II Quantum computation

4 Quantum circuits

The theory of computation has traditionally been studied almost entirely in the abstract, as a topic in pure mathematics. This is to miss the point of it. Computers are physical objects, and computations are physical processes. What computers can or cannot compute is determined by the laws of physics alone, and not by pure mathematics.
– David Deutsch

Like mathematics, computer science will be somewhat different from the other sciences, in that it deals with artificial laws that can be proved, instead of natural laws that are never known with certainty.
– Donald Knuth

The opposite of a profound truth may well be another profound truth.
– Niels Bohr

This chapter begins Part II of the book, in which we explore quantum computation in detail. The chapter develops the fundamental principles of quantum computation, and establishes the basic building blocks for quantum circuits, a universal language for describing sophisticated quantum computations. The two fundamental quantum algorithms known to date are constructed from these circuits in the following two chapters. Chapter 5 presents the quantum Fourier transform and its applications to phase estimation, order-finding and factoring. Chapter 6 describes the quantum search algorithm, and its applications to database search, counting and speedup of solutions to **NP**-complete problems. Chapter 7 concludes Part II with a discussion of how quantum computation may one day be experimentally realized. Two other topics of great interest for quantum computation, quantum noise and quantum error-correction, are deferred until Part III of the book, in view of their wide interest also *outside* quantum computation.

There are two main ideas introduced in this chapter. First, we explain in detail the fundamental model of quantum computation, the quantum circuit model. Second, we demonstrate that there exists a small set of gates which are *universal*, that is, any quantum computation whatsoever can be expressed in terms of those gates. Along the way we also have occasion to describe many other basic results of quantum computation. Section 4.1 begins the chapter with an overview of quantum algorithms, focusing on what algorithms are known, and the unifying techniques underlying their construction. Section 4.2 is a detailed study of single qubit operations. Despite their simplicity, single qubit operations offer a rich playground for the construction of examples and techniques, and it is essential to understand them in detail. Section 4.3 shows how to perform multi-qubit *controlled unitary* operations, and Section 4.4 discusses the description of measurement in the quantum circuits model. These elements are then brought together in Section 4.5 for the statement and proof of the universality theorem. We summarize all the basic elements

of quantum computation in Section 4.6, and discuss possible variants of the model, and the important question of the relationship in computational power between classical and quantum computers. Section 4.7 concludes the chapter with an important and instructive application of quantum computation to the *simulation* of real quantum systems.

This chapter is perhaps the most reader-intensive of all the chapters in the book, with a high density of exercises for you to complete, and it is worth explaining the reason for this intensity. Obtaining facility with the basic elements of the quantum circuit model of computation is quite easy, but requires assimilating a large number of simple results and techniques that must become second nature if one is to progress to the more difficult problem of designing quantum algorithms. For this reason we take an example-oriented approach in this chapter, and ask you to fill in many of the details, in order to acquire such a facility. A less intensive, but somewhat superficial overview of the basic elements of quantum computation may be obtained by skipping to Section 4.6.

4.1 Quantum algorithms

What is a quantum computer good for? We're all familiar with the frustration of needing more computer resources to solve a computational problem. Practically speaking, many interesting problems are impossible to solve on a classical computer, not because they are in principle insoluble, but because of the astronomical resources required to solve realistic cases of the problem.

The spectacular promise of quantum computers is to enable new algorithms which render feasible problems requiring exorbitant resources for their solution on a classical computer. At the time of writing, two broad classes of quantum algorithms are known which fulfill this promise. The first class of algorithms is based upon Shor's *quantum Fourier transform*, and includes remarkable algorithms for solving the factoring and discrete logarithm problems, providing a striking *exponential* speedup over the best known classical algorithms. The second class of algorithms is based upon Grover's algorithm for performing *quantum searching*. These provide a less striking but still remarkable *quadratic* speedup over the best possible classical algorithms. The quantum searching algorithm derives its importance from the widespread use of search-based techniques in classical algorithms, which in many instances allows a straightforward adaptation of the classical algorithm to give a faster quantum algorithm.

Figure 4.1 sketches the state of knowledge about quantum algorithms at the time of writing, including some sample applications of those algorithms. Naturally, at the core of the diagram are the quantum Fourier transform and the quantum searching algorithm. Of particular interest in the figure is the quantum counting algorithm. This algorithm is a clever combination of the quantum searching and Fourier transform algorithms, which can be used to estimate the number of solutions to a search problem more quickly than is possible on a classical computer.

The quantum searching algorithm has many potential applications, of which but a few are illustrated. It can be used to extract statistics, such as the minimal element, from an unordered data set, more quickly than is possible on a classical computer. It can be used to speed up algorithms for some problems in **NP** – specifically, those problems for which a straightforward search for a solution is the best algorithm known. Finally, it can be used to speed up the search for keys to cryptosystems such as the widely used Data Encryption Standard (DES). These and other applications are explained in Chapter 6.

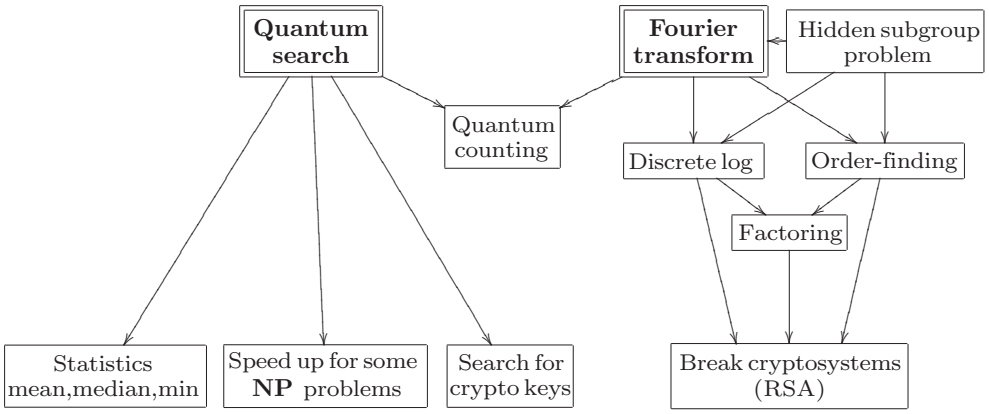


Figure 4.1. The main quantum algorithms and their relationships, including some notable applications.

The quantum Fourier transform also has many interesting applications. It can be used to solve the discrete logarithm and factoring problems. These results, in turn, enable a quantum computer to break many of the most popular cryptosystems now in use, including the RSA cryptosystem. The Fourier transform also turns out to be closely related to an important problem in mathematics, finding a hidden subgroup (a generalization of finding the period of a periodic function). The quantum Fourier transform and several of its applications, including fast quantum algorithms for factoring and discrete logarithm, are explained in Chapter 5.

Why are there so few quantum algorithms known which are better than their classical counterparts? The answer is that coming up with good quantum algorithms seems to be a difficult problem. There are at least two reasons for this. First, algorithm design, be it classical or quantum, is not an easy business! The history of algorithms shows us that considerable ingenuity is often required to come up with near optimal algorithms, even for apparently very simple problems, like the multiplication of two numbers. Finding good quantum algorithms is made doubly difficult because of the additional constraint that we want our quantum algorithms to be *better* than the best known classical algorithms. A second reason for the difficulty of finding good quantum algorithms is that our intuitions are much better adapted to the classical world than they are to the quantum world. If we think about problems using our native intuition, then the algorithms which we come up with are going to be classical algorithms. It takes special insights and special tricks to come up with good quantum algorithms.

Further study of quantum algorithms will be postponed until the next chapter. In this chapter we provide an efficient and powerful language for describing quantum algorithms, the language of quantum circuits – assemblies of discrete sets of components which describe computational procedures. This construction will enable us to quantify the cost of an algorithm in terms of things like the total number of gates required, or the circuit depth. The circuit language also comes with a toolbox of tricks that simplifies algorithm design and provides ready conceptual understanding.

4.2 Single qubit operations

The development of our quantum computational toolkit begins with operations on the simplest quantum system of all – a single qubit. Single qubit gates were introduced in Section 1.3.1. Let us quickly summarize what we learned there; you may find it useful to refer to the notes on notation on page xxiii as we go along.

A single qubit is a vector $|\psi\rangle = a|0\rangle + b|1\rangle$ parameterized by two complex numbers satisfying $|a|^2 + |b|^2 = 1$. Operations on a qubit must preserve this norm, and thus are described by 2×2 unitary matrices. Of these, some of the most important are the Pauli matrices; it is useful to list them again here:

$$X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \quad Y \equiv \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}; \quad Z \equiv \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{4.1}$$

Three other quantum gates will play a large part in what follows, the Hadamard gate (denoted H), phase gate (denoted S), and $\pi/8$ gate (denoted T):

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}; \quad S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}; \quad T = \begin{bmatrix} 1 & 0 \\ 0 & \exp(i\pi/4) \end{bmatrix}. \tag{4.2}$$

A couple of useful algebraic facts to keep in mind are that $H = (X + Z)/\sqrt{2}$ and $S = T^2$. You might wonder why the T gate is called the $\pi/8$ gate when it is $\pi/4$ that appears in the definition. The reason is that the gate has historically often been referred to as the $\pi/8$ gate, simply because up to an unimportant global phase T is equal to a gate which has $\exp(\pm i\pi/8)$ appearing on its diagonals.

$$T = \exp(i\pi/8) \begin{bmatrix} \exp(-i\pi/8) & 0 \\ 0 & \exp(i\pi/8) \end{bmatrix}. \tag{4.3}$$

Nevertheless, the nomenclature is in some respects rather unfortunate, and we often refer to this gate as the T gate.

Recall also that a single qubit in the state $a|0\rangle + b|1\rangle$ can be visualized as a point (θ, φ) on the unit sphere, where $a = \cos(\theta/2)$, $b = e^{i\varphi}\sin(\theta/2)$, and a can be taken to be real because the overall phase of the state is unobservable. This is called the Bloch sphere representation, and the vector $(\cos\varphi\sin\theta, \sin\varphi\sin\theta, \cos\theta)$ is called the Bloch vector. We shall return to this picture often as an aid to intuition.

Exercise 4.1: In Exercise 2.11, which you should do now if you haven't already done it, you computed the eigenvectors of the Pauli matrices. Find the points on the Bloch sphere which correspond to the normalized eigenvectors of the different Pauli matrices.

The Pauli matrices give rise to three useful classes of unitary matrices when they are exponentiated, the *rotation operators* about the $\hat{x}$, $\hat{y}$, and $\hat{z}$ axes, defined by the equations:

$$R_x(\theta) \equiv e^{-i\theta X/2} = \cos\frac{\theta}{2} I - i\sin\frac{\theta}{2} X = \begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix} \tag{4.4}$$

$$R_y(\theta) \equiv e^{-i\theta Y/2} = \cos\frac{\theta}{2} I - i\sin\frac{\theta}{2} Y = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix} \tag{4.5}$$

$$R_z(\theta) \equiv e^{-i\theta Z/2} = \cos\frac{\theta}{2} I - i\sin\frac{\theta}{2} Z = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}. \tag{4.6}$$

Exercise 4.2: Let x be a real number and A a matrix such that $A^2 = I$. Show that

$$\exp(iAx) = \cos(x)I + i\sin(x)A. \tag{4.7}$$

Use this result to verify Equations (4.4) through (4.6).

Exercise 4.3: Show that, up to a global phase, the $\pi/8$ gate satisfies $T = R_z(\pi/4)$.

Exercise 4.4: Express the Hadamard gate H as a product of R_x and R_z rotations and $e^{i\varphi}$ for some φ.

If $\hat{n} = (n_x, n_y, n_z)$ is a real unit vector in three dimensions then we generalize the previous definitions by defining a rotation by θ about the $\hat{n}$ axis by the equation

$$R_{\hat{n}}(\theta) \equiv \exp(-i\theta\,\hat{n}\cdot\vec{\sigma}/2) = \cos\left(\frac{\theta}{2}\right) I - i\sin\left(\frac{\theta}{2}\right)(n_x X + n_y Y + n_z Z), \tag{4.8}$$

where $\vec{\sigma}$ denotes the three component vector (X, Y, Z) of Pauli matrices.

Exercise 4.5: Prove that $(\hat{n}\cdot\vec{\sigma})^2 = I$, and use this to verify Equation (4.8).

Exercise 4.6: (**Bloch sphere interpretation of rotations**) One reason why the $R_{\hat{n}}(\theta)$ operators are referred to as rotation operators is the following fact, which you are to prove. Suppose a single qubit has a state represented by the Bloch vector $\vec{\lambda}$. Then the effect of the rotation $R_{\hat{n}}(\theta)$ on the state is to rotate it by an angle θ about the $\hat{n}$ axis of the Bloch sphere. This fact explains the rather mysterious looking factor of two in the definition of the rotation matrices.

Exercise 4.7: Show that $XYX = -Y$ and use this to prove that $XR_y(\theta)X = R_y(-\theta)$.

Exercise 4.8: An arbitrary single qubit unitary operator can be written in the form

$$U = \exp(i\alpha)R_{\hat{n}}(\theta) \tag{4.9}$$

for some real numbers α and θ, and a real three-dimensional unit vector $\hat{n}$.

1. Prove this fact.
2. Find values for α, θ, and $\hat{n}$ giving the Hadamard gate H.
3. Find values for α, θ, and $\hat{n}$ giving the phase gate

$$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}. \tag{4.10}$$

An arbitrary unitary operator on a single qubit can be written in many ways as a combination of rotations, together with global phase shifts on the qubit. The following theorem provides a means of expressing an arbitrary single qubit rotation that will be particularly useful in later applications to controlled operations.

Theorem 4.1: (**Z-Y decomposition for a single qubit**) Suppose U is a unitary operation on a single qubit. Then there exist real numbers α, β, γ and δ such that

$$U = e^{i\alpha}R_z(\beta)R_y(\gamma)R_z(\delta). \tag{4.11}$$

Proof
Since U is unitary, the rows and columns of U are orthonormal, from which it follows that there exist real numbers α, β, γ,and δ such that

$$U = \begin{bmatrix} e^{i(\alpha-\beta/2-\delta/2)} \cos\frac{\gamma}{2} & -e^{i(\alpha-\beta/2+\delta/2)} \sin\frac{\gamma}{2} \\ e^{i(\alpha+\beta/2-\delta/2)} \sin\frac{\gamma}{2} & e^{i(\alpha+\beta/2+\delta/2)} \cos\frac{\gamma}{2} \end{bmatrix}. \quad (4.12)$$

Equation (4.11) now follows immediately from the definition of the rotation matrices and matrix multiplication. □

Exercise 4.9: Explain why any single qubit unitary operator may be written in the form (4.12).

Exercise 4.10: (X-Y **decomposition of rotations**) Give a decomposition analogous to Theorem 4.1 but using R_x instead of R_z.

Exercise 4.11: Suppose $\hat{m}$ and $\hat{n}$ are non-parallel real unit vectors in three dimensions. Use Theorem 4.1 to show that an arbitrary single qubit unitary U may be written

$$U = e^{i\alpha} R_{\hat{n}}(\beta) R_{\hat{m}}(\gamma) R_{\hat{n}}(\delta), \quad (4.13)$$

for appropriate choices of α, β, γ and δ.

The utility of Theorem 4.1 lies in the following mysterious looking corollary, which is the key to the construction of controlled multi-qubit unitary operations, as explained in the next section.

Corollary 4.2: Suppose U is a unitary gate on a single qubit. Then there exist unitary operators A, B, C on a single qubit such that $ABC = I$ and $U = e^{i\alpha} AXBXC$, where α is some overall phase factor.

Proof
In the notation of Theorem 4.1, set $A \equiv R_z(\beta)R_y(\gamma/2)$, $B \equiv R_y(-\gamma/2)R_z(-(\delta+\beta)/2)$ and $C \equiv R_z((\delta-\beta)/2)$. Note that

$$ABC = R_z(\beta) R_y\left(\frac{\gamma}{2}\right) R_y\left(-\frac{\gamma}{2}\right) R_z\left(-\frac{\delta+\beta}{2}\right) R_z\left(\frac{\delta-\beta}{2}\right) = I. \quad (4.14)$$

Since $X^2 = I$, and using Exercise 4.7, we see that

$$XBX = XR_y\left(-\frac{\gamma}{2}\right) XXR_z\left(-\frac{\delta+\beta}{2}\right) X = R_y\left(\frac{\gamma}{2}\right) R_z\left(\frac{\delta+\beta}{2}\right). \quad (4.15)$$

Thus

$$AXBXC = R_z(\beta) R_y\left(\frac{\gamma}{2}\right) R_y\left(\frac{\gamma}{2}\right) R_z\left(\frac{\delta+\beta}{2}\right) R_z\left(\frac{\delta-\beta}{2}\right) \quad (4.16)$$

$$= R_z(\beta) R_y(\gamma) R_z(\delta). \quad (4.17)$$

Thus $U = e^{i\alpha} AXBXC$ and $ABC = I$, as required. □

Exercise 4.12: Give A, B, C, and α for the Hadamard gate.

Exercise 4.13: (**Circuit identities**) It is useful to be able to simplify circuits by inspection, using well-known identities. Prove the following three identities:

$$HXH = Z; \quad HYH = -Y; \quad HZH = X. \tag{4.18}$$

Exercise 4.14: Use the previous exercise to show that $HTH = R_x(\pi/4)$, up to a global phase.

Exercise 4.15: (**Composition of single qubit operations**) The Bloch representation gives a nice way to visualize the effect of composing two rotations.

(1) Prove that if a rotation through an angle β_1 about the axis $\hat{n}_1$ is followed by a rotation through an angle β_2 about an axis $\hat{n}_2$, then the overall rotation is through an angle β_{12} about an axis $\hat{n}_{12}$ given by

$$c_{12} = c_1 c_2 - s_1 s_2 \, \hat{n}_1 \cdot \hat{n}_2 \tag{4.19}$$

$$s_{12}\hat{n}_{12} = s_1 c_2 \hat{n}_1 + c_1 s_2 \hat{n}_2 - s_1 s_2 \, \hat{n}_2 \times \hat{n}_1 \, , \tag{4.20}$$

where $c_i = \cos(\beta_i/2)$, $s_i = \sin(\beta_i/2)$, $c_{12} = \cos(\beta_{12}/2)$, and $s_{12} = \sin(\beta_{12}/2)$.

(2) Show that if $\beta_1 = \beta_2$ and $\hat{n}_1 = \hat{z}$ these equations simplify to

$$c_{12} = c^2 - s^2 \, \hat{z} \cdot \hat{n}_2 \tag{4.21}$$

$$s_{12}\hat{n}_{12} = sc(\hat{z} + \hat{n}_2) - s^2 \, \hat{n}_2 \times \hat{z} \, , \tag{4.22}$$

where $c = c_1$ and $s = s_1$.

Symbols for the common single qubit gates are shown in Figure 4.2. Recall the basic properties of quantum circuits: time proceeds from left to right; wires represent qubits, and a '/' may be used to indicate a bundle of qubits.

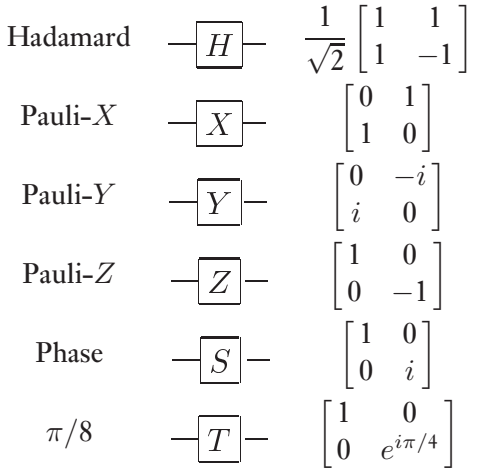


Figure 4.2. Names, symbols, and unitary matrices for the common single qubit gates.

4.3 Controlled operations

'If A is true, then do B'. This type of *controlled operation* is one of the most useful in computing, both classical and quantum. In this section we explain how complex controlled operations may be implemented using quantum circuits built from elementary operations.

The prototypical controlled operation is the controlled-NOT, which we met in Section 1.2.1. Recall that this gate, which we'll often refer to as CNOT, is a quantum gate with two input qubits, known as the *control qubit* and *target qubit*, respectively. It is drawn as shown in Figure 4.3. In terms of the computational basis, the action of the CNOT is given by $|c\rangle|t\rangle \rightarrow |c\rangle|t \oplus c\rangle$; that is, if the control qubit is set to $|1\rangle$ then the target qubit is flipped, otherwise the target qubit is left alone. Thus, in the computational basis $|\text{control}, \text{target}\rangle$ the matrix representation of CNOT is

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{4.23}$$

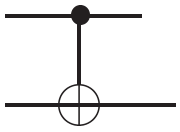


Figure 4.3. Circuit representation for the controlled-NOT gate. The top line represents the control qubit, the bottom line the target qubit.

More generally, suppose U is an arbitrary single qubit unitary operation. A *controlled-U* operation is a two qubit operation, again with a control and a target qubit. If the control qubit is set then U is applied to the target qubit, otherwise the target qubit is left alone; that is, $|c\rangle|t\rangle \rightarrow |c\rangle U^c|t\rangle$. The controlled-$U$ operation is represented by the circuit shown in Figure 4.4.

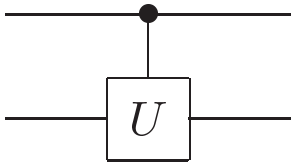
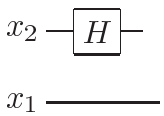
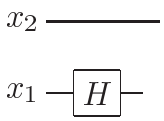


Figure 4.4. Controlled-U operation. The top line is the control qubit, and the bottom line is the target qubit. If the control qubit is set then U is applied to the target, otherwise it is left alone.

Exercise 4.16: (Matrix representation of multi-qubit gates) What is the 4×4 unitary matrix for the circuit



in the computational basis? What is the unitary matrix for the circuit



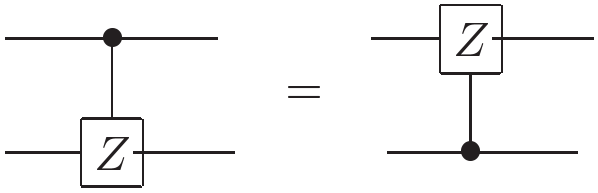
in the computational basis?

Exercise 4.17: (**Building** CNOT **from controlled-Z gates**) Construct a CNOT gate from one controlled-Z gate, that is, the gate whose action in the computational basis is specified by the unitary matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix},$$

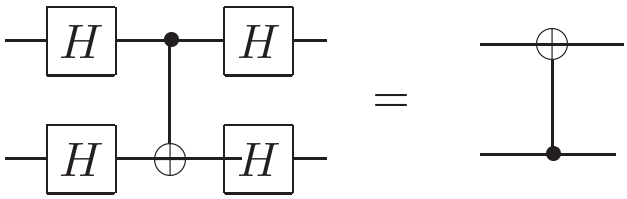
and two Hadamard gates, specifying the control and target qubits.

Exercise 4.18: Show that



Exercise 4.19: (CNOT **action on density matrices**) The CNOT gate is a simple permutation whose action on a density matrix ρ is to rearrange the elements in the matrix. Write out this action explicitly in the computational basis.

Exercise 4.20: (CNOT **basis transformations**) Unlike ideal classical gates, ideal quantum gates do not have (as electrical engineers say) 'high-impedance' inputs. In fact, the role of 'control' and 'target' are arbitrary – they depend on what basis you think of a device as operating in. We have described how the CNOT behaves with respect to the computational basis, and in this description the state of the control qubit is not changed. However, if we work in a different basis then the control qubit *does* change: we will show that its phase is flipped depending on the state of the 'target' qubit! Show that



Introducing basis states $|\pm\rangle \equiv (|0\rangle \pm |1\rangle)/\sqrt{2}$, use this circuit identity to show that the effect of a CNOT with the first qubit as control and the second qubit as target is as follows:

$$|+\rangle|+\rangle \rightarrow |+\rangle|+\rangle \tag{4.24}$$
$$|-\rangle|+\rangle \rightarrow |-\rangle|+\rangle \tag{4.25}$$
$$|+\rangle|-\rangle \rightarrow |-\rangle|-\rangle \tag{4.26}$$
$$|-\rangle|-\rangle \rightarrow |+\rangle|-\rangle. \tag{4.27}$$

Thus, with respect to this new basis, the state of the target qubit is not changed, while the state of the control qubit is flipped if the target starts as $|-\rangle$, otherwise

it is left alone. That is, in this basis, the target and control have essentially interchanged roles!

Our immediate goal is to understand how to implement the controlled-U operation for arbitrary single qubit U, using only single qubit operations and the CNOT gate. Our strategy is a two-part procedure based upon the decomposition $U = e^{i\alpha}AXBXC$ given in Corollary 4.2 on page 176.

Our first step will be to apply the phase shift $\exp(i\alpha)$ on the target qubit, controlled by the control qubit. That is, if the control qubit is $|0\rangle$, then the target qubit is left alone, while if the control qubit is $|1\rangle$, a phase shift $\exp(i\alpha)$ is applied to the target. A circuit implementing this operation using just a single qubit unitary gate is depicted on the right hand side of Figure 4.5. To verify that this circuit works correctly, note that the effect of the circuit on the right hand side is

$$|00\rangle \rightarrow |00\rangle, \quad |01\rangle \rightarrow |01\rangle, \quad |10\rangle \rightarrow e^{i\alpha}|10\rangle, \quad |11\rangle \rightarrow e^{i\alpha}|11\rangle, \tag{4.28}$$

which is exactly what is required for the controlled operation on the left hand side.

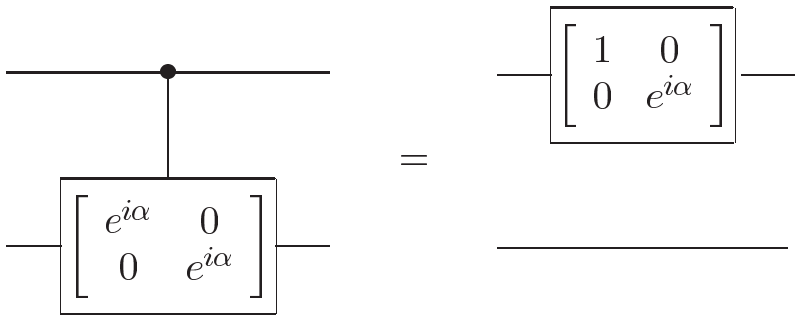


Figure 4.5. Controlled phase shift gate and an equivalent circuit for two qubits.

We may now complete the construction of the controlled-U operation, as shown in Figure 4.6. To understand why this circuit works, recall from Corollary 4.2 that U may be written in the form $U = e^{i\alpha}AXBXC$, where A, B and C are single qubit operations such that $ABC = I$. Suppose that the control qubit is set. Then the operation $e^{i\alpha}AXBXC = U$ is applied to the second qubit. If, on the other hand, the control qubit is not set, then the operation $ABC = I$ is applied to the second qubit; that is, no change is made. That is, this circuit implements the controlled-U operation.

Now that we know how to condition on a single qubit being set, what about conditioning on multiple qubits? We've already met one example of multiple qubit conditioning, the Toffoli gate, which flips the third qubit, the target qubit, conditioned on the first two qubits, the control qubits, being set to one. More generally, suppose we have $n + k$ qubits, and U is a k qubit unitary operator. Then we define the controlled operation $C^n(U)$ by the equation

$$C^n(U)|x_1x_2\ldots x_n\rangle|\psi\rangle = |x_1x_2\ldots x_n\rangle U^{x_1x_2\ldots x_n}|\psi\rangle\,, \tag{4.29}$$

where $x_1x_2\ldots x_n$ in the exponent of U means the *product* of the bits $x_1, x_2, \ldots, x_n$. That is, the operator U is applied to the last k qubits if the first n qubits are all equal to one, and otherwise, nothing is done. Such conditional operations are so useful that we

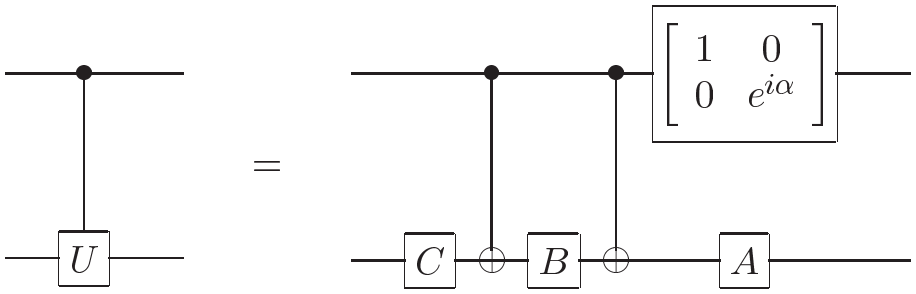


Figure 4.6. Circuit implementing the controlled-U operation for single qubit U. α, A, B and C satisfy $U = \exp(i\alpha)AXBXC$, $ABC = I$.

introduce a special circuit notation for them, illustrated in Figure 4.7. For the following we assume that $k = 1$, for simplicity. Larger k can be dealt with using essentially the same methods, however for $k \geq 2$ there is the added complication that we don't (yet) know how to perform arbitrary operations on k qubits.

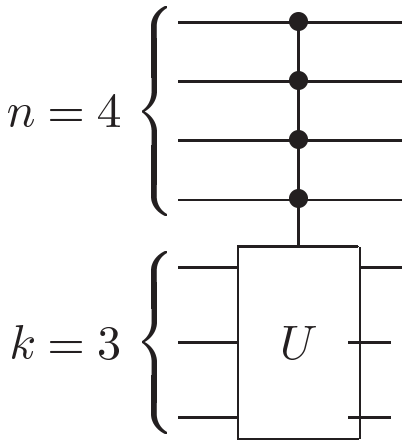


Figure 4.7. Sample circuit representation for the $C^n(U)$ operation, where U is a unitary operator on k qubits, for $n = 4$ and $k = 3$.

Suppose U is a single qubit unitary operator, and V is a unitary operator chosen so that $V^2 = U$. Then the operation $C^2(U)$ may be implemented using the circuit shown in Figure 4.8.

Exercise 4.21: Verify that Figure 4.8 implements the $C^2(U)$ operation.

Exercise 4.22: Prove that a $C^2(U)$ gate (for any single qubit unitary U) can be constructed using at most eight one-qubit gates, and six controlled-NOTs.

Exercise 4.23: Construct a $C^1(U)$ gate for $U = R_x(\theta)$ and $U = R_y(\theta)$, using only CNOT and single qubit gates. Can you reduce the number of single qubit gates needed in the construction from three to two?

The familiar Toffoli gate is an especially important special case of the $C^2(U)$ operation,

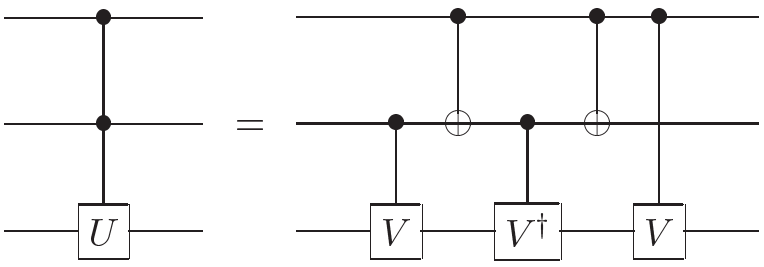


Figure 4.8. Circuit for the $C^2(U)$ gate. V is any unitary operator satisfying $V^2 = U$. The special case $V \equiv (1-i)(I+iX)/2$ corresponds to the Toffoli gate.

the case $C^2(X)$. Defining $V \equiv (1-i)(I+iX)/2$ and noting that $V^2 = X$, we see that Figure 4.8 gives an implementation of the Toffoli gate in terms of one and two qubit operations. From a classical viewpoint this is a remarkable result; recall from Problem 3.5 that one and two bit classical reversible gates are not sufficient to implement the Toffoli gate, or, more generally, universal computation. By contrast, in the quantum case we see that one and two qubit reversible gates are sufficient to implement the Toffoli gate, and will eventually prove that they suffice for universal computation.

Ultimately we will show that any unitary operation can be composed to an arbitrarily good approximation from just the Hadamard, phase, controlled-NOT and $\pi/8$ gates. Because of the great usefulness of the Toffoli gate it is interesting to see how it can be built from just this gate set. Figure 4.9 illustrates a simple circuit for the Toffoli gate made up of just Hadamard, phase, controlled-NOT and $\pi/8$ gates.

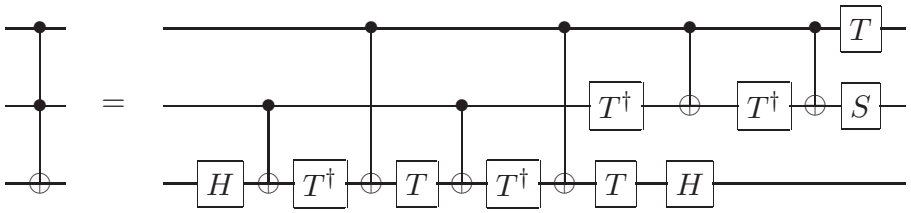


Figure 4.9. Implementation of the Toffoli gate using Hadamard, phase, controlled-NOT and $\pi/8$ gates.

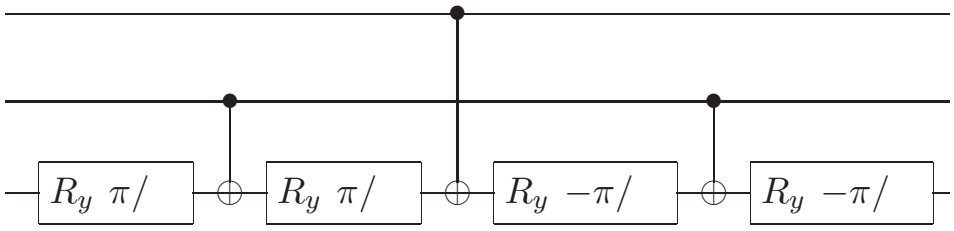
Exercise 4.24: Verify that Figure 4.9 implements the Toffoli gate.

Exercise 4.25: (Fredkin gate construction) Recall that the Fredkin (controlled-swap) gate performs the transform

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.30}$$

(1) Give a quantum circuit which uses three Toffoli gates to construct the Fredkin gate (*Hint*: think of the swap gate construction – you can control each gate, one at a time).
(2) Show that the first and last Toffoli gates can be replaced by CNOT gates.
(3) Now replace the middle Toffoli gate with the circuit in Figure 4.8 to obtain a Fredkin gate construction using only six two-qubit gates.
(4) Can you come up with an even simpler construction, with only five two-qubit gates?

Exercise 4.26: Show that the circuit:



differs from a Toffoli gate only by relative phases. That is, the circuit takes $|c_1, c_2, t\rangle$ to $e^{i\theta(c_1,c_2,t)}|c_1, c_2, t \oplus c_1 \cdot c_2\rangle$, where $e^{i\theta(c_1,c_2,t)}$ is some relative phase factor. Such gates can sometimes be useful in experimental implementations, where it may be much easier to implement a gate that is the same as the Toffoli up to relative phases than it is to do the Toffoli directly.

Exercise 4.27: Using just CNOTs and Toffoli gates, construct a quantum circuit to perform the transformation

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{4.31}$$

This kind of partial cyclic permutation operation will be useful later, in Chapter 7.

How may we implement $C^n(U)$ gates using our existing repertoire of gates, where U is an arbitrary single qubit unitary operation? A particularly simple circuit for achieving this task is illustrated in Figure 4.10. The circuit divides up into three stages, and makes use of a small number $(n-1)$ of working qubits, which all start and end in the state $|0\rangle$. Suppose the control qubits are in the computational basis state $|c_1, c_2, \ldots, c_n\rangle$. The first stage of the circuit is to reversibly AND all the control bits $c_1, \ldots, c_n$ together to produce the product $c_1 \cdot c_2 \ldots c_n$. To do this, the first gate in the circuit ANDs c_1 and c_2 together, using a Toffoli gate, changing the state of the first work qubit to $|c_1 \cdot c_2\rangle$. The next Toffoli gate ANDs c_3 with the product $c_1 \cdot c_2$, changing the state of the second work qubit to $|c_1 \cdot c_2 \cdot c_3\rangle$. We continue applying Toffoli gates in this fashion, until the final work qubit is in the state $|c_1 \cdot c_2 \ldots c_n\rangle$. Next, a U operation on the target qubit is

performed, conditional on the final work qubit being set to one. That is, U is applied if and only if all of c_1 through c_n are set. Finally, the last part of the circuit just reverses the steps of the first stage, returning all the work qubits to their initial state, $|0\rangle$. The combined result, therefore, is to apply the unitary operator U to the target qubit, if and only if all the control bits c_1 through c_n are set, as desired.

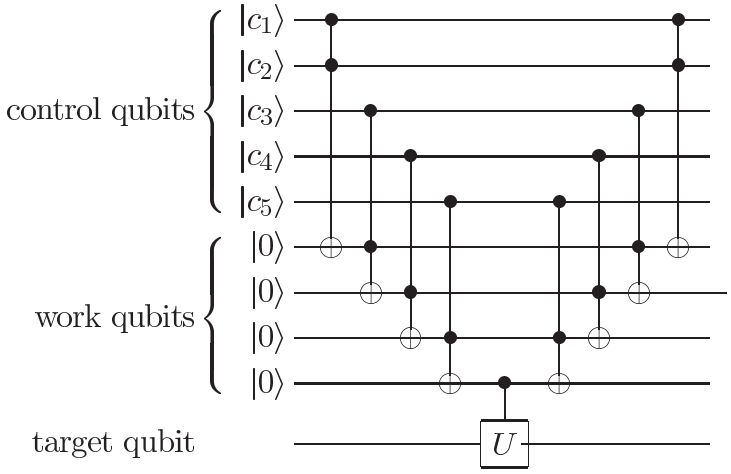


Figure 4.10. Network implementing the $C^n(U)$ operation, for the case $n = 5$.

Exercise 4.28: For $U = V^2$ with V unitary, construct a $C^5(U)$ gate analogous to that in Figure 4.10, but using no work qubits. You may use controlled-V and controlled-$V^\dagger$ gates.

Exercise 4.29: Find a circuit containing $O(n^2)$ Toffoli, CNOT and single qubit gates which implements a $C^n(X)$ gate (for $n > 3$), using no work qubits.

Exercise 4.30: Suppose U is a single qubit unitary operation. Find a circuit containing $O(n^2)$ Toffoli, CNOT and single qubit gates which implements a $C^n(U)$ gate (for $n > 3$), using no work qubits.

In the controlled gates we have been considering, conditional dynamics on the target qubit occurs if the control bits are set to *one*. Of course, there is nothing special about one, and it is often useful to consider dynamics which occur conditional on the control bit being set to zero. For instance, suppose we wish to implement a two qubit gate in which the second ('target') qubit is flipped, conditional on the first ('control') qubit being set to zero. In Figure 4.11 we introduce a circuit notation for this gate, together with an equivalent circuit in terms of the gates we have already introduced. Generically we shall use the open circle notation to indicate conditioning on the qubit being set to zero, while a closed circle indicates conditioning on the qubit being set to one.

A more elaborate example of this convention, involving three control qubits, is illustrated in Figure 4.12. The operation U is applied to the target qubit if the first and third qubits are set to zero, and the second qubit is set to one. It is easy to verify by inspection that the circuit on the right hand side of the figure implements the desired operation. More generally, it is easy to move between circuits which condition on qubits being set

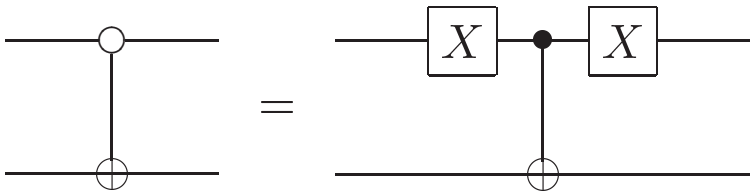


Figure 4.11. Controlled operation with a NOT gate being performed on the second qubit, conditional on the first qubit being set to zero.

to one and circuits which condition on qubits being set to zero, by insertion of X gates in appropriate locations, as illustrated in Figure 4.12.

Another convention which is sometimes useful is to allow controlled-NOT gates to have multiple targets, as shown in Figure 4.13. This natural notation means that when the control qubit is 1, then all the qubits marked with a $\oplus$ are flipped, and otherwise nothing happens. It is convenient to use, for example, in constructing classical functions such as permutations, or in encoders and decoders for quantum error-correction circuits, as we shall see in Chapter 10.

Exercise 4.31: (**More circuit identities**) Let subscripts denote which qubit an operator acts on, and let C be a CNOT with qubit 1 the control qubit and qubit 2 the target qubit. Prove the following identities:

$$CX_1C = X_1X_2 \tag{4.32}$$
$$CY_1C = Y_1X_2 \tag{4.33}$$
$$CZ_1C = Z_1 \tag{4.34}$$
$$CX_2C = X_2 \tag{4.35}$$
$$CY_2C = Z_1Y_2 \tag{4.36}$$
$$CZ_2C = Z_1Z_2 \tag{4.37}$$
$$R_{z,1}(\theta)C = CR_{z,1}(\theta) \tag{4.38}$$
$$R_{x,2}(\theta)C = CR_{x,2}(\theta). \tag{4.39}$$

4.4 Measurement

A final element used in quantum circuits, almost implicitly sometimes, is measurement. In our circuits, we shall denote a projective measurement in the computational basis (Section 2.2.5) using a 'meter' symbol, illustrated in Figure 4.14. In the theory of quantum circuits it is conventional to not use any special symbols to denote more general measurements, because, as explained in Chapter 2, they can always be represented by unitary transforms with ancilla qubits followed by projective measurements.

There are two important principles that it is worth bearing in mind about quantum circuits. Both principles are rather obvious; however, they are of such great utility that they are worth emphasizing early. The first principle is that classically conditioned operations can be replaced by quantum conditioned operations:

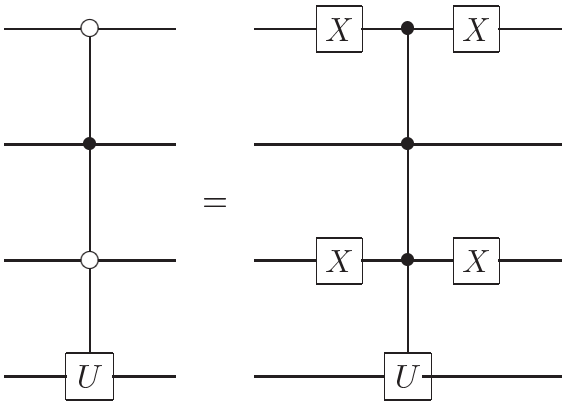


Figure 4.12. Controlled-U operation and its equivalent in terms of circuit elements we already know how to implement. The fourth qubit has U applied if the first and third qubits are set to zero, and the second qubit is set to one.

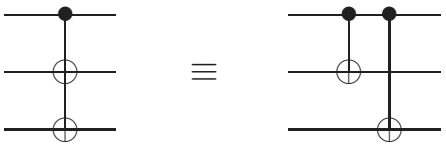


Figure 4.13. Controlled-NOT gate with multiple targets.

Principle of deferred measurement: Measurements can always be moved from an intermediate stage of a quantum circuit to the end of the circuit; if the measurement results are used at any stage of the circuit then the classically controlled operations can be replaced by conditional quantum operations.

Often, quantum measurements are performed as an intermediate step in a quantum circuit, and the measurement results are used to conditionally control subsequent quantum gates. This is the case, for example, in the teleportation circuit of Figure 1.13 on page 27. However, such measurements can *always* be moved to the end of the circuit. Figure 4.15 illustrates how this may be done by replacing all the classical conditional operations by corresponding quantum conditional operations. (Of course, some of the interpretation of this circuit as performing 'teleportation' is lost, because no classical information is transmitted from Alice to Bob, but it is clear that the overall action of the two quantum circuits is the same, which is the key point.)

The second principle is even more obvious – and surprisingly useful!

Figure 4.14. Symbol for projective measurement on a single qubit. In this circuit nothing further is done with the measurement result, but in more general quantum circuits it is possible to change later parts of the quantum circuit, *conditional* on measurement outcomes in earlier parts of the circuit. Such a usage of classical information is depicted using wires drawn with double lines (not shown here).

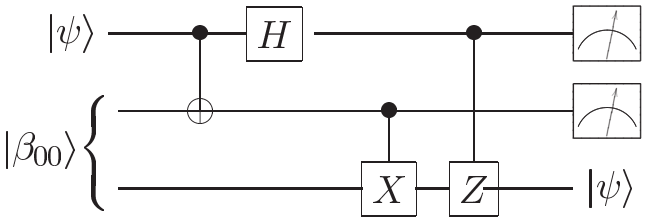


Figure 4.15. Quantum teleportation circuit in which measurements are done at the end, instead of in the middle of the circuit. As in Figure 1.13, the top two qubits belong to Alice, and the bottom one to Bob.

Principle of implicit measurement: Without loss of generality, any unterminated quantum wires (qubits which are not measured) at the end of a quantum circuit may be assumed to be measured.

To understand why this is true, imagine you have a quantum circuit containing just two qubits, and only the first qubit is measured at the end of the circuit. Then the measurement statistics observed at this time are completely determined by the reduced density matrix of the first qubit. However, if a measurement had also been performed on the second qubit, then it would be highly surprising if that measurement could change the statistics of measurement on the first qubit. You'll prove this in Exercise 4.32 by showing that the reduced density matrix of the first qubit is not affected by performing a measurement on the second.

As you consider the role of measurements in quantum circuits, it is important to keep in mind that in its role as an interface between the quantum and classical worlds, measurement is generally considered to be an irreversible operation, destroying quantum information and replacing it with classical information. In certain carefully designed cases, however, this need not be true, as is vividly illustrated by teleportation and quantum error-correction (Chapter 10). What teleportation and quantum error-correction have in common is that in neither instance does the measurement result reveal any information about the identity of the quantum state being measured. Indeed, we will see in Chapter 10 that this is a more general feature of measurement – in order for a measurement to be reversible, it must reveal no information about the quantum state being measured!

Exercise 4.32: Suppose ρ is the density matrix describing a two qubit system. Suppose we perform a projective measurement in the computational basis of the second qubit. Let $P_0 = |0\rangle\langle 0|$ and $P_1 = |1\rangle\langle 1|$ be the projectors onto the $|0\rangle$ and $|1\rangle$ states of the second qubit, respectively. Let ρ' be the density matrix which would be assigned to the system after the measurement by an observer who did not learn the measurement result. Show that

$$\rho' = P_0 \rho P_0 + P_1 \rho P_1 \,. \tag{4.40}$$

Also show that the reduced density matrix for the first qubit is not affected by the measurement, that is, $\mathrm{tr}_2(\rho) = \mathrm{tr}_2(\rho')$.

Exercise 4.33: (Measurement in the Bell basis) The measurement model we have specified for the quantum circuit model is that measurements are performed only