Chapter 1

Cbits and Qbits

1.1 What is a quantum computer?

It is tempting to say that a quantum computer is one whose operation is governed by the laws of quantum mechanics. But since the laws of quantum mechanics govern the behavior of all physical phenomena, this temptation must be resisted. Your laptop operates under the laws of quantum mechanics, but it is not a quantum computer. A quantum computer is one whose operation exploits certain very special transformations of its internal state, whose description is the primary subject of this book. The laws of quantum mechanics allow these peculiar transformations to take place under very carefully controlled conditions.

In a quantum computer the physical systems that encode the individual logical bits must have no physical interactions whatever that are not under the complete control of the program. All other interactions, however irrelevant they might be in an ordinary computer – which we shall call *classical* – introduce potentially catastrophic disruptions into the operation of a quantum computer. Such damaging encounters can include interactions with the external environment, such as air molecules bouncing off the physical systems that represent bits, or the absorption of minute amounts of ambient radiant thermal energy. There can even be disruptive interactions between the computationally relevant features of the physical systems that represent bits and other features of those same systems that are associated with computationally irrelevant aspects of their internal structure. Such destructive interactions, between what matters for the computation and what does not, result in *decoherence*, which is fatal to a quantum computation.

To avoid decoherence individual bits cannot in general be encoded in physical systems of macroscopic size, because such systems (except under very special circumstances) cannot be isolated from their own irrelevant internal properties. Such isolation can be achieved if the bits are encoded in a small number of states of a system of atomic size, where extra internal features do not matter, either because they do not exist, or because they require unavailably high energies to come into play. Such atomic-scale systems must also be decoupled from their surroundings except for the completely controlled interactions that are associated with the computational process itself.

Two things keep the situation from being hopeless. First, because the separation between the discrete energy levels of a system on the atomic scale can be enormously larger than the separation between the levels of a large system, the dynamical isolation of an atomic system is easier to achieve. It can take a substantial kick to knock an atom out of its state of lowest energy. The second reason for hope is the discovery that errors induced by extraneous interactions can actually be corrected if they occur at a sufficiently low rate. While error correction is routine for bits represented by classical systems, quantum error correction is constrained by the formidable requirement that it be done without knowing either the original or the corrupted state of the physical systems that represent the bits. Remarkably, this turns out to be possible.

Although the situation is therefore not hopeless, the practical difficulties in the way of achieving useful quantum computation are enormous. Only a rash person would declare that there will be no useful quantum computers by the year 2050, but only a rash person would predict that there will be. Never mind. Whether or not it will ever become a practical technology, there is a beauty to the theory of quantum computation that gives it a powerful appeal as a lovely branch of mathematics, and as a strange generalization of the paradigm of classical computer science, which had completely escaped the attention of computer scientists until the 1980s. The new paradigm demonstrates that the theory of computation can depend profoundly on the physics of the devices that carry it out. Quantum computation is also a valuable source of examples that illustrate and illuminate, in novel ways, the mysterious phenomena that quantum behavior can give rise to.

For computer scientists the most striking thing about quantum computation is that a quantum computer can be vastly more efficient than anything ever imagined in the classical theory of computational complexity, for certain computational tasks of considerable practical interest. The time it takes the quantum computer to accomplish such tasks scales up much more slowly with the size of the input than it does in any classical computer. Much of this book is devoted to examining the most celebrated examples of this speed-up.

This exposition of quantum computation begins with an introduction to quantum mechanics, specially tailored for this particular application. The quantum-mechanics lessons are designed to give you, as efficiently as possible, the conceptual tools needed to delve into quantum computation. This is done by restating the rules of quantum mechanics, not as the remarkable revision of classical Newtonian mechanics required to account for the behavior of matter at the atomic and subatomic levels, but as a curious generalization of rules describing an ordinary classical digital computer. By focusing exclusively on how quantum mechanics enlarges the possibilities for the physical manipulation of digital information, it is possible to characterize how the quantum theory works in an elementary and quite concise way, which is nevertheless rigorous and complete for this special area of application.

While I assume no prior familiarity with quantum physics (or any other kind of physics), I do assume familiarity with elementary linear algebra and, in particular, with the theory of finite-dimensional vector spaces over the complex numbers. Appendix A summarizes the relevant linear algebra. It is worth examining even if you are well acquainted with the mathematics of such vector spaces, since it also provides a compact summary of the mathematically unconventional language – *Dirac notation* – in which linear algebra is couched in all treatments of quantum computation. Dirac notation is also developed, more informally, throughout the rest of this chapter.

1.2 Cbits and their states

We begin with an offbeat formulation of what an ordinary classical computer does. I frame the elementary remarks that follow in a language which may look artificial and cumbersome, but is designed to accommodate the richer variety of things that a computer can do if it takes full advantage of the possibilities made available by the quantum-mechanical behavior of its constituent parts. By introducing and applying the unfamiliar nomenclature and notation of quantum mechanics in a familiar classical context, I hope to make a little less strange its subsequent extension to the broader quantum setting.

A classical computer operates on strings of zeros and ones, such as 110010111011000, converting them into other such strings. Each position in such a string is called a *bit*, and it contains either a 0 or a 1. To represent such collections of bits the computer must contain a corresponding collection of physical systems, each of which can exist in two unambiguously distinguishable physical states, associated with the value (0 or 1) of the abstract bit that the physical system represents. Such a physical system could be, for example, a switch that could be open (0) or shut (1), or a magnet whose magnetization could be oriented in two different directions, "up" (0) or "down" (1).

It is a common practice in quantum computer science to use the same term "bit" to describe the two-state classical system that represents the value of the abstract bit. But this use of a single term to characterize both the abstract bit (0 or 1) and the physical system whose two states represent the two values is a potential source of confusion. To avoid such confusion, I shall use the term *Cbit* ("C" for "classical") to describe the two-state classical physical system and *Qbit* to describe its quantum generalization. This terminology is inspired by Paul Dirac's early use of *c-number* and *q-number* to describe classical quantities and their quantum-mechanical generalizations. "Cbit" and

"Qbit" are preferable to "c-bit" and "q-bit" because the terms themselves often appear in hyphenated constructions.

Unfortunately the preposterous spelling *qubit* currently holds sway for the quantum system. The term *qubit* was invented and first used in print by the otherwise admirable Benjamin Schumacher.[1] A brief history of the term can be found in the acknowledgments at the end of his paper. Although "qubit" honors the English (German, Italian, . . .) rule that q should be followed by u, it ignores the equally powerful requirement that qu should be followed by a vowel. My guess is that "qubit" has gained acceptance because it visually resembles an obsolete English unit of distance, the homonymic *cubit*. To see its ungainliness with fresh eyes, it suffices to imagine that Dirac had written *qunumber* instead of *q-number*, or that one erased transparencies and cleaned one's ears with *Qutips*.

Because clear distinctions among bits, Cbits, and Qbits are crucial in the introduction to quantum computation that follows, I shall use this currently unfashionable terminology. If you are already addicted to the term *qubit*, please regard *Qbit* as a convenient abbreviation.

To prepare for the extension from Cbits to Qbits, I introduce what may well strike you as a degree of notational overkill in the discussion of Cbits that follows. We shall represent the state of each Cbit as a kind of box, depicted by the symbol $|\ \rangle$, into which we place the value, 0 or 1, represented by that state. Thus the two distinguishable states of a Cbit are represented by the symbols $|0\rangle$ and $|1\rangle$. It is the common practice to call the symbol $|0\rangle$ or $|1\rangle$ itself the *state* of the Cbit, thereby using the same term to refer to both the physical condition of the Cbit and the abstract symbol that represents that physical condition. There is nothing unusual in this. For example one commonly uses the term "position" to refer to the symbol x that represents the physical position of an object. I call this common, if little noted, practice to your attention only because in the quantum case "state" refers *only* to the symbol, there being *no* internal property of the Qbit that the symbol represents. The subtle relation between Qbits and their state symbol will emerge later in this chapter.

Along the same lines, we shall characterize the states of the five Cbits representing 11001, for example, by the symbol

$$|1\rangle|1\rangle|0\rangle|0\rangle|1\rangle, \tag{1.1}$$

and refer to this object as the *state* of all five Cbits. Thus a pair of Cbits can have (or "be in") any of the four possible states

$$|0\rangle|0\rangle,\ \ |0\rangle|1\rangle,\ \ |1\rangle|0\rangle,\ \ |1\rangle|1\rangle, \tag{1.2}$$

1 Benjamin Schumacher, "Quantum coding," *Physical Review* A **51**, 2738–2747 (1995).

three Cbits can be in any of the eight possible states

$$|0\rangle|0\rangle|0\rangle,\ |0\rangle|0\rangle|1\rangle,\ |0\rangle|1\rangle|0\rangle,\ |0\rangle|1\rangle|1\rangle,\ |1\rangle|0\rangle|0\rangle,$$
$$|1\rangle|0\rangle|1\rangle,\ |1\rangle|1\rangle|0\rangle,\ |1\rangle|1\rangle|1\rangle, \tag{1.3}$$

and so on.

As (1.4) already makes evident, when there are many Cbits such products are often much easier to read if one encloses the whole string of zeros and ones in a single bigger box of the form $|\quad\rangle$ rather than having a separate box for each Cbit:

$$|000\rangle,\ |001\rangle,\ |010\rangle,\ |011\rangle,\ |100\rangle,\ |101\rangle,\ |110\rangle,\ |111\rangle. \tag{1.4}$$

We shall freely move between these two equivalent ways of expressing the state of several Cbits that represent a string of bits, boxing the whole string or boxing each individual bit. Whether the form (1.3) or (1.4) is to be preferred depends on the context.

There is also a third form, which is useful when we regard the zeros and ones as constituting the binary expansion of an integer. We can then replace the representations of the 3-Cbit states in (1.4) by the even shorter forms

$$|0\rangle,\ |1\rangle,\ |2\rangle,\ |3\rangle,\ |4\rangle,\ |5\rangle,\ |6\rangle,\ |7\rangle. \tag{1.5}$$

Note that, unlike the forms (1.3) and (1.4), the form (1.5) is ambiguous, unless we are told that these symbols express states of three Cbits. If we are not told, then there is no way of telling, for example, whether $|3\rangle$ represents the 2-Cbit state $|11\rangle$, the 3-Cbit state $|011\rangle$, or the 4-Cbit state $|0011\rangle$, etc. This ambiguity can be removed, when necessary, by adding a subscript making the number of Cbits explicit:

$$|0\rangle_3,\ |1\rangle_3,\ |2\rangle_3,\ |3\rangle_3,\ |4\rangle_3,\ |5\rangle_3,\ |6\rangle_3,\ |7\rangle_3. \tag{1.6}$$

Be warned, however, that, when there is no need to emphasize how many Cbits $|x\rangle$ represents, it can be useful to use such subscripts for other purposes. If, for example, Alice and Bob each possess a single Cbit it can be convenient to describe the state of Alice's Cbit (if it has the value 1) by $|1\rangle_a$, Bob's (if it has the value 0) by $|0\rangle_b$, and the joint state of the two by $|1\rangle_a|0\rangle_b$ or $|10\rangle_{ab}$.

Dirac introduced the $|\ \rangle$ notation (known as Dirac notation) in the early days of the quantum theory, as a useful way to write and manipulate *vectors*. For silly reasons he called such vectors *kets*, a terminology that has survived to this day. In Dirac notation you can put into the box $|\ \rangle$ anything that serves to specify what the vector is. If, for example, we were talking about displacement vectors in ordinary three-dimensional space, we could have a vector

$$|5\text{ horizontal centimeters northeast}\rangle. \tag{1.7}$$

In using Dirac notation to express the state of a Cbit, or a collection of Cbits, I'm suggesting that there might be some utility in thinking of the states as vectors. Is there? Well, in the case of Cbits, not very much, but maybe a little. We now explore this way of thinking about Cbit states, because when we come to the generalization to Qbits, it becomes absolutely essential to consider them to be vectors – so much so that the term *state* is often taken to be synonymous with *vector* (or, more precisely, "vector that represents the state").

We shall briefly explore what one can do with Cbits when one takes the two states $|0\rangle$ and $|1\rangle$ of a single Cbit to be represented by two *orthogonal unit vectors in a two-dimensional space.* While this is little more than a curious and unnecessarily elaborate way of describing Cbits, it is fundamental and unavoidable in dealing with Qbits. Playing unfamiliar and somewhat silly games with Cbits will enable you to become acquainted with much of the quantum-mechanical formalism in a familiar setting.

If you prefer your vectors to be expressed in terms of components, note that we can represent the two orthogonal states of a single Cbit, $|0\rangle$ and $|1\rangle$, as column vectors

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{1.8}$$

In the case of two Cbits the vector space is four-dimensional, with an orthonormal basis

$$|00\rangle,\ |01\rangle,\ |10\rangle,\ |11\rangle. \tag{1.9}$$

The alternative notation for this basis,

$$|0\rangle|0\rangle,\ |0\rangle|1\rangle,\ |1\rangle|0\rangle,\ |1\rangle|1\rangle, \tag{1.10}$$

is deliberately designed to suggest multiplication, since it is, in fact, a short-hand notation for the *tensor product* of the two single-Cbit 2-vectors, written in more formal mathematical notation as

$$|0\rangle \otimes |0\rangle,\ \ |0\rangle \otimes |1\rangle,\ \ |1\rangle \otimes |0\rangle,\ \ |1\rangle \otimes |1\rangle. \tag{1.11}$$

In terms of components, the tensor product $\mathbf{a} \otimes \mathbf{b}$ of an M-component vector $\mathbf{a}$ with components a_μ and an N-component vector $\mathbf{b}$ with components b_ν is the (MN)-component vector with components indexed by all the MN possible pairs of indices (μ, ν), whose (μ, ν)th component is just the product $a_\mu b_\nu$. A broader view can be found in the extended review of vector-space concepts in Appendix A. I shall freely move back and forth between the various ways (1.9)–(1.11) of writing the tensor product and their generalizations to multi-Cbit states, using in each case a form that makes the content clearest.

Once one agrees to regard the two 1-Cbit states as orthogonal unit vectors, the tensor product is indeed the natural way to represent multi-Cbit states, since it leads to the obvious multi-Cbit generalization of the representation (1.8) of 1-Cbit states as column vectors. If we express the states $|0\rangle$ and $|1\rangle$ of each single Cbit as column vectors, then we can get the column vector describing a multi-Cbit state by repeatedly applying the rule for the components of the tensor product of two vectors. The result is illustrated here for a three-fold tensor product:

$$\begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \otimes \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \otimes \begin{pmatrix} z_0 \\ z_1 \end{pmatrix} = \begin{pmatrix} x_0y_0z_0 \\ x_0y_0z_1 \\ x_0y_1z_0 \\ x_0y_1z_1 \\ x_1y_0z_0 \\ x_1y_0z_1 \\ x_1y_1z_0 \\ x_1y_1z_1 \end{pmatrix}. \tag{1.12}$$

On applying this, for example, to the case $|5\rangle_3$, we have

$$|5\rangle_3 = |101\rangle = |1\rangle|0\rangle|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}. \tag{1.13}$$

If we label the vertical components of the 8-vector on the right 0, 1, . . ., 7, from the top down, then the single nonzero component is the 1 in position 5 – precisely the position specified by the state vector in its form on the left of (1.13). This is indeed the obvious multi-Cbit generalization of the column-vector form (1.8) for 1-Cbit states.

This is quite general: the tensor-product structure of multi-Cbit states is just what one needs in order for the 2^n-dimensional column vector representing the state $|m\rangle_n$ to have all its entries zero except for a single 1 in the mth position down from the top.

One can turn this development upside down, taking as one's starting point the simple rule that an integer x in the range $0 \le x < N$ is represented by one of N orthonormal vectors in an N-dimensional space. One can then pick a basis so that 0 is represented by an N-component column vector $|0\rangle$ that has 0 in every position except for a 1 in the top position, and x is to be represented by an N-component column vector $|x\rangle$ that has 0 in every position except for a 1 in the position x down from the top. It then follows from the nature of the tensor product that if $N = 2^n$ and x has the binary expansion $x = \sum_{j=0}^{n-1} x_j 2^j$, then the column vector $|x\rangle_n$ is the tensor product of the n 2-component column vectors $|x_j\rangle$:

$$|x\rangle_n = |x_{n-1}\rangle \otimes |x_{n-2}\rangle \otimes \cdots \otimes |x_1\rangle \otimes |x_0\rangle. \tag{1.14}$$

In dealing with n-Cbit states of the form (1.14) we shall identify each of the n 1-Cbit states, out of which they are composed, by giving the power of 2 associated with the individual bit that the Cbit represents. Thus the 1-Cbit state on the extreme right of (1.14) represents Cbit 0, the state immediately to its left represents Cbit 1, and so on.

This relation between tensor products of vectors and positional notation for integers is not confined to the binary system. Suppose, for example, one represents a decimal digit $x = 0, 1, \ldots, 9$ as a 10-component column vector $\mathbf{v}^{(x)}$ with all components 0 except for a 1, x positions down from the top. If the n-digit decimal number $X = \sum_{j=0}^{n-1} x_j 10^j$ is represented by the tensor product $\mathbf{V} = \mathbf{v}^{(x_{n-1})} \otimes \mathbf{v}^{(x_{n-2})} \otimes \cdots \otimes \mathbf{v}^{(1)} \otimes \mathbf{v}^{(0)}$, then $\mathbf{V}$ will be a 10^n-component column vector with all components 0 except for a 1, x positions down from the top.

Although the representation of Cbit states by column vectors clearly shows why tensor products give a natural description of multi-Cbit states, for almost all other purposes it is better and much simpler to forget about column vectors and components, and deal directly with the state vectors in their abstract forms (1.3)–(1.6).

1.3 Reversible operations on Cbits

Quantum computers do an important part of their magic through *reversible* operations, which transform the initial state of the Qbits into its final form using only processes whose action can be inverted. There is only a single *irreversible* component to the operation of a quantum computer, called *measurement*, which is the only way to extract useful information from the Qbits after their state has acquired its final form. Although measurement is a nontrivial and crucial part of any quantum computation, in a classical computer the extraction of information from the state of the Cbits is so conceptually straightforward that it is not viewed as an inherent part of the computational process, though it is, of course, a nontrivial concern for those who design digital displays or printers. Because the only computationally relevant operations on a classical computer that can be extended to operations on a quantum computer are reversible, only operations on Cbits that are reversible will be of interest to us here.

In a reversible operation every final state arises from a unique initial state. An example of an irreversible operation is ERASE, which forces a Cbit into the state $|0\rangle$ regardless of whether its initial state is $|0\rangle$ or $|1\rangle$. ERASE is irreversible in the sense that, given only the final state and the fact that it was the output of the operation ERASE, there is no way to recover the initial state.

The only nontrivial reversible operation we can apply to a single Cbit is the NOT operation, denoted by the symbol $\mathbf{X}$, which interchanges the two states $|0\rangle$ and $|1\rangle$:

$$\mathbf{X} : |x\rangle \rightarrow |\tilde{x}\rangle; \quad \tilde{1} = 0, \quad \tilde{0} = 1. \tag{1.15}$$

This is sometimes referred to as *flipping* the Cbit. NOT is reversible because it has an inverse: applying $\mathbf{X}$ a second time brings the state of the Cbit back to its original form:

$$\mathbf{X}^2 = \mathbf{1}, \tag{1.16}$$

where $\mathbf{1}$ is the unit (identity) operator. If we represent the two orthogonal states of the Cbit by the column vectors (1.8), then we can express NOT by a linear operator $\mathbf{X}$ on the two-dimensional vector space, whose action on the column vectors is given by the matrix

$$\mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{1.17}$$

So the two reversible things you can do to a single Cbit – leaving it alone and flipping it – correspond to the two linear operators $\mathbf{X}$ and $\mathbf{1}$,

$$\mathbf{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \tag{1.18}$$

on its two-dimensional vector space.

A pedantic digression: since multiplication by the scalar 1 and action by the unit operator $\mathbf{1}$ achieve the same result, I shall sometimes follow the possibly irritating practice of physicists and not distinguish notationally between them. I shall take similar liberties with the scalar 0, the zero vector $\mathbf{0}$, and the zero operator $\mathbf{0}$.

Possibilities for reversible operations get richer when we go from a single Cbit to a pair of Cbits. The most general reversible operation on two Cbits is any permutation of their four possible states. There are 4! = 24 such operations. Perhaps the simplest nontrivial example is the *swap* (or *exchange*) operator $\mathbf{S}_{ij}$, which simply interchanges the states of Cbits i and j:

$$\mathbf{S}_{10}|xy\rangle = |yx\rangle. \tag{1.19}$$

Since the swap operator $\mathbf{S}_{10}$ interchanges $|01\rangle = |1\rangle_2$ and $|10\rangle = |2\rangle_2$, while leaving $|00\rangle = |0\rangle_2$ and $|11\rangle = |3\rangle_2$ fixed, its matrix in the basis $|0\rangle_2, |1\rangle_2, |2\rangle_2, |3\rangle_2$ is

$$\mathbf{S}_{10} = \mathbf{S}_{01} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{1.20}$$

The 2-Cbit operator whose extension to Qbits plays by far the most important role in quantum computation is the *controlled-NOT* or cNOT operator $\mathbf{C}_{ij}$. If the state of the ith Cbit (the *control Cbit*) is $|0\rangle$, $\mathbf{C}_{ij}$ leaves the state of the jth Cbit (the *target Cbit*) unchanged, but, if the state of the control Cbit is $|1\rangle$, $\mathbf{C}_{ij}$ applies the NOT operator $\mathbf{X}$ to the state of the target Cbit. In either case the state of the control Cbit is left unchanged.

We can summarize this compactly by writing

$$\mathbf{C}_{10}|x\rangle|y\rangle = |x\rangle|y \oplus x\rangle, \qquad \mathbf{C}_{01}|x\rangle|y\rangle = |x \oplus y\rangle|y\rangle, \tag{1.21}$$

where $\oplus$ denotes addition modulo 2:

$$y \oplus 0 = y, \qquad y \oplus 1 = \tilde{y} = 1 - y. \tag{1.22}$$

The modulo-2 sum $x \oplus y$ is also called the "exclusive OR" (or XOR) of x and y.

You can construct SWAP out of three cNOT operations:

$$\mathbf{S}_{ij} = \mathbf{C}_{ij}\mathbf{C}_{ji}\mathbf{C}_{ij}. \tag{1.23}$$

This can easily be verified by repeated applications of (1.21), noting that $x \oplus x = 0$. We note some other ways of showing it below.

To construct the matrix for the cNOT operation in the four-dimensional 2-Cbit space, note that if the control Cbit is on the left then cNOT leaves $|00\rangle = |0\rangle_2$ and $|01\rangle = |1\rangle_2$ fixed and exchanges $|10\rangle = |2\rangle_2$ and $|11\rangle = |3\rangle_2$. Therefore the $4 \otimes 4$ matrix representing $\mathbf{C}_{10}$ is just

$$\mathbf{C}_{10} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{1.24}$$

If the control Cbit is on the right, then the states $|01\rangle = |1\rangle_2$ and $|11\rangle = |3\rangle_2$ are interchanged, and $|00\rangle = |0\rangle_2$ and $|10\rangle = |2\rangle_2$ are fixed, so the matrix representing $\mathbf{C}_{01}$ is

$$\mathbf{C}_{01} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}. \tag{1.25}$$

The construction (1.23) of $\mathbf{S}$ out of cNOT operators also follows from (1.20), (1.24), and (1.25), using matrix multiplication. As a practical matter, it is almost always more efficient to establish operator identities by dealing with them directly as operators, avoiding matrix representations.

A very common kind of 2-Cbit operator consists of the tensor product $\otimes$ of two 1-Cbit operators:

$$(\mathbf{a} \otimes \mathbf{b})|xy\rangle = (\mathbf{a} \otimes \mathbf{b})|x\rangle \otimes |y\rangle = \mathbf{a}|x\rangle \otimes \mathbf{b}|y\rangle, \tag{1.26}$$

from which it follows that

$$(\mathbf{a} \otimes \mathbf{b})(\mathbf{c} \otimes \mathbf{d}) = (\mathbf{ac}) \otimes (\mathbf{bd}). \tag{1.27}$$

This tensor-product notation for operators can become quite ungainly when one is dealing with a large number of Cbits and wants to write a 2-Cbit operator that affects only a particular pair of Cbits. If, for example, the 2-Cbit operator in (1.26) acts only on the second and fourth Cbits from the right in a 6-Cbit state, then the operator on the 6-Cbit state has to be written as

$$\mathbf{1} \otimes \mathbf{1} \otimes \mathbf{a} \otimes \mathbf{1} \otimes \mathbf{b} \otimes \mathbf{1}. \tag{1.28}$$

To avoid such typographical monstrosities, we simplify (1.28) to

$$\mathbf{1} \otimes \mathbf{1} \otimes \mathbf{a} \otimes \mathbf{1} \otimes \mathbf{b} \otimes \mathbf{1} = \mathbf{a}_3\mathbf{b}_1 = \mathbf{b}_1\mathbf{a}_3, \tag{1.29}$$

where the subscript indicates which Cbit the 1-Cbit operator acts on, and it is understood that those Cbit states whose subscripts do not appear remain unmodified – i.e. they are acted on by the unit operator. As noted above, we label each 1-Cbit state by the power of 2 it would represent if the n Cbits were representing an integer: the state on the extreme right is labeled 0, the one to its left, 1, etc. Since the order in which **a** and **b** are written is clearly immaterial if their subscripts specify different 1-Cbit states, the order in which one writes them in (1.29) doesn't matter: 1-Cbit operators that act on different 1-Cbit states commute.

Sometimes we deal with 1-Cbit operators that already have subscripts in their names; under such conditions it is more convenient to indicate which Cbit state the operator acts on by a superscript, enclosed in parentheses to avoid confusion with an exponent: thus $\mathbf{X}^{(2)}$ represents the 1-Cbit operator that flips the third Cbit state from the right, but $\mathbf{X}^2$ represents the square of the flip operator (i.e. the unit operator) without reference to which Cbit state it acts on.

To prepare for some of the manipulations we will be doing with operations on Qbits, we now examine a few examples of working with operators on Cbits.

1.4 Manipulating operations on Cbits

It is useful to introduce a 1-Cbit operator **n** that is simply the projection operator onto the state $|1\rangle$:

$$\mathbf{n}|x\rangle = x|x\rangle, \quad x = 0 \text{ or } 1. \tag{1.30}$$

Because $|0\rangle$ and $|1\rangle$ are eigenvectors of **n** with eigenvalues 0 and 1, **n** is called the 1-Cbit *number operator*. We also define the complementary operator,

$$\tilde{\mathbf{n}} = \mathbf{1} - \mathbf{n}, \tag{1.31}$$

which projects onto the state $|0\rangle$, so $|0\rangle$ and $|1\rangle$ are eigenvectors of $\tilde{\mathbf{n}}$ with eigenvalues 1 and 0. These operators have the matrix representations

$$\mathbf{n} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}, \qquad \tilde{\mathbf{n}} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}. \tag{1.32}$$

It follows directly from their definitions that

$$\mathbf{n}^2 = \mathbf{n}, \qquad \tilde{\mathbf{n}}^2 = \tilde{\mathbf{n}}, \qquad \mathbf{n}\tilde{\mathbf{n}} = \tilde{\mathbf{n}}\mathbf{n} = \mathbf{0}, \qquad \mathbf{n} + \tilde{\mathbf{n}} = \mathbf{1}. \tag{1.33}$$

We also have

$$\mathbf{n}\mathbf{X} = \mathbf{X}\tilde{\mathbf{n}}, \qquad \tilde{\mathbf{n}}\mathbf{X} = \mathbf{X}\mathbf{n}, \tag{1.34}$$

since flipping the state of a Cbit and then acting on it with $\mathbf{n}$ ($\tilde{\mathbf{n}}$) is the same as acting on the state with $\tilde{\mathbf{n}}$ ($\mathbf{n}$) and then flipping it. All the simple relations in (1.33) and (1.34) also follow, as they must, from the matrix representations (1.17) and (1.32) for $\mathbf{X}$, $\mathbf{n}$, and $\tilde{\mathbf{n}}$.

Although $\mathbf{n}$ has no interpretation as a physical operation on Cbits – replacing the state of a Cbit by the zero vector corresponds to no physical operation – it can be useful in deriving relations between operations that do have physical meaning. Since, for example, the SWAP operator $\mathbf{S}_{ij}$ acts as the identity if the states of the Cbits i and j are the same, and flips the numbers represented by both Cbits if their states are different, it can be written as

$$\mathbf{S}_{ij} = \mathbf{n}_i\mathbf{n}_j + \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + (\mathbf{X}_i\mathbf{X}_j)(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j). \tag{1.35}$$

At the risk of belaboring the obvious, I note that (1.35) acts as the swap operator because if both Cbits are in the state $|1\rangle$ (so swapping their states does nothing) then only the first term in the sum acts (i.e. each of the other three terms gives 0) and multiplies the state by 1; if both Cbits are in the state $|0\rangle$, only the second term acts and again multiplies the state by 1; if Cbit i is in the state $|1\rangle$ and Cbit j is in the state $|0\rangle$, only the third term acts and the effect of flipping both Cbits is to swap their states; and if Cbit i is in the state $|0\rangle$ and Cbit j is in the state $|1\rangle$, only the fourth term acts and the effect of the two $\mathbf{X}$s is again to swap their states.

To help you become more at home with this notation, you are urged to prove from (1.35) that $\mathbf{S}_{ij}^2 = \mathbf{1}$, using only the relations in (1.33) and (1.34), the fact that $\mathbf{X}^2 = \mathbf{1}$, and the fact that 1-Cbit operators acting on different Cbits commute.

The construction (1.23) of SWAP out of cNOT operators can also be demonstrated using a more algebraic approach. Note first that $\mathbf{C}_{ij}$ can be expressed in terms of $\mathbf{n}$s and $\mathbf{X}$s by

$$\mathbf{C}_{ij} = \tilde{\mathbf{n}}_i + \mathbf{X}_j\mathbf{n}_i, \tag{1.36}$$

since if the state of Cbit i is $|0\rangle$ only the first term acts, which leaves the states of both Cbits unchanged, but if the state of Cbit i is $|1\rangle$ only the second term acts, which leaves the state of Cbit i unchanged, while $\mathbf{X}_j$

flips Cbit j. If you substitute expressions of the form (1.36) for each of the three terms in (1.23), then you can show by purely algebraic manipulations that four of the eight terms into which the products expand vanish and the remaining four can be rearranged to give the swap operator (1.35).

An operator that has no direct role to play in classical computation, but which is as important as the NOT operator $\mathbf{X}$ in quantum computation, is the operator $\mathbf{Z}$ defined by

$$\mathbf{Z} = \tilde{\mathbf{n}} - \mathbf{n} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{1.37}$$

It follows from (1.34) (or from the matrix representations (1.17) and (1.37)) that $\mathbf{X}$ *anticommutes* with $\mathbf{Z}$:

$$\mathbf{ZX} = -\mathbf{XZ}. \tag{1.38}$$

Since $\tilde{\mathbf{n}} + \mathbf{n} = \mathbf{1}$, we can use (1.37) to express the 1-Cbit projection operators $\tilde{\mathbf{n}}$ and $\mathbf{n}$ in terms of $\mathbf{1}$ and $\mathbf{Z}$:

$$\mathbf{n} = \tfrac{1}{2}(\mathbf{1} - \mathbf{Z}), \qquad \tilde{\mathbf{n}} = \tfrac{1}{2}(\mathbf{1} + \mathbf{Z}). \tag{1.39}$$

Using this we can rewrite the cNOT operator (1.36) in terms of $\mathbf{X}$ and $\mathbf{Z}$ operators:

$$\begin{aligned} \mathbf{C}_{ij} &= \tfrac{1}{2}\big(\mathbf{1} + \mathbf{Z}_i\big) + \tfrac{1}{2}\mathbf{X}_j\big(\mathbf{1} - \mathbf{Z}_i\big) \\ &= \tfrac{1}{2}\big(\mathbf{1} + \mathbf{X}_j\big) + \tfrac{1}{2}\mathbf{Z}_i\big(\mathbf{1} - \mathbf{X}_j\big). \end{aligned} \tag{1.40}$$

The second form follows from the first because $\mathbf{X}_j$ and $\mathbf{Z}_i$ commute when $i \neq j$. Note that, if we were to interchange $\mathbf{X}$ and $\mathbf{Z}$ in the second line of (1.40), we would get back the expression directly above it except for the interchange of i and j. So interchanging the $\mathbf{X}$ and $\mathbf{Z}$ operators has the effect of switching which Cbit is the control and which is the target, changing $\mathbf{C}_{ij}$ into $\mathbf{C}_{ji}$. An operator that can produce just this effect is the *Hadamard transformation* (also sometimes called the *Walsh–Hadamard transformation*),

$$\mathbf{H} = \frac{1}{\sqrt{2}}(\mathbf{X} + \mathbf{Z}) = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{1.41}$$

This is another operator of fundamental importance in quantum computation.[2]

2 Physicists should note here an unfortunate clash between the notations of quantum computer science and physics. Quantum physicists invariably use H to denote the Hamiltonian function (in classical mechanics) or Hamiltonian operator (in quantum mechanics). Fortunately Hamiltonian operators, although of crucial importance in the design of quantum computers, play a very limited role in the general theory of quantum computation, being completely overshadowed by the unitary transformations that they generate. So physicists can go along with the computer-science notation without getting into serious trouble.

Since $\mathbf{X}^2 = \mathbf{Z}^2 = \mathbf{1}$ and $\mathbf{XZ} = -\mathbf{ZX}$, one easily shows from the definition (1.41) of $\mathbf{H}$ in terms of $\mathbf{X}$ and $\mathbf{Z}$ that

$$\mathbf{H}^2 = \mathbf{1} \tag{1.42}$$

and that

$$\mathbf{HXH} = \mathbf{Z}, \qquad \mathbf{HZH} = \mathbf{X}. \tag{1.43}$$

This shows how $\mathbf{H}$ can be used to interchange the $\mathbf{X}$ and $\mathbf{Z}$ operators in $\mathbf{C}_{ji}$: it follows from (1.43), together with (1.40) and (1.42), that

$$\mathbf{C}_{ji} = \big(\mathbf{H}_i\mathbf{H}_j\big)\mathbf{C}_{ij}\big(\mathbf{H}_i\mathbf{H}_j\big). \tag{1.44}$$

We shall see that this simple relation can be put to some quite remarkable uses in a quantum computer. While one can achieve this interchange on a classical computer using the SWAP operation, $\mathbf{C}_{ji} = \mathbf{S}_{ij}\mathbf{C}_{ij}\mathbf{S}_{ij}$, the crucial difference between $\mathbf{S}_{ij}$ and $\mathbf{H}_i\mathbf{H}_j$ is that the latter is a product of two 1-Cbit operators, while the former is not.

Of course, the action of $\mathbf{H}$ on the state of a Cbit that follows from (1.41),

$$\mathbf{H}|0\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \qquad \mathbf{H}|1\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle - |1\rangle), \tag{1.45}$$

describes no meaningful transformation of Cbits. Nevertheless, when combined with other operations, as on the right side of (1.44), the Hadamard operations result in the perfectly sensible operation given on the left side. In a quantum computer the action of $\mathbf{H}$ on 1-Qbit states turns out to be not only meaningful but also easily implemented, and the possibility of interchanging control and target Qbits using only 1-Qbit operators in the manner shown in (1.44) turns out to have some striking consequences.

The use of Hadamards to interchange the control and target Qbits of a cNOT operation is sufficiently important in quantum computation to merit a second derivation of (1.44), which further illustrates the way in which one uses the operator formalism. In strict analogy to the definition of cNOT (see (1.21) and the preceding paragraph) we can define a controlled-Z operation, $\mathbf{C}^Z_{ij}$, which leaves the state of the target Cbit j unchanged if the state of the control Cbit i is $|0\rangle$, and operates on the target Cbit with $\mathbf{Z}$ if the state of the control Cbit is $|1\rangle$. As a result $\mathbf{C}^Z_{10}|xy\rangle$ acts as the identity on $|xy\rangle$ unless both x and y are 1, in which case it simply takes $|11\rangle$ into $-|11\rangle$. This behavior is completely symmetric in the two Cbits, so

$$\mathbf{C}^Z_{ij} = \mathbf{C}^Z_{ji}. \tag{1.46}$$

It is a straightforward consequence of (1.42) and (1.43) that sandwiching the target Cbit of a cNOT between Hadamards converts it to a $\mathbf{C}^Z$:

$$\mathbf{H}_j\mathbf{C}_{ij}\mathbf{H}_j = \mathbf{C}^Z_{ij}, \qquad \mathbf{H}_i\mathbf{C}_{ji}\mathbf{H}_i = \mathbf{C}^Z_{ji}. \tag{1.47}$$

In view of (1.46), we then have

$$\mathbf{H}_j\mathbf{C}_{ij}\mathbf{H}_j = \mathbf{H}_i\mathbf{C}_{ji}\mathbf{H}_i, \tag{1.48}$$

which is equivalent to (1.44), since $\mathbf{H}^2 = \mathbf{1}$.

As a final exercise in treating operations on Cbits as linear operations on vectors, we construct an alternative form for the swap operator. If we use (1.39) to reexpress each $\mathbf{n}$ and $\tilde{\mathbf{n}}$ appearing in the swap operator (1.35) in terms of $\mathbf{Z}$, we find that

$$\mathbf{S}_{ij} = \tfrac{1}{2}(\mathbf{1} + \mathbf{Z}_i\mathbf{Z}_j) + \tfrac{1}{2}(\mathbf{X}_i\mathbf{X}_j)(\mathbf{1} - \mathbf{Z}_i\mathbf{Z}_j). \tag{1.49}$$

If we define

$$\mathbf{Y} = i\mathbf{XZ} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad (i = \sqrt{-1}), \tag{1.50}$$

we get the more compact form

$$\mathbf{S}_{ij} = \tfrac{1}{2}(\mathbf{1} + \mathbf{X}_i\mathbf{X}_j + \mathbf{Y}_i\mathbf{Y}_j + \mathbf{Z}_i\mathbf{Z}_j). \tag{1.51}$$

For three quarters of a century physicists have enjoyed grouping the matrix representations of the three operators $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ into a "3-vector" $\vec{\boldsymbol{\sigma}}$ whose "components" are $2 \otimes 2$ matrices:

$$\boldsymbol{\sigma}_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \boldsymbol{\sigma}_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad \boldsymbol{\sigma}_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{1.52}$$

The swap operator then becomes[3]

$$\mathbf{S}_{ij} = \tfrac{1}{2}\big(\mathbf{1} + \vec{\boldsymbol{\sigma}}^{(i)} \cdot \vec{\boldsymbol{\sigma}}^{(j)}\big), \tag{1.53}$$

where "·" represents the ordinary three-dimensional scalar product:

$$\vec{\boldsymbol{\sigma}}^{(i)} \cdot \vec{\boldsymbol{\sigma}}^{(j)} = \boldsymbol{\sigma}_x^{(i)}\boldsymbol{\sigma}_x^{(j)} + \boldsymbol{\sigma}_y^{(i)}\boldsymbol{\sigma}_y^{(j)} + \boldsymbol{\sigma}_z^{(i)}\boldsymbol{\sigma}_z^{(j)}. \tag{1.54}$$

The three components of $\vec{\boldsymbol{\sigma}}$ have many properties that are unchanged under cyclic permutations of x, y, and z. All three are Hermitian.[4] All square to unity,

$$\boldsymbol{\sigma}_x^2 = \boldsymbol{\sigma}_y^2 = \boldsymbol{\sigma}_z^2 = \mathbf{1}. \tag{1.55}$$

3 Physicists might enjoy the simplicity of this "computational" derivation of the form of the exchange operator, compared with the conventional quantum-mechanical derivation, which invokes the full apparatus of angular-momentum theory.

4 The elements of a Hermitian matrix A satisfy $A_{ji} = A^*_{ij}$, where * denotes complex conjugation. A fuller statement in a broader context can be found in Appendix A.

They all anticommute in pairs and the product of any two of them is simply related to the third:

$$\begin{aligned} \boldsymbol{\sigma}_x\boldsymbol{\sigma}_y &= -\boldsymbol{\sigma}_y\boldsymbol{\sigma}_x = i\boldsymbol{\sigma}_z, \\ \boldsymbol{\sigma}_y\boldsymbol{\sigma}_z &= -\boldsymbol{\sigma}_z\boldsymbol{\sigma}_y = i\boldsymbol{\sigma}_x, \\ \boldsymbol{\sigma}_z\boldsymbol{\sigma}_x &= -\boldsymbol{\sigma}_x\boldsymbol{\sigma}_z = i\boldsymbol{\sigma}_y. \end{aligned} \tag{1.56}$$

The three relations (1.56) differ only by cyclic permutations of x, y, and z.

All the relations in (1.55) and (1.56) can be summarized in a single compact and useful identity. Let $\vec{a}$ and $\vec{b}$ be two 3-vectors with components a_x, a_y, a_z and b_x, b_y, b_z that are ordinary real numbers. (They can also be complex numbers, but in most useful applications they are real.) Then one easily confirms that all the relations in (1.55) and (1.56) imply and are implied by the single identity

$$(\vec{a}\cdot\vec{\boldsymbol{\sigma}})(\vec{b}\cdot\vec{\boldsymbol{\sigma}}) = (\vec{a}\cdot\vec{b})\mathbf{1} + i(\vec{a}\times\vec{b})\cdot\vec{\boldsymbol{\sigma}}, \tag{1.57}$$

where $\vec{a}\times\vec{b}$ denotes the vector product (or "cross product") of $\vec{a}$ and $\vec{b}$,

$$\begin{aligned} (\vec{a}\times\vec{b})_x &= a_yb_z - a_zb_y, \\ (\vec{a}\times\vec{b})_y &= a_zb_x - a_xb_z, \\ (\vec{a}\times\vec{b})_z &= a_xb_y - a_yb_x. \end{aligned} \tag{1.58}$$

Together with the unit matrix $\mathbf{1}$, the matrices $\boldsymbol{\sigma}_x$, $\boldsymbol{\sigma}_y$, and $\boldsymbol{\sigma}_z$ form a basis for the four-dimensional algebra of two-dimensional matrices of complex numbers: any such matrix is a unique linear combination of these four with complex coefficients. Because the four are all Hermitian, any two-dimensional Hermitian matrix A of complex numbers must be a *real* linear combination of the four, and therefore of the form

$$A = a_0\mathbf{1} + \vec{a}\cdot\vec{\boldsymbol{\sigma}}, \tag{1.59}$$

where a_0 and the components of the 3-vector $\vec{a}$ are all real numbers.

The matrices $\boldsymbol{\sigma}_x$, $\boldsymbol{\sigma}_y$, and $\boldsymbol{\sigma}_z$ were introduced in the early days of quantum mechanics by Wolfgang Pauli, to describe the angular momentum associated with the spin of an electron. They have many other useful purposes, being simply related to the quaternions invented by Hamilton to deal efficiently with the composition of three-dimensional rotations.[5] It is pleasing to find them here, buried in the interior of the operator that simply swaps two classical bits. We shall have extensive occasion to use Pauli's 1-Qbit operators when we come to the subject of

5 Hamilton's quaternions i, j, k are represented by $i\boldsymbol{\sigma}_x, i\boldsymbol{\sigma}_y, i\boldsymbol{\sigma}_z$. The beautiful and useful connection between Pauli matrices and three-dimensional rotations discovered by Hamilton is developed in Appendix B.

quantum error correction. Some of their properties, developed further in Appendix B, prove to be quite useful in treating Qbits, to which we now turn.

1.5 Qbits and their states

The state of a Cbit is a pretty miserable specimen of a two-dimensional vector. The only vectors with any classical meaning in the whole two-dimensional vector space are the two orthonormal vectors $|0\rangle$ and $|1\rangle$, since those are the only two states a Cbit can have. Happily, nature has provided us with physical systems, Qbits, described by states that do not suffer from this limitation. The state $|\psi\rangle$ associated with a Qbit can be any unit vector in the two-dimensional vector space spanned by $|0\rangle$ and $|1\rangle$ over the complex numbers. The general state of a Qbit is

$$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix}, \tag{1.60}$$

where α_0 and α_1 are two complex numbers constrained only by the requirement that $|\psi\rangle$, like $|0\rangle$ and $|1\rangle$, should be a unit vector in the complex vector space – i.e. only by the normalization condition

$$|\alpha_0|^2 + |\alpha_1|^2 = 1. \tag{1.61}$$

The state $|\psi\rangle$ is said to be a *superposition* of the states $|0\rangle$ and $|1\rangle$ with *amplitudes* α_0 and α_1. If one of α_0 and α_1 is 0 and the other is 1 – i.e. the special case in which the state of the Qbit is one of the two classical states $|0\rangle$ or $|1\rangle$ – it can be convenient to retain the language appropriate to Cbits, speaking of the Qbit "having the value" 0 or 1. More correctly, however, one is entitled to say only that the state of the Qbit is $|0\rangle$ or $|1\rangle$. Qbits, in contrast to Cbits, cannot be said to "have values." They have – or, more correctly, *are described by*, or, better still, are *associated with* – states. We shall often sacrifice correctness for ease of expression. Some reasons for this apparently pedantic terminological hair splitting will emerge below.

Just as the general state of a single Qbit is any normalized superposition (1.60) of the two possible classical states, the general state $|\Psi\rangle$ that nature allows us to associate with two Qbits is any normalized superposition of the four orthogonal classical states,

$$|\Psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle = \begin{pmatrix} \alpha_{00} \\ \alpha_{01} \\ \alpha_{10} \\ \alpha_{11} \end{pmatrix}, \tag{1.62}$$

with the complex amplitudes being constrained only by the normalization condition

$$|\alpha_{00}|^2 + |\alpha_{01}|^2 + |\alpha_{10}|^2 + |\alpha_{11}|^2 = 1. \tag{1.63}$$

This generalizes in the obvious way to n Qbits, whose general state can be any superposition of the 2^n different classical states, with amplitudes whose squared magnitudes sum to unity:

$$|\Psi\rangle = \sum_{0\leq x<2^n} \alpha_x |x\rangle_n, \tag{1.64}$$

$$\sum_{0\leq x<2^n} |\alpha_x|^2 = 1. \tag{1.65}$$

In the context of quantum computation, the set of 2^n classical states – all the possible tensor products of n individual Qbit states $|0\rangle$ and $|1\rangle$ – is called the *computational basis*. For most purposes *classical basis* is a more appropriate term. I shall use the two interchangeably. The states that characterize n Cbits – the classical-basis states – are an extremely limited subset of the states of n Qbits, which can be any (normalized) superposition with complex coefficients of these classical-basis states.

If we have two Qbits, one in the state $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ and the other in the state $|\phi\rangle = \beta_0|0\rangle + \beta_1|1\rangle$, then the state $|\Psi\rangle$ of the pair, in a straightforward generalization of the rule for multi-Cbit states, is taken to be the tensor product of the individual states,

$$\begin{aligned} |\Psi\rangle = |\psi\rangle \otimes |\phi\rangle &= \big(\alpha_0|0\rangle + \alpha_1|1\rangle\big) \otimes \big(\beta_0|0\rangle + \beta_1|1\rangle\big) \\ &= \alpha_0\beta_0|00\rangle + \alpha_0\beta_1|01\rangle + \alpha_1\beta_0|10\rangle + \alpha_1\beta_1|11\rangle \\ &= \begin{pmatrix} \alpha_0\beta_0 \\ \alpha_0\beta_1 \\ \alpha_1\beta_0 \\ \alpha_1\beta_1 \end{pmatrix}. \end{aligned} \tag{1.66}$$

Note that a general 2-Qbit state (1.62) is of the special form (1.66) if and only if $\alpha_{00}\alpha_{11} = \alpha_{01}\alpha_{10}$. Since the four amplitudes in (1.62) are constrained only by the normalization condition (1.63), this relation need not hold, and the general 2-Qbit state, unlike the general state of two Cbits, is *not* a product (1.66) of two 1-Qbit states. The same is true for states of n Qbits. Unlike Cbits, whose general state can only be one of the 2^n products of $|0\rangle$s and $|1\rangle$s, a general state of n Qbits is a superposition of these 2^n product states and cannot, in general, be expressed as a product of any set of 1-Qbit states. Individual Qbits making up a multi-Qbit system, in contrast to individual Cbits, cannot always be characterized as having individual states of their own.[6]

Such nonproduct states of two or more Qbits are called *entangled* states. The term is a translation of Schrödinger's *verschränkt*, which I

6 More precisely, they do not always have what are called *pure states* of their own. It is often convenient to give a statistical description of an individual Qbit (or a group of Qbits) in terms of what is called a *density matrix* or *mixed state*. If one wishes to emphasize that one is not talking about a mixed state, one uses the term "pure state." In this book the term "state" always means "pure state."

am told is rendered more accurately as "entwined" or "enfolded." But Schrödinger himself used the English word "entangled," and may even have used it before coining the German term. When the state of several Qbits is entangled, they can sometimes behave in some very strange ways. An example of such peculiar behavior is discussed in Appendix D. Aside from its intrinsic interest, the appendix provides some further exercise in the analytical manipulation of Qbits.

1.6 Reversible operations on Qbits

The only nontrivial reversible operation a classical computer can perform on a single Cbit is the NOT operation $\mathbf{X}$. Nature has been far more versatile in what it allows us to do to a Qbit. The reversible operations that a quantum computer can perform upon a single Qbit are represented by the action on the state of the Qbit of any *linear* transformation that takes unit vectors into unit vectors. Such transformations $\mathbf{u}$ are called *unitary* and satisfy the condition[7]

$$\mathbf{u}\mathbf{u}^\dagger = \mathbf{u}^\dagger\mathbf{u} = 1. \tag{1.67}$$

Since any unitary transformation has a unitary inverse, such actions of a quantum computer on a Qbit are reversible. The reason why reversibility is crucial for the effective functioning of a quantum computer will emerge in Chapter 2.

The most general reversible n-Cbit operation in a classical computer is a permutation of the $(2^n)!$ different classical-basis states. The most general reversible operation that a quantum computer can perform upon n Qbits is represented by the action on their state of any linear transformation that takes unit vectors into unit vectors – i.e. any 2^n-dimensional unitary transformation $\mathbf{U}$, satisfying

$$\mathbf{U}\mathbf{U}^\dagger = \mathbf{U}^\dagger\mathbf{U} = 1. \tag{1.68}$$

Any reversible operation on n Cbits – i.e. any permutation $\mathbf{P}$ of the 2^n Cbit states – can be associated with a unitary operation $\mathbf{U}$ on n Qbits. One defines the action of $\mathbf{U}$ on the classical-basis states of the Qbit to be identical to the operation of $\mathbf{P}$ on the corresponding classical states of the Cbit. Since the classical basis *is* a basis, $\mathbf{U}$ can be extended to arbitrary n-Qbit states by requiring it to be linear. Since the action of $\mathbf{U}$ on the classical-basis states is to permute them, its effect on any superposition of such states $\sum \alpha_x |x\rangle_n$ is to permute the amplitudes α_x. Such a permutation preserves the value of $\sum |\alpha_x|^2$, so $\mathbf{U}$ takes unit vectors into unit vectors. Being norm-preserving and linear, $\mathbf{U}$ is indeed unitary.

7 These and other facts about linear operators on vector spaces over the complex numbers are also reviewed and summarized in Appendix A.

Many important unitary operations on Qbits that we shall be examining below are defined in this way, as permutations of the classical-basis states, which are implicitly understood to be extended by linearity to all Qbit states. In particular, the transformations NOT, SWAP, and cNOT on Cbits are immediately defined in this way for Qbits as well. But the available unitary transformations on Qbits are, of course, much more general than straightforward extensions of classical operations. We have already encountered two such examples, the operator $\mathbf{Z}$ and the Hadamard transformation $\mathbf{H}$. Both of these take the classical-basis states of a Qbit into another orthonormal basis, so their linear extensions to all Qbit states are necessarily unitary.

In designing quantum algorithms, the class of allowed unitary transformations is almost always restricted to ones that can be built entirely out of products of unitary transformations that act on only one Qbit at a time, called *1-Qbit gates*, or that act on just a pair of Qbits, called *2-Qbit gates*. This restriction is imposed because the technical problems of making higher-order quantum gates are even more formidable than the (already difficult) problems of constructing reliable 1- and 2-Qbit gates.

It turns out that this is not a fundamental limitation, since arbitrary unitary transformations can be approximated to an arbitrary degree of precision by sufficiently many 1- and 2-Qbit gates. We shall not prove this general result,[8] because all of the quantum algorithms to be developed here will be explicitly built up entirely out of 1- and 2-Qbit gates. One very important illustration of the sufficiency of 1- and 2-Qbit gates will emerge in Chapter 2. For a reversible classical computer, it can be shown that at least one 3-Cbit gate is needed to build up general logical operations. But, in a quantum computer, we shall find, remarkably – and importantly for the feasibility of practical quantum computation – that the quantum extension of this 3-Cbit gate can be constructed out of a small number of 1- and 2-Qbit gates.

While unitarity is generally taken to be the hallmark of the transformations nature allows us to perform on quantum states, what is really remarkable about the transformations of Qbit states is their *linearity* (which is, of course, one aspect of their unitarity). It is easy to dream up simple classical models for a Qbit, particularly if one restricts its states to real linear combinations of the two computational basis states. It is not hard to invent classical models for NOT and Hadamard 1-Qbit gates that act linearly on all the 1-Qbit states of the model Qbit. But I know of no classical model that can extend a cNOT on the four computational basis states of two Cbits to an operation that acts

8 The argument is given by David P. DiVincenzo, "Two-bit gates are universal for quantum computation," *Physical Review* A **51**, 1015–1022 (1995), `http://arxiv.org/abs/quant-ph/9407022`.

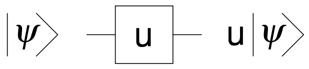


Fig 1.1 A circuit diagram representing the action on a single Qbit of the 1-Qbit gate **u**. Initially the Qbit is described by the input state $|\psi\rangle$ on the left. The thin line (wire) represents the subsequent history of the Qbit. After emerging from the box representing **u**, the Qbit is described on the right by the final state $\mathbf{u}|\psi\rangle$.

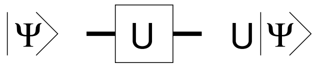


Fig 1.2 A circuit diagram representing the action on n Qbits of the n-Qbit gate **U**. Initially the Qbits ares described by the input state $|\Psi\rangle$ on the left. The thick line (bar) represents the subsequent history of the Qbits. After emerging from the box representing **U**, the Qbits are described on the right by the final state $\mathbf{U}|\Psi\rangle$.

linearly on all the states of two model Qbits. It is a remarkable and highly nontrivial fact about the physical world that nature does allow us, with much ingenuity and hard work, to fabricate unitary cNOT gates for a pair of genuine quantum Qbits.

1.7 Circuit diagrams

It is the practice in quantum computer science to represent the action of a sequence of gates acting on n Qbits by a circuit diagram. The initial state of the Qbits appears on the left, the final state on the right, and the gates themselves in the central part of the figure. Figure 1.1 shows the simplest possible such diagram: a Qbit initially in the state $|\psi\rangle$ is acted on by a 1-Qbit gate **u**, with the result that the Qbit is assigned the new state $\mathbf{u}|\psi\rangle$. Figure 1.2 shows the analogous diagram for an n-Qbit gate **U** and an n-Qbit initial state $|\Psi\rangle$. The line that goes into and out of the box representing the unitary transformation – which becomes useful when one starts chaining together a sequence of gates – is sometimes called a *wire* in the case of a single Qbit, and the thicker line (which represents n wires) associated with an n-Qbit gate is sometimes called a *bar*.

Figure 1.3 reveals a peculiar feature of these circuit diagrams that it is important to be aware of. The diagrams are read from left to right (as one reads ordinary prose in European languages). Part (a) portrays a circuit that acts first with **V** and then with **U** on the initial state $|\Psi\rangle$. The result is the state $\mathbf{UV}|\Psi\rangle$, because it is the convention, in writing equations for linear operators on vector spaces, that the operation appears to the

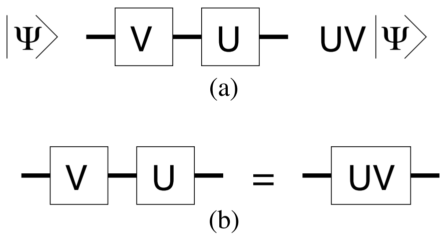


Fig 1.3 (a) A circuit diagram representing the action on n Qbits of two n-Qbit gates. Initially the Qbits are described by the input state $|\Psi\rangle$ on the left. They are acted upon first by the gate **V** and then by the gate **U**, emerging on the right in the final state $\mathbf{UV}|\Psi\rangle$. Note that the order in which the Qbits encounter unitary gates in the figure is opposite to the order in which the corresponding symbols are written in the symbol for the final state on the right. (b) This emphasizes the unfortunate convention that, because gates on the left act before gates on the right in a circuit diagram, a circuit showing **V** on the left and **U** on the right represents the operation conventionally denoted by **UV**.

left of the state on which it acts. Thus the sequence of symbols $|\Psi\rangle$, **V**, and **U** on the left of the circuit diagram in (a) is reversed from the sequence in which they appear in the mathematical representation of the state that is produced on the right. Part (b) shows the consequences of this for the part of the circuit diagram containing just the gates: a diagram in which a gate **V** (on the left) is followed by a gate **U** on the right describes the unitary transformation **UV**.

One should be wary of the possibility for confusion arising from the fact that operators (and states) in circuit diagrams always appear in the diagrams in the opposite sequence from the order in which they appear on the page in the corresponding equations. While several of the most important diagrams we shall encounter are left–right symmetric, many are not, so one should be on guard against getting things backwards when translating equations into circuit diagrams and vice versa.

In lecturing on quantum computation I tried for several years to reverse the computer-science convention, putting the initial state on the right of the circuit diagram and letting the gates on the right act first. This has the great advantage of making the diagram look like the equation it represents. It has, however, a major disadvantage, even setting aside the fact that it flies in the face of well established convention. It requires one to write on the blackboard in the wrong direction, from right to left, whenever one wishes to produce a circuit diagram. Guessing how far to the right one should start is hard to do if the diagram is a lengthy one, and for this reason I gave up after a few years and reverted to the conventional form. A better alternative would be for physicists to start writing their equations with the states on the left (represented by bra vectors rather than ket vectors[9]) and with linear operators appearing to the right of the states on which they act. But this would require abandoning a tradition that goes back three quarters of a century. So we are stuck with a clash of cultures, and must simply keep in mind that confusion can arise if one forgets the elementary fact represented in Figure 1.3(b).

There is little utility to circuit diagrams of the simple form in Figures 1.1–1.3, but they are important as building blocks out of which

9 See Appendix A for the distinction between bras and kets.

larger circuit diagrams are constructed. As the number of operations increases, the diagrams enable one to see at a glance the action of a sequence of 1- and 2-Qbit unitary gates on a collection of many Qbits in a way that is far more transparent and much more easily remembered than the corresponding formulae. Indeed, many calculations that involve rather lengthy equations can be simply accomplished by manipulating circuit diagrams, as we shall see.

When the state vectors entering or leaving a wire or bar in a circuit diagram are computational-basis states like $|x\rangle$, one sometimes omits the symbol $|\ \rangle$ and simply writes x.

1.8 Measurement gates and the Born rule

To give the state of a single Cbit you need only one bit of information: whether the state of the Cbit is $|0\rangle$ or $|1\rangle$. But to specify the state (1.60) of a single Qbit to an arbitrarily high degree of precision, you need arbitrarily many bits of information, since you must specify two complex numbers α and β subject only to the normalization constraint (1.61). Because Qbits not only have a much richer set of states than Cbits, but also can be acted on by a correspondingly richer set of transformations, it might appear obvious that a quantum computer would be vastly more powerful than a classical computer. *But there is a major catch!*

The catch is this: if you have n Cbits, each representing either 0 or 1, you can find out the state of each just by looking. There is nothing problematic about learning the state of a Cbit, and hence learning the result of any calculation you may have built up out of operations on those Cbits. Furthermore – and this is taken for granted in any discussion of a classical computer – the state of Cbits is not altered by the process of reading them. The act of acquiring the information from Cbits is not disruptive. You can read the Cbits at any stage of a computation without messing up subsequent stages.

In stark contrast, if you have n Qbits in a superposition (1.64) of computational basis states, there is nothing whatever you can do to them to extract from those Qbits the vast amount of information contained in the amplitudes α_x. You cannot read out the values of those amplitudes, and therefore you cannot find out what the state is. The state of n Qbits is not associated with any ascertainable property of those Qbits, as it is for Cbits.

There is only one way to extract information from n Qbits in a given state. It is called *making a measurement*.[10] Making a measurement

10 Physicists will note – others need pay no attention to this remark – that what follows is more accurately characterized as "making a (von Neumann) measurement in the computational (classical) basis." There are other ways

consists of performing a certain test on each Qbit, the outcome of which is either 0 or 1. The particular collection of zeros and ones produced by the test is not in general determined by the state $|\Psi\rangle$ of the Qbits; the state determines only the *probability* of the possible outcomes, according to the following rule: the probability of getting a particular result – say 01100, if you have five Qbits – is given by the squared magnitude of the amplitude of the state $|01100\rangle$ in the expansion of the state $|\Psi\rangle$ of the Qbits in the 2^5 computational basis states. More generally, if the state of n Qbits is

$$|\Psi\rangle_n = \sum_{0\leq x<2^n} \alpha_x |x\rangle_n, \tag{1.69}$$

then the probability that the zeros and ones resulting from measurements of all the Qbits will give the binary expansion of the integer x is

$$p(x) = |\alpha_x|^2. \tag{1.70}$$

This basic rule for how information can be extracted from a quantum state was first enunciated by Max Born, and is known as the *Born rule.* It provides the link between amplitudes and the numbers you can actually read out when you test – i.e. measure – the Qbits. The squared magnitudes of the amplitudes give the probabilities of outcomes of measurements. Normalization conditions like (1.65) are just the requirements that the probabilities for all of the 2^n mutually exclusive outcomes add up to 1.

The process of measurement is carried out by a piece of hardware with a digital display, known as an n-Qbit *measurement gate*. Such an n-Qbit measurement gate is depicted schematically in Figure 1.4. In contrast to unitary gates, which have a unique output state for each input state, the state of the Qbits emerging from a measurement gate is only *statistically* determined by the state of the input Qbits. In further contrast to unitary gates, the action of a measurement gate cannot be undone: given the final state $|x\rangle$, there is no way of reconstructing the initial state $|\Psi\rangle$. Measurement is irreversible. Nor is the action of a measurement gate in any sense linear.

To the extent that it suggests that some preexisting property is being revealed, "measurement" is a dangerously misleading term, but it is

to make such a measurement, but they can all be reduced to measurements in the computational basis if an appropriate unitary transformation is applied to the n-Qbit state of the computer just before carrying out the measurement. In this book the term "measurement" always means measurement in the computational basis. Measurements in other bases will always be treated as measurements in the computational basis preceded by suitable unitary transformations. There are also more general forms of measurement than von Neumann measurements, going under the unpleasant acronym *POVM* (for "positive operator-valued measure"). We shall make no explicit use of POVMs.

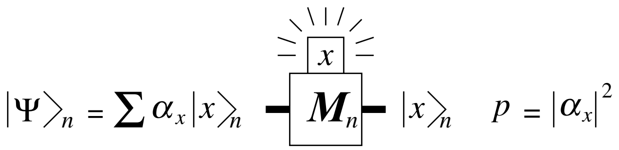


Fig 1.4 A circuit diagram representing an n-Qbit measurement gate. The Qbits are initially described by the n-Qbit state

$$|\Psi\rangle_n = \sum_{0\le x<2^n} \alpha_x |x\rangle_n ,$$

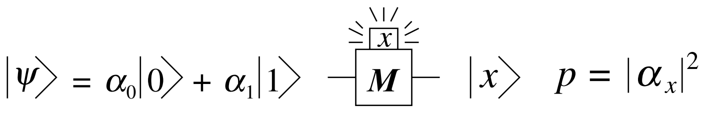
on the left. After the measurement gate M_n has acted, with probability $p = |\alpha_x|^2$ it indicates an integer x, $0 \le x < 2^n$, and the Qbits are subsequently described by the state $|x_n\rangle$ on the right.

hallowed by three quarters of a century of use by quantum physicists, and impossible to avoid in treatments of quantum computation. One should avoid being misled by such spurious connotations of "measurement," though it confused many physicists in the early days of quantum mechanics and may well continue to confuse some to this day. In quantum computation "measurement" means nothing more or less than applying and reading the display of an appropriate measurement gate, whose action is fully specified by the Born rule, as described above, and expanded upon below. While measurement in quantum mechanics is not at all like measuring somebody's weight, it does have some resemblance to measuring Alice's IQ, which, one can argue, reveals no preexisting numerical property of Alice, but only what happens when she is subjected to an IQ test.

The simplest statement of the Born rule is for a single Qbit. If the state of the Qbit is the superposition (1.60) of the states $|0\rangle$ and $|1\rangle$ with amplitudes α_0 and α_1 then the result of the measurement is 0 with probability $|\alpha_0|^2$ and 1 with probability $|\alpha_1|^2$. This measurement is carried out by a 1-Qbit measurement gate, as illustrated in Figure 1.5. We shall see below that n-Qbit measurement gates can be realized by applying 1-Qbit measurement gates to each of the n Qbits. The process of measurement can thus be reduced to applying multiple copies of a *single elementary piece of hardware: the 1-Qbit measurement gate.*

In addition to displaying an n-bit integer with probabilities determined by the amplitudes, there is a second very important aspect of the action of measurement gates: if n Qbits, initially described by a state $|\Psi\rangle$, are sent through an n-Qbit measurement gate, and the display of the measurement gate indicates the integer x, then one must associate with the Qbits emerging from that measurement gate the classical-basis state $|x\rangle_n$, as shown in Figures 1.4 and 1.5. This means that all traces of the amplitudes α_x characterizing the input state have vanished from the output state. The only role they have played in the measurement is to determine the probability of a particular output.

Fig 1.5 A special case of Figure 1.4: a 1-Qbit measurement gate. The reading x of the gate is either 0 or 1.



If the state of the input Qbits is one of the classical-basis states $|x\rangle_n$, then according to the Born rule the probability that the measurement gate will read x and the output state will remain $|x\rangle_n$ is 1. But for superpositions (1.69) with more than a single nonzero amplitude α_x, the output state is not determined. Being a single one of the classical basis states $|x\rangle_n$, the output state no longer carries any information about the amplitudes characterizing the initial state, other than certifying that the particular amplitude α_x was not zero, and, in all likelihood, was not exceedingly small.

So once you send n Qbits through an n-Qbit measurement gate, you remove the possibility of extracting any further information about their original state $|\Psi\rangle$. After such a measurement of five Qbits, if the result is 01100, then the post-measurement state associated with the Qbits is no longer $|\Psi\rangle$, but $|01100\rangle$. The original state $|\Psi\rangle$, with all the rich information potentially available in its amplitudes, is irretrievably lost. Qbits emerging from a measurement gate that indicates the outcome x are characterized by the state $|x\rangle$, regardless of what their pre-measurement state may have been.

This change of state attendant upon a measurement is often referred to as a *reduction* or *collapse* of the state. One says that the pre-measurement state *reduces* or *collapses* to the post-measurement state, as a consequence of the measurement. This should not be taken to imply (though, alas, it often is) that the Qbits themselves suffer a catastrophic "reduction" or "collapse." It is important to keep in mind, in this context, that the state of n Qbits is nothing more than an abstract symbol, used, via the Born rule, to calculate probabilities of measurement outcomes. As has already been noted, there is no internal property of the Qbits that corresponds to their state.

You might well wonder how one can learn anything at all of computational interest under these wretched conditions. The artistry of quantum computation consists of producing, through a cunningly constructed unitary transformation, a superposition in which most of the amplitudes α_x are zero or extremely close to zero, with useful information being carried by *any* of the values of x that have an appreciable probability of being indicated by the measurement. It is thus important to be seeking information that, once possessed, can easily be confirmed, perhaps with an ordinary (classical) computer (e.g. the factors of a large number), so that one is not misled by rare and irrelevant low-probability outcomes. How this is actually accomplished in various cases of interest will be one of our major preoccupations.

It is important to note and immediately reject a possible misunderstanding of the Born rule. One might be tempted to infer from the rule that for a Qbit to be in a superposition, such as the state $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$, means nothing more than that the "actual state" of the Qbit is either $|0\rangle$ with probability $|\alpha_0|^2$ or $|1\rangle$ with probability $|\alpha_1|^2$. Such an assertion goes beyond the rule, of course, which merely asserts that if one subjects a Qbit in the state $|\psi\rangle$ to an appropriate test – a measurement – then the outcome of the test will be 0 or 1 with those probabilities and the post-measurement state of the Qbit can correspondingly be taken to be $|0\rangle$ or $|1\rangle$. This does not imply that prior to the test the Qbit already carried the value revealed by the test and was already described by the corresponding classical-basis state, since, among other possibilities, the action of the test itself might well play a role in bringing forth the outcome.

In fact, it is easy to produce examples that demonstrate that the Qbit, prior to the test, *could not* have been in either of the states $|0\rangle$ and $|1\rangle$. We can see this with the help of the Hadamard transformation (1.41). We have defined the action of the 1-Qbit operators $\mathbf{H}$, $\mathbf{X}$, and $\mathbf{Z}$ only on the computational-basis states $|0\rangle$ and $|1\rangle$, but, as noted above, we can extend their action to arbitrary linear combinations of these states by requiring the extensions to be linear operators. Since the states $|0\rangle$ and $|1\rangle$ form a basis, this determines the action of $\mathbf{H}$, $\mathbf{X}$, and $\mathbf{Z}$ on any 1-Qbit state.

Because it is linear and norm-preserving, $\mathbf{H}$ is unitary, and is therefore the kind of operation a quantum computer can apply to the state of a Qbit: a 1-Qbit gate. The result of the operation of a Hadamard gate is to change the state $|\phi\rangle$ of a Qbit to $\mathbf{H}|\phi\rangle$. Suppose, now, that we apply $\mathbf{H}$ to a Qbit that is initially in the state

$$|\phi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle). \tag{1.71}$$

It follows from (1.45) that the result is just

$$\mathbf{H}|\phi\rangle = |0\rangle. \tag{1.72}$$

So according to the Born rule, if we measure a Qbit described by the state $\mathbf{H}|\phi\rangle$, the result will be 0 with probability 1.

But suppose that a Qbit in the state $|\phi\rangle$ were indeed either in the state $|0\rangle$ with probability $\frac{1}{2}$ or in the state $|1\rangle$ with probability $\frac{1}{2}$. In either case, according to (1.45), the subsequent action of $\mathbf{H}$ would produce a state – either $(1/\sqrt{2})(|0\rangle + |1\rangle)$ or $(1/\sqrt{2})(|0\rangle - |1\rangle)$ – that under measurement yielded 0 or 1 with equal probability. This contradicts the fact just extracted directly from (1.72) that the result of making a measurement on a Qbit in the state $\mathbf{H}|\phi\rangle$ is invariably 0.

So a Qbit in a quantum superposition of $|0\rangle$ and $|1\rangle$ cannot be viewed as being either in the state $|0\rangle$ or in the state $|1\rangle$ with certain probabilities. Such a state represents something quite different. Although the Qbit reveals only a 0 or a 1 when you query it with a measurement gate, prior to the query its state is not in general either $|0\rangle$ or $|1\rangle$, but a superposition of the form (1.60). Such a superposition is as natural and irreducible a description of a Qbit as $|0\rangle$ and $|1\rangle$ are. This point is expanded on in Appendix C.

If the states of n Qbits are restricted to computational-basis states then the process of measurement is just like the classical process of "learning the value" of x without altering the state. Thus a quantum computer can be made to simulate a reversible classical computer by allowing only computational-basis states as input, and using only unitary gates that take computational-basis states into computational-basis states.

The Born rule, relating the amplitudes α_x in the expansion (1.64) of a general n-Qbit state $|\Psi\rangle$ to the probabilities of measuring x, is often stated in terms of inner products or projection operators.[11] The probability of a measurement giving the result x $(0 \leq x < 2^n)$ is

$$p_\Psi(x) = |\alpha_x|^2 = |\langle x|\Psi\rangle|^2. \tag{1.73}$$

It can also be usefully expressed in terms of projection operators:

$$p_\Psi(x) = \langle x|\Psi\rangle\langle\Psi|x\rangle = \langle x|\mathbf{P}_\Psi|x\rangle \tag{1.74}$$

or

$$p_\Psi(x) = \langle\Psi|x\rangle\langle x|\Psi\rangle = \langle\Psi|\mathbf{P}_x|\Psi\rangle, \tag{1.75}$$

where $\mathbf{P}_\Psi = |\Psi\rangle\langle\Psi|$ is the projection operator on the state $|\Psi\rangle$, and $\mathbf{P}_x = |x\rangle\langle x|$ is the projection operator on the state $|x\rangle$.

1.9 The generalized Born rule

There is a stronger version of the Born rule, which plays an important role in quantum computation, even though, surprisingly, it is rarely explicitly mentioned in most standard quantum-mechanics texts. We shall call it the *generalized Born rule.* This stronger form applies when one measures only a single one of $n+1$ Qbits, by sending it through a standard 1-Qbit measurement gate.

To formulate the generalized Born rule, note that any state of all $n+1$ Qbits can be represented in the form

$$|\Psi\rangle_{n+1} = \alpha_0|0\rangle|\Phi_0\rangle_n + \alpha_1|1\rangle|\Phi_1\rangle_n, \qquad |\alpha_0|^2 + |\alpha_1|^2 = 1, \tag{1.76}$$

11 The Dirac notation for inner products and projection operators is described in Appendix A.

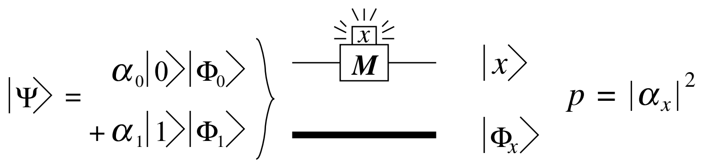

$|\Psi\rangle = \alpha_0|0\rangle|\Phi_0\rangle + \alpha_1|1\rangle|\Phi_1\rangle$
x
M
$|x\rangle$
$|\Phi_x\rangle$
$p = |\alpha_x|^2$

Fig 1.6 The action of a 1-Qbit measurement gate on a single one of $n+1$ Qbits, according to the generalized Born rule. The initial state (on the left) is a general $(n+1)$-Qbit state, expressed in the form $|\Psi\rangle_{n+1} = a_0|0\rangle|\Phi_0\rangle_n + a_1|1\rangle|\Phi_1\rangle_n$. Only the single Qbit on the left of this expression is subjected to a measurement gate.

where $|\Phi_0\rangle_n$ and $|\Phi_1\rangle_n$ are normalized (but not necessarily orthogonal). This follows directly from the general form,

$$|\Psi\rangle_{n+1} = \sum_{x=0}^{2^{n+1}-1} \gamma(x)|x\rangle_{n+1}, \qquad \sum_{x=0}^{2^{n+1}-1} |\gamma(x)|^2 = 1. \tag{1.77}$$

The states $|\Phi_0\rangle_n$ and $|\Phi_1\rangle_n$ are given by

$$|\Phi_0\rangle_n = (1/\alpha_0)\sum_{x=0}^{2^n-1} \gamma(x)|x\rangle_n, \qquad |\Phi_1\rangle_n = (1/\alpha_1)\sum_{x=0}^{2^n-1} \gamma(2^n+x)|x\rangle_n, \tag{1.78}$$

where

$$\alpha_0^2 = \sum_{x=0}^{2^n-1} |\gamma(x)|^2, \qquad \alpha_1^2 = \sum_{x=0}^{2^n-1} |\gamma(2^n+x)|^2. \tag{1.79}$$

(The α_0 and α_1 in (1.78) and (1.79) are real numbers, but can be multiplied by arbitrary phase factors if $|\Phi_0\rangle_n$ and $|\Phi_1\rangle_n$ are multiplied by the inverse phase factors.)

The generalized Born rule asserts that if one measures only the single Qbit whose state symbol is explicitly separated out from the others in the $(n+1)$-Qbit state (1.76), then the 1-Qbit measurement gate will indicate x (0 or 1) with probability $|\alpha_x|^2$, after which the $(n+1)$-Qbit state can be taken to be the product state $|x\rangle|\Phi_x\rangle_n$. (The rule holds for the measurement of any single Qbit – there is nothing special about the Qbit whose state symbol appears on the left in the $(n+1)$-Qbit state symbol.) This action of a 1-Qbit measurement gate on an $(n+1)$-Qbit state is depicted schematically in Figure 1.6.

If the Qbit on which the 1-Qbit gate acts is initially unentangled with the remaining n Qbits, then the action of the gate on the measured Qbit is just that specified by the ordinary Born rule, and the unmeasured Qbits play no role at all, remaining in their original state throughout the process. This is evident from the above statement of the generalized Born rule, specialized to the case in which the two states $|\Phi_0\rangle_n$ and $|\Phi_1\rangle_n$ are identical. It is illustrated in Figure 1.7.

If one applies the generalized Born rule n times to successive 1-Qbit measurements of each of n Qbits, initially in the general n-Qbit state (1.69), one can show by a straightforward argument, given in Appendix E, that the final state of the n Qbits is x with probability $|\alpha_x|^2$, where x is the n-bit integer whose bits are given by the readings

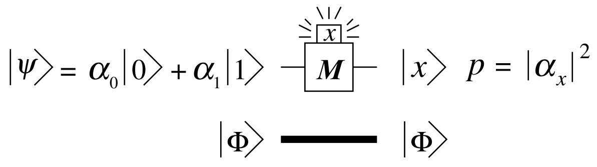

$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$
x
M
$|x\rangle$ $p = |\alpha_x|^2$
$|\Phi\rangle$
$|\Phi\rangle$

Fig 1.7 A simplification of Figure 1.6 when $|\Phi_0\rangle = |\Phi_1\rangle = |\Phi\rangle$. In this case the initial state on the left is just the product state $|\Psi\rangle_n = |\psi\rangle|\Phi\rangle = \big(a_0|0\rangle + a_1|1\rangle\big)|\Phi\rangle$, and the final state of the unmeasured Qbits continues to be $|\Phi\rangle$ regardless of the value of x indicated by the 1-Qbit measurement gate. The unmeasured Qbits are unentangled with the measured Qbit and described by the state $|\Phi\rangle$ throughout the process. The 1-Qbit measurement gate acts on the measured Qbit exactly as it does in Figure 1.5 when no other Qbits are present, and the generalized Born rule of Figure 1.6 reduces to the ordinary Born rule.

on the n 1-Qbit measurement gates. This is nothing but the ordinary Born rule, with the n 1-Qbit measurement gates playing the role of the single n-Qbit measurement gate. There is thus, as remarked upon above, only a single primitive piece of measurement hardware: the 1-Qbit measurement gate. The construction of an n-Qbit measurement gate out of n 1-Qbit measurement gates is depicted in Figure 1.8.

An even more general version of the Born rule follows from the generalized Born rule itself. The general state of $m + n$ Qbits can be written as

$$|\Psi\rangle_{m+n} = \sum_{x=0}^{2^m} \alpha_x |x\rangle_m |\Phi_x\rangle_n, \tag{1.80}$$

where $\sum_x |\alpha_x|^2 = 1$ and the states $|\Phi_x\rangle_n$ are normalized, but not necessarily orthogonal. By applying the generalized Born rule m times to m Qbits in an $(m + n)$-Qbit state, one establishes the rule that if just the m Qbits on the left of (1.80) are measured, then with probability $|\alpha_x|^2$ the result will be x, and after the measurement the state of all $m + n$ Qbits will be the product state

$$|x\rangle_m |\Phi_x\rangle_n \tag{1.81}$$

in which the m measured Qbits are in the state $|x\rangle_m$ and the n unmeasured ones are in the state $|\Phi_x\rangle_n$.

1.10 Measurement gates and state preparation

In addition to providing an output at the end of a computation, measurement gates also play a crucial role (which is not often emphasized) at the beginning. Since there is no way to determine the state of a given collection of Qbits – indeed, in general such a collection might be entangled with other Qbits and therefore not even have a state of its own – how can one produce a set of Qbits in a definite state for the gates of a quantum computer to transform into another computationally useful state?

The answer is by measurement. If one takes n Qbits off the shelf, and subjects them to an n-Qbit measurement gate that registers x, then the Qbits emerging from that gate are assigned the classical-basis state $|x\rangle_n$. If one then applies the 1-Qbit operation **X** to each Qbit that registered a 1 in the measurement, doing nothing to the Qbits that

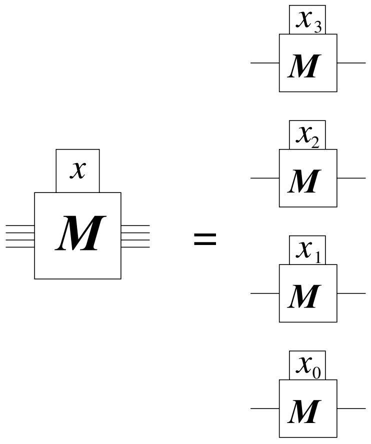


Fig 1.8 Constructing a 4-Qbit measurement gate out of four 1-Qbit measurement gates. The integer x has the binary expansion $x_3x_2x_1x_0$.

registered 0, the resulting set of Qbits will be described by the state $|0\rangle_n$. It is this state that most quantum-computational algorithms take as their input. Such a use of a measurement gate to produce a Qbit described by the state $|0\rangle$ is shown in Figure 1.9.

Measurement gates therefore play *two* roles in a quantum computation. They get the Qbits ready for the subsequent action of the computer, and they extract from the Qbits a digital output after the computer has acted. The initial action of the measurement gates is called *state preparation*, since the Qbits emerging from the process can be characterized by a definite state. The association of unitary operators with the gates that subsequently act on the Qbits permits one to update that initial state assignment into the corresponding unitary transformation of the initial state, thereby making it possible to calculate, using the Born rule, the probabilities of the outcomes of the final measurement gates.

This role of measurement gates in state preparation follows from the Born rule if the Qbits that are to be prepared already have a state of their own, even though that state might not be known to the user of the quantum computer. It also follows from the generalized Born rule if the Qbits already share an entangled state – again, not necessarily known to the user – with additional (unmeasured) Qbits. But one cannot deduce from the Born rules that measurement gates serve to prepare states for Qbits "off the shelf," whose past history nobody knows anything about. In such cases the use of measurement gates to assign a state to the Qbits is a reasonable and plausible extension of the Born rules. It is consistent with them, but goes beyond them.

For particular physical realizations of Qbits, there may be other ways to produce the standard initial state $|0\rangle_n$. Suppose, for example, that each Qbit is an atom, the state $|0\rangle$ is the lowest-energy state (the *ground state*) of the atom, and the state $|1\rangle$ is the atomic state of next-lowest

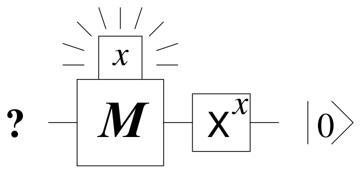


Fig 1.9 Using a 1-Qbit measurement gate to prepare an off-the-shelf Qbit so that its associated state is $|0\rangle$. The input on the left is a Qbit in an unknown condition – i.e. nothing is known of its past history. After the measurement gate is applied, the NOT gate **X** is or is not applied, depending on whether the measurement gate indicates 1 or 0. The Qbit that emerges (on the right) is described by the state $|0\rangle$.

energy (the *first excited state*). Then one can produce the state $|0\rangle_n$ by cooling n such atoms to an appropriately low temperature (determined by the energy difference between the two states – the smaller that energy, the lower the temperature must be).

From the conceptual point of view, state preparation by the use of measurement gates is the simplest way. An acceptable physical candidate for a Qbit must be a system for which measurement gates are readily available. Otherwise there would be no way of extracting information from the computation, however well the unitary gates did their job. So the hardware for state preparation by measurement is already there. Whether one chooses to use it or other (e.g. cryogenic) methods to initialize the Qbits to the state $|0\rangle_n$ is a practical matter that need not concern us here. It is enough to know that it can always be done with measurement gates.

1.11 Constructing arbitrary 1- and 2-Qbit states

The art of quantum computation is to construct circuits out of 1- and 2-Qbit gates that produce final states capable of revealing useful information, when measured. The expectation is that 1-Qbit gates will be comparatively easy to construct. Two-Qbit gates that are not mere tensor products of 1-Qbit gates are likely to be substantially more difficult to make. Attention has focused strongly on the cNOT gate, and gates that can be constructed from it in combination with 1-Qbit unitaries. All of the circuits we shall be examining can, in fact, be reduced to combinations of 1-Qbit gates and 2-Qbit cNOT gates. Given the difficulty in making cNOT gates, it is generally considered desirable to keep their number as small as possible. As an illustration of such constructions, we now examine how to assign arbitrary states to one or two Qbits, starting with the standard 1-Qbit state $|0\rangle$ or the standard 2-Qbit state $|00\rangle$. (Both of these standard states can be produced with the help of measurement gates, as described in Section 1.10.)

The situation for 1-Qbit states is quite simple. Let $|\psi\rangle$ be any 1-Qbit state, and let $|\phi\rangle$ be the orthogonal state (unique to within an overall phase), satisfying $\langle\phi|\psi\rangle = 0$. Since $|0\rangle$ and $|1\rangle$ are linearly independent, there is a unique linear transformation taking them into $|\psi\rangle$ and $|\phi\rangle$. But, since $|\psi\rangle$ and $|\phi\rangle$ are an orthonormal pair (as are $|0\rangle$ and $|1\rangle$), this linear transformation is easily verified to preserve the norm of arbitrary states, so it is a unitary transformation $\mathbf{u}$. Thus, for any $|\psi\rangle$ there is a 1-Qbit unitary gate $\mathbf{u}$ that takes $|0\rangle$ into $|\psi\rangle$:

$$|\psi\rangle = \mathbf{u}|0\rangle. \tag{1.82}$$

Things are more complicated for 2-Qbit states. An unentangled 2-Qbit state, being the product of two 1-Qbit states, can be constructed out of $|00\rangle$ by the application of 1-Qbit unitaries to each of the two Qbits. But a general 2-Qbit state is entangled, and its production requires a 2-Qbit gate that is not just a tensor product of 1-Qbit unitaries. Interestingly, a single cNOT gate, combined with 1-Qbit unitaries, is enough to do the trick.

To see this, note that the general 2-Qbit state,

$$|\Psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle, \tag{1.83}$$

is of the form

$$|\Psi\rangle = |0\rangle \otimes |\psi\rangle + |1\rangle \otimes |\phi\rangle, \tag{1.84}$$

where $|\psi\rangle = \alpha_{00}|0\rangle + \alpha_{01}|1\rangle$ and $|\phi\rangle = \alpha_{10}|0\rangle + \alpha_{11}|1\rangle$. Apply $\mathbf{u} \otimes \mathbf{1}$ to $|\Psi\rangle$, where $\mathbf{u}$ is a linear transformation, whose action on the computational basis is of the form

$$\mathbf{u}|0\rangle = a|0\rangle + b|1\rangle, \qquad \mathbf{u}|1\rangle = -b^*|0\rangle + a^*|1\rangle; \qquad |a|^2 + |b|^2 = 1. \tag{1.85}$$

The transformation $\mathbf{u}$ is unitary because it preserves the orthogonality and normalization of the basis $|0\rangle, |1\rangle$.

We have

$$\begin{aligned}(\mathbf{u} \otimes \mathbf{1})|\Psi\rangle &= \big(a|0\rangle + b|1\rangle\big) \otimes |\psi\rangle + \big(-b^*|0\rangle + a^*|1\rangle\big) \otimes |\phi\rangle \\ &= |0\rangle \otimes |\psi'\rangle + |1\rangle \otimes |\phi'\rangle,\end{aligned} \tag{1.86}$$

where

$$|\psi'\rangle = a|\psi\rangle - b^*|\phi\rangle, \qquad |\phi'\rangle = b|\psi\rangle + a^*|\phi\rangle. \tag{1.87}$$

We would like to choose the complex numbers a and b to make $|\phi'\rangle$ and $|\psi'\rangle$ orthogonal. The inner product $\langle\phi'|\psi'\rangle$ is

$$\langle\phi'|\psi'\rangle = a^2\langle\phi|\psi\rangle - b^{*2}\langle\psi|\phi\rangle + ab^*\big(\langle\psi|\psi\rangle - \langle\phi|\phi\rangle\big). \tag{1.88}$$

If $\langle\phi|\psi\rangle \neq 0$, then setting $\langle\phi'|\psi'\rangle$ to 0 gives a quadratic equation for a/b^*, which has two complex solutions. If a in (1.85) is any nonzero complex number then either solution determines b, which, with a, gives a 1-Qbit unitary $\mathbf{u}$ for which

$$(\mathbf{u} \otimes \mathbf{1})|\Psi\rangle = |0\rangle \otimes |\psi'\rangle + |1\rangle \otimes |\phi'\rangle \tag{1.89}$$

where $|\psi'\rangle$ and $|\phi'\rangle$ are orthogonal. If $\langle\phi|\psi\rangle = 0$ then (1.84) is already of this form with $\mathbf{u} = \mathbf{1}$.

We can pick positive real numbers λ and μ so that $|\psi''\rangle = |\psi'\rangle/\lambda$ and $|\phi''\rangle = |\phi'\rangle/\mu$ are unit vectors, making $|\psi''\rangle$ and $|\phi''\rangle$ an orthonormal pair. They are therefore related to $|0\rangle$ and $|1\rangle$ by a unitary transformation $\mathbf{v}$:

$$|\psi''\rangle = \mathbf{v}|0\rangle, \qquad |\phi''\rangle = \mathbf{v}|1\rangle. \tag{1.90}$$

Equation (1.89) then gives[12]

$$|\Psi\rangle = \big(\mathbf{u}^\dagger \otimes \mathbf{v}\big)\big(\lambda|0\rangle \otimes |0\rangle + \mu|1\rangle \otimes |1\rangle\big). \tag{1.91}$$

We can write this as

$$|\Psi\rangle = \big(\mathbf{u}^\dagger \otimes \mathbf{v}\big)\mathbf{C}_{10}\big(\lambda|0\rangle + \mu|1\rangle\big) \otimes |0\rangle. \tag{1.92}$$

Since $|\Psi\rangle$ is a unit vector and unitary transformations preserve unit vectors, it follows from (1.91) that $\lambda|0\rangle + \mu|1\rangle$ is a unit vector. It can therefore be obtained from $|0\rangle$ by a unitary transformation $\mathbf{w}$. So

$$|\Psi\rangle = \big(\mathbf{u}^\dagger \otimes \mathbf{v}\big)\mathbf{C}_{10}\big(\mathbf{w} \otimes \mathbf{1}\big)\big(|0\rangle \otimes |0\rangle\big) = \mathbf{u}_1^\dagger \mathbf{v}_0 \mathbf{C}_{10} \mathbf{w}_1 |00\rangle. \tag{1.93}$$

We have thus established that a general 1-Qbit state $|\Psi\rangle$ can be constructed out of three 1-Qbit unitaries and a single cNOT gate, acting on the standard state $|00\rangle$. This is an early example of the usefulness of cNOT gates.

1.12 Summary: Qbits versus Cbits

Table 1.1 gives a concise comparison of the elementary properties of Cbits and Qbits. The table uses the term "Bit," with an upper-case B, to mean "Qbit or Cbit," which should be distinguished from "bit," with a lower-case b, which means "0 or 1." Alice (in the fifth line of the table) is anybody who knows the relevant history of the Qbits – their initial state preparation and the unitary gates that have subsequently acted on them.

12 This form for a general vector in a space of 2×2 dimensions is a special case of a more general result for $d \times d$ dimensions known as the *polar* (or Schmidt) *decomposition theorem*.

Table 1.1. A summary of the features of Qbits, contrasted to the analogous features of Cbits

	Cbits	**Qbits**
States of n Bits	$\|x\rangle_n, \quad 0 \le x < 2^n$	$\sum \alpha_x \|x\rangle_n, \sum \|\alpha_x\|^2 = 1$
Subsets of n Bits	Always have states	Generally have no states
Reversible operations on states	Permutations	Unitary transformations
Can state be learned from Bits?	Yes	No
To learn state of Bits	Examine them	Go ask Alice
To get information from Bits	Just look at them	Measure them
Information acquired	x	x with probability $\|\alpha_x\|^2$
State after information acquired	Same: still $\|x\rangle$	Different: now $\|x\rangle$