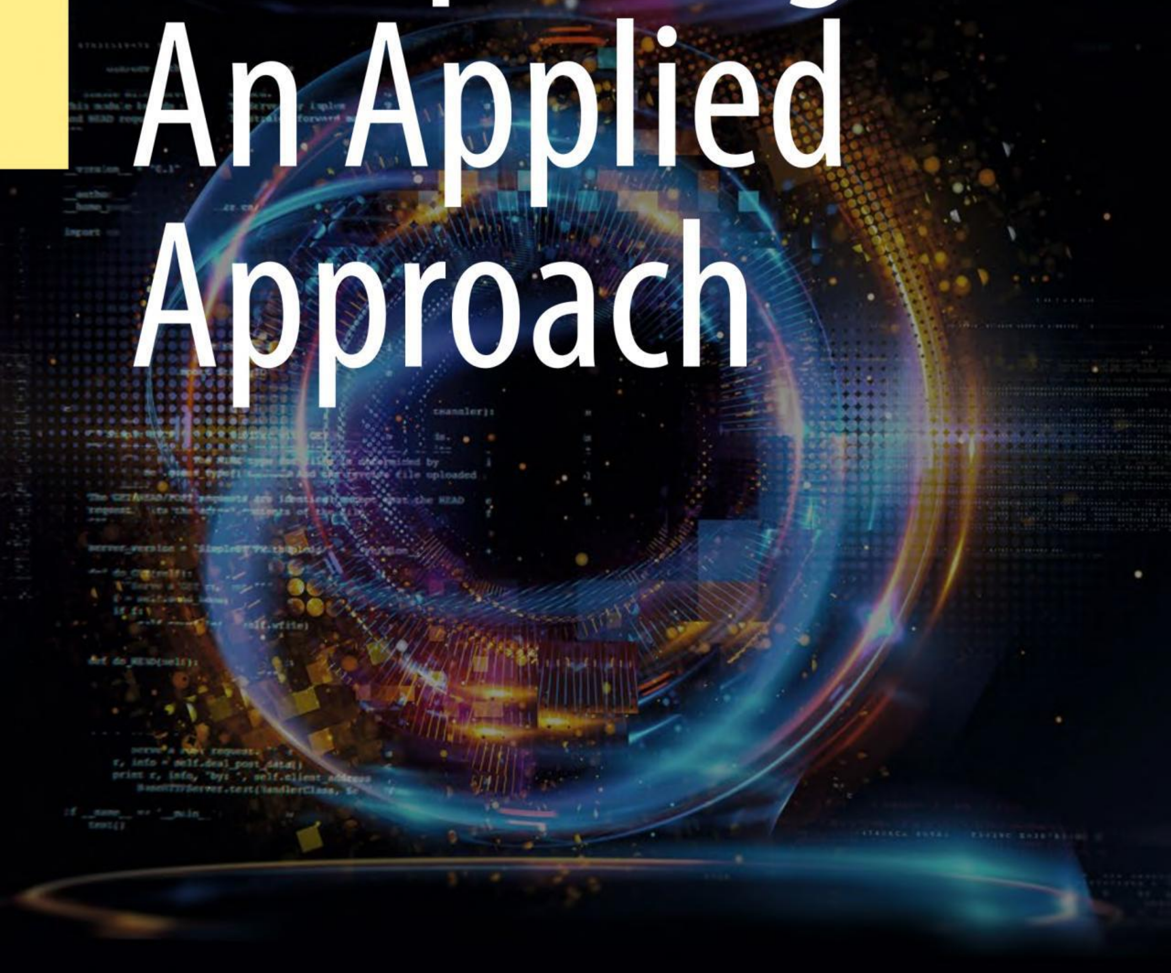
Jack D. Hidary

Quantum Computing: An Applied Approach



Springer

Copyrighted material

Jack D. Hidary

Quantum Computing: An Applied Approach

Springer

Copyrighted material

Jack D. Hidary
Alphabet X
Mountain View, CA
USA

ISBN 978-3-030-23921-3 ISBN 978-3-030-23922-0 (eBook)
https://doi.org/10.1007/978-3-030-23922-0

© Jack D. Hidary under exclusive license to Springer Nature Switzerland AG 2019
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Copyrighted material

Contents

Copyrighted material

Copyrighted material

Copyrighted material

Copyrighted material

Copyrighted material

Copyrighted material

Preface

We are entering a new era of computation that will catalyze discoveries in science and technology. Novel computing platforms will probe the fundamental laws of our universe and aid in solving hard problems that affect all of us. Machine learning programs powered by specialized chips are already yielding breakthrough after breakthrough.

In this book we will explore quantum computing – an emerging platform that is fundamentally different than the way we compute with current digital platforms. To be sure, we are years away from scaled quantum computers. Yet, we now know that such systems are possible; with advances in engineering we are likely to see real impact.

Quantum computing is part of the larger field of quantum information sciences (QIS). All three branches of QIS – computation, communication and sensing – are advancing at rapid rates and a discovery in one area can spur progress in another. Quantum communication leverages the unusual properties of quantum systems to transmit information in a manner that no eavesdropper can read. This field is becoming increasingly critical as quantum computing drives us to a post-quantum cryptography regime. We will cover quantum teleportation and superdense coding, which are both quantum-specific protocols, in chapter 7.

Quantum sensing is a robust field of research which uses quantum devices to move beyond classical limits in sensing magnetic and other fields. For example, there is an emerging class of sensors for detecting position, navigation and timing (PNT) at the atomic scale. These micro-PNT devices can provide highly accurate positioning data when GPS is jammed or unavailable.

In this book we will focus on quantum computation. One of the critical differences between quantum and classical computation is that in quantum computation *we are manipulating quantum states themselves*; this gives us a much larger computing space to work in than in classical computers. In classical computers, if we wish to model a real-world quantum physical

Copyrighted material

system, we can only do so with representations of such a system and we cannot implement the physics itself.

This key difference leads to exciting possibilities for the future of computing and science. All this starts with fundamental truths about our world that were developed during the quantum mechanics revolution in the first half of the 20th century. We will review a number of these core concepts in the first chapter.

I had the fortunate circumstance to have studied quantum mechanics before learning classical physics and therefore relate to quantum physics as the norm – it is my intellectual home. Until we change our educational system, most students will learn the classical before the quantum and so the quantum will seem doubly strange — both from their own human experience as well as from the inculcation of classical ideas before quantum ideas can be introduced.

What is ironic about this state of affairs is that the primary mathematical tool in quantum mechanics is linear algebra, a powerful but very accessible branch of mathematics. Most students, however, only take linear algebra after two or three semesters of calculus, if they take it at all. Yet, no calculus is needed to introduce linear algebra! In any case, we will leave the remedying of mathematics education to another day while we embark here on a journey into a new form of computing.

In this book we will explore how to build a computer of a very different kind than humans have ever built before. What is distinct about this book is that we will go beyond the theoretical into the practical work of how we can build such computers and *how we can write applications for these systems*. There are now several development libraries which we can use to program cloud-based quantum systems. We will walk through code examples and show the reader how to build a quantum circuit comprised of a set of operators to address a particular challenge. We will mainly use Python in this book.

We are currently in the regime of **n**oisy **i**ntermediate-**s**cale **q**uantum (NISQ) computers, a term coined by John Preskill of CalTech [176]. This refers to systems that do not yet have full error-correction (thus noisy) and have dozens to thousands of qubits – well short of the 10^6+ necessary for scaled fault-tolerant computing. Despite the limitations of these initial systems, the theory, algorithms and coding techniques we cover in this book will serve readers as they transition to larger systems that are to come in the future.

Copyrighted material

This work is three books in one: the first part covers the necessary framework that drives the design of quantum computers and circuits. We will also explore what kinds of problems may be amenable to quantum computation in our treatment of complexity classes.

The second part of the book is for those readers who wish to delve into the programming that makes these new machines tick. If you already have a background in quantum mechanics, quantum information theory and theoretical computer science (you know who you are!), you can jump right to the second part and dig into the code. Please refer to the navigation guide in the following pages to chart a course through this material.

In the third part we provide a set of critical tools to use in the journey to master quantum computing (QC). We build up the core concepts of linear algebra and tie them specifically to their use in QC. The table of operators and circuit elements in chapter 14 is a handy reference as you design your own quantum computing protocols.

The book is also a portal to the growing body of literature on the subject. We recommend that the reader use the bibliography to explore both foundational and recent papers in the field.

We will provide further online examples and code tutorials on a continual basis. This a living text that will develop as QC technology matures. We are all travelers together on this new adventure; join us online at this book's GitHub site.[1] We are excited to see what you will develop with these new platforms and tools. Contact us via the site — we look forward to hearing from you.

Jack D. Hidary
June 2019
35,000 ft up

[1] `http://www.github.com/jackhidary/quantumcomputingbook`

Copyrighted material

Acknowledgements

Let me start by thanking my publisher, Elizabeth Loew, who has been so supportive throughout the process, and the entire SpringerNature team for their excellence.

A book like this is a significant undertaking and it took a team of people to help in so many ways. The greatest of appreciation to Stefan Leichenauer who did a great job as book editor and formatter-in-chief. Stefan reviewed large parts of the book and I thank him for his commitment to the project. The entire book is written in TeX and we often pushed the boundaries of what TeX is capable of implementing.

Thank you to Sheldon Axler, author of *Linear Algebra Done Right* (also by Springer) who generously shared his TeX template so that we could format the book correctly for the Springer standards.

Thanks to the many experts who gave of their time to review key sections of the book and provide technical advice. These include (in alphabetical order): Scott Aaronson, Ryan Babbush, Sergio Boixo, Ruffin Evans, Eddie Farhi, Patrick Hayden, Gerry Gilbert, Matt Reagor and Lenny Susskind. Each improved the work materially with their input.

I would also like to recognize the significant achievement of Michael Nielsen and Isaac Chuang in developing their textbook [161]. Recognition as well to John Preskill for his in-depth lecture notes [174]. Nielsen, Chuang and Preskill as well as Mermin [151] and Rieffel [186] have helped many individuals enter the field.

Thanks to my many colleagues at Alphabet X and Google who have encouraged this effort including: Sergey Brin, Astro Teller and Hartmut Neven and his excellent team.

Thank you to the many participants in the workshops I taught on quantum computing as well as my courses on linear algebra and other mathematical topics. Your feedback has been invaluable.

Copyrighted material

Hearty thanks to James Myer who worked with me intensively on the math sections; James' methodical approach assured us of success. We reviewed these sections countless times, continually reworking them. James is not only passionate about mathematics; he also cares deeply about pedagogy and we had productive discussions on the best way to present the core concepts. Thank you as well to Tai-Danae Bradley who also reviewed the math sections and made very helpful suggestions.

Thank you to Ryan LaRose who worked with me on the code sections. In this emerging field where the code frameworks have only been developed within the last few years, solid information and examples can be hard to come by. Ryan combined great skill in research as well as compiling the information in succinct forms. Ryan also did a great job on the book's GitHub site.

Thank you to Ellen Cassidy who did such a professional job proofreading the text for grammar and consistency of format. Ellen has an eagle eye and I commend her work to any author. Joe Tricot also did a wonderful job coordinating the overall process.

Naturally, even with all this help, there will remain items to fix for the next edition. All remaining errors are mine and I will be posting updates on the GitHub site and then include fixes in upcoming versions.

Thank you to my parents, David and Aimee Hidary, and my entire wonderful family for their support through this process. It is a great feeling to share this accomplishment with you.

Copyrighted material

Navigating this Book

Here are our suggestions to make the best use of this book:

1. *University instructors:* You can build several different courses with the material in this book. All code from the book is on the book's website. The math chapters have exercises embedded throughout; for other chapters please consult the online site for coding exercises and other problem sets.
   (a) Course in Quantum Computing for STEM majors:
      i. For this course we recommend assigning chapters 1 and 2 as pre-reading for the course and then proceeding chapter by chapter with the exercises provided on the GitHub site. Solutions are also available on the site.
      ii. If the students do not have sufficient depth in formal linear algebra and related mathematical tools, Part III forms a strong basis for a multi-week treatment with exercises.
   (b) Course in Quantum Computing for physics graduate students:
      i. For this course, we recommend using this book in conjunction with Mike and Ike (which is the way many of us refer to Nielsen and Chuang's excellent textbook [161]) or another suitable text which covers the theoretical concepts in depth. We all owe a huge debt of gratitude to Michael Nielsen, Isaac Chuang and authors of other textbooks over the last twenty years. We also recommend referring to John Preskill's lecture notes as you build your course for advanced physics students [174]. Our work is meant to be complementary to Mike and Ike in several respects:
         A. This work is more focused on coding. For obvious reasons, books written prior to the past few years could not have covered the dev tools and Python-based approaches to quantum computing that now exist.

Copyrighted material

            B. This book does not go into the depth that Mike and Ike does on information theoretic concepts.
            C. This book's mathematical tools section has a more detailed ramp-up for those students who may not have taken a rigorous linear algebra course. The short summaries of linear algebra and other requisite math tools in other textbooks on quantum mechanics are often insufficient in our experience.
        ii. We recommend first assigning chapters 1 and 2 as pre-reading.
        iii. Next, we suggest covering the chapters on unitary operators, measurement and quantum circuits with exercises on the Github site to check knowledge.
        iv. We then recommend spending the bulk of the course in Part II to provide the students with hands-on experience with the code.
    (c) Course in Quantum Computing for CS graduate students:
        i. We suggest assigning the first two chapters as pre-reading and then a review of mathematical tools in Part III. Prior exposure to only undergraduate linear algebra is typically insufficient as it was most likely taught without the full formalism.
        ii. We then recommend chapters 3 and 4 to build up familiarity with unitary operators, measurement and complexity classes in the quantum regime. The instructor can make use of the review questions and answers on the GitHub site.
        iii. The course can then cover the approaches to building a quantum computer followed by all the coding chapters.

    Please check the book's GitHub site to find additional resources including: code from the book, problem sets, solutions, links to videos and other pedagogical resources.

2. *Physicists:* For physicists who specialize in fields outside of quantum computing and wish to ramp up quickly in this area, we recommend reading the brief history of QC as we provide more detail than typical treatments, then the survey of quantum hardware followed by the applications in the second part of the book.
3. *Software engineers:* We recommend starting with the opening two chapters, then reviewing the toolkits in Part III. We then suggest returning to the treatment of qubits and unitary operators in Part I and proceeding from there.

Copyrighted material

4. *Engineering and business leaders:* For readers who will not be doing hands-on coding, we recommend focusing on chapters 1-4. The more adventurous may want to work through some of the code examples to get a tangible feel for the algorithms.
5. *Independent study:* This book can easily be used as a text for independent study. We recommend combining it with online resources. Please consult the GitHub site for an updated list of resources:

   `http://www.github.com/jackhidary/quantumcomputingbook`

   We recommend first assessing your current fluency on the core tools in Part III; there are numerous self-tests throughout the section that can be used for this purpose. The reader can then proceed to Part I.

   For those with a strong background in quantum mechanics and/or information theory we recommend looking up the papers referenced in chapters 2-4 to gain a deeper understanding of the state of the field before proceeding to Part II: Hardware and Applications.

Please consult the book's GitHub site to find a range of resources including: code from the book, problem sets, solutions, links to videos and other pedagogical resources.

Copyrighted material

Part I

Foundations

Copyrighted material

CHAPTER 1

Superposition, Entanglement and Reversibility

What is a quantum computer? The answer to this question encompasses quantum mechanics (QM), quantum information theory (QIT) and computer science (CS).

For our purposes, we will focus on the core of what makes a quantum computer distinct from classical computers.

1.1 Quantum Computer Definition

A quantum computer is a device that leverages specific properties described by quantum mechanics to perform computation.

Every classical (that is, non-quantum) computer can be described by quantum mechanics since quantum mechanics is the basis of the physical universe. However, a classical computer does not take advantage of the specific properties and states that quantum mechanics affords us in doing its calculations.

To delve into the specific properties we use in quantum computers, let us first discuss a few key concepts of quantum mechanics:

- How do we represent the superposition of states in a quantum system?
- What is entanglement?
- What is the connection between reversibility, computation and physical systems?

We will be using Dirac notation, linear algebra and other tools extensively in this text; readers are encouraged to refer to the math chapters later in this work to review as needed.

According to the principles of quantum mechanics, systems are set to a definite state only once they are measured. Before a measurement, systems

© Jack D. Hidary under exclusive license to Springer Nature Switzerland AG 2019
J. D. Hidary, *Quantum Computing: An Applied Approach*,
https://doi.org/10.1007/978-3-030-23922-0_1

Copyrighted material

are in an indeterminate state; after we measure them, they are in a definite state. If we have a system that can take on one of two discrete states when measured, we can represent the two states in Dirac notation as $|0\rangle$ and $|1\rangle$. We can then represent a *superposition of states* as a linear combination of these states, such as

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

1.2 The Superposition Principle

The linear combination of two or more state vectors is another state vector in the same Hilbert space[a] and describes another state of the system.

[a]See Part III for a treatment of Hilbert spaces

As an example, let us consider a property of light that illustrates a superposition of states. Light has an intrinsic property called *polarization* which we can use to illustrate a superposition of states. In almost all of the light we see in everyday life — from the sun, for example — there is no preferred direction for the polarization. Polarization states can be selected by means of a *polarizing filter*, a thin film with an axis that only allows light with polarization parallel to that axis to pass through.

With a single polarizing filter, we can select one polarization of light, for example *vertical polarization*, which we can denote as $|\uparrow\rangle$. *Horizontal polarization*, which we can denote as $|\rightarrow\rangle$, is an orthogonal state to vertical polarization[1]. Together, these states form a basis for any polarization of light. That is, any polarization state $|\psi\rangle$ can be written as linear combination of these states. We use the Greek letter ψ to denote the state of the system

$$|\psi\rangle = \alpha|\uparrow\rangle + \beta|\rightarrow\rangle$$

The coefficients α and β are complex numbers known as *amplitudes*. The coefficient α is associated with vertical polarization and the coefficient β is associated with horizontal polarization. These have an important interpretation in quantum mechanics which we will see shortly.

After selecting vertical polarization with a polarizing filter, we can then introduce a second polarizing filter after the first. Imagine we oriented the

[1]We could have equally used $|0\rangle$ and $|1\rangle$ to denote the two polarization states; the labels used in kets are arbitrary.

Copyrighted material

axis of the second filter perpendicular to the axis of the first. Would we see any light get through the second filter?

If you answered no to this question, you would be correct. The horizontal state $|{\rightarrow}\rangle$ is orthogonal to the first, so there is no amount of horizontal polarization after the first vertical filter.

Suppose now we oriented the axis of the second polarizing filter at 45° (i.e., along the diagonal $\nearrow$ between vertical $\uparrow$ and horizontal $\rightarrow$) to the first instead of horizontally. Now we ask the same question — would we see any light get through the second filter?

If you answered no to this question, you may be surprised to find the answer is *yes*. We would, in fact, see some amount of light get through the second filter. How could this be if all light after the first filter has vertical polarization? The reason is that we can express vertical polarization as a *superposition* of diagonal components. That is, letting $|{\nearrow}\rangle$ denote 45° polarization and $|{\nwarrow}\rangle$ denote −45° polarization, we may write

$$|{\uparrow}\rangle = \frac{1}{\sqrt{2}}|{\nearrow}\rangle + \frac{1}{\sqrt{2}}|{\nwarrow}\rangle$$

As you may expect from geometric intuition, the vertical state consists of equal parts $|{\nearrow}\rangle$ and $|{\nwarrow}\rangle$.

It is for this reason that we see some amount of light get past the second filter. Namely, the vertical polarization can be written as a *superposition* of states, one of which is precisely the 45° diagonal state $|{\nearrow}\rangle$ we are allowing through the second filter. Since the $|{\nearrow}\rangle$ state is only one term in the superposition, not all of the light gets through the filter, but some does. The amount that gets transmitted is precisely $1/2$ in this case. (More formally, the intensity of the transmitted light is $1/2$ that of the incident light.) This value is determined from the amplitudes of the superposition state by a law known as Born's rule, which we now discuss.

Max Born demonstrated in his 1926 paper that **the modulus squared of the amplitude of a state is the probability of that state resulting after measurement** [38]. In this case, since the amplitude is $\frac{1}{\sqrt{2}}$ the probability of obtaining that state is $\left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2}$, so the probability of measuring the light in either the vertical or horizontal polarization state is 50%. Note that we chose an amplitude of $\frac{1}{\sqrt{2}}$ in order to *normalize* the states so that the sum of the modulus squared of the amplitudes will equal one; this enables us to connect the amplitudes to probabilities of measurement with the Born rule.

Copyrighted material

1.3 The Born rule

In a superposition of states, the modulus squared of the amplitude of a state is the probability of that state resulting after measurement. Furthermore, the sum of the squares of the amplitudes of all possible states in the superposition is equal to 1. So, for the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, we have

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

While in the polarization example above we have a 50/50 split in probability for each of two states, if we examined some other physical system it may have a 75/25 split or some other probability distribution. One critical difference between classical and quantum mechanics is that amplitudes (*not probabilities*) can be complex numbers.

In other words, the coefficients α and β which appear in the statement of the Born rule can be complex numbers, such as $i := \sqrt{-1}$ or $(1 + i)/\sqrt{2}$. It is only after we take the square of the modulus of these amplitudes that we get real numbers, hence actual probabilities. Refer to chapter 11 to review complex numbers and how to determine the square of the modulus of a complex number.

As if quantum superposition were not odd enough, QM describes a specific kind of superposition which stretches our imagination even further: *entanglement*. In 1935, when **E**instein worked with **P**odolsky and **R**osen to publish their paper on quantum entanglement [75], their aim was to attack the edifice of QM (this paper is now known as EPR). Even though Einstein earned the Nobel Prize for his 1905 work on the quantum nature of the photoelectric effect, he nevertheless railed against the implications of QM until his later years.

Einstein wrote in 1952 that quantum mechanics appears to him to be "a system of delusion of an exceedingly intelligent paranoiac concocted of incoherent elements of thought" [74]. He hoped that the EPR paper would demonstrate what he perceived to be the deficiencies of QM.

EPR showed that if you take two particles that are entangled with each other then and then measure one of them, this automatically triggers a correlated state of the second — even if the two are at a great distance from each other; this was the seemingly illogical result that EPR hoped to use to show that QM itself must have a flaw. Ironically, we now consider entanglement to be a cornerstone of QM. Entanglement occurs when we have a superposition of states that is not separable. We will put this into a more formal context later on in this text.

Copyrighted material

This "spooky action at a distance" seems at odds with our intuition and with previous physics. Podolsky, the youngest of the co-authors, reportedly leaked the paper to the New York Times to highlight this assault on the tower of QM to the public. The Times ran the story on the front page of the May 4th, 1935 edition with the headline “Einstein Attacks Quantum Theory."

Not only is entanglement accepted as part of standard quantum mechanics, we shall see later in this work that we can leverage entanglement to perform novel types of computation and communication. From an information theoretic point of view, entanglement is a different way of encoding information. If we have two particles that are entangled, the information about them is not encoded locally in each particle, but rather in the correlation of the two.

John Preskill likes to give the analogy of two kinds of books: non-entangled and entangled [176]. In the regular, non-entangled book we can read the information on each page as we normally do. In the entangled book, however, each page contains what appears to be gibberish. The information is encoded in the correlation of the pages, not in each page alone. This captures what Schrödinger expressed when he coined the term entanglement:

> Another way of expressing the peculiar situation is: the best possible knowledge of a whole does not necessarily include the best possible knowledge of all its parts. [196]

Schrödinger further noted that in his opinion entanglement was not just *one* of the phenomena described by quantum mechanics,“but rather *the* characteristic trait of quantum mechanics, the one that enforces its entire departure from classical lines of thought” [196].

1.4 Entanglement

Two systems are in a special case of quantum mechanical superposition called *entanglement* if the measurement of one system is correlated with the state of the other system in a way that is stronger than correlations in the classical world. In other words, the states of the two systems are not *separable*. We will explore the precise mathematical definitions of separability and entanglement later in this book.

Now that we have covered two core ideas of quantum mechanics – superposition and entanglement – let us turn to another fundamental concept that is not treated as often – the physicality of information. Rolf Landauer opened a new line of inquiry when he asked the following question:

Copyrighted material

> The search for faster and more compact computing circuits leads directly to the question: What are the ultimate physical limitations on the progress in this direction? ...we can show, or at the very least strongly suggest, that information processing is inevitably accompanied by a certain minimum amount of heat generation. [128]

In other words, is there a lower bound to the energy dissipated in the process of a basic unit of computation? Due to Landauer and others we now believe that there is such a limit; this is called Landauer's bound (LB). More specifically, the energy cost of erasure of n bits is $nkT \ln 2$ where k is the Boltzmann constant, T is the temperature in Kelvin of the heat sink surrounding the computing device and $\ln 2$ is, of course, the natural log of 2 (~ 0.69315). This limit is the minimum amount of energy dissipated for an irreversible computation.

Landauer acknowledged that this minimum is not necessarily the constraining factor on the energy draw of the system:

> It is, of course, apparent that both the thermal noise and the requirements for energy dissipation are on a scale which is entirely negligible in present-day computer components. The dissipation as calculated, however, is an absolute minimum. [128]

Landauer defined logical irreversibility as a condition in which "the output of a device does not uniquely define the inputs." He then claimed that "logical irreversibility...in turn implies physical irreversibility, and the latter is accompanied by dissipative effects." This follows from the second law of thermodynamics which states that the total entropy of a system cannot decrease and, more specifically, must increase with an irreversible process. For further background on reversibility, thermodynamics and computation see Feynman's *Lectures on Computation* [84].

In classical computing we make use of irreversible computations. For example, the Boolean inclusive *OR* (denoted $\vee$) gate has the following truth table, where 0 denotes "false" and 1 denotes "true":

X	Y	$X \vee Y$
0	0	0
0	1	1
1	0	1
1	1	1

Copyrighted material

Note that an output of value 1 cannot be traced uniquely to a set of inputs. We can arrive at that output through combinations of inputs; the state of the inputs is lost once we move to the output. This does not violate the conservation of information because the information was converted into dissipative heat.

The exclusive *OR* is also irreversible as is the *NAND* gate, which is universal for classical computing. *NAND* stands for "*NOT AND*" and is the inverse of the Boolean *AND* operator. Verify for yourself that *NAND* is irreversible by examining its truth table:

X	Y	X ↑ Y
0	0	1
0	1	1
1	0	1
1	1	0

In quantum computing, we limit ourselves to *reversible* logical operations [161, p. 29]. Later in this book we will consider which combinations of quantum operators are universal. For now, let's focus on the requirement in quantum computation that we limit our set of operators to reversible gates.

This requirement derives from the nature of irreversible operations: if we perform an irreversible operation, we have lost information and therefore a measurement. Our computation cycle then will be done and we can no longer continue with our program. Instead, by limiting all gates to reversible operators, we may continue to apply operators to our set of qubits as long as we can maintain coherence in the system. When we say reversible, we are assuming a theoretical noiseless quantum computer. In a noisy QC that decoheres, we cannot, of course, reverse the operation.

1.5 Reversibility of Quantum Computation

All operators used in quantum computation other than for measurement must be reversible.

In this chapter, we have examined four essential principles of quantum mechanical systems: superposition, the Born rule, entanglement and reversible computation. All four are essential to understanding the difference between classical and quantum computing as we shall see further in the book. We provide references on this book's website to a number of resources for those who wish to deepen their understanding of quantum mechanics.

Copyrighted material

Our generous universe comes equipped with the ability to compute.
—*Dave Bacon* [19]

CHAPTER 2

A Brief History of Quantum Computing

The possibility that we can leverage quantum mechanics to do computation in new and interesting ways has been hiding in plain sight since the field's early days; the principles of superposition and entanglement can form the basis of a very powerful form of computation. The trick is to build such a system that we can easily manipulate and measure.

While Richard Feynman is often credited with the conception of quantum computers, there were several researchers who anticipated this idea. In 1979, Paul Benioff, a young physicist at Argonne National Labs, submitted a paper entitled "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines" [23].[1] In this paper, Benioff demonstrated the theoretical basis for quantum computing and then suggested that such a computer could be built:

> That is, the whole computation process is described by a pure state evolving under the action of a given Hamiltonian. Thus all the component parts of the Turing machine are described by states which have a definite phase relation to one another as the calculation progresses...The existence of such models at least suggests that the possibility of actually constructing such coherent machines should be examined.

Yuri Manin also laid out the core idea of quantum computing in his 1980 book *Computable and Non-Computable* [140]. The book was written in Russian, however, and only translated years later.

[1]Note: Benioff completed and submitted the paper in 1979. It was published in the following year, 1980.

© Jack D. Hidary under exclusive license to Springer Nature Switzerland AG 2019

J. D. Hidary, *Quantum Computing: An Applied Approach*,
https://doi.org/10.1007/978-3-030-23922-0_2
Copyrighted material

In 1981, Feynman gave a lecture entitled "Simulating Physics with Computers" [83].[2] In this talk, he argued that a classical system could not adequately represent a quantum mechanical system:

> ...nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy...

He then set out the features that a quantum computer should have to be useful. At the time of this lecture, however, it was unclear to Feynman and the physics community how one could build such a machine.

Once Benioff, Manin and Feynman opened the doors, researchers began to investigate the nature of the algorithms that could be run on QCs. David Deutsch, a physicist at Oxford, suggested a more comprehensive framework for quantum computing in his 1985 paper [65]. In this work, he describes in detail what a quantum algorithm would look like and anticipates that "one day it will become technologically possible to build quantum computers."

Deutsch then went on to develop an example of an algorithm that would run faster on a quantum computer. He then further generalized this algorithm in collaboration with Richard Jozsa [67]. We will cover these and the other algorithms in more detail with code examples later on in this text.

In computer science and quantum computing, it is often important to evaluate how efficient an algorithm is — that is, how many steps would it take to run such an algorithm. We use big-O notation to represent the upper bound of the worst case of running an algorithm. The O in big-O notation comes from the "order" of the algorithm. We use big-Ω (Omega) notation to indicate the lower bound of the worst-case scenario. So, while Deutsch's problem takes at *worst* $O(n)$ steps to solve on a classical computer, Deutsch-Jozsa's algorithm solves the problem in one step on a quantum computer. Big-O notation will be helpful throughout this book in illuminating the difference between the classical and quantum algorithms.

Umesh Vazirani and his student Ethan Bernstein picked up where Deutsch and Jozsa left off. In 1993, Bernstein and Vazirani (BV) published a paper which described an algorithm that showed clear quantum-classical separation even when small errors are allowed [29]. Why is this significant? While Deutsch-Jozsa demonstrated a *deterministic* quantum advantage, if small errors are allowed in the computation, both classical and quantum versions

[2]Note: Feynman gave his lecture in 1981 and submitted the lecture for publication in May of 1981. The lecture was published by IJTP in 1982.

Copyrighted material

can be run at worst in $O(1)$ steps, showing no separation. By contrast, the Bernstein-Vazirani (BV) algorithm demonstrates separation even when small errors are allowed, thus showing non-deterministic quantum advantage. The problem posed in BV can be solved in $O(n)$ time on a classical computer and in $O(1)$ using the BV circuit on a quantum computer.

BV made a further contribution in their 1993 paper. They described a quantum version of the Fourier transform. This quantum Fourier transform (QFT) would serve as a critical component for Peter Shor when he developed his algorithm to factor large numbers.

The work of BV was quickly followed by Daniel Simon, then a postdoc at the University of Montreal, in 1994. Simon outlined a problem that a quantum computer would clearly solve exponentially faster than a classical one [203]. To be more specific, Simon's algorithm has an upper bound of $O(n)$ on a quantum computer, but a higher $\Omega(2^{n/2})$ on a classical computer. Since the lower bound on the classical computer is of higher order than the upper bound on the quantum computer, there is a clear demonstration of quantum advantage.

Just prior to Daniel Simon's work on algorithms, Seth Lloyd, working at Los Alamos, published a paper in *Science* which described a method of building a working quantum computer [136]. He proposed that a system sending pulses into a unit can represent a quantum state:

> Arrays of pulsed, weakly coupled quantum systems provide a potentially realizable basis for quantum computation. The basic unit in the array could be a quantum dot, a nuclear spin, a localized electronic state in a polymer, or any multistate quantum system that interacts locally with its neighbors and can be compelled to switch between states with resonant pulses of light.

Lloyd realized that:

> The proposed device is capable of purely quantum-mechanical information-processing capacities above and beyond the conventional digital capacities already presented. One of the most important of these capacities is that bits can be placed in superpositions of 0 and 1 by the simple application of pulses at the proper resonant frequencies but at a length different from that required to fully switch the bit. Such bits have a number of uses, including the generation of random numbers.

Copyrighted material

parts of the quantum computer is good; entanglement between the quantum computer and its environment is bad, since it corresponds to decoherence" [69, p. 4].

5. The system is capable of making "strong" measurements of each qubit. By strong measurement, DiVincenzo means that the measurement says "which orthogonal eigenstate of some particular Hermitian operator the quantum state belongs to, while at the same time projecting the wavefunction of the system irreversibly into the corresponding eigenfunction." This means that the measuring technique in the system actually does measure the state of the qubit for the property being measured and leaves the qubit in that state. DiVincenzo wants to prevent systems that have weak measurement, in other words, measuring techniques that do not couple with the qubit sufficiently to render it in that newly measured state. At the time he wrote the paper, many systems did not have sufficient coupling to guarantee projection into the new state.

In upcoming chapters, we will explore in further detail the various methods to physically construct a quantum computer and how to program them once built. Let's now turn our attention to qubits and the operators we use in quantum computation.

Copyrighted material

CHAPTER 3

Qubits, Operators and Measurement

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

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

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

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

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

© Jack D. Hidary under exclusive license to Springer Nature Switzerland AG 2019

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

Copyrighted material

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

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

Let us begin with the definition of a qubit:

3.1 What is a Qubit?

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

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

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

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

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

Copyrighted material

3.2 Quantum Operators

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

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

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

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

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

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

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

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

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

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

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

Copyrighted material

3.4 Representing Superposition of States

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

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

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

and

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

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

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

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

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

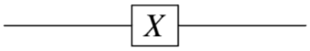
Here, we say that the amplitudes of the $|1\rangle$ state interfere *destructively* — the differing relative phases cause them to sum to zero. On the other hand, the amplitudes of the $|0\rangle$ state interfere *constructively* — they have the same sign (relative phase), so they do not sum to zero,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

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

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

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

Copyrighted material



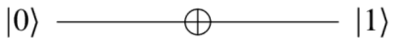
As we have seen, we can represent the X operator in ket notation as

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

and the application of the X operator like so:

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

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



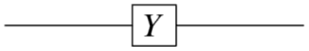
Next we have the Y operator, also denoted σ_y, which rotates the state vector about the y axis[3].

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

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

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

The circuit diagram for the Y operator is



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

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

If we apply Z to the computational basis state we have

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

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

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

Copyrighted material

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

For the case where $j = 1$ we have

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

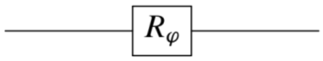
Note that we can multiply the bit-flip operator X by the phase-flip operator Z to yield the Y operator with a global phase shift of i. That is, $Y = iXZ$.

The circuit diagram for the Z operator is

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

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

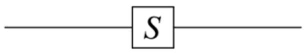
So the Pauli Z operator is just a special case of R_φ where $\varphi = \pi$. Let's recall that $e^{i\pi} = -1$ by Euler's identity (see chapter 13) so we can replace $e^{i\pi}$ with -1 in the Z matrix. The circuit diagram for the R operator is



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

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

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



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

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

Copyrighted material

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

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

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

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

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

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

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

And to state $|1\rangle$ we have

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

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

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

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

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

Copyrighted material

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

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

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

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

Please see chapter 14 for a list of additional identities.

Binary Operators

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

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

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

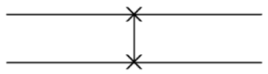
$$SWAP := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

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

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

Copyrighted material

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



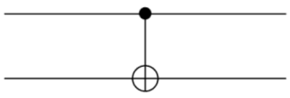
Now we come to a critical operator for quantum computing — controlled-*NOT* (*CNOT*). In this binary operator, we identify the first qubit as the *control qubit* and the second as the *target qubit*. If the control qubit is in state $|0\rangle$ then we do nothing to the target qubit. If, however, the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

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

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

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

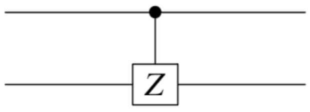
And for the circuit diagram, we depict the *CNOT* in this way



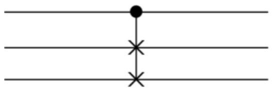
Here is an identity connecting the *SWAP* and *CNOT* operators:

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

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



Copyrighted material



3.2 *Comparison with Classical Gates*

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

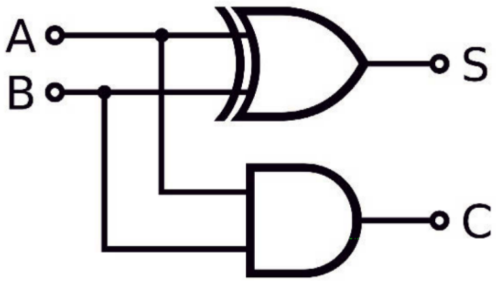


Figure 3.1: Half-adder in classical computing Source: Wikimedia

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

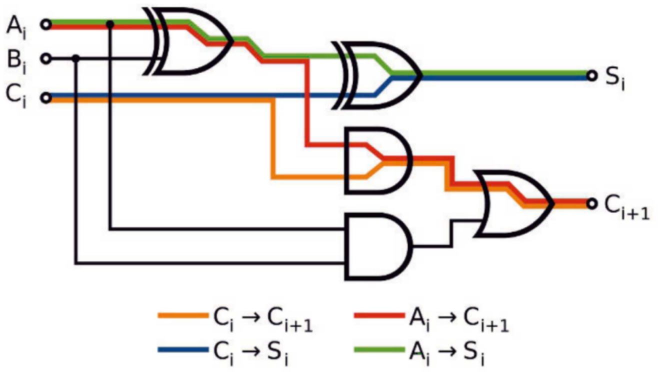


Figure 3.2: Full-adder in classical computing Source: Wikimedia

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

Copyrighted material

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

3.3 *Universality of Quantum Operators*

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

1. The Toffoli gate is universal for QC when paired with a basis-changing unary operator with real coefficients (such as H) [199].

2. Another set of gates which is universal is $\{CNOT, T, H\}$ [44, 161].

3.4 *Gottesman-Knill and Solovay-Kitaev*

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

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

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

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

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

Copyrighted material

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

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

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

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

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

3.5 The Bloch Sphere

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

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

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

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

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

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

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

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

Copyrighted material

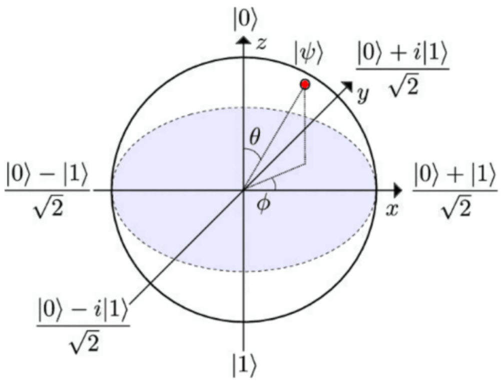

$|0\rangle$
z
$|\psi\rangle$
$\frac{|0\rangle + i|1\rangle}{\sqrt{2}}$
y
θ
$\frac{|0\rangle - |1\rangle}{\sqrt{2}}$
$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$
ϕ
x
$\frac{|0\rangle - i|1\rangle}{\sqrt{2}}$
$|1\rangle$

Figure 3.3: The Bloch sphere Source: [93]

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

3.6 *The Measurement Postulate*

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

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

3.5 Measurement Postulate

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

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

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

Copyrighted material