Chapter 2
Back and Forth Computing

A black cat crossing your path signifies that the animal is going somewhere.

Groucho Marx

The conjunction, disjunction and binary exclusive disjunction logic operators accept binary variables as inputs and then calculate a binary function. As we have already seen, {AND, OR, NOT} is a universal set whose elements can, in theory, be used to build any logical function. We have also seen that a few operators by themselves form a universal set, for example, {NAND} and {NOR}. However, to continue on our journey through the intricacies of computer science, we need to understand a key concept called reversibility.

2.1 Introducing Reversibility

To be able to talk about reversible computing, we need to define two new logic operations called fanout and exchange.

Fanout (FO) is an operation that divides a single input into two or more outputs with the same value as the input value, as illustrated in Fig. 2.1.

Exchange (EX) is an operation in which the pair of input connections in a two-line system is exchanged, as illustrated in Fig. 2.2.

These two simple operations are ideal for a discussion of reversibility concepts and reversible computation and, in the remainder of this chapter, we will broadly draw on *Feynman Lectures on Computation*. Accordingly, we will assume that we have a sufficient number of $|0\rangle$ and $|1\rangle$ bits to design our reversible system.

First of all, the logic operations AND, NAND, OR and XOR are not reversible; in other words, we cannot reconstruct the inputs from the outputs. In fact, it is evident that, with irreversible operations, we lose information that cannot be retrieved. But is there some way to avoid this information loss? Let us try to define a reversible operation as a logic operation that has sufficient information in the outputs to allow the inputs to be reconstructed.

V. Moret-Bonillo, *Adventures in Computer Science*,
DOI 10.1007/978-3-319-64807-1_2

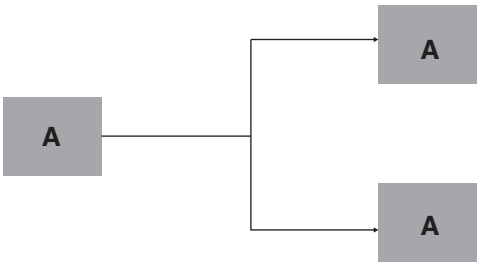

Fig. 2.1 The fanout operation, FO

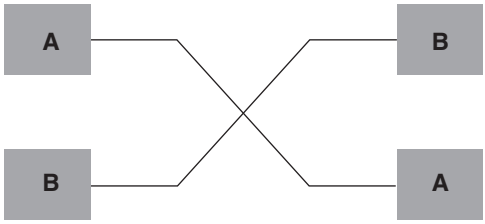

Fig. 2.2 The exchange operation, EX

Reversibility is a basic property that has to be considered if we want to understand certain very important questions related to the thermodynamics of computation. This is because reversibility allows us to calculate free energy (a thermodynamic property) and so determine the physical efficiency of computations. This all makes for a nice analogy between reversible computing and reversible thermodynamics. Charles H. Bennett and Edward Fredkin were the first scientists who independently studied the possibility of building reversible computers, which need to be equipped with reversible logic gates.

The first reversible logic operation we analyze is binary negation (logic operator NOT). To simplify the notation, we will denote the NOT operation by the letter N and represent it using the symbols ♦, or ■, which will remind us of the reversibility of this operation. It is obvious that binary negation is reversible:

$$|1\rangle = \text{NOT}\,|0\rangle = N\,|0\rangle$$
$$|0\rangle = \text{NOT}\,|1\rangle = N\,|1\rangle$$

Similar to the N gate, but not identical, is a gate usually called controlled binary negation or, more simply, controlled-NOT (CN), a device that operates on two inputs to generate two outputs. Figure 2.3 illustrates a standard representation of the CN gate.

In the CN reversible logic gate, the symbol ♦ is a NOT operation that is controlled by the input of the control line •. The way CN operates is as follows:

$$\begin{aligned} |A'\rangle &= |A\rangle, \text{always} \\ \text{If}\,|A\rangle &= |0\rangle, \text{then}\,|B'\rangle = |B\rangle \\ \text{If}\,|A\rangle &= |1\rangle, \text{then}\,|B'\rangle = \text{NOT}\,|B\rangle \end{aligned}$$

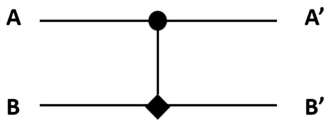

Fig. 2.3 The controlled-NOT gate, CN

Table 2.1 Truth table for the CN gate

$\|A\rangle$	$\|B\rangle$	$\|A'\rangle$	$\|B'\rangle$
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$

We have to interpret the above rules as follows: if the input to the control line A (•) is $|1\rangle$, then the output of the controlled line B (♦) is the negation of the input value of B. Conversely, if the input to the control line A (•) is $|0\rangle$, then the output of the controlled line B (♦) has the same value as its input value. The value of the input in the control line (•) activates a NOT in the controlled line (♦), but the output of the control line (•) is always the same as the input value. Table 2.1 is the truth table of the CN gate.

Clearly we can interpret B' as the output of a binary exclusive disjunction, or XOR, gate ($\bigoplus$), with inputs A and B:

$$B' = A \bigoplus B$$

However, note that the two operations are not the same, since the CN gate generates two outputs, whereas the XOR gate generates only one output. And what about the reversibility of CN? We can easily verify that CN is perfectly reversible since, once its outputs are known, we can always reproduce the inputs. We can check the reversibility of CN just by applying this gate twice, as illustrated in Fig. 2.4 and in Table 2.2.

We can do many things with CN gates, but this gate is not universal. We want to do "everything" with our computers, but in a reversible manner, so, as well as the NOT and CN gates, we need a universal set of operators.

2.2 The Toffoli Gate

The controlled-controlled-NOT (CCN) gate, also called the Toffoli gate, is a reversible device with three lines: two control lines and a conventional NOT line. The CCN gate performs reversible operations and, like the conventional NAND and NOR gates, is in itself a universal set of (reversible) binary operators. The usual representation of the CCN reversible gate is as shown in Fig. 2.5.

The CCN gate works as follows:

There are two control lines, A and B, such that

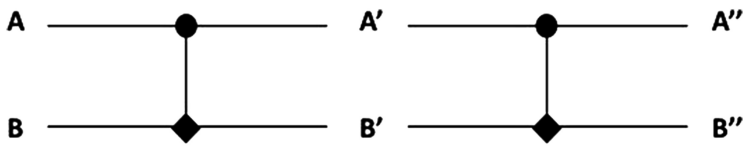

Fig. 2.4 CN reversibility in graphic form

Table 2.2 CN reversibility in tabular form

$\|A\rangle$	$\|B\rangle$	$\|A'\rangle$	$\|B'\rangle$	$\|A''\rangle$	$\|B''\rangle$
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$

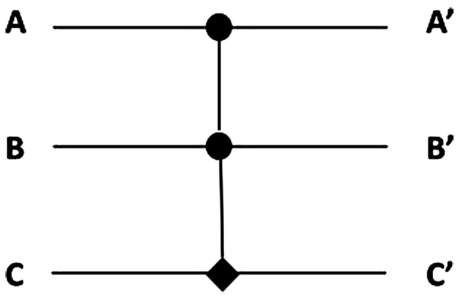

Fig. 2.5 The CCN reversible gate

$|A'\rangle = |A\rangle$ always

$|B'\rangle = |B\rangle$ always

$|A\rangle = |1\rangle$ and $|B\rangle = |1\rangle \rightarrow |C'\rangle = \text{NOT } |C\rangle$

$|A\rangle = |0\rangle$ or $|B\rangle = |0\rangle$ or $(|A\rangle = |0\rangle$ and $|B\rangle = |0\rangle) \rightarrow |C'\rangle = |C\rangle$

Thus:

If we set $|A\rangle = |B\rangle = |1\rangle$, then the corresponding CCN output is the negation of the input state of line C.

If at least one of the control lines, A or B, is in state $|0\rangle$, then the CCN gate does not change the value on line C. Table 2.3 shows the truth table for the CCN gate.

We can check the reversibility of the corresponding operation by applying CCN twice, as illustrated in Fig. 2.6 and Table 2.4.

Table 2.3 Truth table for the CCN gate

$\|A\rangle$	$\|B\rangle$	$\|C\rangle$	$\|A'\rangle$	$\|B'\rangle$	$\|C'\rangle$
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$
$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$

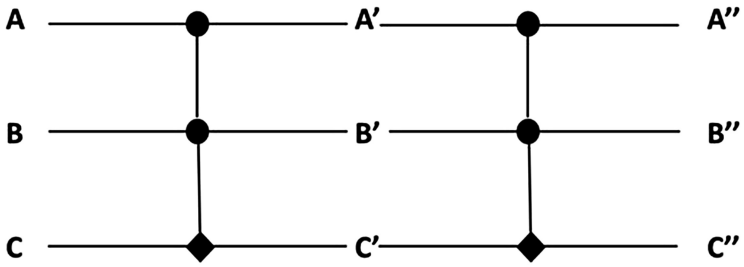

Fig. 2.6 CCN reversibility in graphic form

Table 2.4 CCN reversibility in tabular form

$\|A\rangle$	$\|B\rangle$	$\|C\rangle$	$\|A'\rangle$	$\|B'\rangle$	$\|C'\rangle$	$\|A''\rangle$	$\|B''\rangle$	$\|C''\rangle$
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$
$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$

2.3 The Fredkin Gate

Another gate of interest for reversible computing is the Fredkin controlled exchange (or swap) gate, which also has also three lines. The first line is the control line and the other two lines perform a controlled exchange, as depicted in Fig. 2.7.

The Fredkin reversible gate operates as follows:

$$
\begin{aligned}
|A'\rangle \quad &= \quad |A\rangle \text{ always} \\
\text{If } |A\rangle \quad &= \quad |0\rangle, \text{then } |B'\rangle = |B\rangle \text{ and } |C'\rangle = |C\rangle \\
\text{If } |A\rangle \quad &= \quad |1\rangle, \text{then } |B'\rangle = |C\rangle \text{ and } |C'\rangle = |B\rangle
\end{aligned}
$$

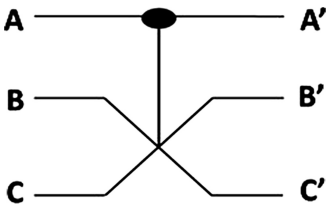

Fig. 2.7 The Fredkin gate

Table 2.5 Truth table for the Fredkin gate

$\|A\rangle$	$\|B\rangle$	$\|C\rangle$	$\|A'\rangle$	$\|B'\rangle$	$\|C'\rangle$
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$
$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$

Table 2.6 Initial Fredkin gate configuration to reproduce a conventional NAND

$\|A_0\rangle$	$\|B_0\rangle$	$\|C_0\rangle$	$\|D_0\rangle$	$\|E_0\rangle$
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$

Arriving at a deep understanding of the Fredkin gate is enjoyable and highly recommendable. Table 2.5 shows the truth table for the Fredkin gate.

Let us demonstrate that the Fredkin gate is universal as well as reversible. We will design a circuit capable of reproducing the NAND gate, which we already know to be universal. For this circuit we will only use Fredkin gates. Assume we have a circuit with five lines (A, B, C, D and E), with the initial configuration shown in Table 2.6.

We will denote by $F_{XY,\,Z}$ the application of the Fredkin gate in such a way that the exchange of the states of lines X and Y is controlled by the state of line Z. Thinking through a little, we can verify that the following sequence of operations gives us the NAND gate in line C: $F_{B0C0,\,A0} \rightarrow F_{C1D1,\,E1} \rightarrow F_{B2C2,\,D2}$. The intermediate results are as shown in Table 2.7 (a–c).

We will leave it to the reader to draw the actual circuit.

Table 2.7 Intermediate results for the Fredkin gate used to reproduce a conventional NAND following the procedure: $F_{B0C0,\,A0} \rightarrow F_{C1D1,\,E1} \rightarrow F_{B2C2,\,D2}$. The NAND gate appears in $|C_3\rangle$

(a)				
$\|A_1\rangle$	$\|B_1\rangle$	$\|C_1\rangle$	$\|D_1\rangle$	$\|E_1\rangle$
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$
(b)				
$\|A_2\rangle$	$\|B_2\rangle$	$\|C_2\rangle$	$\|D_2\rangle$	$\|E_2\rangle$
$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$
(c)				
$\|A_3\rangle$	$\|B_3\rangle$	$\|C_3\rangle$	$\|D_3\rangle$	$\|E_3\rangle$
$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$

Table 2.8 Construction of conventional AND from reversible CCN

$\|A\rangle$	$\|B\rangle$	$\|C\rangle$	$\|A'\rangle$	$\|B'\rangle$	$\|C'\rangle$
$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$	$\|0\rangle$
$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$
$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$

2.4 Building Reversible Gates from Reversible Gates

As mentioned earlier, CCN is a universal reversible gate. We will now configure CCN so that it will function like other, more conventional, logic gates. We will just configure a few of these gates, leaving the remainder as an exercise for the curious reader.

Construction of AND from CCN

The logic operator AND can be built from CCN by setting $|C\rangle = |0\rangle$ and feeding the gate with A and B, as illustrated in Table 2.8. Note that $|C'\rangle$ is the result of $|A\rangle \wedge |B\rangle$.

Construction of NAND from CCN

The logic operation NAND can be built from CCN by setting $|C\rangle = |1\rangle$ and feeding the gate with A and B, as illustrated in Table 2.9.

Table 2.9 Construction of conventional NAND from reversible CCN

$\|A\rangle$	$\|B\rangle$	$\|C\rangle$	$\|A'\rangle$	$\|B'\rangle$	$\|C'\rangle$
$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$
$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$

Table 2.10 Construction of conventional XOR from reversible CCN

$\|A\rangle$	$\|B\rangle$	$\|C\rangle$	$\|A'\rangle$	$\|B'\rangle$	$\|C'\rangle$
$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$
$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$
$\|1\rangle$	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$	$\|1\rangle$
$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|1\rangle$	$\|0\rangle$

Construction of XOR from CCN

The logic operator XOR can be built from CCN, by setting $|A\rangle = |1\rangle$ (or $|B\rangle = |1\rangle$) and feeding the gate with B and C (A and C), as illustrated in Table 2.10.

We will return later to the construction of logic gates from reversible operators.

2.5 Adding in a Reversible Way

We have already looked at the Half Adder, an architecture capable of adding single bits. We also stated that we wanted to generalize the Half Adder and turn it into a full adder, that is to say, a new device that can accept the carry of previous operations. However, we will go further and design a reversible full adder; as a consequence, we will know both the result of the operation and the corresponding inputs.

Figure 2.8 shows the basic architecture of a single-bit full adder, where C is a bit that represents an input carry whose value comes from previous additions (hence the name full adder). Nevertheless, since this operation requires three inputs and generates two outputs, it is not reversible, as it is not possible to reconstruct the three inputs from the result S and the new carry K (the output carry).

The problem is that if we want a reversible full adder, we need more information in the output. More precisely, we need two extra lines in the output, and one extra line in the input configured with a fixed value, for example, $|0\rangle$. We can proceed as follows:

1. Use N, CN and CCN, or only CCN, which, as we already know, is a universal reversible gate.
2. Build the AND, OR and XOR operators with which to design the full adder.
3. Include the two extra lines of output necessary for the system to work properly.

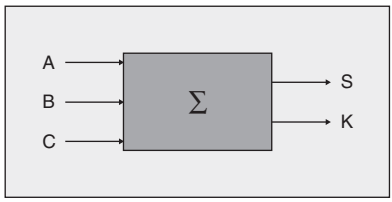

Fig. 2.8 A conventional full adder of single-bit numbers

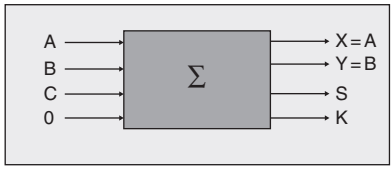

Fig. 2.9 A reversible full adder of single-bit numbers. The number of output and input lines has to be equal

4. Organize the whole system so that the four output lines are the results of the operations S and K and of the original inputs A and B.

Figure 2.9 illustrates the basic architecture of the full adder (to be analyzed later in greater detail). In the meantime, the reader can try to develop the proper internal structure for the full adder, taking into account the explanation above.

2.6 Back to Billiards

Let us return to computation with billiard balls, but now adopting a reversible perspective. We first have to make some changes to allow reversible operations and so will work with the conventional (reversible) FO operation.

Figure 2.10 illustrates how FO is very simple to implement with billiard balls: the only thing to remember is the basic structure for a collision between balls A and B and to include a new restriction: $|A\rangle = |1\rangle$. In other words, we need a ball in A.

With the new restriction, $|A\rangle = |1\rangle$, line A behaves as a control line in the input to the system. Clearly, therefore

$$\text{If } |B\rangle = |1\rangle, \text{then } |W\rangle = |1\rangle \text{ and } |Z\rangle = |1\rangle \text{ (FO in line } B)$$
$$\text{If } |B\rangle = |0\rangle, \text{then } |W\rangle = |0\rangle \text{ and } |Z\rangle = |0\rangle \text{ (nothing happens)}$$

Following the same philosophy as described earlier, it is also possible to configure the billiard ball setup to obtain a reversible CN. However, with the elements we have considered to this point (necessary though they be), a real computer cannot be simulated with billiard balls, as we are missing components that will control our computer from a mechanical perspective.

One such component is a collision gate. Using such gates, when two moving balls collide with two static balls, we have two inputs and four outputs and the net

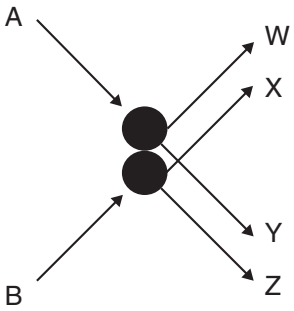

Fig. 2.10 An alternative way of representing the basic structure of a collision between two billiard balls. *A* and *B* are the inputs of the system. *W*, *X*, *Y*, and *Z* are the corresponding possible outputs

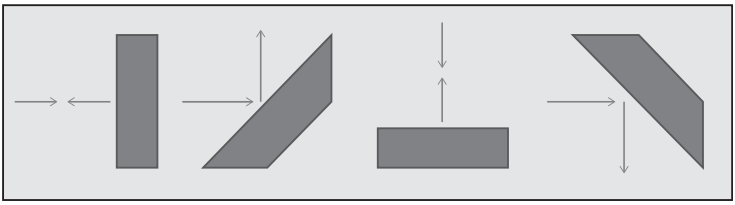

Fig. 2.11 Four basic redirection gates

result is a double FO. Another component is a redirection gate, which reflects a ball that bumps against it and changes its direction. Figure 2.11 illustrates four of these redirection gates.

The devices we still need are not so many and can be easily implemented in our billiard engine. These components will enable us to build all types of gates for the billiard ball setup, including CCN and Fredkin gates, which, as we already know, are universal reversible operators. The conclusion is clear: it is possible to design a reversible computer with billiard balls!

2.7 A Simple Analysis of Reversibility

We have seen that we can build CN, CCN and Fredkin gates with billiard balls. We have also seen that if we can build universal gates, such as the Fredkin gate, then we can build everything, and, besides, the system will be reversible.

Looking now at reversibility from another point of view, how much information do we need in the output of our device to be able to reproduce the input and, therefore, to perform reversible computing? We answer this question by recalling that

1. N accepts one input and gives back one output.

2. CN accepts two inputs and gives back two outputs.
3. CCN accepts three inputs and gives back three outputs.
4. The Fredkin gate accepts three inputs and gives back three outputs.
5. FO accepts one input and gives back two identical outputs.

The above five operations are reversible. Moreover, it seems that to work in a reversible way, we need as a necessary condition the same number of distinct input and output lines. We will discuss this in the next chapter.

2.8 Chapter Summary

We introduced the basic concepts necessary to perform reversible computations. We defined FO and EX as two operations central to reversibility and also described the CN gate, with which we can do many things. But to do everything in a reversible way we need universal reversible gates, such as the CCN (Toffoli) gate, or the Fredkin gate, which is a controlled exchange (or controlled swap) gate. We built new reversible logic gates that operate in a similar way to conventional logic gates, as practice in dealing with the problem of designing devices that are capable, for example, of adding single-bit numbers in a reversible way. We then simulated reversible computation with billiard balls. In a concluding brief analysis of reversibility, we arrived at the conclusion that the number of distinct inputs and outputs has to be the same in a device that operates in a reversible manner.

2.9 Glossary of Terms and Notation Used in This Chapter

Terms

- Free energy: a thermodynamic property that expresses the capacity of a system to perform work under certain conditions.
- Full adder: an adder of binary numbers that accounts for values carried in as well as out. A one-bit full adder adds three one-bit numbers, often written as A, B and C_{in}; A and B are the operands, and C_{in} is a bit carried in from the previous less significant stage. The full adder is usually a component in a cascade of adders that sum binary numbers (eight-, 16-, 32-bit etc.). The circuit produces a two-bit output: output carry and sum.
- Half Adder: a combinational arithmetic circuit that adds two numbers, producing a sum bit (S) and a carry bit (C) as output.
- Reversible computing: a model of computing where the computational process is to some extent reversible (in time). In a computational model that uses transitions from one state of the abstract machine to another, a necessary condition for reversibility is a one-to-one mapping from states to their successors.

- Universal gate: AND, NOT and OR are basic gates which we can combine to create any logic gate or any Boolean expression. NOR and NAND gates have the particular property that either one of them can create any logical Boolean expression if arranged properly. We examine the operation of each gate separately as universal gates.

Notation

FO	fanout operation
EX	exchange operation
N, NOT	NOT operator
♦, ■	NOT operation in a controlled line
(•)	control line
CN	controlled-NOT gate
CCN	controlled-controlled-NOT gate
F	Fredkin gate
S	SUM bit
K	CARRY bit

http://www.springer.com/978-3-319-64806-4

Adventures in Computer Science
From Classical Bits to Quantum Bits
Moret-Bonillo, V.
2017, XIII, 198 p. 55 illus., 21 illus. in color., Hardcover
ISBN: 978-3-319-64806-4