this structure realizes the comparison function of three-bit numbers. One of the simulation results is shown in Fig. 5.19. When inputs $A_2A_1A_0$ and $B_2B_1B_0$ are 101, 010, the outputs are 10, which mean $A > B$.

We can build higher bit binary comparator by the same cascade method. An n-bit ($n \geqslant 2$) binary comparator consists of n one-bit binary comparators and a comprehensive comparison unit with $2n$ inputs. n one-bit binary comparators with decreasing size place one inside the other. The outer one is bigger and compares higher order of

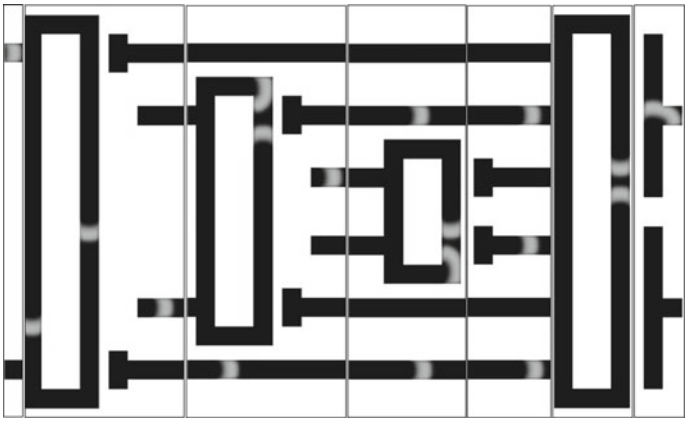

Fig. 5.19 Simulation result of three-bit binary comparator when inputs ($A_2A_1A_0$ and $B_2B_1B_0$) are 101 and 010

the input data, whereas inner one is smaller and compares lower order. The comprehensive comparison unit collects all the comparison results of every bit and compares them to achieve the final result.

5.4.4 Binary Encoder

5.4.4.1 One-Bit and Two-Bit Binary Encoder

Binary encoder compresses 2^n (or fewer) binary inputs into n outputs. In one-bit (2 to 1) binary encoder, the inputs and output are I_1, I_0 and Y_0 respectively, and Y_0 is equal to I_1. In BZ medium, we can build a one-bit encoder by connecting the input channel I_1 to the output channel Y_0 directly.

For two-bit (4 to 2) binary encoder, the Boolean expressions can be represented as:

$$\begin{aligned} Y_1 &= I_2 + I_3, \\ Y_0 &= I_1 + I_3. \end{aligned} \tag{5.6}$$

where I_3, I_2, I_1, I_0 are four inputs, Y_1 and Y_0 are two outputs. Moreover, a group signal (GS) is provided as an additional output for cascade. GS is 1 when one or more of the inputs are active, which can be expressed as:

$$GS = I_0 + I_1 + I_2 + I_3. \tag{5.7}$$

The related inputs are ORed to produce all three outputs. In BZ medium, this device can be constructed by using OR gates and chemical diodes directly. In the simple case, we ignore the order of inputs and outputs in the structure. As shown in Fig. 5.20, input channel I_3 is put between channels I_1 and I_2 to generate outputs Y_0 and Y_1 conveniently; furthermore, channels Y_0 and Y_1 intersect, so that the pulse from channel I_1 can propagate to channel GS. Since two pulses from channels Y_0 and Y_1 never go through the crossover at the same time, a modified crossover structure is employed to achieve crossing in this structure.

Since one and only one of four inputs is 1 at each time, there are four kinds of input combinations for two-bit binary encoder, which are listed as follows. Figure 5.21 shows the simulation results.

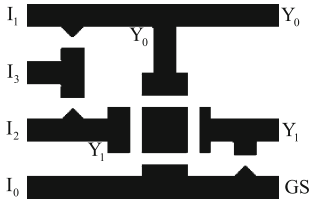

Fig. 5.20 The structure of two-bit binary encoder. Grid size is 220 × 190. A modified crossover structure is used to exchange the order of Y_0 and Y_1

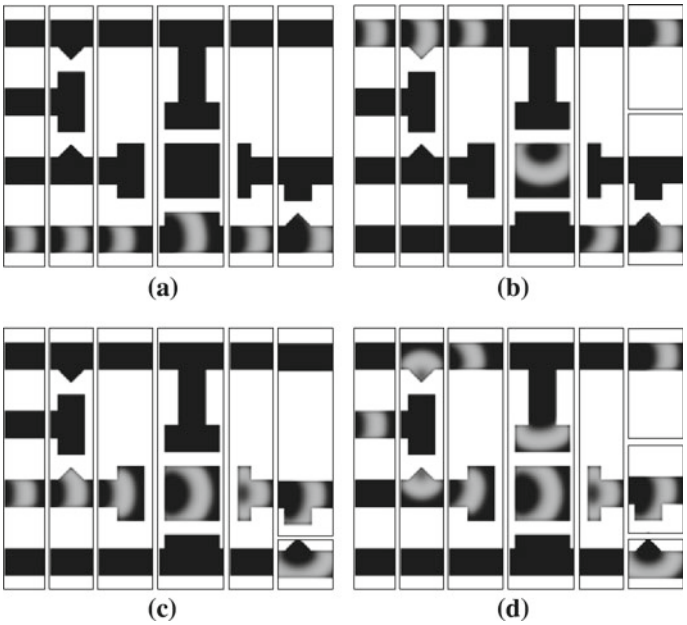

Fig. 5.21 Simulation results of two-bit binary encoder when inputs ($I_3I_2I_1I_0$) are **a** 0001, **b** 0010, **c** 0100, **d** 1000

1. $I_3I_2I_1I_0 = 0001$: As shown in Fig. 5.21a, pulse from channel I_0 propagates and outputs through channel GS. So the outputs ($GS\ Y_1Y_0$) are 100.
2. $I_3I_2I_1I_0 = 0010$: When channel I_1 has pulse (Fig. 5.21b), a single pulse passes the crossover and sends through channels GS and Y_0. So the outputs are 101.
3. $I_3I_2I_1I_0 = 0100$: Similar to 2, pulse from channel I_2 pulse passes the crossover and sends through channels GS and Y_1, as shown in Fig. 5.21c. So the outputs are 110.
4. $I_3I_2I_1I_0 = 1000$: The pulse from channel I_3 generates two pulses in channels Y_0 and Y_1, so two pulses propagate into the crossover. The pulse from channel Y_1 arrives at the crossover earlier than the pulse from channel Y_0, as shown in Fig. 5.21d, then the pulse from channel Y_0 cannot pass the crossover. As a result, there is still one output pulse in channel GS, the outputs are 111.

Table 5.3 summarizes all possible inputs and the corresponding outputs. It is clear that this device implements information encoding of two bits.

5.4.4.2 Three-Bit Binary Encoder

For three-bit (8 to 3) binary encoder, $I_7, I_6, I_5, I_4, I_3, I_2, I_1, I_0$ are eight inputs; Y_2, Y_1, Y_0 are three outputs and GS is the additional output for cascade. Three-bit

Table 5.3 Inputs and corresponding outputs of the structure of two-bit binary encoder

Input				Output		
I_3	I_2	I_1	I_0	GS	Y_1	Y_0
0	0	0	1	1	0	0
0	0	1	0	1	0	1
0	1	0	0	1	1	0
1	0	0	0	1	1	1

binary encoder can be constructed based on two-bit binary encoder by cascade method. The structure of three-bit encoder consists of two two-bit binary encoders, in which the outputs of individual two-bit encoders are ORed to produce Y_0, Y_1 and GS, respectively; and GS of the higher bit encoder is regarded as output Y_2. Three additional crossover structures are introduced in the structure to produce the final outputs, as shown in Fig. 5.22a.

For the structure of three-bit binary encoder, we performed simulations for all eight input combinations. Figure 5.22b shows one of the simulation results, in which the inputs ($I_7I_6I_5I_4I_3I_2I_1I_0$) are 1000000. When I_7 is 1, the outputs of the higher bit encoder are 111. These output pulses pass three additional crossover structures, giving 1111 as the final outputs ($GS\ Y_2Y_1Y_0$). The simulation results indicate that this structure realizes the encoding of three-bit binary numbers.

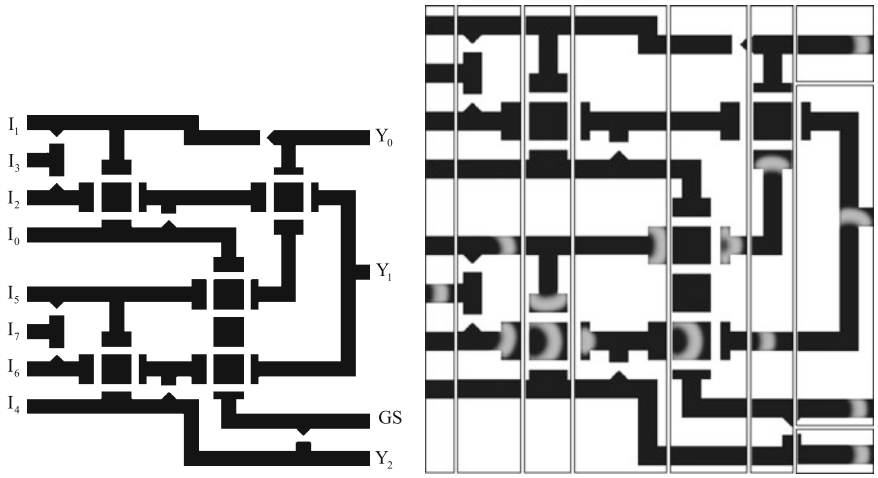

Fig. 5.22 **a** The structure of three-bit binary encoder. Grid size is 460 × 490. This structure consists of two two-bit binary encoders and three additional crossover structures. **b** Simulation result of three-bit binary encoder when inputs ($I_7I_6I_5I_4I_3I_2I_1I_0$) are 1000000

It is easy to form higher bit binary encoders based on BZ reaction from the lower bit encoders and crossover structures. An n-bit ($n \geqslant 2$) binary encoder can be constructed by cascading two $(n-1)$-bit encoders.

5.4.5 *Binary Decoder*

5.4.5.1 One-Bit and Two-Bit Binary Decoder

A decoder is a digital circuit that performs the inverse operation of an encoder. It has n input lines and a maximum of 2^n unique output lines. The logic expressions of one-bit (1 to 2) binary decoder are given by Eq. (5.8), where I_0 is the input, and Y_1, Y_0 are two outputs. It is easy to build one-bit binary decoder in BZ medium by combining an AND gate and a NOT (AND NOT) gate with an auxiliary 1 input. Figure 5.23 shows the structure, in which the auxiliary 1 input is put on the top.

$$\begin{aligned} Y_1 &= I_0 = I_0 1, \\ Y_0 &= \bar{I}_0 = \bar{I}_0 1. \end{aligned} \tag{5.8}$$

A two-bit (2 to 4) binary decoder has two inputs I_1, I_0, and four outputs Y_3, Y_2, Y_1, Y_0. The Boolean expressions of two-bit decoder are expressed as:

$$\begin{aligned} Y_3 &= I_1 I_0, \quad Y_2 = I_1 \bar{I}_0, \\ Y_1 &= \bar{I}_1 I_0, \quad Y_0 = \bar{I}_1 \bar{I}_0. \end{aligned} \tag{5.9}$$

Substituting Eq. (5.8) into Eq. (5.9), we have:

$$\begin{aligned} Y_3 &= I_1 Y_1', \quad Y_2 = I_1 Y_0', \\ Y_1 &= \bar{I}_1 Y_1', \quad Y_0 = \bar{I}_1 Y_0'. \end{aligned} \tag{5.10}$$

where Y_1', Y_0' are the outputs of one-bit binary decoder. Equation (5.10) links two-bit decoder with one-bit decoder. A cascade method is adopt here to build two-bit decoder. Based on the structure of one-bit decoder, two-bit decoder can be divided into two parts: a one-bit decoder with lower bit input I_0 and the auxiliary 1 input

Fig. 5.23 The structure of one-bit binary decoder. Grid size is 130 × 160. It is constructed by combining an AND gate and a NOT (AND NOT) gate with an auxiliary 1 input

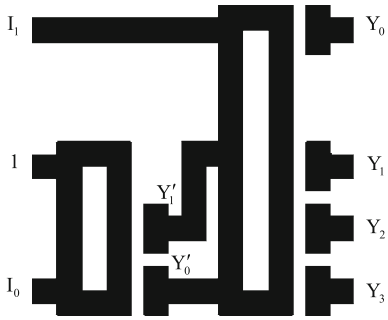

Fig. 5.24 The structure of two-bit binary decoder. Grid size is 260 × 270. This structure consists of a one-bit binary decoder and a rectangular ring structure

as inputs, and a rectangular ring structure with outputs of one-bit decoder Y_1', Y_0' and higher bit input I_1 as inputs. Figure 5.24 shows the structure of two-bit binary decoder, in which, the length of the output channels of one-bit decoder is adjusted to keep the pulses in pace.

Figure 5.25 shows all simulation results of two-bit binary decoder. There are four kinds of input combinations for two-bit decoder, which are listed as follows:

1. $I_1I_0 = 00$: As shown in Fig. 5.25a, with the absence of inputs I_1 and I_0, a pulse from the auxiliary 1 input channel propagates in the structure of one-bit decoder, generating a new pulse in channel Y_0'. This pulse continues propagating in the rectangular ring structure, and send through the output channel Y_0, which means that the final outputs ($Y_3Y_2Y_1Y_0$) are 0001.
2. $I_1I_0 = 01$: When channel I_0 has pulse (Fig. 5.25b), two pulses propagate through the input channels, generating and sending a new pulse through channel Y_1'. Then the pulse from channel Y_1' generates another new pulse in output channel Y_1, suggesting the outputs are 0010.
3. $I_1I_0 = 10$: Similar to 1, channel Y_0' has an output pulse in one-bit decoder. The pulses from input channel I_1 and channel Y_0' collide and generate a new pulse in output channel Y_2, as shown in Fig. 5.25c. So the outputs are 0100.
4. $I_1I_0 = 11$: In Fig. 5.25d, with the presence of inputs I_1, I_0 and auxiliary 1, the pulses are sent through Y_1' and Y_3 in one-bit and two-bit decoders respectively, so the outputs are 1000.

All possible inputs and the corresponding outputs are listed in Table 5.4. The simulation results and the truth table demonstrate that this structure realizes the decoding function of two bits.

5.4.5.2 Three-Bit Binary Decoder

Three-bit binary decoder (3 to 8 decoder) has three inputs I_2, I_1, I_0 and eight outputs $Y_7, Y_6, Y_5, Y_4, Y_3, Y_2, Y_1, Y_0$. Similarly, The logic expressions of three-bit binary decoder can be represented by highest bit input I_2 and the outputs of two-bit decoder:

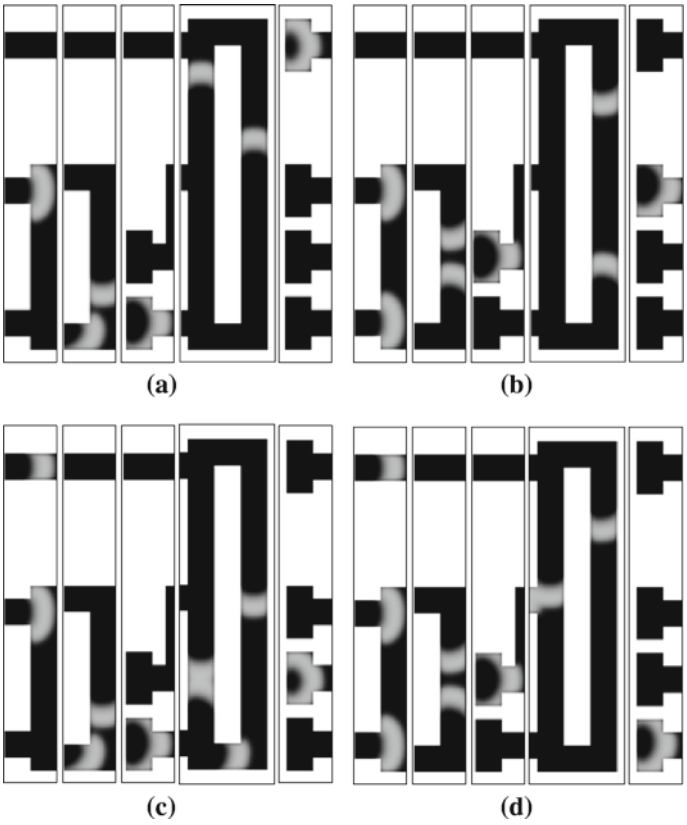

Fig. 5.25 Simulation results of two-bit binary decoder when inputs ($I_1 I_0$) are **a** 00, **b** 01, **c** 10, **d** 11

Table 5.4 Inputs and corresponding outputs of the structure of two-bit binary decoder

Input		Output			
I_1	I_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

$$\begin{aligned} Y_7 &= I_2 Y_3', \quad Y_6 = I_2 Y_2', \\ Y_5 &= I_2 Y_1', \quad Y_4 = I_2 Y_0', \\ Y_3 &= \bar{I}_2 Y_3', \quad Y_2 = \bar{I}_2 Y_2', \\ Y_1 &= \bar{I}_2 Y_1', \quad Y_0 = \bar{I}_2 Y_0' \end{aligned} \tag{5.11}$$

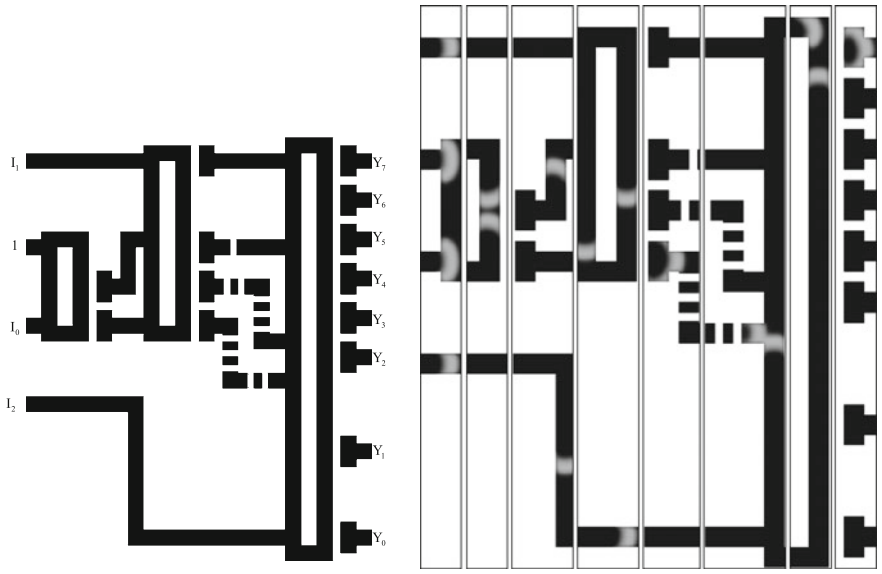

Fig. 5.26 **a** The structure of three-bit binary decoder. Grid size is 440 × 560. This structure consists of a two-bit binary decoder and a rectangular ring structure. **b** Simulation result of three-bit binary decoder when inputs ($I_2I_1I_0$) are 111

where Y_3', Y_2', Y_1', Y_0' are outputs of two-bit decoder. We build three-bit binary decoder by the similar cascade method. As shown in Fig. 5.26a, three-bit decoder consists of a two-bit decoder and a rectangular ring structure, where time delay units are used for pulse delay. In order to arrange the output channels in numerical order, the highest bit input channel I_2 is set to the bottom of the rectangular ring structure.

There are eight input combinations for three-bit binary decoder. We have performed all the simulations to ensure that this structure fulfils three-bit numbers decoding. Figure 5.26b shows one of the simulation results with inputs ($I_2I_1I_0$) 111. With the presence of all inputs, the pulses is sent through Y_3' and Y_7 in two-bit and three-bit decoders. As a result, the final outputs ($Y_7Y_6Y_5Y_4Y_3Y_2Y_1Y_0$) are 10000000.

Using the same basic structure and principle, higher bit decoders can be implemented easily. An n-bit ($n \geqslant 2$) binary decoder can be constructed by combining a $(n-1)$-bit decoder and a rectangular ring structure.

5.5 Universal Combinational Logic Circuits

In digital electronics, the logical function of any combinational logic circuit can be expressed by sum-of-products expressions. Therefore, the universal combinational logic circuits can be constructed based on BZ reaction, if we can achieve the function

of universal sum-of-products expressions. Moreover, the connection structure should be involved in combinational logic circuits, so that the input signals can be transmitted to arithmetical unit synchronously.

In this chapter, we propose a universal method to build combinational logic circuits based on their sum-of-products expressions. Firstly, we design the universal structure of sum-of-products expression to realize the arithmetical function of combinational logic circuits; secondly, we design multi-input multi-output connection structure to fulfill the transmission function. At last, the universal combinational logic circuits are constructed based on these two parts.

5.5.1 Universal Structure of Sum-of-Products Expression

Generally, sum-of-products expression is formed as:

$$Y = A_0A_1A_2A_3 + A_4A_5 + A_6A_7A_8 + \cdots + A_{n-1}A_n \tag{5.12}$$

where, $A_0, A_1, \ldots, A_n$ are binary inputs and Y is binary output. Equation 5.12 is comprised of multi-bit AND, multi-bit OR and NOT operations. These three basic structures are discussed below.

5.5.1.1 Multi-Bit AND Operation

In multi-bit AND operation, only if all the inputs are 1, the output is 1. In Sect. 5.4.1, T-shaped coincidence detector is used to achieve AND operation of two binary numbers; multi-bit AND structure can be built by using multiple T-shaped coincidence detectors. We need $n(n-1)/2$ T-shaped coincidence detectors for n-bit AND operation, which will occupy a lot of space.

In this section, we modify T-shaped coincidence detectors to straight channels separated by passive stripes, and design a multi-bit AND structure, which is simple and easily extensible. The multi-bit AND structure has two part: the inputs are on the left, while the operation and the output are on the right. In the left part of the structure, horizontal channels are applied to receive and transmit input pulses; in the right part, channels are arranged vertically for coincidence detection, the rightmost vertical channel is set as the output channel. The numbers of horizontal channels and vertical channels are both the same as the inputs. Taking 5-bit AND expression $Y = A_0A_1A_2A_3A_4$ as an example, Fig. 5.27 shows the structure. In the figure, the width of the channels and the passive stripes is 20 and 10 grid points respectively; the width of the input channels is added to 40 grid points at the end to fulfill penetration.

When input pulse penetrates to the right part from its input channel, the pulse will fork in the first vertical channel. Two forked pulses generated by different input pulses meet in the first vertical channel, generating a new pulse in the second vertical channel. Similarly, the new pulse forks in the second vertical channel, meets another

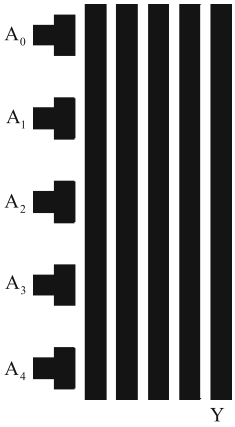

Fig. 5.27 The structure of multi-bit AND operation. Grid size is 200 × 400. This structure consists of two parts: the input part on the left, the operation and output part on the right

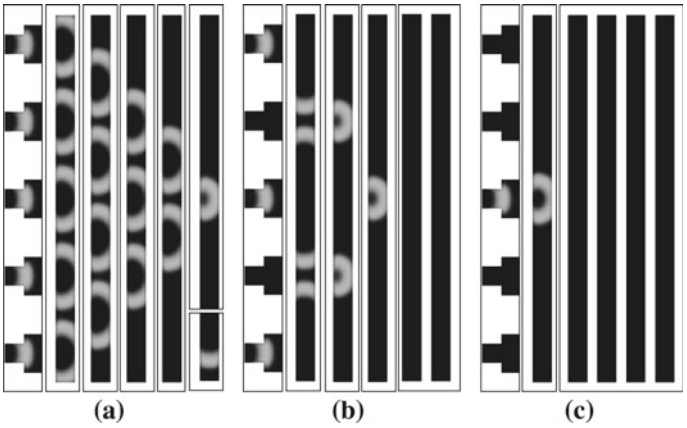

Fig. 5.28 Simulation results of multi-bit AND structure when inputs ($A_4A_3A_2A_1A_0$) are **a** 11111, **b** 10101, **c** 00100

forked pulse, exciting third vertical channel, and so on. Figure 5.28 shows some simulation results of 5-bit AND structure, in which, there are five, three and one input pulses. Simulation results indicate that the number of excited vertical channels is decided by the number of input pulses. If and only if all the inputs are 1, the rightmost vertical channel is excited, which means the output is 1.

5.5.1.2 Multi-Bit OR Operation

In multi-bit OR operation, if one or more than one inputs are 1, the output is 1. Similar to multi-bit AND structure, multi-bit OR structure also contains two parts: the left part is the input part, each input pulse enters the structure from a horizontal input channel; the right part have only one vertical channel, which achieves OR operation and outputting. Figure 5.29a demonstrates the structure of 5-bit OR expression $Y = A_0 + A_1 + A_2 + A_3 + A_4$, in which, the width of the channels is 20 grid points. With the presence of one or more than one input pulses, the vertical output channel is excited, which means the output is 1. The simulation results with five and one input pulses are shown in Fig. 5.29b, c respectively.

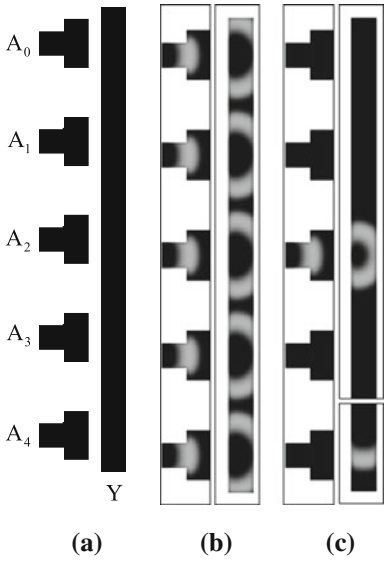

Fig. 5.29 **a** The structure of multi-bit OR operation. Grid size is 80 × 400. Similar to multi-bit AND structure, this structure also consists of two parts. **b** Simulation result of multi-bit OR structure when inputs ($A_4A_3A_2A_1A_0$) are 11111. **c** Simulation result of multi-bit OR structure when inputs ($A_4A_3A_2A_1A_0$) are 00100

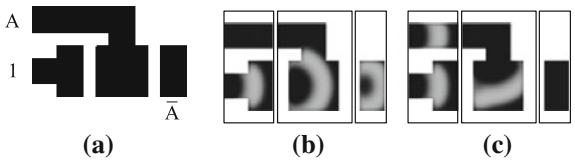

Fig. 5.30 **a** The structure of NOT operation. Grid size is 130 × 90. This structure is modified from crossover structure. **b** Simulation result of NOT structure when input (A) is 0. **c** Simulation result of NOT structure when input (A) is 1

5.5.1.3 NOT Operation

NOT operation is realized in BZ medium by rectangular ring structure in Sect. 5.4.1. In this section, we design a new NOT structure based on modified crossover structure, in order to connect it to the straight channels in multi-bit AND structure and multi-bit OR structure conveniently.

Similar to Sect. 5.4.1, an auxiliary 1 input is needed, except input A of NOT operation. Taking A and auxiliary 1 as two inputs of the crossover structure, we can obtain $\bar{A}$ in the output channel opposite to the input channel of 1. Furthermore, we connect input channel A to square in the middle of the crossover structure, as shown in Fig. 5.30a, so that two input channels are horizontal, and two input pulses can arrive the center square at the same time.

Figure 5.30b, c show the simulation results. When input A is 0, the pulse from the auxiliary 1 input channel penetrates two passive stripes, generating a new pulse in channel $\bar{A}$. So the output $\bar{A}$ is 1. When input A is 1, the pulse from the auxiliary 1 input channel penetrates one passive stripe, meets the input pulse in the center of the modified crossover structure. The mixed pulse can not pass the second passive stripe, there is no pulse in channel $\bar{A}$, giving 0 as the output.

5.5.1.4 Universal Structure of Sum-of-Products Expression

It is easy to achieve the function of sum-of-products expression based on BZ reaction, by combining multi-bit AND structure, multi-bit OR structure and NOT structure. In this section, we take the following logic expression (Eq. 5.13) as an example, and discuss the design of the universal structure of sum-of-products expression.

$$Y = A_0\bar{A_1}A_2 + \bar{A_3}A_4 \tag{5.13}$$

Equation 5.13 is comprised of two multi-bit AND, two NOT and one multi-bit OR operations. So the structure of Eq. 5.13 includes two multi-bit AND structures,two NOT structures and one multi-bit OR structure. The structure is constructed in BZ medium by the following steps:

1. Build the multi-bit AND structures of $A_0A_1A_2$ and A_3A_4 respectively.
2. Connect NOT structures to the inputs of multi-bit AND structures to achieve the logical functions of $A_0\bar{A_1}A_2$ and $\bar{A_3}A_4$.
3. Connect the rightmost channels of two multi-bit AND structures to achieve OR operation of $A_0\bar{A_1}A_2$ and $\bar{A_3}A_4$.

The structure of Eq. 5.13 is shown in Fig. 5.31a. Figure 5.31b, c demonstrate two simulation results.

When inputs ($A_4A_3A_2A_1A_0$) are 11111, the input pulses from channels A_1 and A_3 pass NOT structure and annihilate. There are two input pulses in multi-bit AND structure $A_0A_1A_2$, which excites two vertical channels. Similarly, there is one input

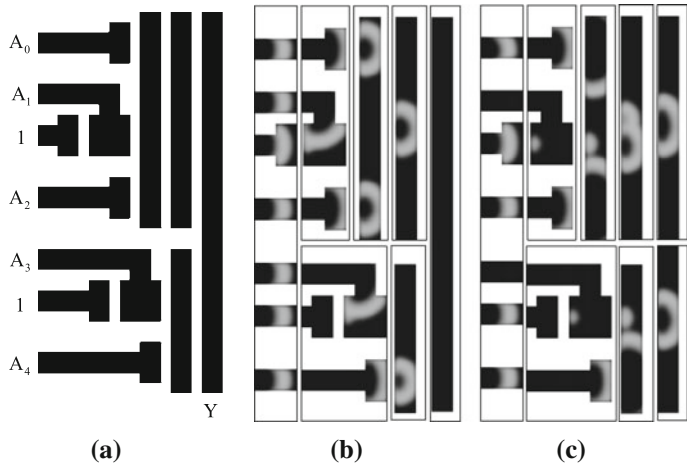

Fig. 5.31 **a** Universal structure of sum-of-products expression of Eq. 5.13. Grid size is 190 × 390. **b** Simulation result when inputs ($A_4A_3A_2A_1A_0$) are 11111. **c** Simulation result when inputs ($A_4A_3A_2A_1A_0$) are 10101

pulse in multi-bit AND structure A_3A_4, so only the leftmost vertical channel is excited. As a result, no new pulse is generated in the rightmost channel, which means the output is 0. When inputs are 10101, there are three and two input pulses in two multi-bit AND structures $A_0A_1A_2$ and A_3A_4 respectively, so the rightmost vertical channel is excited twice, giving 1 as the final output.

5.5.2 *Multi-input Multi-output Connection Structure*

The logical function of sum-of-products expression, such as Eq. 5.13, has been realized based on BZ reaction in previous section. However, in the case that one input is involved in more than one multi-bit AND operations, for example, logic expression $Y = A_0A_1 + A_0A_2$, we should solve the crossing problem of different inputs by designing a new connection structure.

Supposing the input pulses pass the connection structure from left to right synchronously; meanwhile, the pulses output below for logical operation. Furthermore, we try to keep the symmetry of the structure, so that the output pulses leave the structure from right side and underside at the same time. In this section, we extend the crossover structure to build the multi-input multi-output connection structure. The construction steps are listed below:

1. Arrange 40-grid point wide square structures vertically as the input channels of the pulses.

2. Extend crossover structures in the plane by connecting center square of each crossover. The numbers of squares in the horizontal and vertical directions are the same as that of the inputs.
3. Connect the input squares to the crossover array horizontally, so that the pulses can pass through the structure from left to right.
4. Lengthen the proper squares to 90 grid points vertically, so that the pulses can propagate below, since the wave vectors of the pulses have been changed.
5. Lengthen the lengthened squares horizontally for diagonal symmetry of the structure.
6. Fill the blanks in the structure with squares to ensure the pulse propagation.

Taking three inputs as an example, the connection structure is shown in Fig. 5.32, in which, I_0, I_1, I_2 are three input channels, O_0, O_1, O_2 and T_0, T_1, T_2 are output channels for logical operation and transmission respectively.

In Fig. 5.32, when pulses pass through the marked regions horizontally, these regions are in the refractory period of the BZ medium, so the pulses in the upside input channels, such as I_0 and I_1, can not propagate vertically in these regions. We can solve this problem by lengthening the rectangular region above the marked region, until the time interval between two pulses pass the mark region is greater than the refractory period. However, this structure will occupy huge space. For simplicity, we change the concentrations of activator and catalyst to stationary concentrations in the marked regions and the regions below in the process of simulation, so that the pulses can propagate in the structure normally.

Figure 5.33 demonstrates the simulation results with one and three inputs. In the horizontal direction, the pulse penetrates the passive stripes and outputs from the right side. In the vertical direction, when the pulse passes the rectangular region, the wave vector of the pulse changes, so the pulse can penetrate the passive stripes and output below. In Fig. 5.33a, the pulse from input channel I_1 passes through the structure and outputs from channels O_1 and E_1 at the same time. In Fig. 5.33b, three pulses pass their crossover structures, and output synchronously from six output channels.

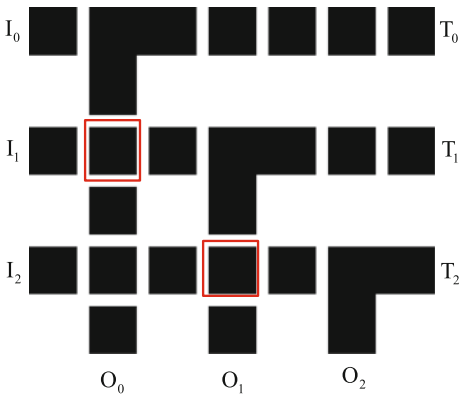

Fig. 5.32 The multi-input multi-output connection structure. Grid size is 360×310. This structure is constructed by extending the crossover structure

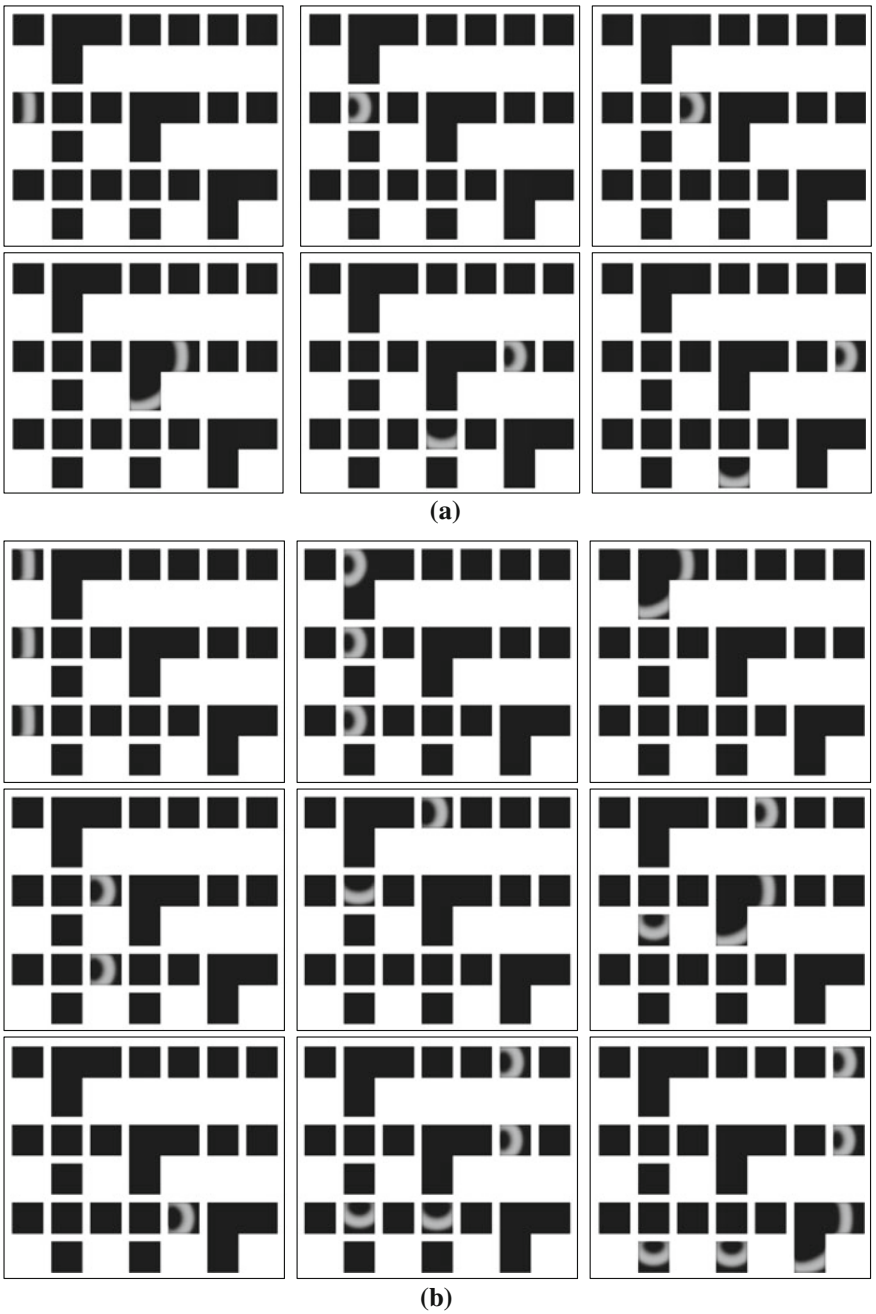

Fig. 5.33 Simulation results of multi-input multi-output connection structure with **a** one input, **b** three inputs

5.5.3 *Universal Structure of Combinational Logic Circuits*

The logical function of any combinational logic circuit can be converted to the corresponding sum-of-products expressions, so we achieve its logical function in BZ medium by combining the sum-of-products structure and connection structure. The universal method to build combinational logic circuit is described below:

1. Make sure the number of inputs in connection structure according to the inputs of sum-of-products expressions.
2. Make sure the number of connection structures according to the sum-of-products expressions.
3. Build the connection structure by the steps listed in Sect. 5.5.2.
4. Make sure the number of sum-of-products structures according to the outputs of sum-of-products expressions.
5. Construct every sum-of-products structures by the steps listed in Sect. 5.5.1.4.
6. Connect all the structures to achieve the logical function of the combinational logic circuit.

The block diagram of the universal structure of combinational logic circuit is shown in Fig. 5.34. In the figure, the inputs on the left propagate to the sum-of-products structures below through connection structures, then the results of the combinational logic circuit are outputted below. We take two-bit binary encoder and one-bit binary adder as two examples to verify this universal construction method.

In two-bit binary encoder, there are four inputs I_3, I_2, I_1, I_0 and two outputs Y_1, Y_0. According to Eq. 5.6, input I_0 is useless to the logic expressions. In the universal structure of two-bit binary encoder, we need two connection structures with three inputs for transmission and two multi-bit OR structures for calculation, as shown in Fig. 5.35a. Figure 5.35b demonstrates the simulation result when inputs ($I_3I_2I_1I_0$) are 1000. We also do simulations with inputs 0100 and 0010, the simulation results indicate that this structure fulfils the logical function of two-bit binary encoder.

In one-bit binary adder, there are two inputs A_i, B_i and two outputs C_i, S_i. According to Eq. 5.3, we need two connection structures with two inputs, two NOT structures, two AND structures and one OR structure to achieve $S_i = \bar{A}_i B_i + A_i \bar{B}_i$; while

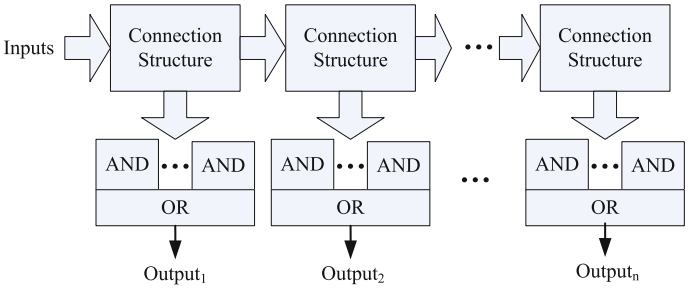

Fig. 5.34 Block diagram of the universal structure of combinational logic circuits

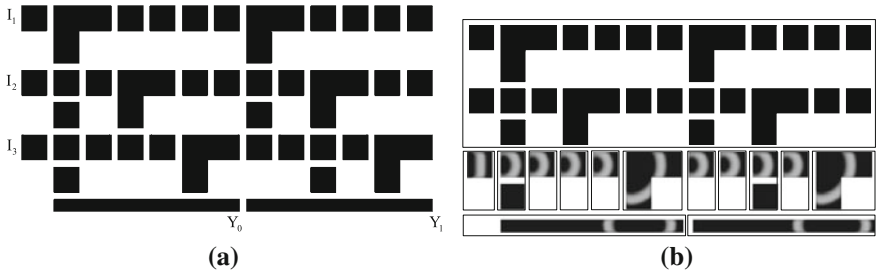

Fig. 5.35 **a** Universal structure of two-bit binary encoder. Grid size is 660 × 340. **b** Simulation result of two-bit encoder when inputs ($I_3I_2I_1I_0$) are 1000

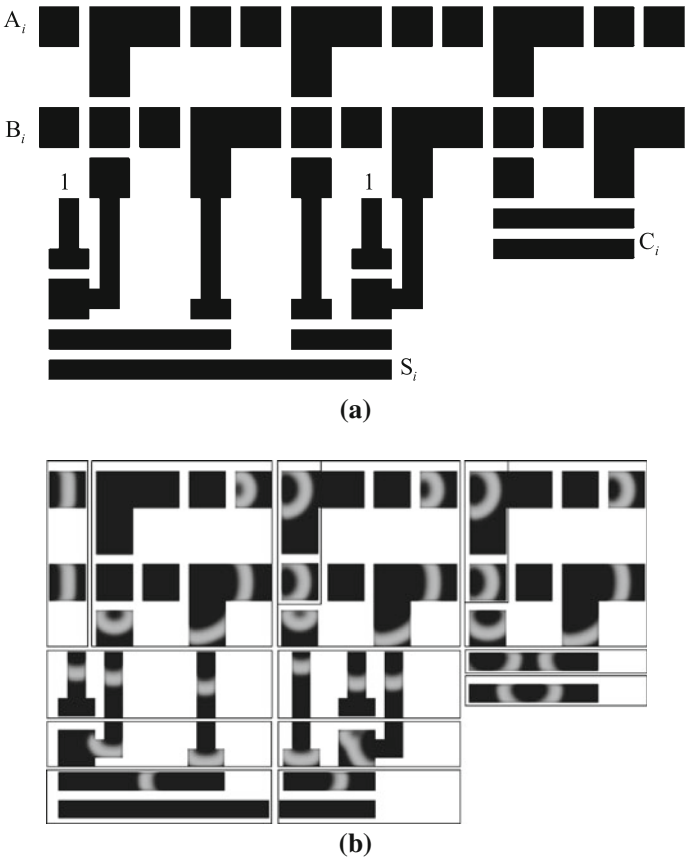

Fig. 5.36 **a** Universal structure of one-bit binary adder. Grid size is 660 × 390. **b** Simulation result of one-bit adder when inputs (A_i, B_i) are 1, 1

another connection structure with two inputs and an AND structure are used for $Ci = A_iB_i$. Figure 5.36 shows the structure and one simulation result with inputs 11. With all the simulations for four input combinations, we conclude that this structure achieve the adding function of two bits.

5.6 Conclusion

It is very important to realize combinational logic circuit based on BZ reaction, since logical functions are the basis for chemical computation. In this chapter, we focus on the design and implementation of combinational logic circuits in BZ medium in two different perspectives. On one hand, we use the conventional method to construct classical combinational logic circuits: low-bit logic circuits are build based on the structure of logic gates, while multi-bit combinational logic circuits are build base on low-bit logic circuits by the cascade method. Four types of classic combinational logic circuits, including binary adder, binary comparator, binary encoder and binary decoder, are implemented and verified in simulation. On the other hand, we propose a universal method to achieve universal combinational logic circuits according to their sum-of-products expressions. Universal structure of sum-of-products expressions and multi-input multi-output structure are formed respectively for logical arithmetic and signal transmission. The universal structures of two-bit binary encoder and one-bit binary adder are designed and implemented in simulation to verify the universal construction method.

The combinational logic circuits based on conventional method have compact structures and can be expended to multi-bit easily, but their fixed structures can only be used to solve fixed problems; While the combinational logic circuits based on universal method have flexible structures, but the structures are huge, and their implementation is restricted by the properties of BZ medium, such as refractory period. By combining these two methods, we can construct different types of combinational logic circuits, which expands the logical functions of chemical computation, facilitating the constructions of chemical computer and other intelligent systems.

Acknowledgments We are grateful to the National Natural Science Foundation of China (NSFC) (61327802, 61273341), Tianjin Research Program of Application Foundation and Advanced Technology (14JCQNJC04700), the National High Technology Research and Development Program (863 Program) of China (2013AA041102).

References

1. Adamatzky, A., Arena, P., Basile, A., Carmona-Galan, R., Costello, B.D.L., Fortuna, L., Frasca, M., Rodriguez-Vazquez, A.: Reaction-diffusion navigation robot control: from chemical to vlsi analogic processors. IEEE Trans. Circuits Syst. I: Regul. Pap. **51**, 926–938 (2004)

2. Adamatzky, A., Costello, B.D.L.: On some limitations of reaction-cdiffusion chemical computers in relation to voronoi diagram and its inversion. Phys. Lett. A **309**, 397–406 (2003)
3. Agladze, K., Aliev, R.R., Yamaguchi, T., Yoshikawa, K.: Chemical diode. J. Phys. Chem. **100**, 13895–13897 (1996)
4. Asai, T., Adamatzky, A., Amemiya, Y.: Towards reaction-diffusion computing devices based on minority-carrier transport in semiconductors. Chaos, Solitons Fract. **20**, 863–876 (2004)
5. Costello, B.D.L., Adamatzky, A., Jahan, I., Zhang, L.: Towards constructing one-bit binary adder in excitable chemical medium. Chem. Phys. **381**, 88–99 (2011)
6. Gorecka, J., Gorecki, J.: Multi-argument logical operations performed with excitable chemical medium. J. Chem. Phys. **124**, 084101 (2006)
7. Gorecka, J.N., Gorecki, J., Igarashi, Y.: One dimensional chemical signal diode constructed with two nonexcitable barriers. J. Phys. Chem. A **111**, 885–889 (2007)
8. Gorecki, J., Gorecka, J.N., Igarashi, Y.: Information processing with structured excitable medium. Nat. Comput. **8**, 473–492 (2009)
9. Gorecki, J., Yoshikawa, K., Igarashi, Y.: On chemical reactors that can count. J. Phys. Chem. A **107**, 1664–1669 (2003)
10. Holley, J., Adamatzky, A., Bull, L., Costello, B.D.L., Jahan, I.: Computational modalities of belousov-zhabotinsky encapsulated vesicles. Nano Commun. Net. **2**, 50–61 (2011)
11. Holley, J., Jahan, I., Costello, B.D.L., Bull, L., Adamatzky, A.: Logical and arithmetic circuits in belousov-zhabotinsky encapsulated disks. Phys. Rev. E **84**, 056110 (2011)
12. Ichino, T., Igarashi, Y., Motoike, I.N., Yoshikawa, K.: Different operations on a single circuit: field computation on an excitable chemical system. J. Chem. Phys. **118**, 8185 (2003)
13. Kuhnert, L., Agladze, K.I., Krinsky, V.I.: Image processing using light-sensitive chemical waves. Nature **337**, 244–247 (1989)
14. Motoike, I., Yoshikawa, K.: Information operations with an excitable field. Phys. Rev. E **59**, 5354–5360 (1999)
15. Motoike, I.N., Adamatzky, A.: Three-valued logic gates in reaction-diffusion excitable media. Chaos, Solitons Fract. **24**, 107–114 (2005)
16. Motoike, I.N., Yoshikawa, K.: Information operations with multiple pulses on an excitable field. Chaos, Solitons Fract. **17**, 455–461 (2003)
17. Murray, J.D.: Mathematical Biology: I. An Introduction. Springer, Berlin (2002)
18. Nagahara, H., Ichino, T., Yoshikawa, K.: Direction detector on an excitable field: Field computation with coincidence detection. Phys. Rev. E **70**, 036221 (2004)
19. Rambidi, N.G., Shamayaev, K.E., Peshkov, G.Y.: Image processing using light-sensitive chemical waves. Phys. Lett. A **298**, 375–382 (2002)
20. Rambidi, N.G., Yakovenchuk, D.: Finding paths in a labyrinth based on reaction-diffusion media. BioSystems **51**, 67–72 (1999)
21. Rambidi, N.G., Yakovenchuk, D.: Chemical reaction-diffusion implementation of finding the shortest paths in a labyrinth. Phys. Rev. E **63**, 026607 (2001)
22. Sielewiesiuk, J., Gorecki, J.: Logical functions of a cross junction of excitable chemical media. J. Phys. Chem. A **105**, 8189–8195 (2001)
23. Steinbock, O., Kettunen, P., Showalter, K.: Chemical wave logic gates. J. Phys. Chem. **100**, 18970–18975 (1996)
24. Steinbock, O., Toth, A., Showalter, K.: Navigating complex labyrinths: optimal paths from chemical waves. Science **267**, 868–871 (1995)
25. Stevens, W.M., Adamatzky, A., Jahan, B.D.L., Costello, I.: Time-dependent wave selection for information processing in excitable media. Phys. Rev. E **85**, 066129 (2012)
26. Sun, M.Z., Zhao, X.: Multi-bit binary decoder based on Belousov-Zhabotinsky reaction. J. Chem. Phys. **138**, 114106 (2013)
27. Szymanski, J., Gorecka, J.N., Igarashi, Y., Gizynski, K., Gorecki, J., Zauner, K., Planque, M.: Droplets with information processing ability. Int. J. Unconv. Comput. **7**, 185–200 (2011)
28. Toth, A., Gaspar, V., Showalter, K.: Signal transmission in chemical systems: propagation of chemical waves through capillary tubes. J. Phys. Chem. **98**, 522–531 (1994)
29. Toth, A., Showalter, K.: Logic gates in excitable media. J. Chem. Phys. **103**, 2058 (1995)
30. Zhang, G.M., Wong, I., Chou, M., Zhao, X.: Towards constructing multi-bit binary adder based on belousov-zhabotinsky reaction. J. Chem. Phys. **136**, 164108 (2012)