CHAPTER 3

Qubits, Operators and Measurement

In this chapter we will cover qubits and the core set of operators we use to manipulate the state of qubits.

A *qubit* is a quantum bit. A qubit is similar to a classical bit in that it can take on 0 or 1 as states, but it differs from a bit in that it can also take on a continuous range of values representing a superposition of states. In this text we will use qubit to refer to quantum bits and the word bit to refer to classical bits.

While in general we use two-level qubit systems to build quantum computers we can also choose other types of computing architectures. For example, we could build a QC with *qutrits* which are three-level systems. We can think of these as having states of 0, 1 or 2 or a superposition of these states.

The more general term for such a unit is *qudit*; qubits and qutrits are specific instances of qudits which can be computing units of any number of states. The Siddiqi Lab at UC Berkeley, for example, has designed a qutrit-based QC [34]. In a qutrit system we can represent more states than a qubit system with the same number of computational units.

A qubit system of say 100 qubits can handle 2^{100} states (1.26765E+30), while a qutrit system can handle 3^{100} states (5.15378E+47), a number which is 17 orders of magnitude larger. Put another way, to represent the same number space as a 100 qubit system, we only need ~ 63 qutrits ($\log_3(2^{100})$). Since it is more difficult to build qutrit systems, the mainstream QCs are currently based on qubits. Whether we choose qubits, qutrits or some other

J. D. Hidary, *Quantum Computing: An Applied Approach*,
https://doi.org/10.1007/978-3-030-23922-0_3

qudit number, each of these systems can run any algorithm that the others can, i.e., they can simulate each other.[1]

In QM we represent states as vectors, operators as matrices and we use Dirac notation instead of traditional linear algebra symbols to represent vectors and other abstractions. Chapter 11 contains a review of linear algebra, Dirac notation and other mathematical tools that are crucial for our inquiry in this book. In this chapter we will assume knowledge of these mathematical tools; we encourage the reader to use the math chapters to review these concepts in the context of quantum computing.

Let us begin with the definition of a qubit:

3.1 What is a Qubit?

A physical qubit is a two-level quantum mechanical system. As we will see in the chapter on building quantum computers, there are many ways to construct a physical qubit. We can *represent* a qubit as a two-dimensional complex Hilbert space, $\mathbb{C}^2$. The *state* of the qubit at any given time can be represented by a vector in this complex Hilbert space.

The Hilbert space is equipped with the inner product which allows us to determine the relative position of two vectors representing qubit states. We denote the inner product of vectors $|u\rangle$, $|v\rangle$ as $\langle u|v\rangle$; this will equal 0 if $|u\rangle$ and $|v\rangle$ are orthogonal and 1 if $|u\rangle = |v\rangle$. To represent two or more qubits we can tensor product Hilbert spaces together to represent the combined states of the qubits. As we shall see, we have methods to represent separable states, where the qubits are independent of one another, and entangled states such as a Bell state, where we cannot separate the two qubit states.

We can represent the states $|0\rangle$ and $|1\rangle$ with vectors as shown below. We call these two the computational basis of a two-level system. We can then apply operators in the form of matrices to the vectors in the state space.

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

[1]Note that we could consider the same question in classical systems, i.e., we could have used a 3-state "trit" instead of the bit, but we choose to use bits as there are distinct advantages to the binary system.

3.2 Quantum Operators

In gate-based quantum computers, the operators which we use to evolve the state of the qubits are unitary and therefore reversible. Some of the operators are unitary, reversible *and* involutive (i.e., they are their own inverses); others are not involutive. A measurable quantity, or observable, is a Hermitian operator; thus the measurement in a quantum computer outputs real values from the system. We use the terms operators and gates interchangeably.

In addition to an inner product of two vectors, linear algebra gives us the outer product. This is when we take two vectors and form a matrix (whereas an inner product gives us a scalar). If we take the outer product $|0\rangle\langle 0|$, for example, we produce the following operator

$$|0\rangle\langle 0| = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

Similarly, we can take the outer product of the other three combinations to produce these matrices

$$|0\rangle\langle 1| = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

$$|1\rangle\langle 0| = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$$

$$|1\rangle\langle 1| = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

We can take the sum of two of these matrices to form a unitary matrix, like so

$$|0\rangle\langle 1| + |1\rangle\langle 0| = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{3.3}$$

This, in fact, is the X or NOT operator which we will encounter shortly in this chapter.

We have established that a qubit can be in one of the computational basis states of 0 or 1 or in a superposition of these two states. How can we represent the superposition of multiple states? We can do so as a linear combination of the computational bases of the state space.

3.4 Representing Superposition of States

We represent a superposition of states as the linear combination of computational bases of the state space. Each term in the superposition has a complex coefficient or *amplitude*.

Using the two computational basis vectors in the case of a single qubit, two examples of superpositions of states are

$$|+\rangle := \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)$$

and

$$|-\rangle := \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right)$$

These two states differ by a minus sign on the $|1\rangle$ state. More formally, we call this difference a *relative phase*. The term phase has numerous meanings in physics — in this context, it refers to an angle. The minus sign is related to the angle π (180°) by Euler's identity[2]

$$e^{i\pi} = -1$$

Relative phases are of fundamental importance for quantum algorithms in that they allow for *constructive interference* and *destructive interference*. For example, if we evaluate the sum of the above states, we obtain

$$\frac{1}{\sqrt{2}}(|+\rangle + |-\rangle) = \frac{1}{2}(|0\rangle + |1\rangle) + \frac{1}{2}(|0\rangle - |1\rangle) = |0\rangle$$

Here, we say that the amplitudes of the $|1\rangle$ state interfere *destructively* — the differing relative phases cause them to sum to zero. On the other hand, the amplitudes of the $|0\rangle$ state interfere *constructively* — they have the same sign (relative phase), so they do not sum to zero, and thus we are left with the state $|0\rangle$ as the result.

We can also consider subtracting the two superposition states. We leave it to the reader to verify that

$$\frac{1}{\sqrt{2}}(|-\rangle - |+\rangle) = -|1\rangle$$

[2]For more on Euler's identity, refer to chapter 13.

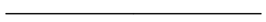
Here, the amplitudes of the $|0\rangle$ state interfere destructively while the amplitudes of the $|1\rangle$ state interfere constructively. In this example, we do not end up with the $|1\rangle$ state exactly — it is multiplied by a minus sign. As we saw above, we can interpret this minus sign as an angle (or phase) $e^{i\pi}$. Here, it is applied to the entire state, not just one term in the superposition. We refer to this type of phase as a *global phase*.

While it is true that the $-|1\rangle$ state is not exactly the $|1\rangle$ state, we will see in future chapters that a global phase change has no impact on quantum measurements. That is, the measurement statistics obtained by measuring the $-|1\rangle$ state and the $|1\rangle$ state are exactly identical. In this case, we often say that the two states are *equal, up to global phase*.

Quantum Circuit Diagrams

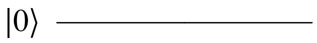
We use circuit diagrams to depict quantum circuits. We construct and read these diagrams from left to right; we can think of circuit diagrams like a staff of music which we read in the same direction. Barenco et al. set forth a number of the foundational operators that we use today in QC [22]. Fredkin and Toffoli [217, 89] added to this set with two ternary operators.

We begin the construction of a quantum circuit diagram with the circuit wire which we represent as a line

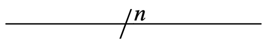


A line with no operator on it implies that the qubit remains in the state in which it was previously prepared. This means that we are relying on the quantum computer to maintain the state of the qubit.

We denote the initial prepared state with a ket and label on the left of the wire

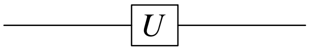


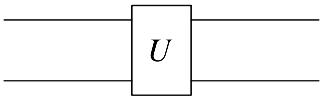
We denote n number of qubits prepared in that state with a slash n symbol across the wire.



3.1 *Quantum Operators*

Let us now turn to the set of commonly used quantum operators. We denote a single-qubit operator with a box containing the letter representing that operator straddling the line. We denote a binary gate with an operator box spanning two quantum wires and spanning three wires for a ternary operator, etc. Note that we could have chosen a different set of operators to accomplish universal quantum computation; the set of operators chosen is arbitrary and is sufficient as long as it meets the test of universality which we will cover later in this chapter. Here are representations for unary and binary operators





Unary Operators

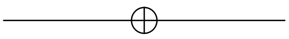
Let us now cover the set of one-qubit, or unary, quantum operators. The first three operators we will examine are the Pauli matrices. These three matrices along with the identity matrix and all of their ± 1 and $\pm i$ multiples constitute what is known as the *Pauli group*. First, we have X, which is the *NOT* operator (also known as the bit flip operator and can be referred to as σ_x)

$$X := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

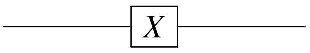
If we apply X to $|0\rangle$ then we have

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0+0 \\ 1+0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle$$

We can represent the initial state of a qubit and the operators we apply to them with a circuit diagram. We use the following symbol to represent the X operator in circuit diagrams



This is different from the convention of the operator name in a box, which one may also encounter in circuit diagrams



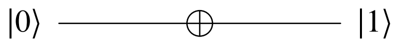
As we have seen, we can represent the X operator in ket notation as

$$X := |0\rangle\langle 1| + |1\rangle\langle 0|$$

and the application of the X operator like so:

$$X\,|j\rangle = |j \oplus 1\rangle$$

where $j \in \{0, 1\}$. Here the $\oplus$ operation denotes addition modulo-2, and $j \oplus 1$ is equivalent to the *NOT* operation. So if we start with the qubit in state $|0\rangle$ and apply *NOT* then we have



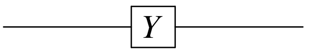
Next we have the Y operator, also denoted σ_y, which rotates the state vector about the y axis[3].

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

So that if we apply it to the $|1\rangle$ state we have

$$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 - i \\ 0 + 0 \end{pmatrix} = \begin{pmatrix} -i \\ 0 \end{pmatrix} = -i\,|0\rangle$$

The circuit diagram for the Y operator is



And the Z operator, also denoted σ_z, which rotates the state vector about the z axis (also called the phase flip operator since it flips it by π radians or 180 degrees)

$$Z := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

If we apply Z to the computational basis state we have

$$Z\,|j\rangle = (-1)^j\,|j\rangle$$

or to show this in matrix form for the special case $j = 0$

[3]The x, y and z axes in this section refer to representation of the qubit's state on a Bloch sphere, which we will cover later in this chapter.

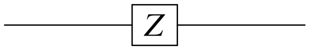
$$\begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1+0\\0+0\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix} = (-1)^0\,|0\rangle = |0\rangle$$

For the case where $j = 1$ we have

$$\begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0+0\\0-1\end{pmatrix} = \begin{pmatrix}0\\-1\end{pmatrix} = (-1)^1\,|1\rangle = -\,|1\rangle$$

Note that we can multiply the bit-flip operator X by the phase-flip operator Z to yield the Y operator with a global phase shift of i. That is, $Y = iXZ$.

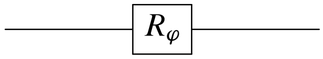
The circuit diagram for the Z operator is



Next we turn to the more general phase shift operator. When we apply this operator we leave the state $|0\rangle$ as is and we take the state $|1\rangle$ and rotate it by the angle (or phase) denoted by φ, as specified in the matrix

$$R_\varphi := \begin{pmatrix}1 & 0\\ 0 & e^{i\varphi}\end{pmatrix}$$

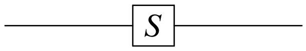
So the Pauli Z operator is just a special case of R_φ where $\varphi = \pi$. Let's recall that $e^{i\pi} = -1$ by Euler's identity (see chapter 13) so we can replace $e^{i\pi}$ with -1 in the Z matrix. The circuit diagram for the R operator is



Let's discuss two additional phase shift operators that are special cases of the R_φ matrix. First, the S operator, where $\varphi = \pi/2$

$$S := \begin{pmatrix}1 & 0\\ 0 & i\end{pmatrix}$$

The S operator thus rotates the state about the z-axis by 90°. The circuit diagram for the S operator is



Next let's turn to the T operator which rotates the state about the z-axis by 45°. If we give φ the value of $\pi/4$ then[4]

[4]Note that the T gate is also known as the $\pi/8$ gate, since if we factor out $e^{i\pi/8}$, the diagonal components each have $|\varphi| = \pi/8$, but this is of course the same operator.

$$T := \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$$

Note that $S = T^2$. In other words, if we apply the T matrix to the vector representing the state and then apply T again to the resulting vector from the first operation we have accomplished the same result as applying S once ($45° + 45° = 90°$). The circuit diagram for the T operator is

$$\text{———}\boxed{T}\text{———}$$

Now let's turn to the Hadamard operator. **This operator is crucial in quantum computing since it enables us to take a qubit from a definite computational basis state into a superposition of two states.** The Hadamard matrix is

$$H := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

It was actually the mathematician John Sylvester who developed this matrix, but we name it after Jacques Hadamard (see Stigler's law of eponymy which, of course, was probably conceived by Merton and others). The circuit diagram for the H operator is

$$\text{———}\boxed{H}\text{———}$$

If we apply the Hadamard to state $|0\rangle$ we obtain

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1+0 \\ 1+0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

And to state $|1\rangle$ we have

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 0+1 \\ 0-1 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

So we can see that the H operator takes a computational basis state and projects it into a superposition of states $(|0\rangle + |1\rangle)/\sqrt{2}$ or $(|0\rangle - |1\rangle)/\sqrt{2}$, depending on the initial state.

What is the $\sqrt{2}$ doing in this state? Let us recall the Born rule that the square of the modulus of the amplitudes of a quantum state is the probability of that state. Furthermore, for all amplitudes α, β, etc. of a state

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

That is, the probabilities must sum to one since one of the states will emerge from the measurement.

Before moving on to the binary operators, let us define the identity operator and then determine which operators can be expressed as sequences of other operators. The identity operator is simply the matrix which maintains the current state of the qubit. So for one qubit we can use

$$I := \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Having covered the set of unary operators, we can show the following identities:

$$\begin{aligned} HXH &= Z \\ HZH &= X \\ HYH &= -Y \\ H^\dagger &= H \\ H^2 &= I \end{aligned}$$

Please see chapter 14 for a list of additional identities.

Binary Operators

Let us now consider two qubit, or *binary*, operators. In a two-qubit system, by convention, we use the following computational basis states:

$$|00\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad |01\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \quad |10\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \quad |11\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

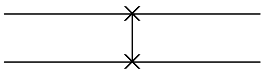
Let us first discuss the *SWAP* operator. The *SWAP* takes the state $|01\rangle$ to $|10\rangle$ and, of course, $|10\rangle$ to $|01\rangle$. We can represent this operator with the following matrix

$$SWAP := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

And apply it to a 4-d vector representing the state $|01\rangle$ as follows

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0+0+0+0 \\ 0+0+0+0 \\ 0+1+0+0 \\ 0+0+0+0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = |10\rangle$$

Satisfy yourself that this operator applied to one of the two-qubit computational basis vectors will have the desired result. For the circuit diagram of the *SWAP* operator we use



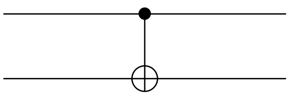
Now we come to a critical operator for quantum computing — controlled-*NOT* (*CNOT*). In this binary operator, we identify the first qubit as the *control qubit* and the second as the *target qubit*. If the control qubit is in state $|0\rangle$ then we do nothing to the target qubit. If, however, the control qubit is in state $|1\rangle$ then we apply the *NOT* operator (X) to the target qubit. **We use the *CNOT* gate to entangle two qubits in the QC**. We can represent *CNOT* with the following matrix

$$CNOT := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

So, for example, we compute the action of *CNOT* on the state $|10\rangle$ as follows

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0+0+0+0 \\ 0+0+0+0 \\ 0+0+0+0 \\ 0+0+1+0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = |11\rangle$$

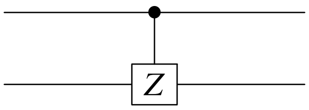
And for the circuit diagram, we depict the *CNOT* in this way



Here is an identity connecting the *SWAP* and *CNOT* operators:

$$SWAP_{ij} = CNOT_{ij} CNOT_{ji} CNOT_{ij}$$

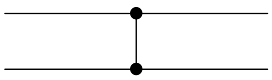
Now let's turn to another control operator: *CZ*. Here we have a control qubit and a target qubit just as with *CNOT*; however, in this operation if the control qubit is in state $|1\rangle$ then we will apply the Z operator to the target qubit. We can represent the *CZ* operator in a circuit diagram as



and as a matrix

$$CZ := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

We can represent the *CZ* operator in circuit diagrams as



Note that unlike the *CNOT* gate, the *CZ* gate is symmetric: we can choose either qubit as the control or the target and we will end up with the same result. This is why we can represent the *CZ* gate with a dot on both circuit wires.

Ternary Operators

We have discussed both unary and binary operators. Now let's consider the ternary or 3-qubit operators. First, we have the Toffoli operator, also known as the *CCNOT* gate [217]. Just as in the *CNOT* operator, we have control and target qubits. In this case, the first two qubits are control and the third is the target qubit. Both control qubits have to be in state $|1\rangle$ for us to modify the target qubit. Another way of thinking about this is that the first two qubits (x and y) have to satisfy the Boolean *AND* function — if that equals TRUE then we apply *NOT* to the target qubit, z. We can represent this action as

$$(x, y, z) \mapsto (x, y, (z \oplus xy))$$

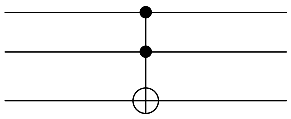
or, as a matrix,

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

As an example, we apply this gate to the state $|110\rangle$

$$\begin{pmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1\\0&0&0&0&0&0&1&0\end{pmatrix}\begin{pmatrix}0\\0\\0\\0\\0\\0\\1\\0\end{pmatrix}=\begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\1\end{pmatrix}=|111\rangle$$

In circuit diagrams, we use the following to denote the Toffoli



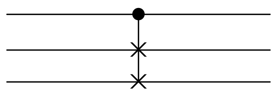
Next, let's consider the Fredkin gate, also known as the *CSWAP* gate [89]. When we apply this operator, the first qubit is the control and the other two are the target qubits. If the first qubit is in state $|0\rangle$ we do nothing and if it is in state $|1\rangle$ then we *SWAP* the other two qubits with each other. The matrix representing this operations is

$$\begin{pmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1\end{pmatrix}$$

For example, the Fredkin gate applied to $|110\rangle$ gives

$$\begin{pmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1\end{pmatrix}\begin{pmatrix}0\\0\\0\\0\\0\\0\\1\\0\end{pmatrix}=\begin{pmatrix}0\\0\\0\\0\\0\\1\\0\\0\end{pmatrix}=|101\rangle$$

In circuit diagrams we use this symbol for the Fredkin operator



3.2 *Comparison with Classical Gates*

In classical computing we have a set of commonly used gates: *AND*, *NOT*, *OR*, *NAND*, *XOR*, *FANOUT*, etc. We can use combinations of these gates to perform any computation in classical computing. A classical computer that can run these gates is Turing-complete or universal. In fact, we can prove that the *NAND* gate alone is sufficient to construct all other classical operators [198]. We can construct classical circuits with these basic building blocks such as a circuit for the half-adder

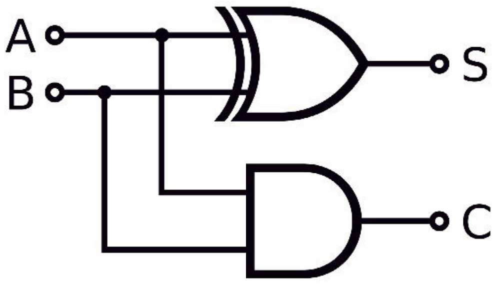


Figure 3.1: Half-adder in classical computing Source: Wikimedia

We can then build a full-adder from those elements:

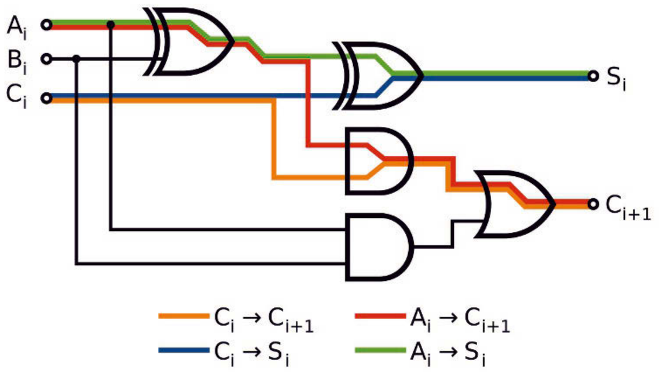


Figure 3.2: Full-adder in classical computing Source: Wikimedia

Neither *AND*, *OR*, *XOR*, *NAND* or *FANOUT* can be used in quantum computing. The *AND*, *OR*, *XOR* and *NAND* gates are not reversible. The

FANOUT gate would not be allowed in quantum computing since it involves the duplication, or cloning, of a state; this would violate the no-cloning theorem. Of the primary classical gates, only the *NOT* operator can be used in the quantum computing regime as it is reversible and does not involve cloning.

3.3 *Universality of Quantum Operators*

If *NAND* is universal for classical computing, is there such a gate or set of gates that are universal for quantum computing? In fact, there are several combinations of unary and binary operators that lead to universality. No set of unary gates on their own can achieve universal QC. Two of the gate sets that yield universality are:

1. The Toffoli gate is universal for QC when paired with a basis-changing unary operator with real coefficients (such as H) [199].
2. Another set of gates which is universal is $\{CNOT, T, H\}$ [44, 161].

3.4 *Gottesman-Knill and Solovay-Kitaev*

The Gottesman-Knill theorem states that circuits built with only Clifford gates can be simulated efficiently on classical computers assuming the following conditions:

- state preparation in the computational basis
- measurements in the standard basis
- any classical control conditioned on the measurement outcomes

The Clifford group of operators is generated by the set $C = \{CNOT, S, H\}$ [96] [168].

A further theorem that is worth considering at this junction is that of Solovay-Kitaev. This theorem states that if a set of single-qubit quantum gates generates a dense subset of $SU(2)$, which is the special unitary group of unitary matrices which are 2 x 2, then that set is guaranteed to fill $SU(2)$ quickly, i.e., it is possible to obtain good approximations to any desired gate using surprisingly short sequences of gates from the given generating set [62]. The theorem generalizes to multi-qubit gates and for operators from SU(d) [62].

A simplified version of this statement is that all finite universal gate sets can simulate a given gate set to a degree δ of precision. More precisely, if L is the size of the circuit (i.e., the number of gates) then the approximation L' of

L has a bounded number of gates; this can be specified in big-O notation by

$$L' = O\left(L \log^4\left(\frac{L}{\delta}\right)\right)$$

If D denotes the depth of the circuit, i.e., the number of computational steps, then the approximation D' of D has a bounded depth specified in big-O notation by

$$D' = O\left(L \log^4\left(\frac{D}{\delta}\right)\right)$$

So, these expressions demonstrate that the simulation is quite efficient and better than polynomial time.

3.5 *The Bloch Sphere*

There are several ways to represent the state of a qubit:

1. We can write out the state in Dirac notation. For example, if we have a qubit that is prepared in state $|0\rangle$ and then apply the X operator, we will then find the qubit in state $|1\rangle$ (assuming no outside noise)

$$X|0\rangle \rightarrow |1\rangle$$

2. We can use the Bloch sphere to represent the state of a single qubit. Any state in a quantum computation can be represented as a vector that begins at the origin and terminates on the surface of the unit Bloch sphere. By applying unitary operators to the state vectors, we can move the state around the sphere. We take as convention that the two antipodes of the sphere are $|0\rangle$ on the top of the sphere and $|1\rangle$ on the bottom.

As we can see in Figure 3.3, one of the advantages of visualization with the Bloch sphere is that we can represent superposition states such as

$$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

as we see at the X axis. We can also differentiate between states that contain different phases as is shown in the states along the X and Y axes.

Let us return to computational universality which we treated above. Now that we have introduced the Bloch sphere, another way to think about a set of gates that satisfies universal computation is one which enables us to reach any point on the Bloch sphere.

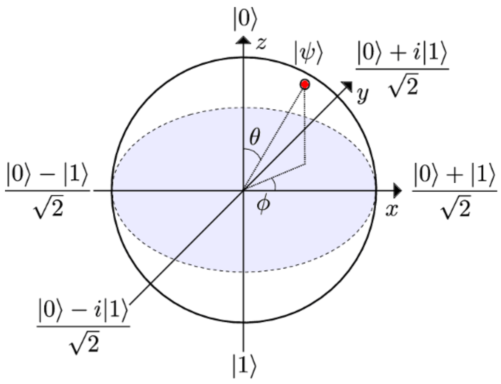

$|0\rangle$
z
$|\psi\rangle$
$\frac{|0\rangle + i|1\rangle}{\sqrt{2}}$
y
θ
$\frac{|0\rangle - |1\rangle}{\sqrt{2}}$
$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$
ϕ
x
$\frac{|0\rangle - i|1\rangle}{\sqrt{2}}$
$|1\rangle$

Figure 3.3: The Bloch sphere Source: [93]

For interactive visualizations of qubits on the Bloch sphere, see the book's online website. Now that we have covered the main unitary operators which we use in QC, let's turn to the measurement of the QC's state.

3.6 *The Measurement Postulate*

Measurement in classical physics is a seemingly straightforward process. The act of measurement is assumed to have no effect on the item that we are measuring. Furthermore, we have the ability to measure one property of a system, get a reading, then measure another property and be confident that the first property measured still retains its observed value. Not so in quantum mechanics; in this regime, the act of measurement has a profound effect on the observation.

Building on the principles of quantum mechanics, we can state the measurement postulate as:

3.5 Measurement Postulate

Every measurable physical quantity, o, is described by a corresponding Hermitian operator, O, acting on the state Ψ.

According to this postulate, there exists a Hermitian operator, which we call an observable, associated with each property. So, for example, the observable $\hat{x}$ is associated with the position of a particle.

We recall that a Hermitian operator is equal to its adjoint (which is its complex conjugate transpose). If O is Hermitian then we can state that $O = O^\dagger$ (see chapter 12 for more discussion on Hermitians).

Hermitian operators have the desirable property that their eigenvalues are guaranteed to be real numbers. When measuring a physical system for properties such as momentum or position we need to specify a real number.

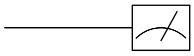
Each possible outcome of the measurement is an eigenvalue, λ, of the observable and is characterized by $|P\,|\psi\rangle\,|^2$ where P is the projector onto the eigenspace of the observable. Since we normalize the vectors, the state after measurement is represented by a unit eigenvector of the observable with eigenvalue λ_i.

We discussed earlier how a system can be in a superposition of two or more states. We can represent this as a linear combination of the orthonormal basis vectors. For example,

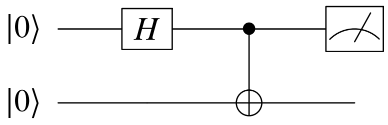
$$|\Psi\rangle = \frac{1}{\sqrt{2}}\,|0\rangle + \frac{1}{\sqrt{2}}\,|1\rangle$$

This leads to the question of how we can set up the measurement in such a way as to obtain the output we require. This in turn is dependent on the amplitudes of each state since, as discussed previously, the square of the modulus of the amplitude is the probability of that state appearing as the output upon measurement (Born's rule).

We can represent a measurement in a quantum circuit like this:



Now that we have our unitary and measurement operators, we will construct some basic quantum circuits. Let's figure out what this one does:



We start with two qubits, let's call them $q0$ and $q1$, each prepared in state $|0\rangle$. We then apply the Hadamard operator to $q0$ which puts it into the superposition of states

$$|q0\rangle = \frac{1}{\sqrt{2}}\,|0\rangle + \frac{1}{\sqrt{2}}\,|1\rangle$$

We then apply a *CNOT* across $q0$ and $q1$. This entangles the two qubits so we now have the combined, non-separable state of the two qubits of

$$\frac{1}{\sqrt{2}}\,|00\rangle + \frac{1}{\sqrt{2}}\,|11\rangle$$

We have thus created a Bell state, or an EPR pair. We then measure $q0$ with a 50/50 chance of finding a 0 or a 1 value for the real-valued output.

3.7 *Computation-in-Place*

We can depict the circuit in a 3-dimensional manner as in Figure 3.4. Here you can see the qubits beginning in a prepared state followed by a Hadamard step to put them all into a superposition; then they undergo a series of one and two-qubit operations such as X, Y, T and CZ. Measurement is then applied in the final step.

In this diagram, we see a crucial difference between classical and quantum computing.

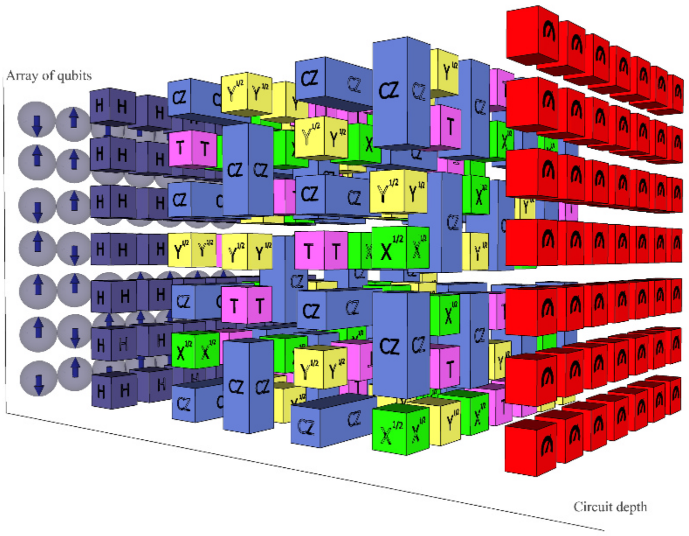


Figure 3.4: 3-D Quantum circuit diagram Source: Google

3.6 Computation-in-Place

In most forms of gate-based quantum computing, the information is represented in the states of the qubits as they evolve over time with the successive application of unitary operators.

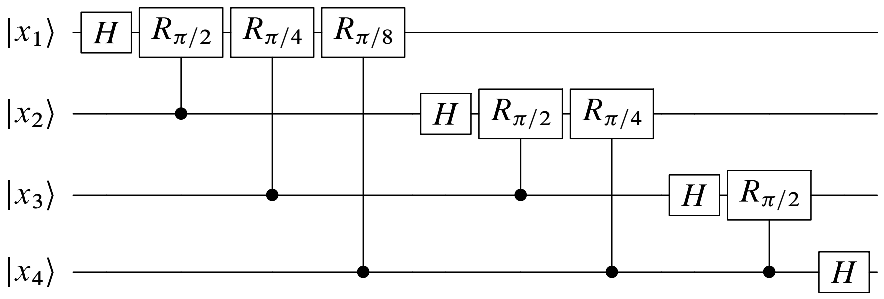
Computation-in-place is in stark contrast to classical computing, where we shuttle data around the processor to various memory and calculation registers. In most forms of quantum computers, all processing takes place on the qubits themselves. After measurement in a QC we output real-valued bits which we can share with the CPU that is controlling the quantum processor and, if necessary, incorporate in further processing on classical machines.

We now come to one of the key questions in quantum computing: if measurement is based on the modulus squared of the amplitude of each qubit state, then how can we pre-determine which of the qubits will be the output? Deutsch, Jozsa, Bernstein, Vazirani, Shor and others realized that we can influence the output by setting up the amplitudes prior to measurement to favor the output we need for that computational task.

One method for accomplishing this goal is the quantum Fourier transform (QFT – not to be confused with quantum field theory!). By applying a QFT across all qubits prior to measurement, we can obtain phase information upon measurement instead of amplitude information.

The QFT is an efficient process on a quantum computer: the discrete Fourier transform on 2^n amplitudes only takes $O(n^2)$ applications of Hadamard and phase-shift operators (where n is the number of qubits). We will cover the QFT in greater detail later in this text.

Here we display the circuit for QFT for $n = 4$:



Before we turn our attention to quantum hardware, let's delve into computational complexity in the next chapter. This will give us the foundation to understand which sorts of problems are appropriate for a quantum computer.