CHAPTER 4

Boolean Algebra, Logic Gates, and Quantum Information Processing

Boolean algebra uses symbols to represent statements or propositions. A *proposition* may be either true or false but not both. Boolean Algebra uses symbol 1 to represent a proposition that is true, and symbol 0 to represent the inverse of the proposition that is false. If symbols, *a, b, c*,... stand for elementary propositions that can be either true or false, then the logic connectives: *conjunction* (AND), *disjunction* (OR), and *negation* (NOT), can be used to combine these propositions to make *complex* propositions.

4.1 Boolean Algebra

Boolean algebra [1] is defined for a set $\mathcal{A}$ in terms of two binary operations $\cdot$ and +. The symbols $\cdot$ and + are called the AND and *inclusive* OR, respectively. The operations in Boolean Algebra are based on the following *axioms* or *postulates*:

1. If $x, y \varepsilon \mathcal{A}$, then $x + y \varepsilon \mathcal{A}; x \cdot y \varepsilon \mathcal{A}$

   This is known as the *closure property*.
2. If $x, y \varepsilon \mathcal{A}$, then $x + y = y + x; x \cdot y = y \cdot x$

   that is, + and $\cdot$ operations are *commutative*

3. If $x, y, z \varepsilon \mathcal{A}$, then

$$x + (y \cdot z) = (x + y) \cdot (x + z)$$
$$x \cdot (y + z) = (x \cdot y) + (x \cdot z)$$

   that is, + and · operations are *distributive*.

4. Identity elements, denoted as 0 and 1 must exist such that $x + 0 = x$ and $x \cdot 1 = x$ for all elements of $\mathcal{A}$.

5. For every element x in $\mathcal{A}$ there exists an element x', called the *complement* of x such that

$$x + x' = 1 \qquad x \cdot x' = 0$$

Note that the basic postulates are grouped in pairs. One postulate can be obtained from the other by simply interchanging all OR and AND operations, and the identity elements 0 and 1. This property is known as *duality*.

There are several theorems that can be used for manipulating Boolean functions.

Theorem 1. The identity elements 0 and 1 are unique.

Theorem 2. The *idempotent laws*

i. $x + x = x$ *ii.* $x \cdot x = x$

Theorem 3.

i. $x + 1 = 1$ *ii.* $x \cdot 0 = 0$

Theorem 4. The *absorption laws*

i. $x + xy = x$ *ii.* $x \cdot (x + y) = x$

Theorem 5. Every element in $\mathcal{A}$ has a unique complement

Theorem 6. *Involution law*

$(x')' = x$

Theorem 7.

i. $x + x'y = x + y$ *ii.* $x(x' + y) = xy$

Theorem 8. *DeMorgan's law*

i. $(x + y)' = x' \cdot y'$ *ii.* $(xy)' = x' + y'$

If a 1 is used to denote a true proposition and a 0 is used to denote a false proposition, then the AND (·) combination of two propositions can be written as follows:

$$0 \cdot 0 = 0$$
$$0 \cdot 1 = 0$$
$$1 \cdot 0 = 0$$
$$1 \cdot 1 = 1$$

The AND combinations of two propositions are known as the *product* of two propositions. The OR combinations of two statements known as the *sum* of two propositions, can be written as follows:

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 1$$

The NOT operation of a statement is true if and only if the operation is false. The NOT operation also known as *complementation* or *negation* can be stated as follows:

$$(0)' = 1$$
$$(1)' = 0$$

An AND gate implements the logical *product* operation and an OR gate implements the logical *sum* operation. The complementation is performed by an inverter (NOT) gate. Any complex proposition can be decomposed into a collection of elementary propositions, each of which can then be realized by an appropriate gate and connected together as implied in the proposition. Thus, a complex proposition can be converted into a Boolean circuit using *gates* and *wires*. The gates perform simple logic operations and the wires carry information around the circuit. A common basis for Boolean circuits is the set {AND, OR, NOT}, from which all other Boolean functions can be constructed. In addition to the operators AND, OR, and NOT, three other operators can be added to the set of elementary logic operators: *identity*, *fanout*, and *exchange*. The identity operator does not change any bit it operates on, that is, a 0 remains a 0 and a 1 remains a 1. Thus the identity operator can be considered as a wire that takes a signal from one place to another. *Fanout* splits a signal into two identical copies of itself. *Exchange* swaps two input signals when they are not equal.

4.2 Classical Circuit Computation Model

Several models have been investigated over the years for the study of classical computation, namely, Turing machines, high-level programming languages, and Boolean circuits [2]. The Boolean circuit model is not only the most appropriate because logic circuits are the basic building blocks of real-world computers, it is also the easiest to generalize for the study of quantum computation. The Boolean circuit model can be represented by the block diagram of Fig. 4.1.

It shows f as a function of n variables $(x_1, x_2, \ldots, x_n)$. If each variable can independently assume either a true (1) or a false (0) value then they are known as binary variables, and the function is referred to as a *Boolean function* of n variables. A classical circuit models of computation evaluates function f and produces an m-bit output. Thus, it computes a binary function

$$f: (0, 1)^n \rightarrow (0, 1)^m$$

that maps its n input variables to the values of its m outputs.

A Boolean function can be described by a truth table. Since each variable can be a 0 or a 1, there can be 2^ncombinations of values for n variables. For each combination of values, a function can have a value of either 0 or 1. A truth table displays the value of a function for all possible 2^ncombinations of its variables. Figure 4.2 shows the truth table for the function

A small set of circuit elements known as *logic gates* can be used to implement Boolean functions; this set is called the *basis*. The most common basis contains the following three gates: AND, OR, and NOT; they are sufficient to realize any Boolean function of the form shown in Fig. 4.1. Each gate maps its inputs to a 1-bit output, that is, $n = 2$, $m = 1$ in Fig. 4.1:

$$f: (0, 1)^2 \rightarrow (0, 1)^1$$

The AND gate produces a 1 output if and only if all input variables are 1s. Figure 4.3 shows a two-input AND gate with four possible input combinations.

Figure 4.4 shows a two input OR gate. The OR gate produces a 0 output only if all the inputs are 0.

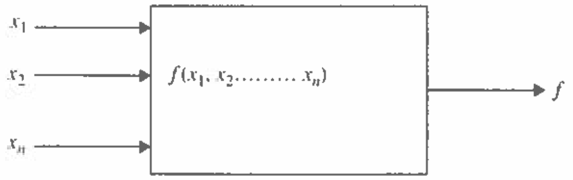
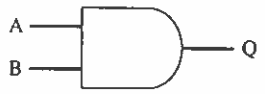


Figure 4.1 A function of n variables.

$f(a, b, c) = ab + bc + ac$

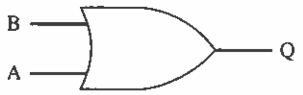
a	b	c	$f(a, b, c)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

FIGURE 4.2 Truth table for $f(a, b, c) = ab + bc + ac$.



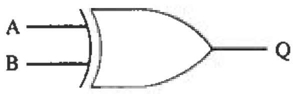
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

FIGURE 4.3 AND gate.



A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

FIGURE 4.4 OR gate.



A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

Figure 4.5 EX-OR gate.

A variant of the OR gate known as EXCLUSIVE-OR or XOR gate, has also been found to be very useful. The only difference between XOR and the conventional OR gate is that XOR produces an output 0 when both inputs are 1. Figure 4.5 shows the symbol and the truth table of the XOR gate.

The NOT gate produces an output of 1 when the input is 0, and an output of 0 when the input is 1. Figure 4.6 shows the symbol for the NOT gate. It is sometimes referred to as an *inverting buffer* or simply an *inverter*.

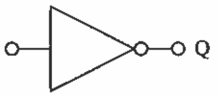
4.3 Universal Logic Gates

A set of gates is considered *universal* if every Boolean function can be implemented using only gates in this set; thus universal gates are *functionally complete*. For example, the following sets of gates are universal

(AND, NOT), (OR, NOT), (AND, XOR)

Since AND and NOT can be combined into a single gate, that is, NAND (Fig. 4.7*a*), and also OR and NOT can be combined into a NOR gate (Fig. 4.7*b*); both NAND and NOR gates are universal.

The truth table and the symbol for the NAND and the NOR gate are shown in Fig. 4.7. The bubble on the output in the graphic symbol in Fig. 4.7*a* and 4.7*b* denotes a complement operation that is performed on the output of the gates.



A	Q
0	1
1	0

Figure 4.6 NOT (inverter).

A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0

(a)

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

(b)

Figure 4.7 (*a*) NAND, (*b*) NOR.

4.4 Quantum Computation

Quantum computers utilize certain unique properties of subatomic particles in conjunction with the theories of computer science to process and store information. This merging of quantum mechanics and computer science has been extensively explored during the last three decades, and has led to the development of techniques for a class of computational problems, for example, deciphering codes, factoring large numbers, searching an unsorted collection etc. that can be solved much more efficiently using a quantum computer.

Such advances in information processing capability can be attributed to the fact that the data bits in a quantum computer, unlike their counterparts in classical computers can simultaneously exist in more than one state at a time and can be manipulated simultaneously.

Information in conventional digital representation uses a sequence of *bits*. Each bit is basically the charge of an electron. If the electron is charged, the bit is assumed to carry a value 1; alternatively the bit carries a value 0 if the electron is not charged. Thus a bit also known as a *classical bit* can be in state 0 or state 1, and measuring a bit at any time results in one of two possible outcomes.

Bit is basically the charge of electron.

4.5 The Quantum Bit and Its Representations

In quantum computing systems as in classical systems, two distinguishable states of the system are needed to represent a single bit of data. For example, consider the electron in a hydrogen atom. It can be in its ground state or in an excited state as depicted in Fig. 4.8.

If this were a classical system, it could be assumed as shown in the figure the *excited* state represent a $|1>$ and the *ground* state represent a $|0>$. In general, the electron is a quantum system, may exist in a linear superposition of the ground and excited state. It is in the ground state (0) with probability amplitude α and in the

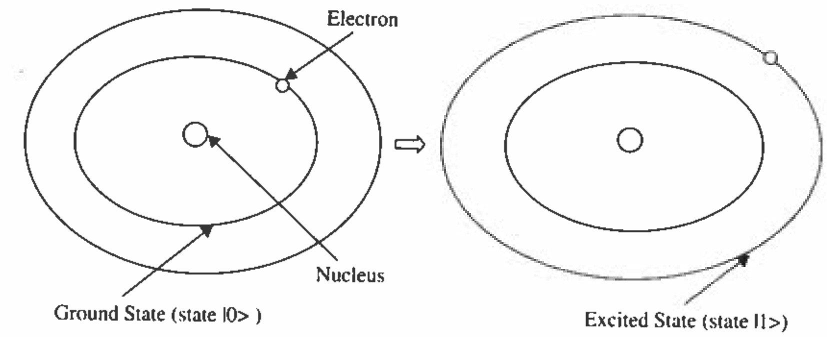


Figure 4.8 Electron in a hydrogen atom.

excited state (1) with probability amplitude β. Such a two-state quantum system is referred to as a *qubit*, and its actual state ψ can also be any linear combination (or *superposition*) of these basis states.

The state space of a qubit can be visualized by using an imaginary sphere (Fig. 4.9) known as the *Bloch sphere*. It has a unit radius. The arrow on the sphere represents the state of the qubit. Its north and south poles are selected to represent the basis states |1> and |0>, respectively; the other locations are superpositions of |0> and |1>. While the state of a classical bit can be either the north and the south pole of the equator, a qubit can be any point on the sphere.

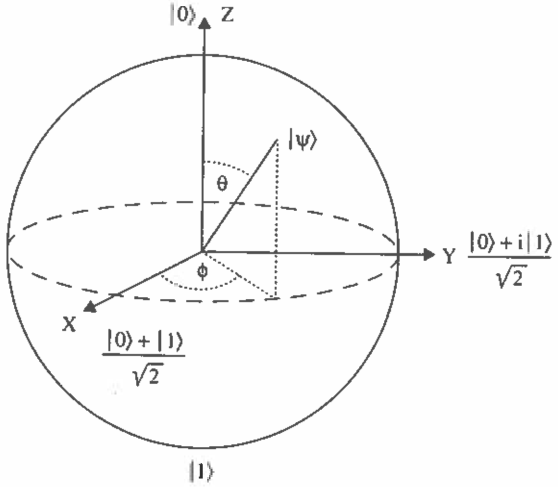

|0⟩ Z
|ψ⟩
θ
φ
Y
$\frac{|0\rangle + i|1\rangle}{\sqrt{2}}$
X
$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$
|1⟩

Figure 4.9 A qubit represented as a Bloch sphere.

The Bloch sphere allows the state of a qubit to be represented using unit spherical coordinates, for example, the polar angle θ and the azimuth angle ϕ. The Bloch sphere representation of a qubit is

$$|\psi> = \cos\frac{\theta}{2}\,|0> + e^{i\varphi}\sin\frac{\theta}{2}\,|1>$$

where $0 \le \theta < \pi$ and $0 \le \phi < 2\pi$.

The normalization constraint is

$$\left|\cos\frac{\theta}{2}\right|^2 + \left|\sin\frac{\theta}{2}\right|^2 = 1$$

Note that $|\psi> = |0>$ when $\theta = 0$, and $|\psi> = |1>$ when $\theta = \pi$, regardless of ϕ. In the Bloch sphere representation a qubit can not only be in either the north or the south pole of the sphere but also in states that are blend of these two states. In other words, a qubit can exist in multiple states simultaneously. This is basically the essence of the *principle of superposition* that happens because of the wave nature of subatomic particles.

A qubit can be physically implemented by two states of an electron orbiting a hydrogen atom (as shown in Fig. 4.8), by the spin-1/2 system with the two states $|\uparrow>$ and $|\downarrow>$, by the horizontal and the vertical polarizations of a photon or by any other two-state quantum system. A qubit responds in the same way as a classical bit when measured, that is, produces an output 0 or 1.

A unique property that makes quantum computing so special and offers such an unparallel potential, is that qubits unlike classical bits can also work with the overlap of both 0 and 1 states. For example, a 4-bit (classical) register can store one number from 0 to 15 at a time, whereas a 4-qubit register can store all 16 numbers in a superposition. All values in a qubit register can be simultaneously accessed and operated on, thus allowing truly parallel computation. A quantum state in superposition can be written as a linear combinations of $|0>$ and $|1>$

$$|\psi> = \alpha\,|0> + \beta\,|1> \qquad (4.1)$$

where $|\psi>$ is the state of the qubit and $|0>$ and $|1>$ are the *computational basis states*. The coefficients α and β are complex numbers, they are called *probability amplitudes*; the actual probabilities are given by the absolute value squared of the associated amplitude. That is, if α is the probability amplitude of 0 state then the probability of the qubit being in 0 state is $\alpha\alpha^* = |\alpha|^2$, where α^* is the complex conjugate of α. Similarly the probability of the qubit being in 1-state is $|\beta|^2$. Normalization requires that the sum of the probabilities must be 1:

$$|\alpha|^2 + |\beta|^2 = 1$$

The quantum state ψ in Eq. (4.1) can be written as a *unit column* vector in a two-dimensional complex plane spanned by the two basis states. Figure 4.10 illustrates the two basis states; these states are called *normal basis*. Note that vectors | 0 > and | 1 > are *orthogonal*, that is, perpendicular to each other. A qubit with state | 0 > is represented by the column vector $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and a qubit with state | 1 > is represented by the column vector $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$, that is

$$|\,0> = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad |\,1> = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Thus

$$\begin{aligned} \psi &= \alpha\,|\,0> + \beta\,|\,1> \\ &= \alpha\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta\begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} \alpha \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ \beta \end{pmatrix} \\ &= \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \end{aligned}$$

In other words, an arbitrary qubit state is represented by the vector $\begin{pmatrix} \alpha \\ \beta \end{pmatrix}$.

Figure 4.10 illustrates the two basis states; these two states are called *normal basis*. Note that vectors | 0 > and | 1 > are *orthogonal* that is, perpendicular to each other. The arrow in the diagram is the hypotenuse of the right-angled triangle formed by | 0 > and | 1 >, and the square of the hypotenuse equals the sum of the squares of the vertical and horizontal sides. Since this sum has the value one so the hypotenuse has the length one. Thus given an arrow of unit length its projection on the vertical and horizontal directions gives a pair of numbers, the sum of the squares of these numbers is 1. In other words, the arrow provides

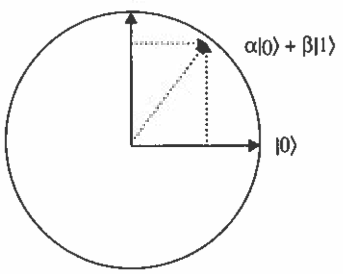


Figure 4.10 Basis states of a qubit.

all the necessary information about state of the configuration and is called the *state vector* [3].

Two other states can be derived from Eq. (4.1) by assuming $\alpha = \frac{1}{\sqrt{2}}$ and $\beta = \frac{1}{\sqrt{2}}$

$$|+> = \frac{1}{\sqrt{2}}|0> + \frac{1}{\sqrt{2}}|1>$$

$$|-> = \frac{1}{\sqrt{2}}|0> - \frac{1}{\sqrt{2}}|1>$$

$|+>$ and $|->$ also form a computational basis. The basis states $|0>$ and $|1>$ in Eq. (4.1) can be rewritten as

$$|0> = \frac{1}{\sqrt{2}}|+> + \frac{1}{\sqrt{2}}|->$$

$$|1> = \frac{1}{\sqrt{2}}|+> - \frac{1}{\sqrt{2}}|->$$

Hence, a qubit state in Eq. (4.1) can be expressed using basis $|+>$ and $|->$.

As stated earlier classical bit can only be in a single state whereas a *qubit* cannot only be in one of the two discrete states, it can also exist simultaneously in a blend of some of these states. The proportions of $|0>$ and $|1>$ in the blend need not be equal, and can be arbitrary. Thus an infinite number of possible combinations of $|0>$ and $|1>$ is possible in a qubit provided the constraint: $|\alpha|^2 + |\beta|^2 = 1$ is satisfied. Thus, in principle, it is possible to store a vast amount of information on a single qubit but it is impossible to retrieve the information. When the value in a qubit is measured, it returns $|0>$ with probability α^2 or it returns $|1>$ with probability β^2, and then the qubit assumes the state just returned.

4.6 Superposition in Quantum Systems

Superposition is a fundamental principle of quantum physics. It states that all states of a quantum system may be superimposed, that is, combined together like waves in classical physics to yield a *coherent* quantum state that is distinct from its component states. The state however collapses into a random state once it is measured.

For example, assume an electron as a qubit with *spin-up* orientation representing state $|0>$ and *spin-down* state $|1>$. However, unlike a classical bit that can only be in a single state at any time, a *qubit* can be in state *up*, *down*, or a combination of both states at the same time because of the wave-like characteristics of subatomic particles.

A qubit in superposition behaves as if it were in both | 0 > and | 1 > states simultaneously. This new state | ψ > of the qubit can be written as:

$$|\psi> = \alpha|\uparrow> + \beta|\downarrow> = \alpha|0> + \beta|1>$$

where α and β are complex numbers and are known as *probability amplitudes* as indicated earlier, satisfying the relation

$$\alpha^2 + \beta^2 = 1$$

This indicates that a qubit has the probability α^2 of being in *spin-up* (classical 0-state) and has probability β^2 of being in state *spin-down* (classical 1-state); it can also be in a coherent superposition of both. Thus, | ψ > can be considered as a vector in the two-dimensional complex vector space C^2 spanned by two basis states | 0 > and | 1 >. In other words, a quantum bit can be in all of its positions at the same time.

It should be clear from the above that the major advantage of a qubit over its classical counterpart is that an operation on a qubit while it is in a superposed state, can simultaneously affect both of its values. However, when a qubit in superposition is measured it irreversibly collapses into either 0 or 1 state, thereby destroying the superposition. After that if the qubit is measured again it gives the same result. This implies that no additional information can be obtained by repeating the measurement. It should be mentioned here that whether a particle is in two places at the same time can never in practice be observed, only a measurement can identify one state or another.

4.7 Quantum Register

A quantum register is composed of a number of qubits; the size of the register is determined by the number of qubits. For example, a quantum register of size 4 can store individual number from 0 to 15. At any particular time the 4 qubits can be in any one of 16 possible configurations:

$$0000, 0001, 0010, \ldots\ldots\ldots\ldots 1111$$

Thus, a 4-qubit register can be represented in a superposition of the above 16 states:

$$|\psi> = c_0|0000> + c_1|0001> + c_2|0010>, \ldots\ldots\ldots + c_{14}|1110> + c_{15}|1111>$$