Chapter 1

Introduction to Quantum Bits and Quantum Gates on IBM's Quantum Computer

Today from the viewpoint of computing characteristic, "Computer Science" in fact consists of traditional digital computers [Turing 1937, von Neumann 1956], bio-molecular computers [Adleman 1994] and quantum computers [Deutsch 1985]. Today we can build traditional digital computers from integrated circuits that include thousands of millions of individual transistors. We call all of these traditional digital computers as *classical*. To traditional digital computers, quantum supremacy is the watershed moment where a quantum computer completes one computation that would be intractable on a classical supercomputer and is imminent [Aaronson and Chen 2017, Coles et al 2018].

Because **IBM**'s quantum computers have become available as a cloud service to the public, the need of training a cohort of quantum programmers who have been developing classic computer programs for most of their career has arisen. With quantum assembly language and Python on **IBM**'s quantum computers [Cross et al 2017, **IBM Q** 2016], we plan to study **quantum algorithms** that consists of some beautiful ideas that everyone interested in computation should know. Our goal is to explain quantum algorithms with vectors and matrices in linear algebra that is accessible to almost everyone. In this introductory chapter, we describe quantum bits and quantum gates operating quantum bits, and we explain how to use quantum assembly language and Python on **IBM**'s quantum computers to implement them to solve any given a problem.

1.1 Quantum Bits

A *classical bit* has a state that either 0 or 1. A *quantum bit* (or a *qubit* for short) also has a state. Two possible states for a quantum bit are the states |0> and |1>. Notation like '|>' is called the *Dirac notation*. The states |0> and |1> for a quantum bit correspond to the states 0 and 1 for a classical bit and are known as '*computational basis state vectors*' of the two-dimensional Hilbert space. The computational basis state vector $|0\rangle$ of a quantum bit is represented as a (2 × 1) column vector $\begin{pmatrix}1\\0\end{pmatrix}$ and the computational basis state vector $|1\rangle$ of a quantum bit is also represented as a (2 × 1)

column vector $\binom{0}{1}$. They form an orthonormal basis for the two-dimensional Hilbert space.

The main difference between bits and quantum bits is that a quantum bit can be in a state other than |0> or |1>. A quantum bit has two 'computational basis state vectors' $|0\rangle$ and $|1\rangle$ of the two-dimensional Hilbert space. Its arbitrary state |Φ> is nothing else than a linearly weighted combination of the following computational basis state vectors, often called *superposition*:

$$|\Phi\rangle = l_0 |0\rangle + l_1 |1\rangle = l_0 \binom{1}{0} + l_1 \binom{0}{1} = \binom{l_0}{l_1}. \qquad (1.1)$$

The weighted factors l_0 and l_1 that are complex numbers are the so-called *probability amplitudes*. Thus they must satisfy $| l_0 |^2 + | l_1 |^2 = 1$. Put another way, the state of a quantum bit is a *unit* vector in the two-dimensional Hilbert space.

All the time, classical computers do examination of a bit to decide whether it is in the state 0 or 1 when they retrieve the content of the memory. However, quantum computers cannot check a quantum bit to decide its quantum state, that is, the values of l_0 and l_1. Instead, when after a quantum bit is measured from quantum computers, either the result 0 with the probability $|l_0|^2$ or the result 1 with the probability $|l_1|^2$ is obtained. This is to say that reading quantum bits is to *measure*, and the readout is in classical bits.

1.1.1 Multiple Quantum Bits

In a system of two classical bit, there are four possible states 00, 01, 10, and 11. Similarly, a system of two quantum bits has four states |00>, |01>, |10> and |11> that correspond to the classical four states 00, 01, 10, and 11 and are known as "*computational basis state vectors*" of the four-dimensional Hilbert space. The computational basis state vector |00> of two quantum bits is represented as a (4 × 1) column vector $\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$, the computational basis state vector |01> of two quantum bits is

represented as a (4 × 1) column vector $\begin{pmatrix}0\\1\\0\\0\end{pmatrix}$, the computational basis state vector |10> of two quantum bits is represented as a (4 × 1) column vector $\begin{pmatrix}0\\0\\1\\0\end{pmatrix}$ and The computational basis state vector |11> of two quantum bits is represented as a (4 × 1) column vector $\begin{pmatrix}0\\0\\0\\1\end{pmatrix}$. They form an orthonormal basis for the four-dimensional Hilbert space. The arbitrary state of two quantum bits is nothing else than a linearly weighted combination of the following computational basis state vectors, often called *superposition*:

$$|\Phi\rangle = l_0 |00\rangle + l_1 |01\rangle + l_2 |10\rangle + l_3 |11\rangle = l_0 \begin{pmatrix}1\\0\\0\\0\end{pmatrix} + l_1 \begin{pmatrix}0\\1\\0\\0\end{pmatrix} + l_2 \begin{pmatrix}0\\0\\1\\0\end{pmatrix} + l_3 \begin{pmatrix}0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}l_0\\l_1\\l_2\\l_3\end{pmatrix}. \quad (1.2)$$

The weighted factors l_0, l_1, l_2 and l_3 that are complex numbers are the so-called *probability amplitudes*. Therefore they must satisfy $| l_0 |^2 + | l_1 |^2 + | l_2 |^2 + | l_3 |^2 = \Sigma_{k \in \{0,1\}^2} | l_k |^2 = 1$, where the notation "$\{0, 1\}^2$" means "the set of strings of length two with each letter being either zero or one". Put another way, the state of two quantum bits is a *unit* vector in the four-dimensional Hilbert space.

Similarly, all the time, classical computers do examination of two bits to judge whether it is in the state 00, 01, 10 or 11 when they retrieve the content of the memory. However, quantum computers cannot do examination of two quantum bits to judge its quantum state, that is, the values of l_0, l_1, l_2 and l_3. Instead, when after two quantum bits are measured from quantum computers, the result 00 with the probability $|l_0|^2$, the result 01 with the probability $|l_1|^2$, the result 10 with the probability $|l_2|^2$ or the result 11 with the probability $|l_3|^2$ is obtained. This is to say that reading quantum bits is to *measure*, and the readout is in classical bits.

More generally, In a system of n classical bit, there are 2^n possible states 0, 1, 2, …

and (2^n-1) that are the decimal representation of *n* classical bits. Similarly, a system of *n* quantum bits has 2^n states |0>, |1>, |2>, … and $|2^n-1>$ that are the decimal representation of *n* quantum bits and correspond to the classical 2^n states 0, 1, 2, … and (2^n-1). The 2^n states are known as '*computational basis state vectors*' of the 2^n-dimensional Hilbert space. The first computational basis state vector |0> of *n* quantum bits is represented as a $(2^n \times 1)$ column vector $\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$, the second computational basis state vector |1> of *n* quantum bits is represented as a $(2^n \times 1)$ column vector $\begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}$ and the last computational basis state vector $|2^n-1>$ of *n* quantum bits is represented as a $(2^n \times 1)$ column vector $\begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$. They form an orthonormal basis for the 2^n-dimensional Hilbert space. The arbitrary state of *n* quantum bits is nothing else than a linearly weighted combination of the following computational basis state vectors, often called *superposition*:

$$|\Phi> = \sum_{k=0}^{2^n-1} l_k \, |k\rangle. \tag{1.3}$$

Each weighted factor l_k for $0 \le k \le (2^n-1)$ that is a complex number is the so-called *probability amplitudes*. Hence they must satisfy $\sum_{k=0}^{2^n-1} |l_k|^2 = \sum_{k \in \{0,1\}^n} |l_k|^2 = 1$, where the notation "$\{0, 1\}^n$" means "the set of strings of length *n* with each letter being either zero or one". Put another way, the state of *n* quantum bit is a *unit* vector in the 2^n -dimensional Hilbert space.

Similarly, all the time, classical computers do examination of *n* bits to determine whether it is in the state 0, 1, 2, … and (2^n-1) when they retrieve the content of the memory. However, quantum computers cannot do examination of *n* quantum bits to determine its quantum state, that is, the value of each l_k for $0 \le k \le (2^n-1)$. Instead, when after *n* quantum bits are measured from quantum computers, the result 0 with the probability $|l_0|^2$, the result 1 with the probability $|l_1|^2$, the result 2 with the probability $|l_2|^2$ or the last result (2^n-1) with the probability $|l_{2^n-1}|^2$ is obtained. This is to say that reading *n* quantum bits is to *measure*, and the readout is in classical bits.

1.1.2 Declaration and Measurement of Multiple Quantum Bits

QASM is the abbreviation of quantum assembly language. Open QASM is a simple text language that illustrates generic quantum circuits. In Open QASM, the syntax of the human-readable form has elements of **C** and assembly languages. For an Open QASM program, the *first* (*non-comment*) line must be "**OPENQASM M.m;**" that indicates a *major* version **M** and *minor* version **m**. Because in the cloud on **IBM**'s quantum computers it supports version 2.0, we describe version 2.0 and use version 2.0 to write a quantum program. The version *keyword* cannot occur multiple times in a file. Statements are separated by semicolons and whitespace is ignored. Comments begin with a pair of forward slashes and end with a new line. The statement "**include** "**filename**";" continues parsing **filename** as if the contents of the file were pasted at the location of the **include** statement. The path is specified relative to the current working directory.

In Open QASM (version 2.0) the only storage types are classical and quantum registers that are, respectively, one-dimensional arrays of bits and quantum bits. The statement "**qreg name**[**size**]**;**" declares an array of quantum bits (quantum register) with the given **name** and **size** that is the number of quantum bits to this quantum register. Identifiers, such as **name**, must start with a *lowercase* letter and can contain alpha-numeric characters and underscores. The label (variable) **name**[*k*] refers to the *k*th quantum bit of this register for $0 \leq k \leq$ (**size** − 1). Each quantum bit of this register is initialized to |0>. Similarly, the statement "**creg name**[**size**]**;**" declares an array of bits (classical register) with the given **name** and **size** that is the number of bits to this classical register. The label (variable) **name**[*k*] refers to the *k*th bit of this register for $0 \leq k \leq$ (**size** − 1). Each bit of this classical register is initialized to 0. The statement "**measure qubit|qreg -> bit|creg;**" measures the quantum bit(s) and records the measurement outcome(s) by overwriting the classical bit(s). Both arguments must be register-type, or both must be bit-type. If both arguments are register-type and have the same size, the statement "**measure a -> b;**" means use measure **a**[*k*] **->** **b**[*k*]; for each index *k* into register **a**.

For **IBM Q** Experience, the graphical representation of the measurement gate is as follows:

.

It takes a quantum bit in a superposition of states as input and spits either a 1 or 0.

Moreover, the output is not random. There is a probability of a 1 or 0 as output which depends on the original state of the quantum bit. It records the measurement outcome(s) by overwriting the classical bit(s).

In Listing 1.1, the program in the backend *ibmqx4* with five quantum bits in **IBM**'s

```
1 OPENQASM 2.0;
2 include "qelib1.inc";
3 qreg q[5];
4 creg c[5];
5 measure q[0] -> c[0];
6 measure q[1] -> c[1];
7 measure q[2] -> c[2];
8 measure q[3] -> c[3];
9 measure q[4] -> c[4];
```

Listing 1.1: The program of declared and measured statements of five quantum bits.

quantum computer is the first example in which we describe how to declare quantum bits and to measure quantum bits. Figure 1.1 is the corresponding quantum circuit of the program in Listing 1.1. The statement "OPENQASM 2.0;" on line one of Listing 1.1 is to indicate that the program is written with version 2.0 of Open QASM. Next, the statement "include "qelib1.inc";" on line two of Listing 1.1 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line three of Listing 1.1 is to declare that in the program there

Figure 1.1: The quantum circuit of declaring and measuring five quantum bits.

are five quantum bits. In the left top of Figure 1.1, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next,

the statement “creg c[5];” on line four of Listing 1.1 is to declare that in the program there are five classical bits. In the left bottom of Figure 1.1, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0. The statement “measure q[0] -> c[0];” on line five of Listing 1.1 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. Next, the statement “measure q[1] -> c[1];” on line six of Listing 1.1 is to measure the second quantum bit q[1] and to record the measurement outcome by overwriting the second classical bit c[1]. The statement “measure q[2] -> c[2];” on line seven of Listing 1.1 is to measure the third quantum bit q[2] and to record the measurement outcome by overwriting the third classical bit c[2]. Next, the statement “measure q[3] -> c[3];” on line eight of Listing 1.1 is to measure the fourth quantum b-

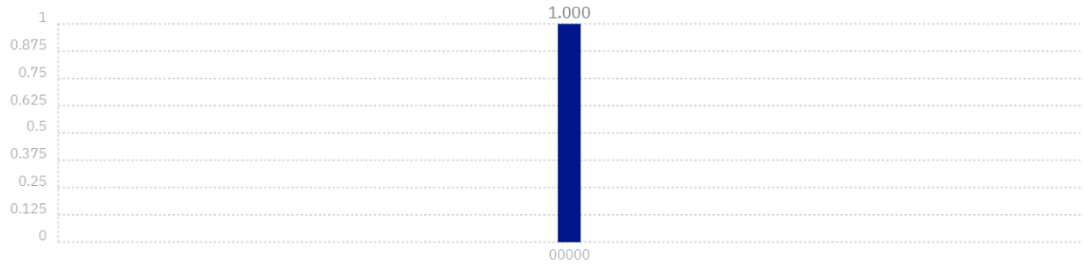

Figure 1.2: After the measurement to the program in Listing 1.1 is completed, we obtain the answer 00000 with the probability 1.000.

it q[3] and to record the measurement outcome by overwriting the fourth classical bit c[3]. The statement “measure q[4] -> c[4];” on line nine of Listing 1.1 is to measure the fifth quantum bit q[4] and to record the measurement outcome by overwriting the fifth classical bit c[4]. In the backend *ibmqx4* with five quantum bits in **IBM**’s quantum computers, we use the command “simulate” to execute the program in Listing 1.1. The result appears in Figure 1.2. From Figure 1.2, we obtain the answer 00000 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>) with the probability one.

1.2 NOT Gate of Single Quantum Bit

A classical computer is built from an electrical circuit including wires and logic gates. Similarly, a quantum computer is built from a quantum circuit consisting of wires and elementary quantum gates to complete and manipulate the quantum information. Occurring of changing a classical state to another classical state can be illustrated by means of using the language of *classical* computation. Analogous to the way, occurring of changing a quantum state to another quantum state can be introduced by means of

using the language of *quantum* computation. In this section and in the later sections, we introduce quantum gates on **IBM**'s quantum computers, propose quantum circuits of many examples describing their application and describe how to use Open QASM to write a program for implementing those examples.

For a classical computer, its circuits contain *wires* and *logic gates.* We use the wires to carry information around the circuit and use the logic gates to complete manipulation of information that is to convert it from one state to another state. For example, we consider that one logic gate of classical single bit, **NOT** gate, whose operation is to convert the state 0 to another state 1 and the state 1 to another state 0. This is to say that the classical states 0 and 1 are interchanged.

Similarly, the quantum **NOT** gate takes the state $l_0 |0> + l_1 |1>$ to the corresponding state $l_0 |1> + l_1 |0>$, where the role of |0> and |1> have been interchanged. It is assumed that we denote a matrix **X** to represent the quantum **NOT** gate as follows:

$$\mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{1.4}$$

It is also assumed that $\mathbf{X}^+$ is the conjugate-transpose matrix of **X** and is equal to $(\mathbf{X}^*)^t = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, where the * indicates complex conjugation and the *t* points out the transpose operation. Because $\mathbf{X} \times (\mathbf{X}^*)^t = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = (\mathbf{X}^*)^t \times \mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, **X** is a unitary matrix or a unitary operator. If the quantum state $l_0 |0> + l_1 |1>$ is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \tag{1.5}$$

with the top entry is the amplitude for |0> and the bottom entry is the amplitude for |1>, then the corresponding output from the quantum **NOT** gate is

$$\begin{pmatrix} l_1 \\ l_0 \end{pmatrix} = l_1 |0> + l_0 |1>. \tag{1.6}$$

Notice that the action of the quantum **NOT** gate is to that the state |0> is replaced by

the state corresponding to the *first* column of the matrix **X** and the state |1> is also replaced by the state corresponding to the *second* column of the matrix **X**. Because $\mathbf{X}^2 = \mathbf{X} \times \mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, applying **X** twice to a state does nothing to it. For **IBM Q** Experience, the graphical representation of the quantum **NOT** gate is as follows:

.

1.2.1 Programming with NOT Gate of Single Quantum Bit

In Listing 1.2, the program in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer is the second example in which we introduce how to program with **NOT** gate operating one quantum bit. Figure 1.3 is the corresponding quantum circuit of the program in Listing 1.2. The statement "OPENQASM 2.0;" on line one of Listing 1.2 is to indicate that the program is written with version 2.0 of Open QASM. Then, the statement "include "qelib1.inc";" on line two of Listing 1.2 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line three of Listing 1.2 is to declare that in the program there

```
1.  OPENQASM 2.0;
2.  include "qelib1.inc";
3.  qreg q[5];
4.  creg c[5];
5.  x q[0];
6.  x q[1];
7.  x q[2];
8.  x q[3];
9.  x q[4];
10. measure q[0] -> c[0];
11. measure q[1] -> c[1];
12. measure q[2] -> c[2];
13. measure q[3] -> c[3];
14. measure q[4] -> c[4];
```

Listing 1.2: The program to the use of five **NOT** gates operating five quantum bits.

are five quantum bits. In the left top of Figure 1.3, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement "creg c[5];" on line four of Listing 1.2 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.3, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

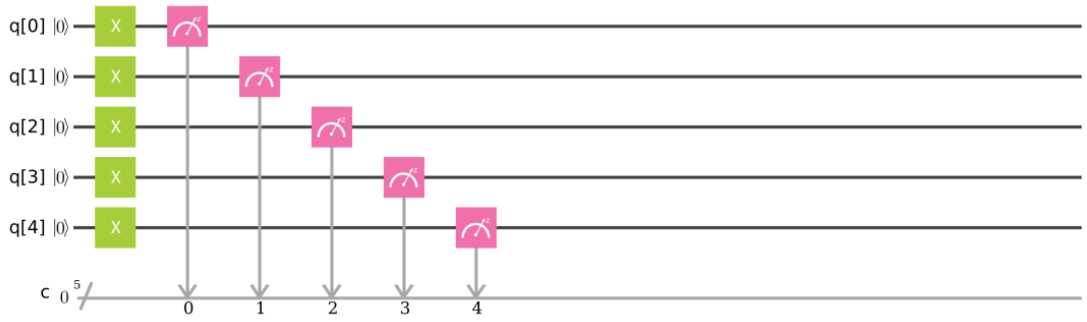

Figure 1.3: The quantum circuit to five **NOT** gates operating five quantum bits.

The statement "x q[0];" on line five of Listing 1.2 actually implements $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. This indicates that the statement "x q[0];" on line five of Listing 1.2 is to use **NOT** gate to convert q[0] from one state |0> to another state |1>, where "x" is to represent **NOT** gate. Next, the statement "x q[1];" on line six of Listing 1.2 actually completes $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. This is to say that the statement "x q[1];" on line six of Listing 1.2 is to make use of **NOT** gate to convert q[1] from one state |0> to another state |1>. Next, the statement "x q[2];" on line seven of Listing 1.2 actually implements $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. This implies that the statement "x q[2];" on line seven of Listing 1.2 is to apply **NOT** gate to convert q[2] from one state |0> to another state |1>. Next, the statement "x q[3];" on line eight of Listing 1.2 actually completes $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. This indicates that the statement "x q[3];" on line eight of Listing 1.2 is to use **NOT** gate to convert q[3] from one state |0> to another state |1>. Next, the statement "x q[4];" on line nine of Listing 1.2 actually performs $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \times$

$\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}$. This is to say that the statement “x q[4];” on line nine of Listing 1.2 is to make use of **NOT** gate to convert q[4] from one state |0> to another state |1>. After the five statements above are completed, the state |0> of each quantum bit is converted as the state |1>.

Next, the statement “measure q[0] -> c[0];” on line ten of Listing 1.2 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. The statement “measure q[1] -> c[1];” on line eleven of Listing 1.2 is to measure the second quantum bit q[1] and to record the measurement outcome by overwriting the second classical bit c[1]. Next, the statement “measure q[2] -> c[2];” on line 12 of Listing 1.2 is to measure the third quantum bit q[2] and to record the measurement outcome by overwriting the third classical bit c[2]. The statement “measure q[3] -> c[3];” on line 13 of Listing 1.1 is to measure the fourth quantum bit q[3] and to record the measurement outcome by overwriting the fourth classical bit c[3]. Next, the statement “measure q[4] -> c[4];” on line 14 of Listing 1.1 is to measure the fifth quantum bit q[4] and to record the measurement outcome by overwriting the fifth classical bit c[4]. In the backend *ibmqx4* with five quantum bits in **IBM**’s quantum computers, we use the command “simulate” to execute the program in Listing 1.2. The result appears in Figure 1.4. From Figure 1.4, we obtain the answer 11111 (c[4] = q[4] = |1>, c[3] = q[3] = |1>, c[2] = q[2] = |1>, c[1] = q[1] = |1> and c[0] = q[0] = |1>) with the probability one.

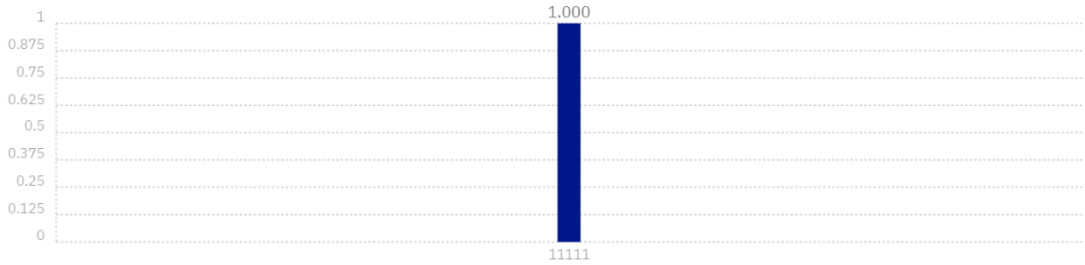

Figure 1.4: After the measurement to the program in Listing 1.2 is completed, we obtain the answer 11111 with the probability 1.000.

1.3 The Hadamard Gate of Single Quantum Bit

The Hadamard gate of single quantum bit is

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}. \quad (1.7)$$

It is supposed that H^+ is the conjugate-transpose matrix of H and is equal to $(H^*)^t = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$, where the * indicates complex conjugation and the *t* indicates the transpose operation. Since $H \times (H^*)^t = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} = (H^*)^t \times H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, H is a unitary matrix or a unitary operator. This is to say that the Hadamard gate H is one of quantum gates with single quantum bit. If the quantum state $l_0 |0> + l_1 |1>$ is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \quad (1.8)$$

with the top entry is the amplitude for |0> and the bottom entry is the amplitude for |1>, then the corresponding output from the Hadamard gate H is

$$\begin{pmatrix} \frac{l_0+l_1}{\sqrt{2}} \\ \frac{l_0-l_1}{\sqrt{2}} \end{pmatrix} = \frac{l_0+l_1}{\sqrt{2}} |0> + \frac{l_0-l_1}{\sqrt{2}} |1>. \quad (1.9)$$

If in (1.8) the value of l_0 is equal to one and the value of l_1 is equal to zero, then the Hadamard gate H turns a |0> into $\frac{1}{\sqrt{2}}$ (|0> + |1>) (the first column of H), which is 'halfway' between |0> and |1>. Similarly, if in (1.8) the value of l_0 is equal to zero and the value of l_1 is equal to one, then the Hadamard gate H turns a |1> into $\frac{1}{\sqrt{2}}$ (|0> – |1>) (the second column of H), which also is 'halfway' between |0> and |1>. Because $H^2 = H \times H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, using H twice to a state does nothing to it. For **IBM Q** Experience, the graphical representation of the Hadamard

gate *H* is as follows:

1.3.1 Programming with the Hadamard Gate of Single Quantum Bit

In Listing 1.3, the program in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer is the third example in which we describe how to program with the Hadamard gate operating one quantum bit. Figure 1.5 is the corresponding quantum circuit of the program in Listing 1.3. The statement "OPENQASM 2.0;" on line one of Listing 1.3 is to point out that the program is written with version 2.0 of Open QASM.

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
6. measure q[0] -> c[0];
```

Listing 1.3: The program to the use of the Hadamrad gate operating a quantum bit.

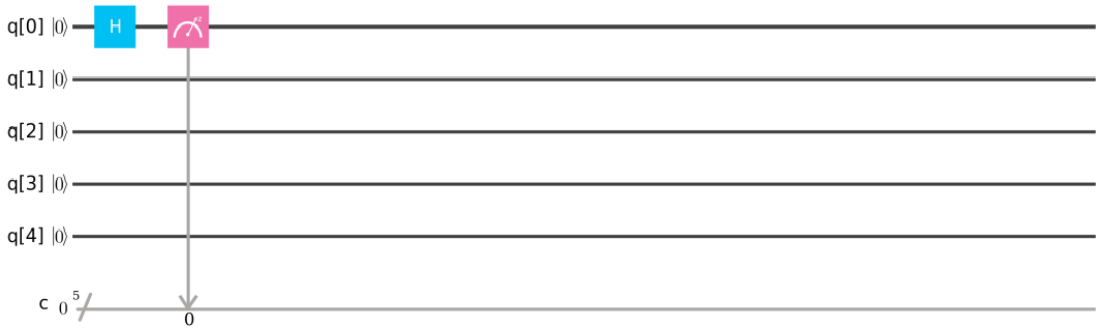

Figure 1.5: The quantum circuit to the Hadamrad gate operating a quantum bit.

Then, the statement "include "qelib1.inc";" on line two of Listing 1.3 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line three of Listing 1.3 is to declare that in the program there are five quantum bits. In the left top of Figure 1.5, five quantum bits are

subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>.

Next, the statement “creg c[5];” on line four of Listing 1.3 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.5, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0. The statement “h q[0];” on line five of Listing 1.3 actually completes $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}}$ (|0> + |1>). This is to say that the statement “h q[0];” on line five of Listing 1.3 is to use the Hadamard gate to convert q[0] from one state |0> to another state $\frac{1}{\sqrt{2}}$ (|0> + |1>) (its superposition), where “h” is to represent the Hadamard gate.

Next, the statement “measure q[0] -> c[0];” on line six of Listing 1.3 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**’s quantum computers, we apply the command “simulate” to execute the program in Listing 1.3. The result appears in Figure 1.6. From Figure 1.6, we obtain the answer 00001 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |1>) with the probability 0.520. Or we obtain the answer 00000 with the probability 0.480 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>).

Figure 1.6: After the measurement to the program in Listing 1.3 is completed, we obtain the answer 00001 with the probability 0.520 or the answer 00000 with the probability 0.480.

1.4 The Z Gate of Single Quantum Bit

The *Z* gate of single quantum bit is

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\pi} \end{pmatrix}, \qquad (1.10)$$

where $e^{\sqrt{-1}\times\pi}$ is equal to $cos(\pi) + \sqrt{-1} \times sin(\pi) = -1$. It is assumed that Z^+ is the conjugate-transpose matrix of Z and is equal to $(Z^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$, where the * indicates complex conjugation and the t is the transpose operation. Because $Z \times (Z^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = (Z^*)^t \times Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, Z is a unitary matrix or a unitary operator. This implies that the Z gate is one of quantum gates with single quantum bit. If the quantum state $l_0 |0> + l_1 |1>$ is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \qquad (1.11)$$

with the top entry is the amplitude for |0> and the bottom entry is the amplitude for |1>, then the corresponding output from the Z gate is

$$\begin{pmatrix} l_0 \\ -l_1 \end{pmatrix} = l_0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + (-l_1) \begin{pmatrix} 0 \\ 1 \end{pmatrix} = l_0 |0> + (-l_1) |1>. \qquad (1.12)$$

This indicates that the Z gate leaves |0> unchanged, and flips the sign of |1> to give $-|1>$. Since $Z^2 = Z \times Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, applying Z twice to a state does nothing to it. For **IBM Q** Experience, the graphical representation of the Z gate is as follows:

.

1.4.1 Programming with the Z Gate of Single Quantum Bit

In Listing 1.4, the program in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer is the fourth example in which we illustrate how to program with the Z gate that leaves |0> unchanged and flips the sign of |1> to give $-|1>$. Figure 1.7 is

the corresponding quantum circuit of the program in Listing 1.4. The statement "OPENQASM 2.0;" on line one of Listing 1.4 is to indicate that the program is written with version 2.0 of Open QASM. Next, the statement "include "qelib1.inc";" on line two of Listing 1.4 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
6. z q[0];
7. measure q[0] -> c[0];
```

Listing 1.4: The program to the use of the *Z* gate.

current working directory. The statement "qreg q[5];" on line three of Listing 1.4 is to declare that in the program there are five quantum bits. In the left top of Figure 1.7, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement "creg c[5];" on line four of Listing 1.4 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.7, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

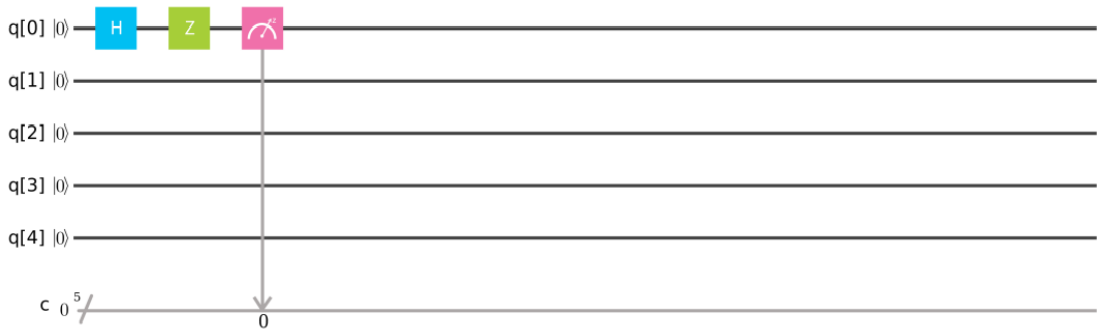

Figure 1.7: The corresponding quantum circuit of the program in Listing 1.4.

The statement "h q[0];" on line five of Listing 1.4 actually implements $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This

indicates that the statement “h q[0];” on line five of Listing 1.4 is to use the Hadamard gate to convert q[0] from one state |0> to another state $\frac{1}{\sqrt{2}}$ (|0> + |1>) (its superposition), where “h” is to represent the Hadamard gate. Next, the statement “z q[0];” on line six of Listing 1.4 actually completes $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ -1 \end{pmatrix}) = (\frac{1}{\sqrt{2}}$ |0>) + ($-\frac{1}{\sqrt{2}}$) (|1>) = $\frac{1}{\sqrt{2}}$ (|0> – |1>). This is to say that the statement “z q[0];” on line six of Listing 1.4 is to apply the *Z* gate to convert q[0] from one state $\frac{1}{\sqrt{2}}$ (|0> + |1>) to another state $\frac{1}{\sqrt{2}}$ (|0> – |1>).

Next, the statement “measure q[0] -> c[0];” on line seven of Listing 1.4 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**’s quantum computers, we use the command “simulate” to execute the program in Listing 1.4. The result appears in Figure 1.8. From Figure 1.8, we obtain the answer 00001 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |1>) with the probability 0.490. Or we obtain the answer 00000 with the probability 0.510 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>).

Figure 1.8: After the measurement to the program in Listing 1.4 is completed, we obtain the answer 00001 with the probability 0.490 or the answer 00000 with the probability 0.510.

1.5 The Y Gate of Single Quantum Bit

The *Y* gate of single quantum bit is

$$Y = \begin{pmatrix} 0 & -\sqrt{-1} \\ \sqrt{-1} & 0 \end{pmatrix} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} = \begin{pmatrix} 0 & e^{-\sqrt{-1}\times\frac{\pi}{2}} \\ e^{\sqrt{-1}\times\frac{\pi}{2}} & 0 \end{pmatrix}, \quad (1.13)$$

where $i = \sqrt{-1}$ is known as the imaginary unit and $e^{\sqrt{-1}\times\frac{\pi}{2}} = cos(\frac{\pi}{2}) + \sqrt{-1} \times sin(\frac{\pi}{2}) = \sqrt{-1}$ and $e^{-\sqrt{-1}\times\frac{\pi}{2}} = cos(-\frac{\pi}{2}) + \sqrt{-1} \times sin(-\frac{\pi}{2}) = -\sqrt{-1}$. It is supposed that Y^+ is the conjugate-transpose matrix of Y and is equal to $(Y^*)^t = \begin{pmatrix} 0 & -\sqrt{-1} \\ \sqrt{-1} & 0 \end{pmatrix} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$, where the * indicates complex conjugation and the t is the transpose operation. Since $Y \times (Y^*)^t = \begin{pmatrix} 0 & -\sqrt{-1} \\ \sqrt{-1} & 0 \end{pmatrix} \times \begin{pmatrix} 0 & -\sqrt{-1} \\ \sqrt{-1} & 0 \end{pmatrix} = (Y^*)^t \times Y = \begin{pmatrix} 0 & -\sqrt{-1} \\ \sqrt{-1} & 0 \end{pmatrix} \times \begin{pmatrix} 0 & -\sqrt{-1} \\ \sqrt{-1} & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, Y is a unitary matrix or a unitary operator. This is to say that the Y gate is one of quantum gates with single quantum bit. If the quantum state $l_0 |0> + l_1 |1>$ is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \tag{1.14}$$

with the top entry is the amplitude for |0> and the bottom entry is the amplitude for |1>, then the corresponding output from the Y gate is

$$\begin{pmatrix} -\sqrt{-1}l_1 \\ \sqrt{-1}l_0 \end{pmatrix} = (-\sqrt{-1}l_1)\begin{pmatrix} 1 \\ 0 \end{pmatrix} + (\sqrt{-1}l_0)\begin{pmatrix} 0 \\ 1 \end{pmatrix} = (-\sqrt{-1}l_1)\,|0> + (\sqrt{-1}l_0)\,|1>. \tag{1.15}$$

This indicates that the Y gate converts single quantum bit from one state $l_0 |0> + l_1 |1>$ to another state $(-\sqrt{-1}l_1) |0> + (\sqrt{-1}l_0) |1>$. Since $Y^2 = Y \times Y = \begin{pmatrix} 0 & -\sqrt{-1} \\ \sqrt{-1} & 0 \end{pmatrix} \times \begin{pmatrix} 0 & -\sqrt{-1} \\ \sqrt{-1} & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, using Y twice to a state does nothing to it. For **IBM Q** Experience, the graphical representation of the Y gate is as follows:

.

1.5.1 Programming with the Y Gate of Single Quantum Bit

In Listing 1.5, the program in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer is the fifth example in which we introduce how to program with the *Y* gate that converts single quantum bit from one state $\frac{1}{\sqrt{2}}$ |0> + $\frac{1}{\sqrt{2}}$ |1> to another state $(-\sqrt{-1}\frac{1}{\sqrt{2}})$ |0> + $(\sqrt{-1}\frac{1}{\sqrt{2}})$ |1>. Figure 1.9 is the corresponding quantum circuit of the program in Listing 1.5. The statement "OPENQASM 2.0;" on line one of Listing 1.5 is to point out that the program is written with version 2.0 of Open QASM. Next, the statement "include "qelib1.inc";" on line two of Listing 1.5 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line three of Listing 1.5 is to declare that in the program there

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
6. y q[0];
7. measure q[0] -> c[0];
```

Listing 1.5: The program to the use of the *Y* gate.

are five quantum bits. In the left top of Figure 1.9, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement "creg c[5];" on line four of Listing 1.5 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.9, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

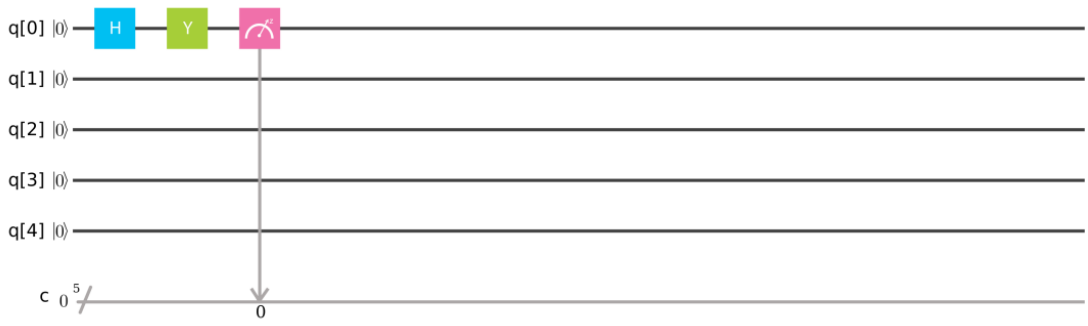

Figure 1.9: The corresponding quantum circuit of the program in Listing 1.5.

The statement “h q[0];” on line five of Listing 1.5 actually completes $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This is to say that the statement “h q[0];” on line five of Listing 1.5 is to apply the Hadamard gate to convert q[0] from one state |0> to another state $\frac{1}{\sqrt{2}} (|0> + |1>)$ (its superposition), where “h” is to represent the Hadamard gate. Next, the statement “y q[0];” on line six of Listing 1.5 actually implements $\begin{pmatrix} 0 & -\sqrt{-1} \\ \sqrt{-1} & 0 \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} -\sqrt{-1} \times \frac{1}{\sqrt{2}} \\ \sqrt{-1} \times \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} -\sqrt{-1} \\ \sqrt{-1} \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} -\sqrt{-1} \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ \sqrt{-1} \end{pmatrix}) = (-\sqrt{-1} \times \frac{1}{\sqrt{2}}) \begin{pmatrix} 1 \\ 0 \end{pmatrix} + (\sqrt{-1} \times \frac{1}{\sqrt{2}}) \begin{pmatrix} 0 \\ 1 \end{pmatrix} = (-\sqrt{-1} \times \frac{1}{\sqrt{2}}) \ |0> + (\sqrt{-1} \times \frac{1}{\sqrt{2}}) \ |1> = \frac{1}{\sqrt{2}} (-\sqrt{-1} \, |0> + \sqrt{-1} \ |1>)$. This indicates that the statement “y q[0];” on line six of Listing 1.5 is to apply the *Y* gate to convert q[0] from one state $\frac{1}{\sqrt{2}} (|0> + |1>)$ to another state $\frac{1}{\sqrt{2}} (-\sqrt{-1} \, |0> + \sqrt{-1} \ |1>)$.

Next, the statement “measure q[0] -> c[0];” on line seven of Listing 1.5 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**’s quantum computers, we apply the command “simulate” to execute the program in Listing 1.5. The result appears in Figure 1.10. From Figure 1.10, we obtain the answ-

Figure 1.10: After the measurement to the program in Listing 1.5 is completed, we obtain the answer 00001 with the probability 0.520 or the answer 00000 with the probability 0.480.

er 00001 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |1>) with the probability 0.520. Or we obtain the answer 00000 with the probability 0.480 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>).

1.6 The S Gate of Single Quantum Bit

The *S* gate of single quantum bit that is the square root of the *Z* gate is

$$S = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{2}} \end{pmatrix} \qquad (1.16)$$

where $i = \sqrt{-1}$ is known as the imaginary unit and $e^{\sqrt{-1}\times\frac{\pi}{2}} = cos(\frac{\pi}{2}) + \sqrt{-1} \times sin(\frac{\pi}{2}) = \sqrt{-1}$. It is assumed that S^+ is the conjugate-transpose matrix of *S* and is equal to $(S^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & -\sqrt{-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -i \end{pmatrix}$, where the * points out complex conjugation and the *t* indicates the transpose operation. Because $S \times (S^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{-1} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & -\sqrt{-1} \end{pmatrix} = (S^*)^t \times S = \begin{pmatrix} 1 & 0 \\ 0 & -\sqrt{-1} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, *S* is a unitary matrix or a unitary operator. This implies that the *S* gate is one of quantum gates with single quantum bit. If the quantum state l_0 |0> + l_1 |1> is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \qquad (1.17)$$

with the top entry is the amplitude for |0> and the bottom entry is the amplitude for |1>, then the corresponding output from the *S* gate is

$$\begin{pmatrix} l_0 \\ \sqrt{-1}l_1 \end{pmatrix} = (l_0)\begin{pmatrix} 1 \\ 0 \end{pmatrix} + (\sqrt{-1}l_1)\begin{pmatrix} 0 \\ 1 \end{pmatrix} = (l_0)|0> + (\sqrt{-1}l_1)\,|1> = (l_0)\,|0> + (e^{\sqrt{-1}\times\frac{\pi}{2}} \times l_1)|1>. \qquad (1.18)$$

This indicates that the *S* gate converts single quantum bit from one state l_0 |0> + l_1 |1> to another state (l_0) |0> + $(\sqrt{-1}l_1)$ |1> = (l_0) |0> + $(e^{\sqrt{-1}\times\frac{\pi}{2}} \times l_1)$ |1>. This is also to say that the *S* gate leaves |0> unchanged and modifies the phase of |1> to give $(\sqrt{-1})$ |1> $(e^{\sqrt{-1}\times\frac{\pi}{2}}$ |1>). The probability of measuring a |0> or |1> is unchanged after applying

the *S* gate, however it modifies the phase of the quantum state. Because $S^2 = S \times S = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{-1} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$, applying *S* twice to a state is equivalent to do the *Z* gate to it. For **IBM Q** Experience, the graphical representation of the *S* gate is as follows:

.

1.6.1 Programming with the S Gate of Single Quantum Bit

In Listing 1.6, the program in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer is the sixth example in which we describe how to program with the *S* gate that converts single quantum bit from one state $\frac{1}{\sqrt{2}}$ (|0> + |1>) to another state $(\frac{1}{\sqrt{2}})$ (|0> + $(\sqrt{-1})$ |1>). Figure 1.11 is the corresponding quantum circuit of the program in Listing 1.6. The statement "OPENQASM 2.0;" on line one of Listing 1.6 is to indicate that the program is written with version 2.0 of Open QASM. Next, the statement "include "qelib1.inc";" on line two of Listing 1.6 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line three of Listing 1.6 is to declare that in the program there

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
6. s q[0];
7. measure q[0] -> c[0];
```

Listing 1.6: The program to the use of the *S* gate.

are five quantum bits. In the left top of Figure 1.11, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement "creg c[5];" on line four of Listing 1.6 is to declare that there are five

classical bits in the program. In the left bottom of Figure 1.11, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

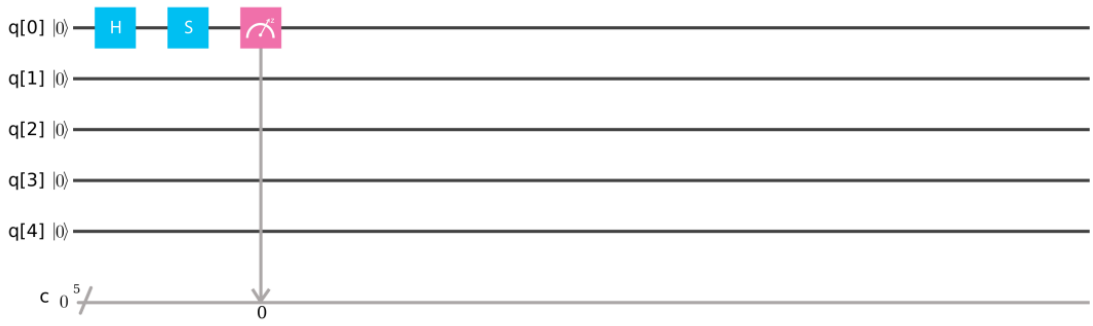

Figure 1.11: The corresponding quantum circuit of the program in Listing 1.6.

The statement "h q[0];" on line five of Listing 1.6 actually implements $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}}$ (|0> + |1>). This indicates that the statement "h q[0];" on line five of Listing 1.6 is to use the Hadamard gate to convert q[0] from one state |0> to another state $\frac{1}{\sqrt{2}}$ (|0> + |1>) (its superposition), where "h" is to represent the Hadamard gate. Next, the statement "s q[0];" on line six of Listing 1.6 actually completes $\begin{pmatrix} 1 & 0 \\ 0 & \sqrt{-1} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \sqrt{-1} \times \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ \sqrt{-1} \end{pmatrix}$ $= \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ \sqrt{-1} \end{pmatrix}) = (\frac{1}{\sqrt{2}}) (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + (\sqrt{-1}) \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}}$ (|0 > + $\sqrt{-1}$ |1 >). This is to say that the statement "s q[0];" on line six of Listing 1.6 is to use the *S* gate to convert q[0] from one state $\frac{1}{\sqrt{2}}$ (|0> + |1>) to another state $\frac{1}{\sqrt{2}}$ (|0 > + $\sqrt{-1}$ |1 >).

Next, the statement "measure q[0] -> c[0];" on line seven of Listing 1.6 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers, we use the command "simulate" to execute the program in Listing 1.6. The result appears in Figure 1.12. From Figure 1.12, we obtain the answer 00001 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |1>) with the probability 0.550. Or we obtain the answer 00000 with the

probability 0.450 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>).

Figure 1.12: After the measurement to the program in Listing 1.6 is completed, we obtain the answer 00001 with the probability 0.550 or the answer 00000 with the probability 0.450.

1.7 The S^+ Gate of Single Quantum Bit

The S^+ gate of single quantum bit that is the conjugate-transpose matrix of the S gate is

$$S^+ = \begin{pmatrix} 1 & 0 \\ 0 & -\sqrt{-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -i \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -e^{\sqrt{-1}\times\frac{\pi}{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\frac{\pi}{2}} \end{pmatrix}, \quad (1.19)$$

where $i = \sqrt{-1}$ is known as the imaginary unit and $e^{-\sqrt{-1}\times\frac{\pi}{2}} = cos(-\frac{\pi}{2}) + \sqrt{-1} \times sin(-\frac{\pi}{2}) = -\sqrt{-1}$. It is supposed that $(S^+)^+$ is the conjugate-transpose matrix of S^+ and is equal to $((S^+)^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{2}} \end{pmatrix}$, where the * indicates complex conjugation and the t is the transpose operation. Since $S^+ \times ((S^+)^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & -\sqrt{-1} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{-1} \end{pmatrix} = ((S^+)^*)^t \times S^+ = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{-1} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & -\sqrt{-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, S^+ is a unitary matrix or a unitary operator. This is to say that the S^+ gate is one of quantum gates with single quantum bit. If the quantum state l_0 |0> + l_1 |1> is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \quad (1.20)$$

with the top entry is the amplitude for |0> and the bottom entry is the amplitude for |1>,

then the corresponding output from the S^+ gate is

$$\begin{pmatrix} l_0 \\ -\sqrt{-1}l_1 \end{pmatrix} = (l_0)\begin{pmatrix}1\\0\end{pmatrix} + (-\sqrt{-1}l_1)\begin{pmatrix}0\\1\end{pmatrix} = (l_0)\,|0\rangle + (-\sqrt{-1}l_1)\,|1\rangle = (l_0)\,|0\rangle + (e^{-\sqrt{-1}\times\frac{\pi}{2}} \times l_1)\,|1\rangle. \quad (1.21)$$

This implies that the S^+ gate converts single quantum bit from one state $l_0\,|0\rangle + l_1\,|1\rangle$ to another state $(l_0)\,|0\rangle + (-\sqrt{-1}l_1)\,|1\rangle = (l_0)\,|0\rangle + (e^{-\sqrt{-1}\times\frac{\pi}{2}} \times l_1)\,|1\rangle$. This also indicates that the S^+ gate leaves $|0\rangle$ unchanged and modifies the phase of $|1\rangle$ to give $(-\sqrt{-1})\,|1\rangle$ $(e^{-\sqrt{-1}\times\frac{\pi}{2}}\,|1\rangle)$. The probability of measuring a $|0\rangle$ or $|1\rangle$ is unchanged after applying the S^+ gate, however it modifies the phase of the quantum state. Because $(S^+)^2 = S^+ \times S^+ = \begin{pmatrix}1 & 0\\ 0 & -\sqrt{-1}\end{pmatrix} \times \begin{pmatrix}1 & 0\\ 0 & -\sqrt{-1}\end{pmatrix} = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}$, applying S^+ twice to a state is equivalent to do the Z gate to it. For **IBM Q** Experience, the graphical representation of the S^+ gate is as follows:

.

1.7.1 Programming with the S⁺ Gate of Single Quantum Bit

In Listing 1.7, the program in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer is the seventh example in which we illustrate how to program with the S^+ gate that converts single quantum bit from one state $\frac{1}{\sqrt{2}}\,(|0\rangle + |1\rangle)$ to another state $(\frac{1}{\sqrt{2}})\,(|0\rangle + (-\sqrt{-1})\,|1\rangle)$. Figure 1.13 is the corresponding quantum circuit of the program in Listing 1.7. The statement "OPENQASM 2.0;" on line one of Listing 1.7 is to point out that the program is written with version 2.0 of Open QASM. Next, the statement "include "qelib1.inc";" on line two of Listing 1.7 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line three of Listing 1.7 is to declare that in the program there

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
6. sdg q[0];
7. measure q[0] -> c[0];
```

Listing 1.7: The program to the use of the S^+ gate.

are five quantum bits. In the left top of Figure 1.13, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement "creg c[5];" on line four of Listing 1.7 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.13, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

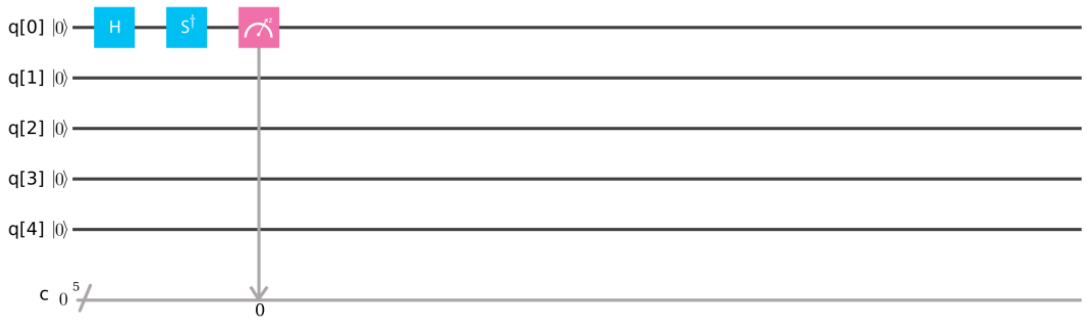

Figure 1.13: The corresponding quantum circuit of the program in Listing 1.7.

The statement "h q[0];" on line five of Listing 1.7 actually completes $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This is to say that the statement "h q[0];" on line five of Listing 1.7 is to apply the Hadamard gate to convert q[0] from one state |0> to another state $\frac{1}{\sqrt{2}}$ (|0> + |1>) (its superposition), where "h" is to represent the Hadamard gate. Next, the statement "sdg q[0];" on line six of

Listing 1.7 actually performs $\begin{pmatrix} 1 & 0 \\ 0 & -\sqrt{-1} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\sqrt{-1} \times \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -\sqrt{-1} \end{pmatrix} = \frac{1}{\sqrt{2}} \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ -\sqrt{-1} \end{pmatrix} \right) = \left(\frac{1}{\sqrt{2}} \right) \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} + (-\sqrt{-1}) \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \frac{1}{\sqrt{2}} (|0> + (-\sqrt{-1}) |1>)$. This indicates that the statement "sdg q[0];" on line six of Listing 1.7 is to apply the S^+ gate to convert q[0] from one state $\frac{1}{\sqrt{2}}$ (|0> + |1>) to another state $\frac{1}{\sqrt{2}}$ $(|0> + (-\sqrt{-1}) |1>)$.

Next, the statement "measure q[0] -> c[0];" on line seven of Listing 1.7 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers, we apply the command "simulate" to execute the program in Listing 1.7. The result appears in Figure 1.14. From Figure 1.14, we obtain the answer 00001 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |1>) with the probability 0.500. Or we obtain the answer 00000 with the probability 0.500 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>).

Figure 1.14: After the measurement to the program in Listing 1.7 is completed, we obtain the answer 00001 with the probability 0.500 or the answer 00000 with the probability 0.500.

1.8 The T Gate of Single Quantum Bit

The *T* gate of single quantum bit that is the square root of the *S* gate is

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1} \times \frac{\pi}{4}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1+\sqrt{-1}}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1+i}{\sqrt{2}} \end{pmatrix}, \quad (1.22)$$

where $i = \sqrt{-1}$ is known as the imaginary unit and $e^{\sqrt{-1} \times \frac{\pi}{4}} = cos(\frac{\pi}{4}) + \sqrt{-1} \times sin(\frac{\pi}{4})$

$= \frac{1+\sqrt{-1}}{\sqrt{2}} = \frac{1+i}{\sqrt{2}}$. It is assumed that T^+ is the conjugate-transpose matrix of T and is equal to $(T^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1-\sqrt{-1}}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1-i}{\sqrt{2}} \end{pmatrix}$, where the * indicates complex conjugation and the t is the transpose operation. Since $T \times (T^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} = (T^*)^t \times T = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, T is a unitary matrix or a unitary operator. This indicates that the T gate is one of quantum gates with single quantum bit. If the quantum state $l_0 |0> + l_1 |1>$ is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \tag{1.23}$$

with the top entry is the amplitude for |0> and the bottom entry is the amplitude for |1>, then the corresponding output from the T gate is

$$\begin{pmatrix} l_0 \\ e^{\sqrt{-1}\times\frac{\pi}{4}} \times l_1 \end{pmatrix} = (l_0) \begin{pmatrix} 1 \\ 0 \end{pmatrix} + (e^{\sqrt{-1}\times\frac{\pi}{4}} \times l_1) \begin{pmatrix} 0 \\ 1 \end{pmatrix} = (l_0) |0> + (e^{\sqrt{-1}\times\frac{\pi}{4}} \times l_1) |1>. \tag{1.24}$$

This is to say that the T gate converts single quantum bit from one state $l_0 |0> + l_1 |1>$ to another state $(l_0) |0> + (e^{\sqrt{-1}\times\frac{\pi}{4}} \times l_1) |1>$. This also is to say that the T gate leaves |0> unchanged and modifies the phase of |1> to give $(e^{\sqrt{-1}\times\frac{\pi}{4}}) |1>$. The probability of measuring a |0> or |1> is unchanged after using the T gate, however it modifies the phase of the quantum state. Because $(T)^2 = T \times T = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{-1} \end{pmatrix}$, using T twice to a state is equivalent to do the S gate to it. For **IBM Q** Experience, the graphical representation of the T gate is as follows:

.

1.8.1 Programming with the T Gate of Single Quantum Bit

In Listing 1.8, the program in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer is the eighth example in which we describe how to program with the *T* gate that converts single quantum bit from one state $\frac{1}{\sqrt{2}}$ (|0> + |1>) to another state $(\frac{1}{\sqrt{2}})$ (|0> + $(e^{\sqrt{-1}\times\frac{\pi}{4}})$ |1>). Figure 1.15 is the corresponding quantum circuit of the program in Listing 1.8. The statement "OPENQASM 2.0;" on line one of Listing 1.8 is to indicate that the program is written with version 2.0 of Open QASM. Next, the statement "include "qelib1.inc";" on line two of Listing 1.8 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line three of Listing 1.8 is to declare that in the program there

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
6. t q[0];
7. measure q[0] -> c[0];
```

Listing 1.8: The program to the use of the *T* gate.

are five quantum bits. In the left top of Figure 1.15, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement "creg c[5];" on line four of Listing 1.8 is to declare that there are five cla-

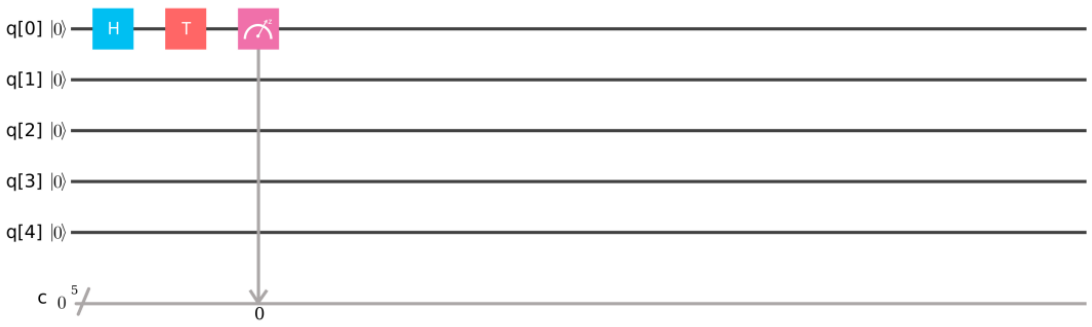

Figure 1.15: The corresponding quantum circuit of the program in Listing 1.8.

ssical bits in the program. In the left bottom of Figure 1.15, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

The statement "h q[0];" on line five of Listing 1.8 actually implements $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This implies that the statement "h q[0];" on line five of Listing 1.8 is to use the Hadamard gate to convert q[0] from one state |0> to another state $\frac{1}{\sqrt{2}}$ (|0> + |1>) (its superposition), where "h" is to represent the Hadamard gate. Next, the statement "t q[0];" on line six of Listing 1.8 actually completes $\begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ e^{\sqrt{-1}\times\frac{\pi}{4}} \times \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ e^{\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ e^{\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix}) = (\frac{1}{\sqrt{2}}) (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + (e^{\sqrt{-1}\times\frac{\pi}{4}}) \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|0> + (e^{\sqrt{-1}\times\frac{\pi}{4}}) |1>)$. This is to say that the statement "t q[0];" on line six of Listing 1.8 is to use the *T* gate to convert q[0] from one state $\frac{1}{\sqrt{2}}$ (|0> + |1>) to another state $\frac{1}{\sqrt{2}} (|0> + (e^{\sqrt{-1}\times\frac{\pi}{4}}) |1>)$.

Next, the statement "measure q[0] -> c[0];" on line seven of Listing 1.8 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers, we use the command "simulate" to execute the program in Listing 1.8. The result appears in Figure 1.16. From Figure 1.16, we obtain the answer

Figure 1.16: After the measurement to the program in Listing 1.8 is completed, we

obtain the answer 00001 with the probability 0.480 or the answer 00000 with the probability 0.520.

00001 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |1>) with the probability 0.480. Or we obtain the answer 00000 with the probability 0.520 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>).

1.9 The T^+ Gate of Single Quantum Bit

The T^+ gate of single quantum bit that is the conjugate-transpose matrix of the T gate is

$$T^+ = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1-\sqrt{-1}}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1-i}{\sqrt{2}} \end{pmatrix}, \quad (1.25)$$

where $i = \sqrt{-1}$ is known as the imaginary unit and $e^{-\sqrt{-1}\times\frac{\pi}{4}} = cos(-\frac{\pi}{4}) + \sqrt{-1} \times sin(-\frac{\pi}{4}) = \frac{1-\sqrt{-1}}{\sqrt{2}} = \frac{1-i}{\sqrt{2}}$. It is supposed that $(T^+)^+$ is the conjugate-transpose matrix of T^+ and is equal to $((T^+)^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1+\sqrt{-1}}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1+i}{\sqrt{2}} \end{pmatrix}$, where the * is the complex conjugation and the t is the transpose operation. Since $T^+ \times ((T^+)^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} = ((T^+)^*)^t \times T^+ = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, T^+ is a unitary matrix or a unitary operator. This is to say that the T^+ gate is one of quantum gates with single quantum bit. If the quantum state l_0 |0> + l_1 |1> is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \quad (1.26)$$

with the top entry is the amplitude for |0> and the bottom entry is the amplitude for |1>, then the corresponding output from the T^+ gate is

$$\begin{pmatrix} l_0 \\ e^{-\sqrt{-1}\times\frac{\pi}{4}} \times l_1 \end{pmatrix} = (l_0)\begin{pmatrix} 1 \\ 0 \end{pmatrix} + (e^{-\sqrt{-1}\times\frac{\pi}{4}} \times l_1)\begin{pmatrix} 0 \\ 1 \end{pmatrix} = (l_0)\,|0> + (e^{-\sqrt{-1}\times\frac{\pi}{4}} \times l_1)\,|1>. \quad (1.27)$$

This indicates that the T^+ gate converts single quantum bit from one state l_0 |0> + l_1 |1> to another state (l_0) |0> + $(e^{-\sqrt{-1}\times\frac{\pi}{4}} \times l_1)$ |1>. This also is to say that the T^+ gate leaves |0> unchanged and modifies the phase of |1> to give $(e^{-\sqrt{-1}\times\frac{\pi}{4}})$ |1>. The probability of measuring a |0> or |1> is unchanged after applying the T^+ gate, however it modifies the phase of the quantum state. Because $(T^+)^2 = T^+ \times T^+ = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\frac{\pi}{4}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\frac{\pi}{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -\sqrt{-1} \end{pmatrix}$, applying T^+ twice to a state is equivalent to do the S^+ gate to it. For **IBM Q** Experience, the graphical representation of the T^+ gate is as follows:

.

1.9.1 Programming with the T+ Gate of Single Quantum Bit

In Listing 1.9, the program in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer is the ninth example in which we introduce how to program with the T^+ gate that converts single quantum bit from one state $\frac{1}{\sqrt{2}}$ (|0> + |1>) to another state $(\frac{1}{\sqrt{2}})$ (|0> + $(e^{-\sqrt{-1}\times\frac{\pi}{4}})$ |1>). Figure 1.17 is the corresponding quantum circuit of the program in Listing 1.9. The statement "OPENQASM 2.0;" on line one of Listing 1.9 is to point out that the program is written with version 2.0 of Open QASM. Next, the statement "include "qelib1.inc";" on line two of Listing 1.9 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line three of Listing 1.9 is to declare that in the program there

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
```

```
6. tdg q[0];
7. measure q[0] -> c[0];
```

Listing 1.9: The program to the use of the T^+ gate.

are five quantum bits. In the left top of Figure 1.17, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement "creg c[5];" on line four of Listing 1.9 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.17, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

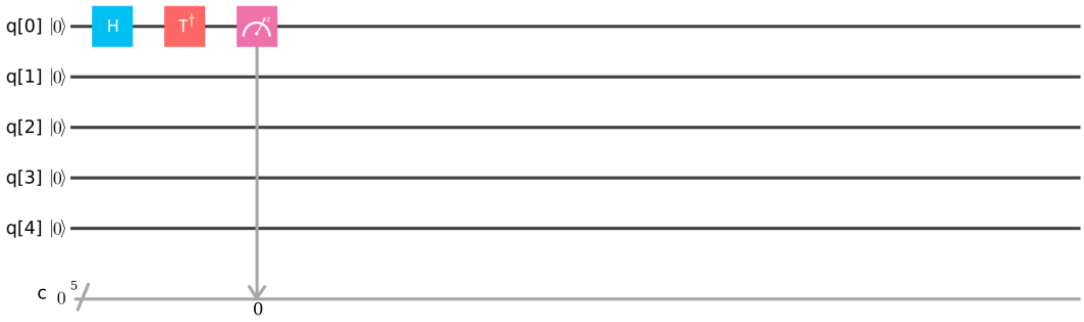

Figure 1.17: The corresponding quantum circuit of the program in Listing 1.9.

The statement "h q[0];" on line five of Listing 1.9 actually completes $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This is to say that the statement "h q[0];" on line five of Listing 1.9 is to use the Hadamard gate to convert q[0] from one state |0> to another state $\frac{1}{\sqrt{2}} (|0> + |1>)$ (its superposition), where "h" is to represent the Hadamard gate. Next, the statement "tdg q[0];" on line six of Listing 1.9 actually implements $\begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1} \times \frac{\pi}{4}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ e^{-\sqrt{-1} \times \frac{\pi}{4}} \times \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ e^{-\sqrt{-1} \times \frac{\pi}{4}} \end{pmatrix} = \frac{1}{\sqrt{2}} \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ e^{-\sqrt{-1} \times \frac{\pi}{4}} \end{pmatrix} \right) = \left(\frac{1}{\sqrt{2}} \right) \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \left(e^{-\sqrt{-1} \times \frac{\pi}{4}} \right) \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \frac{1}{\sqrt{2}} (|0> + \left(e^{-\sqrt{-1} \times \frac{\pi}{4}} \right) |1>)$. This indicates that the statement "tdg q[0];" on line six of Listing 1.9

is to use the T^+ gate to convert q[0] from one state $\frac{1}{\sqrt{2}}$ (|0> + |1>) to another state $\frac{1}{\sqrt{2}}$ ($|0> + (e^{-\sqrt{-1}\times\frac{\pi}{4}})\ |1>$).

Next, the statement "measure q[0] -> c[0];" on line seven of Listing 1.9 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers, we apply the command "simulate" to execute the program in Listing 1.9. The result appears in Figure 1.18. From Figure 1.18, we obtain the answer 00001 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |1>) with the probability 0.550. Or we obtain the answer 00000 with the probability 0.450 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>).

Figure 1.18: After the measurement to the program in Listing 1.9 is completed, we obtain the answer 00001 with the probability 0.550 or the answer 00000 with the probability 0.450.

1.10 The Identity Gate of Single Quantum Bit

The *identity* gate *id* of single quantum bit is

$$id = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{1.28}$$

It is assumed that id^+ is the conjugate-transpose matrix of *id* and is equal to $((id)^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, where the * is the complex conjugation and the *t* is the transpose operation. Because $id \times ((id)^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = ((id)^*)^t \times id = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, *id* is a unitary matrix or a unitary operator. This indicates that the identity gate

id is one of quantum gates with single quantum bit. If the quantum state l_0 |0> + l_1 |1> is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \tag{1.29}$$

with the top entry is the amplitude for |0> and the bottom entry is the amplitude for |1>, then the corresponding output from the identity gate *id* is

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix} = (l_0) \begin{pmatrix} 1 \\ 0 \end{pmatrix} + (l_1) \begin{pmatrix} 0 \\ 1 \end{pmatrix} = (l_0)\,|0> + (l_1)\,|1>. \tag{1.30}$$

This is to say that the identity gate *id* converts single quantum bit from one state l_0 |0> + l_1 |1> to another state (l_0) |0> + (l_1) |1>. This also is to say that the identity gate *id* does not change |0> and |1> and only performs an idle operation on single quantum bit for a time equal to one unit of time. The probability of measuring a |0> or |1> is unchanged after using the identity gate *id*. Since $(id)^2 = id \times id = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, applying the identity gate *id* twice to a state is equivalent to do nothing to it. For **IBM Q** Experience, the graphical representation of the identity gate *id* is as follows:

.

1.10.1 Programming with the Identity Gate of Single Quantum Bit

In Listing 1.10, the program in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer is the tenth example in which we introduce how to program with the identity gate *id* that converts single quantum bit q[0] from one state $\frac{1}{\sqrt{2}}$ (|0> + |1>) to another state $(\frac{1}{\sqrt{2}})$ (|0> + |1>). Actually, the identity gate *id* only completes an idle operation on q[0] for a time equal to one unit of time. Figure 1.19 is the corresponding quantum circuit of the program in Listing 1.10. The statement "OPENQASM 2.0;" on line one of Listing 1.10 is to indicate that the program is written with version 2.0 of Open QASM. Next, the statement "include "qelib1.inc";" on line two of Listing 1.10 is

to continue parsing the file “qelib1.inc” as if the contents of the file were pasted at the location of the include statement, where the file “qelib1.inc” is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement “qreg q[5];” on line three of Listing 1.10 is to declare that in the program there are five quantum bits. In the left top of Figure 1.19, five quantum bits

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
6. id q[0];
7. measure q[0] -> c[0];
```

Listing 1.10: The program to the use of the identity gate.

are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement “creg c[5];” on line four of Listing 1.10 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.19, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

Figure 1.19: The corresponding quantum circuit of the program in Listing 1.10.

The statement “h q[0];” on line five of Listing 1.10 actually performs $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This implies that the statement “h q[0];” on line five of Listing 1.10 is to apply the Hadamard gate to convert

q[0] from one state |0> to another state $\frac{1}{\sqrt{2}}$ (|0> + |1>) (its superposition), where “h” is to represent the Hadamard gate. Next, the statement “id q[0];” on line six of Listing 1.10 actually completes $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This is to say that the statement “id q[0];” on line six of Listing 1.10 is to use the identity gate *id* to convert q[0] from one state $\frac{1}{\sqrt{2}}$ (|0> + |1>) to another state $\frac{1}{\sqrt{2}}$ (|0> + |1>). This also is to say that the identity gate *id* only completes an idle operation on q[0] for a time equal to one unit of time.

Next, the statement “measure q[0] -> c[0];” on line seven of Listing 1.10 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**’s quantum computers, we apply the command “simulate” to execute the program in Listing 1.10. The result appears in Figure 1.20. From Figure 1.20, we obtain the answer 00001 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |1>) with the probability 0.460. Or we obtain the answer 00000 with the probability 0.540 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>).

Figure 1.20: After the measurement to the program in Listing 1.10 is completed, we obtain the answer 00001 with the probability 0.460 or the answer 00000 with the probability 0.540.

1.11 The Controlled-NOT Gate of Two Quantum Bits

The ***controlled-NOT*** or ***CNOT*** gate of two quantum bits is

$$U_{CN} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \quad (1.31)$$

It is supposed that U_{CN}^{+} is the conjugate-transpose matrix of U_{CN} and is equal to $((U_{CN})^{*})^{t} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$, where the * is the complex conjugation and the t is the transpose operation. Since $U_{CN} \times ((U_{CN})^{*})^{t} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} =$ $((U_{CN})^{*})^{t} \times U_{CN} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$, U_{CN} is a unitary matrix or a unitary operator. This is to say that the ***controlled-NOT*** or ***CNOT*** gate U_{CN} is one of quantum gates with two quantum bits. If the quantum state $l_0 |00> + l_1 |01> + l_2 |10> + l_3 |11>$ is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \\ l_2 \\ l_3 \end{pmatrix}, \quad (1.32)$$

with the first entry l_0 is the amplitude for |00>, the second entry l_1 is the amplitude for |01>, the third entry l_2 is the amplitude for |10> and the fourth entry l_3 is the amplitude for |11>, then the corresponding output from the ***CNOT*** gate U_{CN} is

$$\begin{pmatrix} l_0 \\ l_1 \\ l_3 \\ l_2 \end{pmatrix} = l_0 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} + l_1 \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} + l_3 \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} + l_2 \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = l_0 |00> + l_1 |01> + l_3 |10> + l_2 |11>. \quad (1.33)$$

This indicates that the ***CNOT*** gate U_{CN} converts two quantum bits from one state $l_0 |00> + l_1 |01> + l_2 |10> + l_3 |11>$ to another state $l_0 |00> + l_1 |01> + l_3 |10> + l_2 |11>$. This is to say that in the ***CNOT*** gate U_{CN} if the control quantum bit (the *first* quantum bit) is set to 0, then the target quantum bit (the *second* quantum bit) is left alone. If the control quantum bit (the *first* quantum bit) is set to 1, then the target quantum bit (the *second* quantum bit) is flipped. The probability of measuring a |00> or |01> is unchanged, the

probability of measuring a |10> is $|l_3|^2$ and the probability of measuring a |11> is $|l_2|^2$ after applying the ***CNOT*** gate U_{CN}. Because $(U_{CN})^2 = U_{CN} \times U_{CN} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$, applying the ***CNOT*** gate U_{CN} twice to a state is equivalent to do nothing to it. For **IBM Q** Experience, the graphical representation of the ***CNOT*** gate U_{CN} is as follows:

.

In the graphical representation of the ***CNOT*** gate U_{CN}, the top wire carries the controlled quantum bit and the bottom wire carries the target quantum bit.

1.11.1 Connectivity of the Controlled-NOT Gate in IBMQX4

Those authors that wrote textbooks write quantum algorithms with a fully connected hardware in which one can apply a quantum gate of two quantum bits to any pair of two quantum bits. In practice, *ibmqx4* that is a real quantum computer may not have full connectivity. In the *ibmqx4* with five quantum bits, there are six connections. Six connections of a **CNOT** gate appears in Figure 1.21.The *first* **CNOT** gate in Figure 1.21 has the controlled quantum bit q[1] and the target quantum bit q[0] and the corresponding instruction in version 2.0 of Open QASM is “cx q[1],q[0];”, where cx is to represent the **CNOT** gate. The *second* **CNOT** gate in Figure 1.21 has the controlled

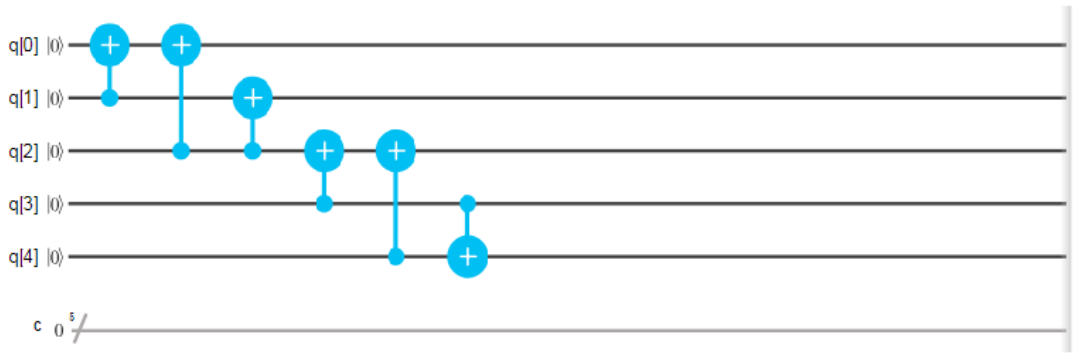

Figure 1.21: A **CNOT** gate has six connections in *ibmqx4* on **IBM** quantum computers.

quantum bit q[2] and the target quantum bit q[0] and the corresponding instruction in version 2.0 of Open QASM is “cx q[2],q[0];”, where cx is to represent the **CNOT** gate.

The *third* **CNOT** gate in Figure 1.21 has the controlled quantum bit q[2] and the target quantum bit q[1] and the corresponding instruction in version 2.0 of Open QASM is “cx q[2],q[1];”, where cx is to represent the **CNOT** gate. The *fourth* **CNOT** gate in Figure 1.21 has the controlled quantum bit q[3] and the target quantum bit q[2] and the corresponding instruction in version 2.0 of Open QASM is “cx q[3],q[2];”, where cx is to represent the **CNOT** gate. The *fifth* **CNOT** gate in Figure 1.21 has the controlled quantum bit q[4] and the target quantum bit q[2] and the corresponding instruction in version 2.0 of Open QASM is “cx q[4],q[2];”, where cx is to represent the **CNOT** gate. The *sixth* **CNOT** gate in Figure 1.21 has the controlled quantum bit q[3] and the target quantum bit q[4] and the corresponding instruction in version 2.0 of Open QASM is “cx q[3],q[4];”, where cx is to represent the **CNOT** gate.

In contrast, a fully connected hardware with five quantum bits would allow a **CNOT** gate to apply to twenty different pairs of any two-quantum bits. This indicates that there are fourteen “missing connections”. Fortunately, there are different ways to yield connections by means of using clever gate sequences. For example, a **CNOT** gate that has the controlled quantum bit q[j] and the target quantum bit q[k] for $0 \leq j$ and $k \leq 4$ can be reversed by means of applying Hadamard gates on each quantum bit both before and after the **CNOT** gate. This is to say that the new instruction (the new connection) “cx q[k], q[j]” is implemented by means of applying the five instructions that are subsequently “h q[j];”, “h q[k]”, “cx q[j], q[k]”, “h q[j]” and “h q[k]” for $0 \leq j$ and $k \leq 4$. Similarly, there exists a gate sequence to make a **CNOT** gate with the controlled bit q[j] and the target bit q[l] if one has connections between the controlled bit q[j] and the target bit q[k], and the controlled bit q[k] and the target bit q[l] for $0 \leq j$, k and $l \leq 4$. This indicates that the new instruction (the new connection) “cx q[j], q[l]” is implemented by means of applying the four instructions that are subsequently “cx q[k], q[l]”, “cx q[j], q[k]”, “cx q[k], q[l]” and “cx q[j], q[k]” for $0 \leq j$, k and $l \leq 4$.

1.11.2 Implementing a Copy Machine of one bit with the CNOT Gate

Offered its data input is initialized permanently with |0>. Then, the **CNOT** gate emits a copy of the controlled input on each output. Therefore, the **CNOT** gate actually is a copy machine of one bit. In Listing 1.11, the program in the backend *ibmqx4* with five quantum bits in **IBM**’s quantum computer is the eleventh example in which we

describe how to program with the **CNOT** gate that converts the controlled bit q[3] and the target bit q[3] from one state ($\frac{1}{\sqrt{2}}$ (|0> + |1>)) (|0>) ($\frac{1}{\sqrt{2}}$ (|00> + |10>) to another state ($\frac{1}{\sqrt{2}}$) (|00> + |11>). Actually, the **CNOT** gate emits a copy of the controlled input on each output. Figure 1.22 is the corresponding quantum circuit of the program in Listing 1.11. The statement “OPENQASM 2.0;” on line one of Listing 1.11 is to point out that the program is written with version 2.0 of Open QASM. Then, the statement “include "qelib1.inc";” on line two of Listing 1.11 is to continue parsing the file “qelib1.inc” as if the contents of the file were pasted at the location of the include statement, where the file “qelib1.inc” is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement “qreg q[5];” on line three of Listing 1.11 is to declare that in the program there are five quantum bits. In the

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[3];
6. cx q[3],q[4];
7. measure q[3] -> c[3];
8. measure q[4] -> c[4];
```

Listing 1.11: Implementing a copy machine of one bit with the **CNOT** gate.

left top of Figure 1.22, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement “creg c[5];” on line four of Listing 1.11 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.22, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

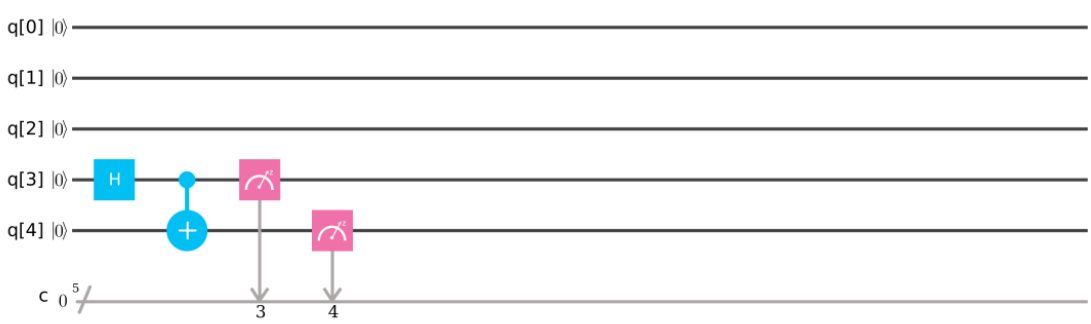

Figure 1.22: The corresponding quantum circuit of the program in Listing 1.11.

The statement “h q[3];” on line five of Listing 1.11 actually completes $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This is to say that the statement “h q[3];” on line five of Listing 1.11 is to use the Hadamard gate to convert q[3] from one state |0> to another state $\frac{1}{\sqrt{2}} (|0> + |1>)$ (its superposition), where “h” is to represent the Hadamard gate. Next, the statement “cx q[3],q[4];” on line six of Listing 1.11 actually completes $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|00> + |11>)$. This indicates that the statement “cx q[3],q[4];” on line six of Listing 1.11 is to apply the **CNOT** gate to emit a copy of the controlled input q[3] on each output.

Next, the statement “measure q[3] -> c[3];” on line seven of Listing 1.11 is to measure the fourth quantum bit q[3] and to record the measurement outcome by overwriting the fourth classical bit c[3]. The statement “measure q[4] -> c[4];” on line eight of Listing 1.11 is to measure the fifth quantum bit q[4] and to record the measurement outcome by overwriting the fifth classical bit c[4].In the backend *ibmqx4* with five quantum bits in **IBM**’s quantum computers, we use the command “simulate” to execute the program in Listing 1.11. The result appears in Figure 1.23. From Figure

Figure 1.23: After the measurement to the program in Listing 1.11 is completed, we obtain the answer 00000 with the probability 0.540 or the answer 11000 with the

probability 0.460.

1.23, we obtain the answer 00000 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>) with the probability 0.540. Or we obtain the answer 11000 with the probability 0.460 (c[4] = q[4] = |1>, c[3] = q[3] = |1>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>). If the answer is 00000, then this imply that the **CNOT** gate copy the value 0 of the controlled input q[3] to the target bit q[4]. If the answer is 11000, then this indicates that the **CNOT** gate copy the value 1 of the controlled input q[3] to the target bit q[4].

1.12 The U1(λ) Gate of Single Quantum Bit with One Parameter

The $U1(\lambda)$ gate that is the first physical gate of the Quantum Experience and is a phase gate of single quantum bit of one parameter with zero duration is

$$U1(\lambda) = U1(\text{lambda}) = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\lambda} \end{pmatrix}, \tag{1.34}$$

where λ (lambda) is a real value. It is assumed that $(U1(\lambda))^+$ ($U1$(lambda))$^+$ is the conjugate-transpose matrix of $U1(\lambda)$ ($U1$(lambda)) and is equal to $(((U1(\lambda)))^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\lambda} \end{pmatrix}$, where the * is the complex conjugation and the t is the transpose operation. Because $U1(\lambda) \times (U1(\lambda))^+ = U1(\lambda) \times (((U1(\lambda)))^*)^t = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\lambda} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\lambda} \end{pmatrix} = (U1(\lambda))^+ \times U1(\lambda) = (((U1(\lambda)))^*)^t \times U1(\lambda) = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\sqrt{-1}\times\lambda} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\lambda} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $U1(\lambda)$ ($U1$(lambda)) is a unitary matrix or a unitary operator. This implies that the phase gate $U1(\lambda)$ ($U1$(lambda)) is one of quantum gates that is a phase gate of single quantum bit of one parameter with zero duration. If the quantum state l_0 |0> + l_1 |1> is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \tag{1.35}$$

with the first entry l_0 is the amplitude for |0> and the second entry l_1 is the amplitude for |1>, then the corresponding output from the phase gate $U1(\lambda)$ ($U1$(lambda)) is

$$\begin{pmatrix} l_0 \\ e^{\sqrt{-1}\times\lambda} \times l_1 \end{pmatrix} = l_0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \left(e^{\sqrt{-1}\times\lambda} \times l_1\right) \begin{pmatrix} 0 \\ 1 \end{pmatrix} = l_0 |0\rangle + \left(e^{\sqrt{-1}\times\lambda} \times l_1\right) |1\rangle. \quad (1.36)$$

This is to say that the phase gate *U1*(λ) (*U1*(lambda)) converts one quantum bit from one state l_0 |0> + l_1 |1> to another state l_0 |0> + $\left(e^{\sqrt{-1}\times\lambda} \times l_1\right)$ |1>. This indicates that the phase gate *U1*(λ) (*U1*(lambda)) leaves |0> unchanged and modifies the phase of |1> to give ($e^{\sqrt{-1}\times\lambda}$) |1>. The probability of measuring a |0> or |1> is unchanged after using the phase gate *U1*(λ) (*U1*(lambda)), however it modifies the phase of the quantum state. Because $(U1(\lambda))^2 = U1(\lambda) \times U1(\lambda) = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\lambda} \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times\lambda} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times 2\times\lambda} \end{pmatrix}$, using the phase gate *U1*(λ) (*U1*(lambda)) twice to a state is equivalent to do that leaves |0> unchanged and modifies the phase of |1> to give ($e^{\sqrt{-1}\times 2\times\lambda}$) |1> to it. For **IBM Q** Experience, the graphical representation of the phase gate *U1*(λ) (*U1*(lambda)) is as follows:

.

1.12.1 Programming with the U1(λ) Gate with One Parameter

In Listing 1.12, in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer, the program is the *twelfth* example in which we introduce how to program with the phase gate *U1*(2 * pi) that converts single quantum bit q[0] from one state ($\frac{1}{\sqrt{2}}$) (|0> + |1>) to another state ($\frac{1}{\sqrt{2}}$) (|0> + |1>). Because the input value of the first parameter lambda for the phase gate *U1*(lambda) is (2 * pi) and *U1*(2 * pi) is equal to $\begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1}\times 2\times\pi} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, the phase gate *U1*(2 * pi) actually implements one identity gate. Figure 1.24 is the corresponding quantum circuit of the program in Listing 1.12.

The statement "OPENQASM 2.0;" on line one of Listing 1.12 is to indicate that the program is written with version 2.0 of Open QASM. Next, the statement "include "qelib1.inc";" on line two of Listing 1.12 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line

three of Listing 1.12 is to declare that in the program there are five quantum bits. In the left top of Figure 1.24, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Then, the statement "creg c[5];" on line four of Listing 1.12 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.24, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
6. u1(2 * pi) q[0];
7. measure q[0] -> c[0];
```

Listing 1.12: Program of using the phase gate *U1*(2 * pi) with one parameter.

The statement "h q[0];" on line five of Listing 1.12 actually implements $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}}$ (|0> + |1>). This is

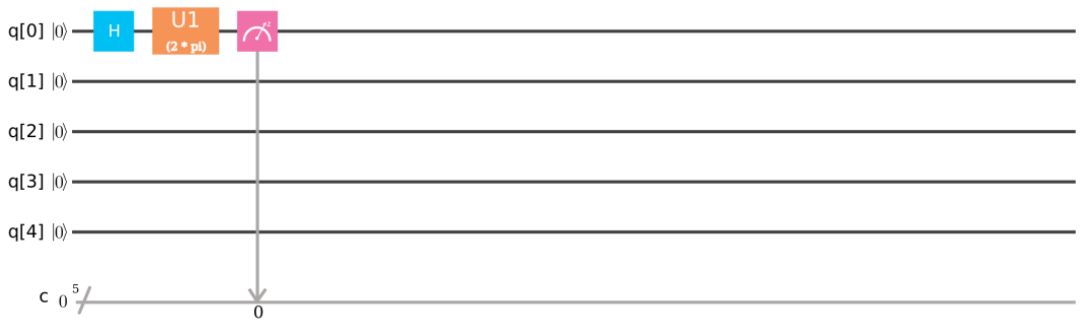

Figure 1.24: The corresponding quantum circuit of the program in Listing 1.12.

to say that the statement "h q[0];" on line five of Listing 1.12 is to apply the Hadamard gate to convert q[0] from one state |0> to another state $\frac{1}{\sqrt{2}}$ (|0> + |1>) (its superposition), where "h" is to represent the Hadamard gate. Next, the statement "u1(2 * pi) q[0];" on line six of Listing 1.12 actually completes $\begin{pmatrix} 1 & 0 \\ 0 & e^{\sqrt{-1} \times 2 \times \pi} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \times$

$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This is to say that the statement "u1(2 * pi) q[0];" on line six of Listing 1.12 is to implement one identity gate to q[0].

Next, the statement "measure q[0] -> c[0];" on line seven of Listing 1.12 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers, we apply the command "simulate" to execute the program in Listing 1.12. The result appears in Figure 1.25. From Figure 1.25, we obtain the answer 00000 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>) with the probability 0.530. Or we obtain the answer 00001 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |1>) with the probability 0.470.

Figure 1.25: After the measurement to the program in Listing 1.12 is completed, we obtain the answer 00000 with the probability 0.530 or the answer 00001 with the probability 0.470.

1.13 The U2(ϕ, λ) Gate of Single Quantum Bit with Two Parameters

The phase gate *U2*(ϕ, λ) (*U2*(phi, lambda)) that is the second physical gate of the Quantum Experience and is a phase gate of single quantum bit of two parameters with duration one unit of gate time is

$$U2(\phi, \lambda) = U2(\text{phi, lambda}) = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-e^{\sqrt{-1}\times\lambda}}{\sqrt{2}} \\ \frac{e^{\sqrt{-1}\times\phi}}{\sqrt{2}} & \frac{e^{\sqrt{-1}\times(\lambda+\phi)}}{\sqrt{2}} \end{pmatrix}, \qquad (1.37)$$

where ϕ and λ (phi and lambda) are both real numbers. It is assumed that $(U2(\phi, \lambda))^+$ ($(U2(\text{phi, lambda}))^+$) is the conjugate-transpose matrix of $U2(\phi, \lambda)$ (*U2*(phi,

lambda)) and is equal to $((U2(\phi,\ \lambda))^{*})^{t} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{e^{-\sqrt{-1}\times\phi}}{\sqrt{2}} \\ \frac{-e^{-\sqrt{-1}\times\lambda}}{\sqrt{2}} & \frac{e^{-\sqrt{-1}\times(\lambda+\phi)}}{\sqrt{2}} \end{pmatrix}$, where the * is the complex conjugation and the *t* is the transpose operation. Because $U2(\phi,\ \lambda) \times (U2(\phi,\ \lambda))^{+} = U2(\phi,\ \lambda) \times ((U2(\phi,\ \lambda))^{*})^{t} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-e^{\sqrt{-1}\times\lambda}}{\sqrt{2}} \\ \frac{e^{\sqrt{-1}\times\phi}}{\sqrt{2}} & \frac{e^{\sqrt{-1}\times(\lambda+\phi)}}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{e^{-\sqrt{-1}\times\phi}}{\sqrt{2}} \\ \frac{-e^{-\sqrt{-1}\times\lambda}}{\sqrt{2}} & \frac{e^{-\sqrt{-1}\times(\lambda+\phi)}}{\sqrt{2}} \end{pmatrix} = (U2(\phi,\ \lambda))^{+} \times U2(\phi,\ \lambda) = ((U2(\phi,\ \lambda))^{*})^{t} \times U2(\phi,\ \lambda) = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{e^{-\sqrt{-1}\times\phi}}{\sqrt{2}} \\ \frac{-e^{-\sqrt{-1}\times\lambda}}{\sqrt{2}} & \frac{e^{-\sqrt{-1}\times(\lambda+\phi)}}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-e^{\sqrt{-1}\times\lambda}}{\sqrt{2}} \\ \frac{e^{\sqrt{-1}\times\phi}}{\sqrt{2}} & \frac{e^{\sqrt{-1}\times(\lambda+\phi)}}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $U2(\phi,\ \lambda)$ (*U2*(phi, lambda)) is a unitary matrix or a unitary operator. This implies that the phase gate $U2(\phi,\ \lambda)$ (*U2*(phi, lambda)) is one of quantum gates that is a phase gate of single quantum bit of two parameters with duration one unit of time. If the quantum state l_0 |0> + l_1 |1> is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \tag{1.38}$$

with the first entry l_0 is the amplitude for |0> and the second entry l_1 is the amplitude for |1>, then the corresponding output from the phase gate $U2(\phi,\ \lambda)$ (*U2*(phi, lambda)) is

$$\begin{pmatrix} \frac{l_0 - e^{\sqrt{-1}\times\lambda}\times l_1}{\sqrt{2}} \\ \frac{e^{\sqrt{-1}\times\phi}\times l_0 + e^{\sqrt{-1}\times(\lambda+\phi)}\times l_1}{\sqrt{2}} \end{pmatrix} = \frac{l_0 - e^{\sqrt{-1}\times\lambda}\times l_1}{\sqrt{2}}\ |0> + \frac{e^{\sqrt{-1}\times\phi}\times l_0 + e^{\sqrt{-1}\times(\lambda+\phi)}\times l_1}{\sqrt{2}}\ |1>. \tag{1.39}$$

This is to say that the phase gate $U2(\phi,\ \lambda)$ (*U2*(phi, lambda)) converts single quantum bit from one state l_0 |0> + l_1 |1> to another state $\frac{l_0 - e^{\sqrt{-1}\times\lambda}\times l_1}{\sqrt{2}}$ |0> + $\frac{e^{\sqrt{-1}\times\phi}\times l_0 + e^{\sqrt{-1}\times(\lambda+\phi)}\times l_1}{\sqrt{2}}$ |1>. Since $(U2(\phi,\ \lambda))^{2} = U2(\phi,\ \lambda) \times U2(\phi,\ \lambda) = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-e^{\sqrt{-1}\times\lambda}}{\sqrt{2}} \\ \frac{e^{\sqrt{-1}\times\phi}}{\sqrt{2}} & \frac{e^{\sqrt{-1}\times(\lambda+\phi)}}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-e^{\sqrt{-1}\times\lambda}}{\sqrt{2}} \\ \frac{e^{\sqrt{-1}\times\phi}}{\sqrt{2}} & \frac{e^{\sqrt{-1}\times(\lambda+\phi)}}{\sqrt{2}} \end{pmatrix} =$

$\begin{pmatrix} \frac{1-e^{\sqrt{-1}\times(\lambda+\phi)}}{2} & \frac{-e^{\sqrt{-1}\times\lambda}\times(1+e^{\sqrt{-1}\times(\lambda+\phi)})}{2} \\ \frac{e^{\sqrt{-1}\times\phi}\times(1+e^{\sqrt{-1}\times(\lambda+\phi)})}{2} & \frac{\left(1-e^{\sqrt{-1}\times(\lambda+\phi)}\right)\times(-e^{\sqrt{-1}\times(\lambda+\phi)})}{2} \end{pmatrix}$, using the phase gate $U2(\phi,\ \lambda)$ (*U2*(phi, lambda)) twice to a state is equivalent to modify the amplitude to it. For **IBM Q** Experience, the graphical representation of the phase gate $U2(\phi,\ \lambda)$ (*U2*(phi, lambda)) is as follows:

.

1.13.1 Programming with the U2($\phi,\ \lambda$) Gate with Two Parameters

For the phase gate *U2*(phi, lambda) the input values of the first parameter phi and the second parameter lambda are respectively (0 * pi) and (1 * pi), so *U2*(0*pi, 1*pi) is equal to $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-e^{\sqrt{-1}\times1\times\pi}}{\sqrt{2}} \\ \frac{e^{\sqrt{-1}\times0\times\pi}}{\sqrt{2}} & \frac{e^{\sqrt{-1}\times(1\times\pi+0\times\pi)}}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$. Therefore, the phase gate *U2*(0*pi, 1*pi) actually implements one Hadamard gate. In Listing 1.13, in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer, the program is the *thirteenth* example in which we illustrate how to program with the phase gate *U2*(0*pi, 1*pi) that converts single quantum bit q[0] from one state ($\frac{1}{\sqrt{2}}$) (|0> + |1>) to another state (|0>). Figure 1.26 is the corresponding quantum circuit of the program in Listing 1.13.

The statement "OPENQASM 2.0;" on line one of Listing 1.13 is to point out that the program is written with version 2.0 of Open QASM. Then, the statement "include "qelib1.inc";" on line two of Listing 1.13 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line three of Listing 1.13 is to declare that in the program there are five quantum bits. In the left top of Figure 1.26, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement "creg c[5];" on line four of Listing 1.13 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.26, five classical bits are subsequently c[0], c[1], c[2], c[3] and c[4]. The initial value of each classical bit is set to 0.

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
6. u2(0*pi,1*pi) q[0];
7. measure q[0] -> c[0];
```

Listing 1.13: Program of using the phase gate *U2*(0*pi, 1*pi) with two parameters.

The statement "h q[0];" on line five of Listing 1.13 actually completes $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This is to say that the statement "h q[0];" on line five of Listing 1.13 is to use the Hadamard gate to convert q[0] from one state |0> to another state $\frac{1}{\sqrt{2}} (|0> + |1>)$ (its superposition), where

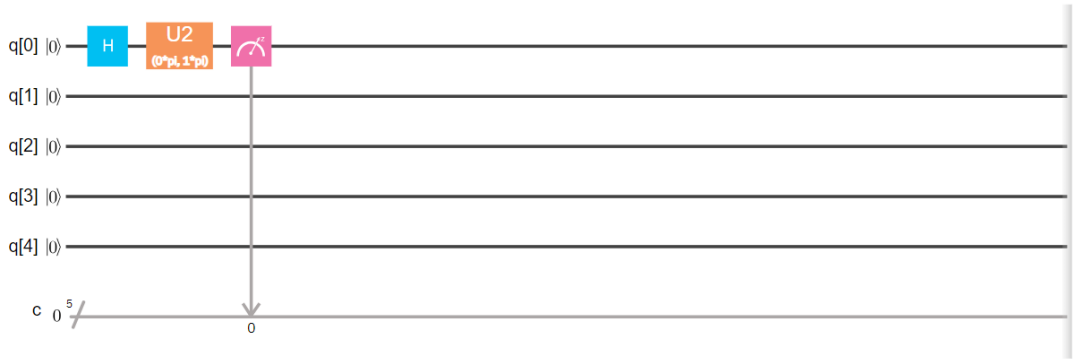

Figure 1.26: The corresponding quantum circuit of the program in Listing 1.13.

"h" is to represent the Hadamard gate. Next, the statement "u2(0*pi,1*pi) q[0];" on line six of Listing 1.13 actually implements $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-e^{\sqrt{-1}\times 1\times \pi}}{\sqrt{2}} \\ \frac{e^{\sqrt{-1}\times 0\times \pi}}{\sqrt{2}} & \frac{e^{\sqrt{-1}\times (1\times \pi + 0\times \pi)}}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0>$. This indicates that the statement "u2(0*pi,1*pi) q[0];" on line six of Listing 1.13 is to complete one Hadamard gate to q[0]. Therefore, applying the Hadamard gate twice from line five and line six of Listing 1.13 to q[0]

does nothing to it.

Next, the statement "measure q[0] -> c[0];" on line seven of Listing 1.13 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers, we use the command "simulate" to execute the program in

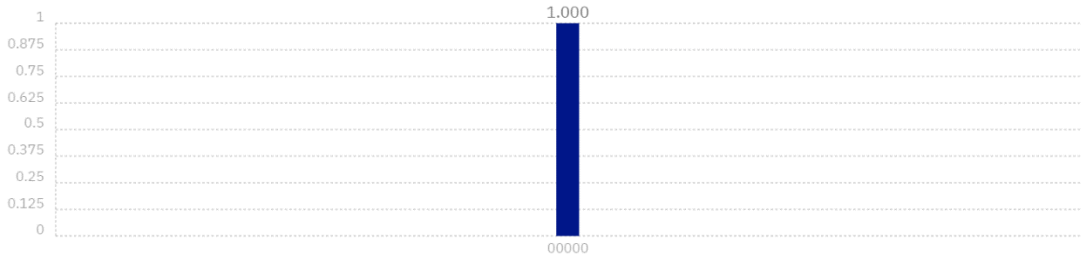

Figure 1.27: After the measurement to the program in Listing 1.13 is completed, we obtain the answer 00000 with the probability 1.000.

Listing 1.13. The result appears in Figure 1.27. From Figure 1.27, we obtain the answer 00000 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>) with the probability 1.000.

1.14 The U3(θ,ϕ,λ) Gate of Single Quantum Bit of Three Parameters

The phase gate *U3*(θ, ϕ, λ) (*U3*(theta, phi, lambda)) that is the third physical gate of the Quantum Experience and is a phase gate of single quantum bit of three parameters with duration two units of gate time is

$$U3(\theta,\ \phi,\ \lambda) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{\sqrt{-1}\times\lambda} \times \sin\left(\frac{\theta}{2}\right) \\ e^{\sqrt{-1}\times\phi} \times \sin\left(\frac{\theta}{2}\right) & e^{\sqrt{-1}\times(\lambda+\phi)} \times \cos\left(\frac{\theta}{2}\right) \end{pmatrix}, \quad (1.40)$$

where θ, ϕ and λ (theta, phi and lambda) are all real numbers. It is supposed that $(U3(\theta,\ \phi,\ \lambda))^{+}$ ($(U3(\text{theta, phi, lambda}))^{+}$) is the conjugate-transpose matrix of *U3*(θ, ϕ, λ) (*U3*(theta, phi, lambda)) and is equal to $((U3(\theta,\ \phi,\ \lambda))^{*})^{t}$ = $\begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & e^{-\sqrt{-1}\times\phi} \times \sin\left(\frac{\theta}{2}\right) \\ -e^{-\sqrt{-1}\times\lambda} \times \sin\left(\frac{\theta}{2}\right) & e^{-\sqrt{-1}\times(\lambda+\phi)} \times \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$, where the * is the complex conjugation and the *t* is the transpose operation. Because *U3*(θ, ϕ, λ) × ((*U3*(θ, ϕ,

λ))*)^t = $\begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{\sqrt{-1}\times\lambda} \times \sin\left(\frac{\theta}{2}\right) \\ e^{\sqrt{-1}\times\phi} \times \sin\left(\frac{\theta}{2}\right) & e^{\sqrt{-1}\times(\lambda+\phi)} \times \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$ ×

$\begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & e^{-\sqrt{-1}\times\phi} \times \sin\left(\frac{\theta}{2}\right) \\ -e^{-\sqrt{-1}\times\lambda} \times \sin\left(\frac{\theta}{2}\right) & e^{-\sqrt{-1}\times(\lambda+\phi)} \times \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$ = ((*U3*(θ, ϕ, λ))*)^t × *U3*(θ, ϕ, λ)

= $\begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & e^{-\sqrt{-1}\times\phi} \times \sin\left(\frac{\theta}{2}\right) \\ -e^{-\sqrt{-1}\times\lambda} \times \sin\left(\frac{\theta}{2}\right) & e^{-\sqrt{-1}\times(\lambda+\phi)} \times \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$ ×

$\begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{\sqrt{-1}\times\lambda} \times \sin\left(\frac{\theta}{2}\right) \\ e^{\sqrt{-1}\times\phi} \times \sin\left(\frac{\theta}{2}\right) & e^{\sqrt{-1}\times(\lambda+\phi)} \times \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$ = $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, *U3*(θ, ϕ, λ) (*U3*(theta, phi, lambda)) is a unitary matrix or a unitary operator. This implies that the phase gate *U3*(θ, ϕ, λ) (*U3*(theta, phi, lambda)) is one of quantum gates that is a phase gate of single quantum bit of three parameters with duration two units of gate time. If the quantum state l_0 |0> + l_1 |1> is written in a vector notation as

$$\begin{pmatrix} l_0 \\ l_1 \end{pmatrix}, \tag{1.41}$$

with the first entry l_0 is the amplitude for |0> and the second entry l_1 is the amplitude for |1>, then the corresponding output from the phase gate *U3*(θ, ϕ, λ) (*U3*(theta, phi, lambda)) is

$\begin{pmatrix} l_0 \times \cos\left(\frac{\theta}{2}\right) - l_1 \times e^{\sqrt{-1}\times\lambda} \times \sin\left(\frac{\theta}{2}\right) \\ l_0 \times e^{\sqrt{-1}\times\phi} \times \sin\left(\frac{\theta}{2}\right) + l_1 \times e^{\sqrt{-1}\times(\lambda+\phi)} \times \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$ = ($l_0 \times \cos\left(\frac{\theta}{2}\right) - l_1 \times$

$e^{\sqrt{-1}\times\lambda} \times \sin\left(\frac{\theta}{2}\right)$) |0> + ($l_0 \times e^{\sqrt{-1}\times\phi} \times \sin\left(\frac{\theta}{2}\right) + l_1 \times e^{\sqrt{-1}\times(\lambda+\phi)} \times \cos\left(\frac{\theta}{2}\right)$) |1>.

(1.42)

This indicates that the phase gate *U3*(θ, ϕ, λ) (*U3*(theta, phi, lambda)) converts single quantum bit from one state l_0 |0> + l_1 |1> to another state ($l_0 \times \cos\left(\frac{\theta}{2}\right) - l_1 \times e^{\sqrt{-1}\times\lambda} \times \sin\left(\frac{\theta}{2}\right)$) |0> + ($l_0 \times e^{\sqrt{-1}\times\phi} \times \sin\left(\frac{\theta}{2}\right) + l_1 \times e^{\sqrt{-1}\times(\lambda+\phi)} \times \cos\left(\frac{\theta}{2}\right)$) |1>. Because (*U3*(θ, ϕ, λ))^2 =

$$\begin{pmatrix} \cos^2\left(\frac{\theta}{2}\right) - e^{\sqrt{-1}\times(\lambda+\phi)} \times \sin^2\left(\frac{\theta}{2}\right) & -\cos\left(\frac{\theta}{2}\right) \times \sin\left(\frac{\theta}{2}\right) \times e^{\sqrt{-1}\times\lambda} \times (1 + e^{\sqrt{-1}\times(\lambda+\phi)}) \\ \cos\left(\frac{\theta}{2}\right) \times \sin\left(\frac{\theta}{2}\right) \times e^{\sqrt{-1}\times\phi} \times (1 + e^{\sqrt{-1}\times(\lambda+\phi)}) & \left(\sin^2\left(\frac{\theta}{2}\right) - \cos^2\left(\frac{\theta}{2}\right) \times e^{\sqrt{-1}\times(\lambda+\phi)}\right) \times (-e^{\sqrt{-1}\times(\lambda+\phi)}) \end{pmatrix},$$

applying the phase gate *U3*(θ, ϕ, λ) (*U3*(theta, phi, lambda)) twice to a state is equivalent to modify the amplitude to it. For **IBM Q** Experience, the graphical representation of the phase gate *U3*(θ, ϕ, λ) (*U3*(theta, phi, lambda)) is as follows:

.

1.14.1 Programming with the U3(θ, ϕ, λ) Gate with Three Parameters

For the phase gate *U3*(theta, phi, lambda), the input value of the first parameter theta is (0.5 * pi), the input value of the second parameter phi is (0 * pi) and the input value of the third parameter lambda is (1 * pi), so *U3*(0.5*pi, 0*pi, 1*pi) is equal to $\begin{pmatrix} \cos\left(\frac{\pi}{4}\right) & -e^{\sqrt{-1}\times1\times\pi} \times \sin\left(\frac{\pi}{4}\right) \\ e^{\sqrt{-1}\times0\times\pi} \times \sin\left(\frac{\pi}{4}\right) & e^{\sqrt{-1}\times(1\times\pi+0\times\pi)} \times \cos\left(\frac{\pi}{4}\right) \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-e^{\sqrt{-1}\times1\times\pi}}{\sqrt{2}} \\ \frac{e^{\sqrt{-1}\times0\times\pi}}{\sqrt{2}} & \frac{e^{\sqrt{-1}\times(1\times\pi+0\times\pi)}}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$. Hence, the phase gate *U3*(0.5*pi, 0*pi, 1*pi) actually completes one Hadamard gate. In Listing 1.14, in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computer, the program is the *fourteenth* example in which we describe how to program with the phase gate *U3*(0.5*pi, 0*pi, 1*pi) that converts single quantum bit q[0] from one state ($\frac{1}{\sqrt{2}}$) (|0> + |1>) to another state (|0>). Figure 1.28 is the corresponding quantum circuit of the program in Listing 1.14.

The statement "OPENQASM 2.0;" on line one of Listing 1.14 is to indicate that the program is written with version 2.0 of Open QASM. Next, the statement "include "qelib1.inc";" on line two of Listing 1.14 is to continue parsing the file "qelib1.inc" as if the contents of the file were pasted at the location of the include statement, where the file "qelib1.inc" is **Quantum Experience (QE) Standard Header** and the path is specified relative to the current working directory. The statement "qreg q[5];" on line three of Listing 1.14 is to declare that in the program there are five quantum bits. In the left top of Figure 1.28, five quantum bits are subsequently q[0], q[1], q[2], q[3] and q[4]. The initial value of each quantum bit is set to |0>. Next, the statement "creg c[5];" on line four of Listing 1.14 is to declare that there are five classical bits in the program. In the left bottom of Figure 1.28, five classical bits are subsequently c[0], c[1], c[2],

c[3] and c[4]. The initial value of each classical bit is set to 0.

```
1. OPENQASM 2.0;
2. include "qelib1.inc";
3. qreg q[5];
4. creg c[5];
5. h q[0];
6. u3(0.5*pi,0*pi,1*pi) q[0];
7. measure q[0] -> c[0];
```

Listing 1.14: Program of using the phase gate *U3*(0.5*pi, 0*pi, 1*pi) of three parameters.

The statement "h q[0];" on line five of Listing 1.14 actually performs $\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} (\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}) = \frac{1}{\sqrt{2}} (|0> + |1>)$. This indicates that the statement "h q[0];" on line five of Listing 1.14 is to apply the Hadamard gate to convert q[0] from one state |0> to another state $\frac{1}{\sqrt{2}} (|0> + |1>)$ (its superposition), where "h" is to represent the Hadamard gate. Next, the statement "u3(0.5*pi,0*pi,1*pi) q[0];"

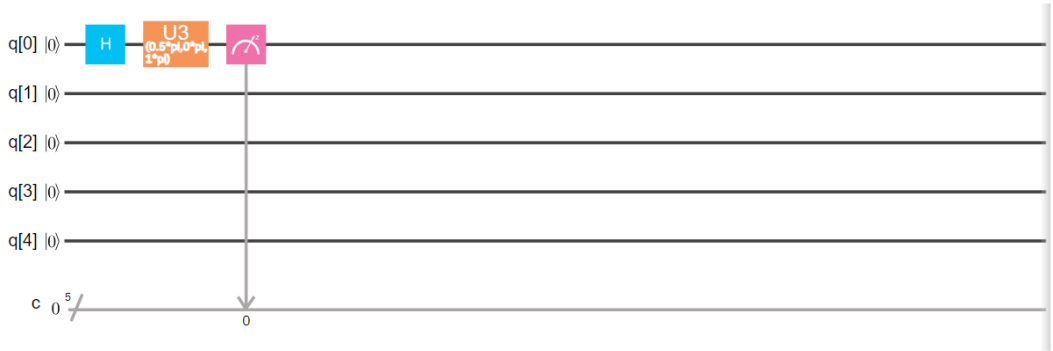

Figure 1.28: The corresponding quantum circuit of the program in Listing 1.14.

on line six of Listing 1.14 actually completes $\begin{pmatrix} \cos\left(\frac{\pi}{4}\right) & -e^{\sqrt{-1}\times 1\times\pi} \times \sin\left(\frac{\pi}{4}\right) \\ e^{\sqrt{-1}\times 0\times\pi} \times \sin\left(\frac{\pi}{4}\right) & e^{\sqrt{-1}\times(1\times\pi+0\times\pi)} \times \cos\left(\frac{\pi}{4}\right) \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} =$

$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-e^{\sqrt{-1}\times 1\times\pi}}{\sqrt{2}} \\ \frac{e^{\sqrt{-1}\times 0\times\pi}}{\sqrt{2}} & \frac{e^{\sqrt{-1}\times(1\times\pi+0\times\pi)}}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \times \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0>$. This is to say that the statement "u3(0.5*pi,0*pi,1*pi) q[0];" on line six of Listing 1.14 is to complete one Hadamard gate to q[0]. Hence, using the Hadamard gate twice from line five and line six of Listing 1.14 to q[0] does nothing to it.

Next, the statement "measure q[0] -> c[0];" on line seven of Listing 1.14 is to measure the first quantum bit q[0] and to record the measurement outcome by overwriting the first classical bit c[0]. In the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers, we use the command "simulate" to execute the program in Listing 1.14. The result appears in Figure 1.29. From Figure 1.29, we obtain the answer 00000 (c[4] = q[4] = |0>, c[3] = q[3] = |0>, c[2] = q[2] = |0>, c[1] = q[1] = |0> and c[0] = q[0] = |0>) with the probability 1.000.

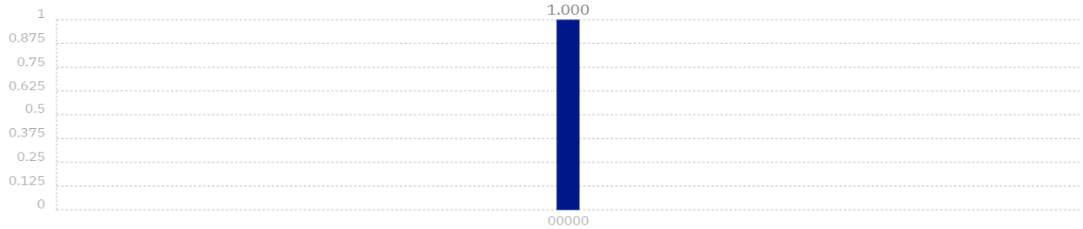

Figure 1.29: After the measurement to the program in Listing 1.14 is completed, we obtain the answer 00000 with the probability 1.000.

1.15 Summary

In this chapter, we introduced single quantum bit, multiple quantum bits and their superposition. We also described two statements of declaration and measurement for quantum bits and classical bits in Open QASM (version 2.0) in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers. We illustrated all of the quantum gates with single quantum bit and the ***controlled-NOT*** or ***CNOT*** gate of two quantum bits in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers. Simultaneously, we also in detail introduced connectivity of the ***controlled-NOT*** gate in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers. We introduced how to program with each quantum gate of single quantum bit completing each different kind of application and how to execute each quantum program in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers. We also described how to program with the ***controlled-NOT*** gate to implement a copy machine

of one bit.

1.16 Bibliographical Notes

A famous article that gives a detailed technical definition for quantum supremacy is [Aaronson and Chen 2017]. Popular textbooks [Nielsen and Chuang 2000; Imre and Balazs 2007; Lipton and Regan 2014] give an excellent introduction for quantum bits and quantum gates. A popular textbook [Silva 2018], a famous project [**IBM Q** 2016] and two famous articles [Cross et al 2017; Coles et al 2018] give many excellent examples to write quantum programs with quantum assembly language in Open QASM (version 2.0) in the backend *ibmqx4* with five quantum bits in **IBM**'s quantum computers.

1.17 Exercises

1.1 Please write a quantum program in which we use the U3(θ, ϕ, λ) gate with three parameters to implement the ***NOT*** gate.

1.2 Please write a quantum program in which we apply the U3(θ, ϕ, λ) gate with three parameters to implement the Hadamard gate.

1.3 Please write a quantum program in which we make use of the U3(θ, ϕ, λ) gate with three parameters to implement the ***Z*** gate.

1.4 Please write a quantum program in which we use the U3(θ, ϕ, λ) gate with three parameters to implement the ***Y*** gate.

1.5 Please write a quantum program in which we use the U3(θ, ϕ, λ) gate with three parameters to implement the ***S*** gate.

1.6 Please write a quantum program in which we apply the U3(θ, ϕ, λ) gate with three parameters to implement the $\boldsymbol{S}^{+}$ gate.

1.7 Please write a quantum program in which we make use of the U3(θ, ϕ, λ) gate with three parameters to implement the ***T*** gate.

1.8 Please write a quantum program in which we use the U3(θ, ϕ, λ) gate with three parameters to implement the $\boldsymbol{T}^{+}$ gate.

1.9 Please write a quantum program in which we apply the U3(θ, ϕ, λ) gate with three parameters to implement the *identity* gate.

1.10 Please write a quantum program in which we make use of the U3(θ, ϕ, λ) gate with three parameters to implement the U1(λ) gate with one parameter.

1.11 Please write a quantum program in which we use the U3(θ, ϕ, λ) gate with three parameters to implement the U2(ϕ, λ) gate with two parameters.