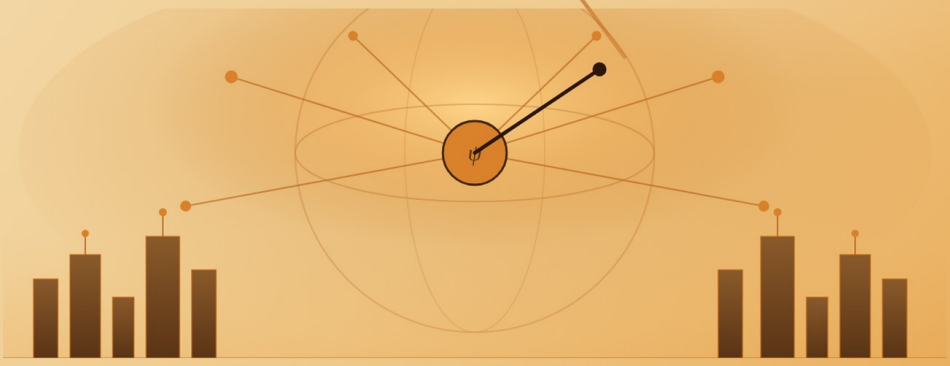
ANDHRA PRADESH STATE COUNCIL OF HIGHER EDUCATION (APSCHE)
SRI VENKATESWARA UNIVERSITY, TIRUPATI

SCIENCE & ENGINEERING TEXTBOOK SERIES

QUANTUM COMPUTING

Science & Engineering Edition

VOLUME I



Foundations, Algorithms & Real-World Applications

Prof. S. Varadarajan
Professor, Dept. of ECE, SVU College of Engineering, Tirupati • QAIC Faculty Mentor, Amaravati Quantum Valley

Prof. V. Vasudeva Rao
University Research Advisor, SVU Tirupati • Adjunct Professor, IIT Tirupati • HAG Professor (Retd.), IIT Kharagpur

Quantum Computing — Science & Engineering Edition, Vol. 1

Edition: First

Prof. S. Varadarajan | Prof. V. Vasudeva Rao

Publisher's Note

© 2026 Andhra Pradesh State Council of Higher Education and Sri Venkateswara University, Tirupati. All rights reserved.

ISBN: 978-93-345-9241-2
DOI: Sept'2026

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the Indian Copyright Law, and permission for use must always be obtained from SV University. Violations are liable to prosecution under the Indian Copyright Law.

The use of general descriptive names, registered names, trademarks, 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.

Prepared with AI assistance under the direction and technical review of the authors; the authors take full responsibility for the content.

Published by

Andhra Pradesh State Council of Higher Education and Sri Venkateswara University, Tirupati, Andhra Pradesh, India

About the Cover Design

The cover is meant to say, at a glance, what this book is about and who stands behind it.

Ivory, saffron and brown. Ivory stands for clarity, the plain page on which difficult ideas become understandable. Saffron has long been associated in Indian tradition with learning, aspiration and the spirit of enquiry. Brown grounds the design in the patience and rigour of the laboratory and the classroom.

The glowing qubit. At the centre is a single qubit, drawn as a Bloch sphere with its state vector, the starting point of this book. The lines that join it to neighbouring points suggest entanglement, the resource that gives quantum computing its power.

The circuit skyline. The towers on either side echo the circuitry of a quantum chip and a city rising from the ground, a reminder of Amaravati Quantum Valley, where this knowledge is being turned into industry.

The three emblems. The Government of Andhra Pradesh, APSCHE and Sri Venkateswara University appear together to reflect the partnership behind this series, with APSCHE and SVU as co-publishers.

The rising line. The bronze diagonal is a path from theory to application, the journey a reader makes from the first qubit to the case studies of the final unit.

FOREWORD

Quantum Computing — Science & Engineering Edition

— ✦ —

It is a privilege to introduce this volume, Quantum Computing — Science & Engineering Edition, authored by Prof. S. Varadarajan and Prof. V. Vasudeva Rao of Sri Venkateswara University, Tirupati. Quantum technology occupies a place of particular importance for the Andhra Pradesh State Council of Higher Education at present, and this book arrives at a moment when that importance is only growing.



Sri Vijaya Bhaskara Rao
Chairman (In-charge), APSCHE
Govt. of Andhra Pradesh

Andhra Pradesh has taken a deliberate and visible lead in this field. The Quantum Valley Amaravati initiative, built through the partnership of the Government of Andhra Pradesh, IBM, and Tata Consultancy Services, is fast becoming one of the country's principal centres for quantum research, industry engagement, and skilled talent. APSCHE, through its collaboration with E&ICT Academy MNIT Jaipur and IIT Madras on a certified quantum technology training programme, and in step with the aims of India's National Quantum Mission, has worked to ensure that our university and college students are not spectators to this development but active participants in it. A book of this kind, one that opens quantum computing to any diligent engineering or computer science student rather than reserving it for specialists in physics, is precisely the kind of resource that effort has needed.

What distinguishes this volume is its refusal to take shortcuts with either rigour or accessibility. It builds its subject from a single qubit to working familiarity with real algorithms, real hardware, and real industrial deployments — including a detailed, well-documented treatment of Quantum Valley Amaravati itself — without assuming a reader already fluent in the mathematics of quantum mechanics. Each unit pairs historical context with worked examples and hands-on laboratory exercises in Qiskit, so that a student finishes not with borrowed enthusiasm for the subject, but with the ability to reason about it, code it, and judge which claims made about it in the wider world are actually well-founded.

I note, too, the candour with which the authors have described their own process, including the disciplined and transparent use of an AI writing aid under their continuous technical direction and review. That transparency, and the care evident in the book's sourcing of every historical claim and image, reflect exactly the standard of academic integrity this Council expects of the material it recommends to institutions across the State.

I commend Prof. Varadarajan and Prof. Vasudeva Rao for undertaking this work, and I am confident it will serve students, faculty, and institutions across Andhra Pradesh — and well beyond it — as a dependable and genuinely teachable foundation in one of the defining technologies of this decade. I recommend it warmly to every reader who wishes to understand, rather than merely admire, quantum computing.

Vijaya Bhaskara Rao
Chairman (In-charge)
Andhra Pradesh State Council of Higher Education (APSCHE)

FOREWORD

Quantum Computing — Science & Engineering Edition

— ✦ —

Sri Venkateswara University was founded in 1954 with a clear and enduring purpose: to bring the frontiers of knowledge within reach of every student in this region, and to ensure that the sons and daughters of Andhra Pradesh are never left behind when the world changes. That purpose has guided us through successive technological revolutions — from the early computing era to the age of the internet — and it guides us still today, as a transformation deeper and more consequential than any of those earlier ones begins to take shape. Quantum technology is not a distant prospect. It is here, it is advancing rapidly, and it belongs in the hands of our students.



Prof. Tata Narasinga Rao
Hon'ble Vice-Chancellor
Sri Venkateswara University

Quantum Computing — Science and Engineering Edition, Volume 1, authored by Prof. S. Varadarajan, Professor of Electronics & Communication Engineering, and Prof. V. Vasudeva Rao, Adjunct Professor, Sri Venkateswara University, is our own community's answer to that call. It is a book written by teachers who know their students — who understand the questions they bring to the classroom, the background they arrive with, and the careers they are preparing for. The result is a text that does not demand prior mastery of quantum mechanics as a precondition for entry. Instead, it builds the subject from first principles, step by step, from a single qubit to real algorithms, real hardware platforms, and real industrial applications, pairing every concept with a hands-on laboratory exercise in Qiskit so that learning and doing are never separated.

This book is also deeply aligned with the national and State contexts in which our students will build their careers. India's National Quantum Mission, launched with a commitment of approximately ₹6,003 crore over five years, has set clear goals: to develop indigenous quantum computing platforms, build a workforce that can design and deploy quantum systems, and position India as one of the world's leading nations in this technology within the decade. Andhra Pradesh, through the Amaravati Quantum Valley (AQV) — the State's flagship quantum initiative, built in collaboration with IBM, Tata Consultancy Services, and other leading partners — has aligned itself fully with that national ambition. Prof. Varadarajan, Professor of Electronics & Communication Engineering at SVU and a QAIC Faculty Mentor with AQV, and Prof. Vasudeva Rao, Adjunct Professor at SVU, who brings four decades of research in superconductivity and cryogenics at IIT Kharagpur, represent exactly the combination of mission-connected practice and foundational scientific depth that this moment requires. A textbook written by faculty who are themselves inside the mission is not merely a study aid; it is a living link between the university and the national effort.

Sri Venkateswara University takes its responsibility in this regard seriously. We have made quantum technology a priority not simply because the policy context demands it, but because we believe in the potential of our students. The young men and women who study within these walls possess the curiosity, the

mathematical preparation, and the determination to engage with the hardest problems science and engineering present. What they have sometimes lacked is a resource that meets them at their level of preparation and walks with them all the way to genuine competence. This book is designed to do precisely that. It demands real effort — but it rewards that effort with real capability, not borrowed enthusiasm. A student who works through this volume will not merely have heard about quantum computing; they will be able to reason about it, code it, test it, and judge which claims about it are actually well-founded.

I am particularly pleased that the authors have been honest about limits as well as promise. Quantum computing is not a universal accelerant: most everyday computational problems gain no advantage from it whatsoever, and the machines available today are noisy and work in close partnership with classical systems. This book teaches students to ask the right diagnostic question — is this genuinely a quantum-shaped problem? — before reaching for quantum tools. That intellectual discipline, the discipline of asking first whether the technology fits the problem rather than fitting the problem to the technology, is one of the most valuable things a young engineer can develop.

The University's aspiration is straightforward: we want every engineering and science student at SVU — and, through the resources this book makes available, students at institutions across Andhra Pradesh — to have access to a sound, rigorous, and genuinely teachable introduction to quantum computing. We want them to move from learning to experimentation, from experimentation to use cases, and from use cases to meaningful contribution to the national mission. This book is an important step in that direction.

I congratulate Prof. Varadarajan and Prof. Vasudeva Rao on this significant and timely contribution. I commend their willingness to undertake a work of this scope, their intellectual integrity in presenting the subject honestly, and their commitment to keeping the Qiskit laboratories current and runnable — a detail that matters more to a working student than any amount of eloquent prose. I look forward to Volume II, which will take the mathematical foundations of quantum computing into deeper territory, and I look forward with equal confidence to the uses to which the students of this University and this State will put what they learn from these pages.

To every reader who opens this book: the technology it describes will shape the world you inhabit and the careers you build. Begin with it honestly, engage with it rigorously, and never mistake familiarity with understanding. The questions it raises are among the most interesting humanity has ever asked of nature. Sri Venkateswara University is proud to offer you this guide.

Prof. Tata Narasinga Rao
Hon'ble Vice-Chancellor
Sri Venkateswara University, Tirupati
Andhra Pradesh, India

FOREWORD

Quantum Computing — Science & Engineering Edition

— ✦ —

AQV
AMARAVATI QUANTUM VALLEY

Quantum computing is moving from the physics laboratory into engineering practice. Its promise lies not in replacing classical computers, but in opening new ways to address a specific class of problems – molecular and materials simulation, and selected optimisation and cryptographic tasks – that conventional machines handle poorly. For students and faculty, what is needed now is not awareness alone but a sound working understanding: of the principles, of the hardware, and, just as importantly, of the limits.



Venkata Swamy Tadikonda
Advisor, Amaravati Quantum Valley
Govt. of Andhra Pradesh

Quantum Computing – Science & Engineering Edition, Volume 1, meets that need well. It takes the reader from qubits, superposition, measurement and entanglement, through gates, circuits and the major hardware platforms, to the algorithms of Deutsch, Shor and Grover, and on to noise, error correction, quantum cryptography and teleportation. Every unit pairs the mathematics with worked examples and a hands-on Qiskit laboratory, so that students write and run circuits alongside the equations rather than treating the subject as an abstraction.

What I value most is the book's honesty. It states plainly that a quantum computer is not simply a faster classical computer, that most problems gain no quantum speedup at all, and that today's machines are noisy and work in partnership with classical systems. Its simple test – is this a quantum-shaped problem? – is the same question we ask when assessing use cases at Amaravati Quantum Valley. Engineers who learn to ask it early will spend less time on hype and find the real opportunities sooner.

This book also comes from within the ecosystem it describes. Under the vision of Hon'ble Chief Minister Sri N. Chandrababu Naidu, and in alignment with India's National Quantum Mission, Andhra Pradesh is building Amaravati Quantum Valley as a place where infrastructure, research, industry, startups and academia come together. Under the leadership of Sri Sridhar C V, Mission Director, Amaravati Quantum Valley, AQV's focus has moved from awareness to capability: building faculty and student capability across the State, and turning real problems from government and industry into practical quantum and AI use cases that students, startups and institutions can work on. Prof. Varadarajan is a QAIC Faculty Mentor with AQV, and Prof. Vasudeva Rao brings a long career in superconductivity and cryogenics research, including four decades at IIT Kharagpur – the physics on which today's leading quantum processors run. A textbook written by mentors for the students they teach is exactly the kind of contribution this mission depends on.

To every student who opens this book, I offer a simple progression: from learning to experimentation, from experimentation to use cases, and from use cases to meaningful innovation. Write circuits, test them, refine them and learn from where they fail. Compare hardware approaches. Work in teams that combine physics, computing and domain knowledge. Above all, begin with a real problem rather than with the technology for its own sake.

I congratulate Prof. Varadarajan and Prof. Vasudeva Rao on this timely work, and look forward to Volume II on the mathematical foundations of quantum computing. My best wishes to the authors and to every reader who begins their quantum journey with this book.

Venkata Swamy Tadikonda
Advisor, Amaravati Quantum Valley
Government of Andhra Pradesh

Preface

Why This Book Exists

This book began with a simple conviction: quantum computing should not belong only to physicists. It is fast becoming a subject every engineer, computer scientist, and technologist needs at least a working command of – not because everyone will build a quantum computer, but because quantum computing is quietly becoming one of the instruments through which the next stage of human problem-solving will be conducted. Drug discovery, materials design, secure communication, logistics at planetary scale – each of these is a place where classical computation runs into a genuine wall, and quantum computation offers, for the first time, a principled way past it.

This volume was written for a reader who may have no formal background in physics, no prior exposure to linear algebra beyond an undergraduate course, and no assumption that quantum mechanics is intuitive – because it isn't, for anyone, at first. What such a reader does need, and what this book tries to supply, is a patient, honest, worked-through path from a single qubit to a genuine understanding of where the field stands today, including its real industrial deployments and its real limitations.

The Challenge This Subject Poses

Quantum computing is unusually difficult to teach well, for a specific reason: it sits at the intersection of three things a student rarely has all at once – comfort with linear algebra, an intuitive feel for probability, and a willingness to accept that nature genuinely does not behave the way everyday experience suggests it should. Most treatments solve this by picking one audience and writing past everyone else: a physics text that assumes prior quantum mechanics, or a computer-science text that treats the physics as a black box. This book tries a different balance – introducing every piece of mathematics at the point it is needed, explaining every physical idea in plain language before formalizing it, and using worked examples and runnable code throughout so that no idea is left as an abstraction. Whether that balance succeeds is, honestly, for the reader and the instructor using this book to judge – not for its authors to declare.

What This Book Offers, Unit by Unit

Volume 1 moves through six units. Units 1 and 2 build the qubit itself – superposition, measurement, entanglement, and the gate toolkit – from first principles. Unit 3 grounds that mathematics in real hardware: superconducting circuits, trapped ions, photonics, spin, and neutral atoms, and the quantum circuit model that describes all of them uniformly. Unit 4 assembles this toolkit into complete algorithms – Deutsch's, Deutsch–Jozsa, Shor's, and Grover's – and is honest throughout about what kind of advantage each one actually delivers. Unit 5 removes the assumption of perfect hardware, covering noise, error correction, the information theory that underlies both classical and quantum computing, quantum cryptography, and teleportation. Unit 6 turns from theory to practice, presenting real, sourced, current case studies – including a detailed look at India's own Amaravati Quantum Valley – so that the reader finishes not with textbook abstractions alone, but with a working sense of where this technology already touches industry today.

Every unit follows the same structure deliberately: a historical case study to open, worked examples throughout, a hands-on Qiskit laboratory, and – new to this hybrid model – a short applied use case scoped to that unit's own toolkit, with the deepest, most integrative use cases reserved for Unit 6.

What You Should Be Able to Do After Reading This Book

A reader who works through this volume seriously – meaning attempting the exercises, not merely reading past them – should be able to: read a quantum circuit diagram and predict its output by hand for small numbers of qubits; explain, without hand-waving, what problem each major quantum algorithm in this book solves and why it is faster than the best known classical method; write and run a working Qiskit program implementing any protocol covered here; distinguish a well-founded claim about quantum computing's current capabilities from an overstated one; and hold an informed conversation with a specialist about where the field genuinely stands, rather than where headlines suggest it stands.

Measuring What You Have Learned

Each unit closes with a set of exercises graded in difficulty, and Unit 6 closes the volume with questions that require drawing on every earlier unit at once – a natural, cumulative self-test. Beyond this book's own exercises, three external benchmarks are worth using: the ability to reproduce this book's worked examples from memory and extend them to a new case the book did not cover; the ability to read a recent, non-technical article about a quantum computing announcement and correctly judge which of its claims are well-supported; and, for readers wanting formal certification, structured programmes such as IBM's Qiskit certification or the quantum technology training courses offered through India's National Quantum Mission – the same certification track, in fact, that one of this book's own authors completed en route to becoming a district quantum mentor.

Quantum Computing and Artificial Intelligence: Two Technologies, One Future

A question this book is asked often enough to answer directly here: does quantum computing compete with artificial intelligence, or complement it? The honest answer is complement, in at least two concrete ways. First, quantum computers may eventually accelerate specific machine learning subroutines – Unit 6, Section 6.6 treats this honestly as a promising but still unproven research direction, not a settled result. Second, and more immediately relevant to how this book itself was made: large language models are already a practical aid to writing, structuring, and explaining technical material – including, transparently, this book, discussed in the closing section below. Quantum computing and artificial intelligence are not rival technologies racing for the same seat; they are more likely to become partners, with classical AI systems designing and interpreting the outputs of quantum circuits long before quantum computers are powerful enough to run large AI models themselves.

From the Laboratory to the Market: How Reliable Is Quantum Computing Today?

Unit 6 answers this question in detail with named companies and dated events, but the short version bears stating plainly here: quantum computing today is real, funded, and running genuine pilots – IBM's partnership with Moderna, Google's with Boehringer Ingelheim, Volkswagen's quantum-optimized bus routing in Lisbon, and India's own Amaravati Quantum Valley all exist and are documented, not hypothetical. What quantum computing is not, today, is broadly, unambiguously superior to classical computation for practical business problems at scale. Every credible industry voice cited in this book describes the technology as hybrid, exploratory, and narrowing toward a smaller set of proven use cases over the next several years – not as a mature, drop-in replacement for classical infrastructure. Readers should trust specific, sourced claims over general enthusiasm, in this book and everywhere else.

Is Quantum Computing the Only Path Forward?

No, and this book would be doing its reader a disservice to imply otherwise. Quantum computing is one of several active approaches to problems classical silicon struggles with, and a well-informed technologist should know where it sits among its neighbours:

Approach	Core Idea	Where It Competes With Quantum Computing
Classical high-performance computing (HPC)	More transistors, more parallelism, better algorithms	Still the default choice for the overwhelming majority of large computational problems
Quantum-inspired classical algorithms	Classical algorithms borrowing structure from quantum methods	Often match quantum performance today on optimization problems, without needing quantum hardware at all
Neuromorphic computing	Hardware modelled on biological neural circuits	Targets energy-efficient AI inference, a different goal than quantum computing's targeted problems
Photonic classical computing	Light instead of electrons for classical computation	Aims at speed and energy efficiency in classical, not quantum, computation
DNA and molecular computing	Massively parallel chemical reactions as computation	Extremely early-stage; suited to specific combinatorial and data-storage problems

Quantum computing's distinguishing claim is not that it is faster in general – Unit 1, Section 1.7's “myth vs. reality” box makes this point early and this book returns to it repeatedly – but that for a specific, growing list of problems with the right structure, no classical approach, however well engineered, can close the gap. Knowing which problems belong on that list, and which do not, is arguably the single most valuable judgment this book tries to build.

A Note on How This Book Was Written

In the interest of the transparency this preface has argued for throughout, its authors want to be direct about their process. This volume was drafted with the assistance of an AI language model, working under the continuous direction, structural design, and technical review of its two named authors. Every unit's pedagogical structure – the opening case study, the pioneer profiles, the worked examples, the applied use case, the Qiskit laboratory – was specified and refined by the authors across many rounds of review; every historical photograph was sourced and verified by the authors, and any image whose authenticity could not be confirmed was deliberately excluded rather than risk misattribution; every real-world claim in Unit 6 was checked against named, dated, primary sources; and every error caught during review – including several cross-reference mistakes and one incorrect scientific anecdote flagged as unverified rather than stated as fact – was corrected by the authors before publication.

The authors' contribution, concretely, is this: the decision that this book should exist and for whom; the pedagogical philosophy that shapes every unit; the technical review and correction of its content, particularly in the areas of quantum hardware, superconductivity, and the authors' own certified quantum training and curriculum design experience; and the accountability for its accuracy and suitability for classroom use, which rests with the authors and not with any tool used in its drafting. Readers, instructors, and reviewers are encouraged to treat this volume exactly as they would any other technical textbook – verifying claims that matter to their own use, and raising errors directly with the authors so that future editions improve.

A Word on Ethics

Three practices guided this book's development, stated here explicitly rather than left implicit. First, no fact about a named person – living or historical – was fabricated; where a claim could not be verified, this book says so in the text itself rather than presenting it with false confidence. Second, no image was used unless its authenticity as a genuine photograph of the person it claims to depict could be reasonably established; an AI-generated illustration and an unsourced social-media graphic were both identified and rejected during this book's development for exactly this reason. Third, this book does not present AI-assisted drafting as if it were unassisted original prose – the note above exists because concealment, not assistance, is what would have compromised this book's integrity and its authors' standing.

What Lies Ahead

Quantum computing's near-term future, as Unit 6 documents in detail, is one of expanding hybrid pilots rather than sudden, universal disruption: more organisations testing narrower, better-chosen problems, more processors crossing from physical to logical, error-corrected qubits, and a slow but real migration of the world's cryptographic infrastructure to quantum-safe standards regardless of when large-scale quantum computers actually arrive. For a student beginning today, this is, if anything, good timing – the field needs trained people more than it needs any single breakthrough, and the reader who finishes this volume with genuine understanding, rather than borrowed enthusiasm, is exactly the kind of contributor this moment calls for.

Prof. S. Varadarajan
Prof. V. Vasudeva Rao
Sri Venkateswara University, Tirupati

Contents

Preface 7
Acronyms and Abbreviations 14
Acknowledgements 16

Unit 1: From Classical Limits to the Quantum Paradigm 17
1.1 How to Read This Unit 18
1.2 Pioneers Behind This Unit 18
1.3 Case Study: The Molecule No Computer Can See 19
1.4 From Classical to Quantum: A Change in Paradigm 20
1.5 Why Classical Computers Hit a Wall 21
1.6 The Quantum Mechanics You Actually Need 22
1.7 Bits vs. Qubits: The Comparison That Matters 23
1.8 Case Study Revisited 24
1.9 Realization Lab: Your First Qubit, For Real 24
1.10 Before You Move On 26
1.11 References and Further Study 26
Bridge to Unit 2 27

Unit 2: States, Measurement, Entanglement and Gates 28
2.1 Pioneers Behind This Unit 29
2.2 Case Study: The Correlation Einstein Couldn't Accept 30
2.3 Quantum States and Qubits, Revisited as Vectors 31
2.4 Superposition and Measurement, Made Precise 32
2.5 Quantum Entanglement and Its Importance 33
2.6 The Gate Toolkit: Pauli, Hadamard, Phase, CNOT, Toffoli 34
2.7 The Bloch Sphere 35
2.8 Case Study Revisited: Building the Correlation 36
2.9 Realization Lab: Building and Verifying a Bell Pair 36
2.10 Before You Move On 37
2.11 References and Further Study 38
Bridge to Unit 3 39

Unit 3: Qubits in Hardware, Quantum Circuits, and Bell States 40
3.1 Pioneers Behind This Unit 41
3.2 Case Study: Turning a Circuit Diagram Into a Real Machine 42
3.3 The Qubit as Physical Reality 43
3.4 The Bloch Sphere, Revisited for Circuit Design 45
3.5 Building Quantum Circuits 46
3.6 The Four Bell States 48
3.7 Case Study Revisited: From Diagram to Machine 49
3.8 Realization Lab: Preparing and Verifying All Four Bell States 49
3.9 Applied Use Case: Quantum Random Number Generation 51

3.10 Before You Move On....52
3.11 References and Further Study....52
Bridge to Unit 4....53

Unit 4: Quantum Algorithms....54

4.1 Pioneers Behind This Unit....55
4.2 Case Study: The Problem That Could Break the Internet....56
4.3 What Makes a Quantum Algorithm Different....57
4.4 Classical Logic, Made Reversible....58
4.5 Quantum vs Classical Complexity Classes....59
4.6 Deutsch's Algorithm....60
4.7 The Deutsch–Jozsa Algorithm....61
4.8 Shor's Factorization Algorithm....62
4.9 Grover's Search Algorithm....63
4.10 Comparing the Four Algorithms....65
4.11 Case Study Revisited: Breaking, and Rebuilding, Trust....65
4.12 Realization Lab: The Deutsch–Jozsa Algorithm in Qiskit....66
4.13 Applied Use Case: Grover Search for a Supply-Chain Record Lookup....67
4.14 Before You Move On....68
4.15 References and Further Study....69
Bridge to Unit 5....69

Unit 5: Noise, Error Correction, Information Theory and Cryptography....70

5.1 Pioneers Behind This Unit....71
5.2 Case Study: You Cannot Copy What You Do Not Know....72
5.3 Noise and the Limits of Fragile Information....73
5.4 Graph States and Their Use in Codes....74
5.5 Quantum Error-Correcting Codes....75
5.6 Fault-Tolerant Computation....76
5.7 Classical vs Quantum Information Theory....77
5.8 Quantum Cryptography: The BB84 Protocol....79
5.9 Quantum Teleportation....80
5.10 Case Study Revisited: Turning a Limitation Into a Tool....81
5.11 Realization Lab: The Three-Qubit Bit-Flip Code in Qiskit....82
5.12 Applied Use Case: Quantum Key Distribution for Secure Communication....83
5.13 Before You Move On....84
5.14 References and Further Study....85
Bridge to Unit 6....86

Unit 6: Real-World Quantum Computing....87

6.1 Introduction: From Theory to Deployment....88
6.2 Case Study: Quantum Valley Amaravati – Building India's Quantum Ecosystem....88
6.3 Case Study: Quantum Chemistry and Drug Discovery....89
6.4 Case Study: Post-Quantum Cryptography Migration....90
6.5 Case Study: Quantum Optimization for Industry....91

6.6 Case Study: Quantum Machine Learning, Briefly....92
6.7 The Road Ahead: From NISQ to Fault Tolerance....92
6.8 Realization Lab: A Toy VQE for the Hydrogen Molecule....93
6.9 Before You Move On....94
6.10 References and Further Study....95
Closing Note: The End of Volume 1....96

Subject Index....97

References....97

Acronyms and Abbreviations

Technical and organizational acronyms used across this volume, listed alphabetically.

ADMET – Absorption, Distribution, Metabolism, Excretion, Toxicity (drug-discovery evaluation criteria)

AICTE – All India Council for Technical Education

APAS – Andhra Pradesh Akademi of Sciences

APSCHE – Andhra Pradesh State Council of Higher Education

AQRF – Amaravati Quantum Reference Facility

BB84 – Bennett–Brassard 1984 (the first quantum key distribution protocol)

BPP – Bounded-error Probabilistic Polynomial time (classical complexity class)

BQP – Bounded-error Quantum Polynomial time (quantum complexity class)

BTech / MTech – Bachelor of Technology / Master of Technology

CAS – Career Advancement Scheme

C-DAC / C-DOT – Centre for Development of Advanced Computing / Centre for Development of Telematics

CNOT – Controlled-NOT gate

CoE – Centre of Excellence

CSIR – Council of Scientific and Industrial Research

CZ – Controlled-Z gate

DRDO – Defence Research and Development Organisation

DST – Department of Science and Technology (Government of India)

E&ICT – Electronics and Information and Communication Technology (Academy)

ECE – Electronics and Communication Engineering

EPR – Einstein–Podolsky–Rosen (1935 thought experiment)

FIPS – Federal Information Processing Standard

GST – Goods and Services Tax

HAG – Higher Administrative Grade

HHT – Hilbert–Huang Transform

IEEE – Institute of Electrical and Electronics Engineers

IETE – Institution of Electronics and Telecommunication Engineers

IPR – Institute for Plasma Research

ISRO – Indian Space Research Organisation

ISTE – Indian Society for Technical Education

IoT – Internet of Things

MeitY – Ministry of Electronics and Information Technology (Government of India)

ML-DSA / ML-KEM – Module-Lattice Digital Signature Algorithm / Module-Lattice Key Encapsulation Mechanism

MoU – Memorandum of Understanding

NAAC – National Assessment and Accreditation Council

NBA – National Board of Accreditation

NISQ – Noisy Intermediate-Scale Quantum (hardware era)

NIST – National Institute of Standards and Technology (United States)

NMR – Nuclear Magnetic Resonance

NPTEL – National Programme on Technology Enhanced Learning

NQM – National Quantum Mission (India)

PO – Programme Outcome

PQC – Post-Quantum Cryptography

PSPACE – Polynomial Space (classical complexity class)

QAOA – Quantum Approximate Optimization Algorithm

QFT – Quantum Fourier Transform

QKD – Quantum Key Distribution

QML – Quantum Machine Learning

SLH-DSA – Stateless Hash-Based Digital Signature Algorithm

SVU / SVUCE – Sri Venkateswara University / Sri Venkateswara University College of Engineering

SWAYAM – Study Webs of Active Learning for Young Aspiring Minds (Government of India platform)

TCS – Tata Consultancy Services

TEQIP – Technical Education Quality Improvement Programme

VQE – Variational Quantum Eigensolver

Acknowledgements

The authors gratefully acknowledge Sri Venkateswara University, Tirupati, and Sri Venkateswara University College of Engineering, for the institutional support that made this volume possible, and IIT Kharagpur and IIT Tirupati for four decades of research grounding that shaped its treatment of quantum hardware and superconductivity.

We thank E&ICT Academy MNIT Jaipur, IIT Madras, and APSCHE for the certified quantum technology training programme that directly informed this book's structure and syllabus, and the Quantum Valley Amaravathi initiative – the Government of Andhra Pradesh, IBM, and Tata Consultancy Services – for building the real-world ecosystem this book's closing unit draws upon.

We acknowledge India's National Quantum Mission for its role in coordinating and funding the broader skilling effort this book grew out of, and the students and faculty of the Chittoor district cluster whose questions, in the classroom and beyond it, continue to sharpen how this material is explained.

Finally, we thank our families for their patience through the preparation of this volume, and each reader who takes the time to point out an error or suggest an improvement – every such note makes the next edition better.

Prof. S. Varadarajan
Prof. V. Vasudeva Rao

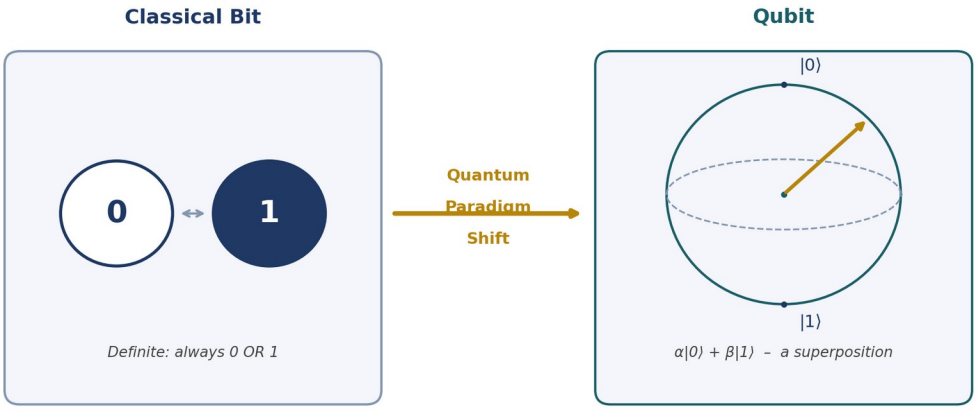
Quantum Computing

Science & Engineering Edition

Unit 1: From Classical Limits to the Quantum Paradigm

"What I cannot create, I do not understand."

— Richard Feynman



Every classical bit is a fixed choice. Every qubit is a standing question – and this unit is about learning to ask it.

1.1 How to Read This Unit

This unit is written so that a reader can move through it in a single, connected line, rather than hunting for what matters. A Realization Lab near the end turns the theory into runnable code. Historical portraits, worked examples, and a running case study are woven throughout – the case study opens the unit as a question and closes it as an answer.

1.2 Pioneers Behind This Unit

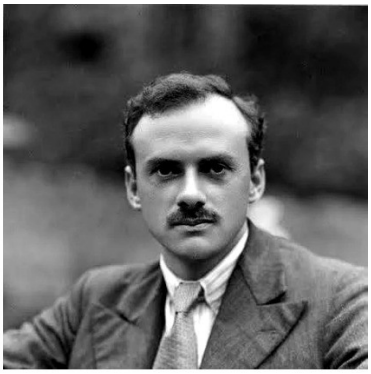
Every idea in this unit stands on the shoulders of a handful of people who changed how we think about computation and about nature itself.



Richard Feynman
1918-1988



Alan Turing
1912-1954



Paul Dirac
1902-1984

Richard Feynman – The Question That Started It All (1981)

In 1981, physicist Richard Feynman posed a deceptively simple question at a conference at MIT: classical computers struggle to simulate quantum systems because the required memory grows exponentially with system size – so why not build a computer that is itself quantum mechanical, and let nature simulate nature? This single observation is usually credited as the conceptual birth of quantum computing [8]. Feynman never built a quantum computer himself – the engineering would take another four decades to catch up with the idea.

What makes Feynman worth remembering here is not just the question but where it came from: he was a physicist trying to run ordinary simulations of nature on ordinary computers, and kept hitting the same exponential wall this unit opens with. Rather than accept the wall as a limitation of programming, he asked whether the computer itself was built on the wrong model of physics. That willingness to question the machine rather than the method is exactly the habit of mind this book hopes to pass on.

Alan Turing – The Classical Foundation (1936)

Decades before Feynman's question, Alan Turing gave computation its first rigorous mathematical description: the Turing machine [4]. Every classical computer you have ever used –

and every limitation of classical computers discussed in this unit – traces back to Turing's 1936 model. Understanding the classical paradigm precisely is what lets us see, precisely, where the quantum paradigm departs from it.

Turing was 24 when he wrote this down, working out what it even means for a procedure to be "computable" years before any electronic computer existed to test the idea. That is the pattern worth noticing: the deepest limits on a technology are often discovered mathematically, long before the hardware that will bump up against them is ever built. Quantum computing has followed exactly the same arc – Feynman's question preceded working quantum hardware by forty years, just as Turing's machine preceded the first real computer.

Paul Dirac – The Notation We Still Use (1939)

Every $|0\rangle$ and $|1\rangle$ you will write in this book uses "bra-ket" notation, introduced by Paul Dirac in 1939 to describe quantum states compactly [5]. It is a small piece of notation with an outsized legacy – it is the language every quantum computing paper, textbook, and piece of Qiskit code still speaks today.

Dirac introduced this notation not to invent something new about physics, but to make existing quantum mechanics easier to write and think in – a reminder that good notation is itself a kind of discovery. A student who masters bra-ket notation early is not just learning symbols; they are inheriting a way of thinking about quantum states that took the field's founders years to settle on. Every worked example in the rest of this book leans on the fact that you can now read it fluently.

Where we are going: this unit builds the bridge from Turing's classical world to Feynman's quantum question, using Dirac's notation as the language for everything that follows in Units 2 onward.

1.3 Case Study: The Molecule No Computer Can See

A Real Bottleneck in Drug Discovery

Caffeine has 24 atoms. Simulating its exact quantum mechanical behaviour – the behaviour that determines how it binds to a receptor in your brain – is already a serious computational challenge. Now consider a candidate cancer drug with a few hundred atoms, designed to bind precisely to a tumour protein without damaging surrounding healthy tissue. Pharmaceutical researchers routinely hit a wall: even the world's fastest supercomputers cannot simulate such a molecule exactly, and are forced to use approximations that can miss the very interaction the drug depends on.

Keep these two questions in mind as you read this unit – we will answer them directly at the end:

Q1. Why, precisely, does simulating a molecule get exponentially harder as it grows by just a few atoms, rather than growing steadily?

Q2. What would a computer need to look like, in principle, to represent nature's own quantum behaviour directly instead of approximating it?

1.4 From Classical to Quantum: A Change in Paradigm

1.4.1 What Do We Mean by a "Computational Paradigm"?

A computational paradigm is simply a general way of thinking about how computation is performed – what information looks like, and what operations are allowed on it. Changing paradigm does not mean "getting a faster computer"; it means changing the very representation of information itself.

1.4.2 The Classical Paradigm

In the classical paradigm, information is encoded as discrete symbols – in digital computers, as bits, each holding a definite value of 0 or 1 at any instant. These bits are manipulated by Boolean logic gates (AND, OR, NOT, XOR) built from transistors. This model traces back to Alan Turing's 1936 mathematical description of computation (the Turing machine) [4] and to the von Neumann architecture – memory, a processing unit, and input/output, all built around definite, readable bit values.

Why the Classical Model Succeeded

Two properties explain its success. **Noise tolerance:** a voltage close to the value representing 1 is still read as 1, so small imperfections don't corrupt the answer. **Scalability:** as transistors shrank, more of them fit on a chip, so computers became faster, smaller, and cheaper for decades – a trend known as Moore's Law.

1.4.3 The Quantum Paradigm

In the quantum paradigm, information is represented not as a definite 0 or 1, but as a quantum state that can exist in superposition – a combination of both possibilities at once, in a precise mathematical sense we define in Section 1.6 – and can be transformed by quantum gates. This is the shift Feynman's question pointed toward: using nature's own quantum behaviour as the computational resource, rather than fighting against it.

Aspect	Classical Paradigm	Quantum Paradigm
Basic unit of information	Bit: 0 or 1	Qubit: $\alpha \|0\rangle + \beta \|1\rangle$
State before reading	Definite, directly readable	Described by amplitudes; measurement gives a probabilistic outcome
Operations	Boolean logic gates	Unitary quantum gates
Copying	Bits can be copied freely	Unknown quantum states cannot be copied exactly (no-cloning)
Main strength	Reliable, general-purpose computation	Potential speedups for select structured problems

Crucial Clarification

The shift from classical to quantum computation is a change of model, not merely an increase in speed. A quantum computer is not "a faster classical computer" – it is a different kind of machine altogether, useful for a different class of problems.

1.5 Why Classical Computers Hit a Wall

1.5.1 Physical and Hardware Limits

Shrinking transistors indefinitely runs into real physics: leakage current, heat, and manufacturing precision all become serious problems at very small scales. Classical computers are not becoming obsolete – but simply making them smaller and faster forever is not realistic.

1.5.2 The Exponential Wall: Simulating Quantum Systems

This is the heart of Feynman's 1981 argument, and the answer to Case-Study Question 1.

Why 2^n, Exactly?

A single qubit's state needs 2 complex numbers (amplitudes) to describe fully. Two qubits together need 4. Three need 8. In general, a system of n quantum two-level particles needs 2^n complex amplitudes for a complete description – not because we chose a wasteful representation, but because that is how many independent possibilities a genuinely quantum system can be in superposition of, simultaneously.

A 300-atom drug molecule, modelled even crudely at the electron level, can easily involve hundreds of interacting quantum two-level degrees of freedom. 2^{300} is a number vastly larger than the number of atoms in the observable universe – no classical computer, however large, can store that state exactly. This is why the difficulty is exponential, not merely large: each additional quantum particle doubles the classical resources needed, rather than adding to them steadily.

Worked Example – Counting the Wall

Problem: How many complex amplitudes are needed to exactly describe the quantum state of 10 qubits? Of 20 qubits?

Solution: By the rule above, n qubits require 2^n amplitudes. For n = 10: 2^{10} = 1,024 amplitudes – trivial for a laptop. For n = 20: 2^{20} = 1,048,576 amplitudes – still fine. For n = 50: $2^{50} \approx 1.1 \times 10^{15}$ amplitudes – already beyond the memory of most classical supercomputers if each amplitude needs even a few bytes of storage.

Takeaway: the wall is not about today's computers being underpowered – it is structural. Every

additional qubit doubles the problem.

1.5.3 What This Does Not Mean

Classical limitations do not mean classical computers are weak, nor that quantum computers solve every hard problem efficiently. Quantum advantage appears only when a problem has structure that superposition, interference, and entanglement can actually exploit. Even quantum computers rely on classical computers to compile circuits, control hardware, and interpret results – a point we return to in Section 1.7.

1.6 The Quantum Mechanics You Actually Need

You do not need a full physics degree to work with quantum computing – but you do need to precisely understand a handful of principles. We define every symbol the moment it appears. The treatment in this section follows the standard presentation used in the field's two most widely adopted references [1, 2].

1.6.1 The Quantum State and Dirac Notation

The state of a qubit is written using the ket notation Dirac introduced: $|0\rangle$ and $|1\rangle$ are the two basic "reference" states (directly analogous to classical 0 and 1). A general qubit state is written:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle, \qquad \alpha, \beta \in \mathbb{C}$$

Every Symbol, Explained

- **$|\psi\rangle$ ("ket psi"):** the name we give to this particular qubit's quantum state – ψ is just a label, like naming a variable.
- **α, β:** probability amplitudes – complex numbers (numbers with a real and an imaginary part) that determine how likely each outcome is. They are not probabilities themselves.
- **$\mathbb{C}$:** the symbol for "the set of all complex numbers" – stating $\alpha, \beta \in \mathbb{C}$ just means "α and β are each some complex number."

1.6.2 From Amplitudes to Probabilities

The actual probability of measuring 0 or 1 comes from the squared magnitude of each amplitude:

$$P(0) = |\alpha|^2, \quad P(1) = |\beta|^2, \quad \text{with } |\alpha|^2 + |\beta|^2 = 1$$

The last equation is the normalization condition: since the qubit must be measured as something, all probabilities must add up to exactly 1.

Worked Example – Checking a Valid Quantum State

Problem: Is $|\psi\rangle = \sqrt{(1/2)}\,|0\rangle + \sqrt{(1/2)}\,|1\rangle$ a valid quantum state? What are P(0) and P(1)?

Solution: Here $\alpha = \beta = 1/\sqrt{2}$. $P(0) = |\alpha|^2 = (1/\sqrt{2})^2 = 1/2$. $P(1) = |\beta|^2 = 1/2$. Check normalization: P(0)

+ P(1) = 1/2 + 1/2 = 1 ✓ Valid.

Interpretation: this qubit, if measured, gives 0 or 1 with exactly equal (50%/50%) probability – an equal superposition, the quantum equivalent of a fair coin, though as we'll see next, it is far more than that.

1.6.3 Superposition, Precisely

Superposition is the principle that a quantum system can exist in a linear combination of basis states, as in $|\psi\rangle$ above. A common source of confusion: superposition does not mean the qubit secretly holds both 0 and 1 the way a classical memory might hold two separate values. It means the qubit's amplitudes are not yet committed to one outcome – and crucially, those amplitudes can be transformed by quantum gates before that commitment happens at measurement.

The Hadamard gate (H) is the standard gate used to create an equal superposition from $|0\rangle$:

$$H\,|0\rangle = (|0\rangle + |1\rangle) / \sqrt{2}$$

This is exactly the state checked in the worked example above.

1.6.4 The Bloch Sphere

The Bloch sphere is a geometric picture of a single qubit's state: the north pole represents $|0\rangle$, the south pole represents $|1\rangle$, and every other point on the sphere's surface represents some superposition, described by two angles θ and φ. Its importance: two quantum states can give identical measurement probabilities yet sit at different points on the sphere (different phase) – and behave differently once further gates are applied. Probability alone does not fully describe a qubit; phase matters too.

1.6.5 Measurement

Measurement converts a quantum state into a classical outcome (0 or 1), with probability given by the squared amplitudes above. Measurement is fundamentally different from reading a classical bit in one critical way: reading a classical bit does not disturb it, but measuring a qubit collapses its superposition – after measurement, the qubit is definitely in the measured state, and the other amplitude's information is gone. This is why quantum algorithms carefully delay measurement until all the useful interference has already happened (Unit 4 builds directly on this idea).

1.7 Bits vs. Qubits: The Comparison That Matters

Feature	Classical Bit	Qubit
Basic values	0 or 1	$\|0\rangle$, $\|1\rangle$, or superposition $\alpha\,\|0\rangle + \beta\,\|1\rangle$
Before reading	Definite value	Described by amplitudes
Reading	Non-destructive	Collapses the state (measurement)
Copying	Freely copyable	Cannot be copied exactly (no-cloning principle)
n units together	Store one of 2^n strings at a time	Can hold amplitudes over all 2^n basis

Feature	Classical Bit	Qubit
		states simultaneously

Myth vs. Reality

Myth: *"A quantum computer with n qubits tries all 2^n possibilities at once, so it is automatically exponentially faster at everything."*

Reality: Superposition gives you amplitudes over 2^n possibilities, but a single measurement reveals only one outcome, chosen probabilistically – you cannot simply "read off" all 2^n answers. Quantum speedups come from carefully designed algorithms that use interference to make wrong answers cancel out and right answers reinforce, before measurement happens. Most known problems get no quantum speedup at all. This is precisely why quantum algorithm design (Unit 4) is its own deep field, not an automatic consequence of owning quantum hardware.

1.8 Case Study Revisited: Answering Our Two Questions

Closing the Loop

Q1. Why does simulating a molecule get exponentially harder? Because an exact quantum description of n interacting quantum particles needs 2^n complex amplitudes (Section 1.5.2) – not a matter of insufficient engineering effort, but a structural property of how quantum systems are described. A 300-particle molecule's classical description is astronomically larger than any classical memory that will ever exist.

Q2. What would a computer need to look like to represent this directly? Exactly what Feynman proposed: a computer whose own information carriers – qubits – are themselves quantum mechanical, so that simulating an n-particle quantum system takes roughly n qubits, not 2^n classical bits. This is quantum simulation, one of the most promising near-term applications of quantum computing, and it is precisely the tool that could one day make the "unsolvable molecule" from our case study solvable.

1.9 Realization Lab: Your First Qubit, For Real

1.9.1 Is This a Quantum-Shaped Problem? A Checklist

Before Reaching for a Quantum Computer, Ask:

- Does the problem involve simulating an inherently quantum system (molecules, materials, quantum devices)? If yes – strong candidate.
- Does the problem have exploitable mathematical structure (e.g. periodicity, as in Shor's

algorithm, Unit 4) rather than being a generic brute-force search?

- Is the problem size actually large enough that classical methods struggle – or would a classical laptop solve it instantly anyway?
- Can the problem tolerate today's noisy, error-prone qubits (Unit 5), or does it need perfect fault-tolerant hardware that doesn't exist yet?

If the honest answers lean "yes, yes, yes, tolerable" – you likely have a quantum-shaped problem. Our case study's molecule simulation scores well on all four counts.

1.9.2 Hello, Qubit: Your First Quantum Circuit in Qiskit

The following creates exactly the superposition state from Section 1.6.3, runs it on a simulator, and shows the measurement statistics – turning Section 1.6 from equations into something you can execute yourself.

Qiskit Code – Equal Superposition and Measurement

```
# Install once: pip install qiskit qiskit-aer
from qiskit import QuantumCircuit
from qiskit_aer import AerSimulator

# Step 1: one qubit, one classical bit to store the measurement result
qc = QuantumCircuit(1, 1)

# Step 2: apply the Hadamard gate -- creates (|0> + |1>) / sqrt(2)
qc.h(0)

# Step 3: measure the qubit into the classical bit
qc.measure(0, 0)

# Step 4: run 1000 times on a simulator and count outcomes
sim = AerSimulator()
result = sim.run(qc, shots=1000).result()
counts = result.get_counts()
print(counts)  # expect roughly {"0": ~500, "1": ~500}
```

Line-by-line, in words: we start every qubit in $|0\rangle$ by default (Section 1.6.1). qc.h(0) applies the Hadamard gate from Section 1.6.3, producing the equal superposition we checked by hand in the worked example. qc.measure(0,0) performs the collapse described in Section 1.6.5. Because measurement is probabilistic, we repeat the whole experiment 1000 times (shots=1000) and count how often each outcome occurs – verifying, experimentally, the P(0) = P(1) = 0.5 we calculated by hand.

1.9.3 Where to Run This

- **No installation:** IBM Quantum Learning – free, browser-based, and can run on real IBM quantum hardware, including access routes relevant to the Heron-class processor at Amaravati Quantum Valley (Unit 5 covers this hardware in depth).
- **Local install:** qiskit.org – full documentation and tutorials.
- **This book's code:** every listing in this book is available at our companion repository – [GitHub link – to be finalized with your co-author] – so you can download, run, and modify every example yourself.

1.10 Before You Move On

Explain It to a 10-Year-Old

In two or three sentences, with no equations and no jargon, explain to someone who has never heard of quantum computing: (a) what a qubit is, and (b) why scientists are excited about quantum computers for simulating molecules. If you can do this without saying "superposition" or "amplitude," you have truly understood Section 1.6 – not just memorised it. (Try recording yourself in 60 seconds – it's harder than it sounds.)

Exercises

1. A qubit is in the state $|\psi\rangle = (\sqrt{3}/2)\,|0\rangle + (1/2)\,|1\rangle$. Verify this is a valid quantum state, and compute P(0) and P(1).

2. Explain, in your own words, why n qubits require 2^n amplitudes but n classical bits require only n bits to describe a definite state.

3. A research group claims their new 500-qubit quantum computer can instantly solve any problem a classical computer cannot. Using the Myth vs. Reality box in Section 1.7, identify what is wrong with this claim.

4. Apply the checklist in Section 1.9.1 to (a) sorting a list of 10 numbers, and (b) simulating a 40-atom catalyst molecule. Which is quantum-shaped, and why?

5. Case-study extension: suggest one other real-world domain (beyond drug discovery) where exponential classical simulation cost is a genuine bottleneck today.

Solutions to all exercises are provided via the book's digital companion (Section 1.9.3).

1.11 References and Further Study

Numbered to match the consolidated References at the end of this volume; the same number identifies the same source everywhere it is cited.

1.11.1 Textbooks

1. M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, 10th Anniversary Edition, Cambridge University Press, 2010.
2. Eleanor Rieffel and Wolfgang Polak, Quantum Computing: A Gentle Introduction, MIT Press, 2011.

1.11.2 Original and Historical Papers

4. A. M. Turing, "On Computable Numbers, with an Application to the Entscheidungsproblem", Proceedings of the London Mathematical Society, 1936.
5. P. A. M. Dirac, "A New Notation for Quantum Mechanics", Mathematical Proceedings of the Cambridge Philosophical Society, 1939.
8. R. P. Feynman, "Simulating Physics with Computers", International Journal of Theoretical Physics, 1982.

1.11.3 Video Lectures

22. IBM Quantum – Qiskit learning platform (qiskit.org), including Qiskit Nature modules on VQE and quantum chemistry.

23. MIT OpenCourseWare – Quantum Information Science lecture series.

1.11.4 R&D Organisations Active in This Area

- Amaravati Quantum Valley (Andhra Pradesh, India) – IBM Heron-class 156-qubit processor deployment, in partnership with TCS and IBM.
- IBM Quantum – one of the largest superconducting-qubit research and cloud-access programmes worldwide.
- India's National Quantum Mission – national programme coordinating quantum research across Indian institutions.
- Google Quantum AI, IonQ, PsiQuantum – leading alternative hardware approaches (superconducting, trapped-ion, and photonic qubits respectively), introduced fully in Unit 5.

Bridge to Unit 2

Where We Go Next

This unit established what a qubit is and why the quantum paradigm exists. Unit 2 asks the next question: if a single qubit is this rich, what happens when qubits interact? We will meet quantum gates precisely (Pauli, Hadamard, Phase, CNOT), the strange and powerful phenomenon of entanglement, and the Bloch sphere in full mathematical detail – the toolbox every quantum circuit in this book, from Unit 3 onward, is built from.

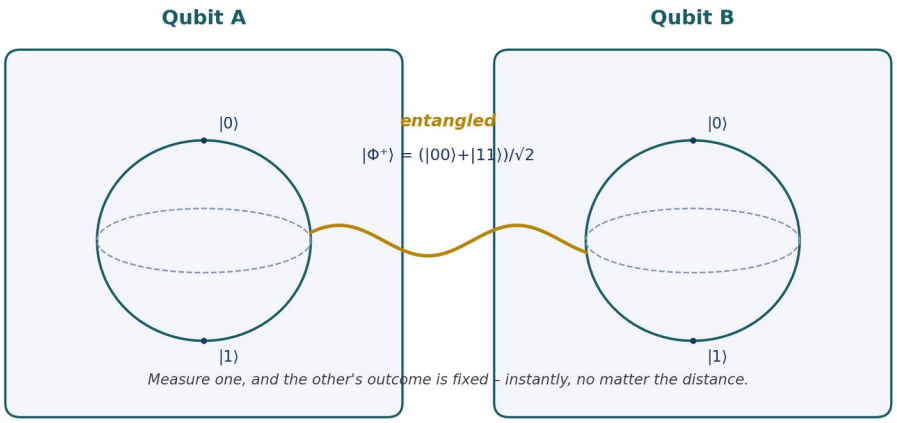
Quantum Computing

Science & Engineering Edition

Unit 2: States, Measurement, Entanglement and Gates

"Spooky action at a distance."

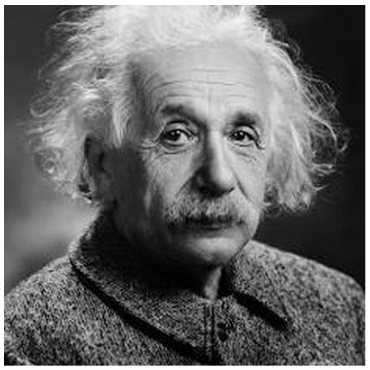
— Albert Einstein, describing entanglement (1947 letter to Max Born)



A correlation with no classical explanation – created, not discovered, the moment two qubits meet.

2.1 Pioneers Behind This Unit

Unit 1 gave us a single qubit. This unit asks what happens when qubits meet – and the answers were fought over, proved, and finally drawn on paper by the people below.



Albert Einstein



Boris Podolsky



Nathan Rosen

EPR – The Paradox That Named the Problem (1935)

In 1935, Albert Einstein, Boris Podolsky and Nathan Rosen published a thought experiment showing that quantum mechanics predicts correlations between distant particles so strong that measuring one instantly constrains the other [3]. They intended it as a criticism – evidence that quantum theory must be incomplete, since nothing should travel faster than light. Their paper did not settle the question, but it gave the phenomenon its name and its first precise description: what Einstein later called 'spooky action at a distance.' We now call it entanglement.

What makes the EPR paper worth reading even today is that its authors were right about the mathematics and wrong about the conclusion – a rare and instructive combination. They correctly derived that quantum mechanics predicts these strange correlations; they simply assumed no sane theory could really behave that way, and concluded something must be missing from quantum mechanics itself. It took until 1964, and John Bell below, to show that nature really does behave exactly as strangely as the equations said. Being technically correct and still wrong about what it means is a lesson every scientist eventually learns.



John Stewart Bell
1928–1990

John Bell – Proving Entanglement Is Real (1964)

Nearly thirty years after EPR, physicist John Bell found a way to settle the argument experimentally rather than philosophically [7]. Bell derived an inequality that any theory with hidden, pre-agreed local values must obey. Quantum mechanics predicts that entangled particles can violate it. Decades of increasingly rigorous experiments – honoured by the 2022 Nobel Prize in Physics – have confirmed the quantum prediction. Bell's theorem is why physicists can say with confidence that entanglement is a genuine physical resource, not a gap in our knowledge.

Bell did this work almost as a side project, without expecting it to be experimentally testable any time soon, let alone in his own lifetime – he passed away in 1990, twelve years before the Nobel Prize eventually recognised the line of research his 1964 paper opened. It is worth remembering that some of the most consequential physics is not a flashy new machine but a single, carefully reasoned inequality on paper – one that took the rest of the field decades to build the technology to actually test.



Felix Bloch
1905–1983

Felix Bloch – The Sphere We Still Draw (1946)

Felix Bloch introduced his geometric picture of a two-level quantum system while working on nuclear magnetic resonance, years before anyone spoke of qubits. Every time a quantum computing paper, textbook, or piece of Qiskit code draws a single qubit as a point on a sphere, it is using Bloch's coordinate system. Section 2.7 of this unit is built entirely on his construction.

Bloch was not trying to invent a tool for a future technology – he was trying to understand how atomic nuclei behave in a magnetic field, work that later won him the 1952 Nobel Prize in Physics. That his geometric picture turned out to be exactly the right way to visualize any two-level quantum system, qubits included, is a good example of how the most useful ideas in a new field often arrive secondhand, borrowed from a completely different problem solved decades earlier.

Where we are going: this unit turns the single qubit of Unit 1 into a working toolkit – multi-qubit states, measurement rules, entanglement, the five gates named in the syllabus, and the Bloch sphere – everything Unit 3's quantum circuits are built from.

2.2 Case Study: The Correlation Einstein Couldn't Accept

Two Coins That Are Not Coins

Prepare two qubits together, separate them – to opposite ends of a lab, or in real experiments, opposite ends of a city – and measure each one in the same basis. Individually, each result is a perfect coin flip: 0 or 1, fifty-fifty, with no way to predict it. Yet the two results always agree. Flip the first "coin" and get 0, and the second, measured independently, kilometres away, is also 0. This is not a manufacturing trick and not a hidden pre-agreed answer – Bell's theorem (Section 2.1) rules that out. The correlation is created fresh, by the physics of measurement itself.

Keep these two questions in mind as you read this unit – we will answer them directly at the

end:

Q1. If nothing travels between the two qubits at measurement time, where does the correlation come from?

Q2. What specific sequence of quantum gates builds this correlation from two ordinary $|0\rangle$ qubits – and how would we verify it actually happened?

2.3 Quantum States and Qubits, Revisited as Vectors

A quantum state is the mathematical object that describes everything the theory allows us to predict about a system. To calculate with qubits rather than just talk about them, we now treat that state as a vector.

2.3.1 The Vector Picture of a Quantum State

The state of one qubit is a vector in a two-dimensional complex vector space, spanned by the basis vectors $|0\rangle$ and $|1\rangle$. In column form:

$$|0\rangle = [1\ \ 0]^T, \quad |1\rangle = [0\ \ 1]^T, \quad |\psi\rangle = \alpha|0\rangle + \beta|1\rangle = [\alpha\ \ \beta]^T$$

Writing states as column vectors is what makes gates calculable: applying a gate is nothing more than multiplying its matrix by this column. Section 2.6 relies on exactly this mechanic for every gate in the syllabus.

2.3.2 Multi-Qubit State Space and the Tensor Product

Two qubits have four computational basis states: $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$. In general, n qubits have 2^n basis states, and the register's state is a linear combination over all of them:

$$|\psi\rangle = \Sigma^x \alpha^x |x\rangle, \quad \Sigma^x |\alpha^x|^2 = 1$$

Individual qubit states are combined with the tensor product, written ⊗: if the first qubit is $|a\rangle$ and the second is $|b\rangle$, their joint state is $|a\rangle\otimes|b\rangle$, usually abbreviated $|ab\rangle$. Not every two-qubit state can be split back into this product form – and that gap is precisely what Section 2.5 calls entanglement.

2.3.3 Global Phase vs Relative Phase

A Distinction Worth Getting Right

Global phase: $|\psi\rangle$ and $e^{i\vartheta}|\psi\rangle$ are the same physical state – the factor multiplies everything equally and is never measurable by itself.

Relative phase: $(|0\rangle+|1\rangle)/\sqrt{2}$ and $(|0\rangle-|1\rangle)/\sqrt{2}$ give identical computational-basis probabilities, yet they are different states – later gates (and later measurements in other bases) tell them apart. Relative phase is a real, usable resource; global phase is not.

Worked Example – Checking Qubit States

$|\psi\rangle = (3/5)|0\rangle + (4/5)|1\rangle$ is valid: $(3/5)^2 + (4/5)^2 = 9/25 + 16/25 = 1$. Measuring gives 0 with probability 9/25 and 1 with probability 16/25.

$|\chi\rangle = |0\rangle + |1\rangle$ is not normalized as written – the squared magnitudes add to 2. Dividing by $\sqrt{2}$ fixes it. Normalization is not cosmetic; unnormalized "states" do not correspond to real probabilities.

2.4 Superposition and Measurement, Made Precise

2.4.1 Equal and Unequal Superpositions

An equal superposition gives every basis state the same probability – $|+\rangle = (|0\rangle+|1\rangle)/\sqrt{2}$, produced from $|0\rangle$ by a Hadamard gate, is the standard example. An unequal superposition, such as $\sqrt{0.8}|0\rangle + \sqrt{0.2}|1\rangle$, still satisfies normalization but assigns 0.8 and 0.2 probability to the two outcomes. Superposition should never be read as "storing several classical answers to read out at once" – a single measurement still returns exactly one classical outcome.

2.4.2 Measurement as Projection and State Update

For a computational-basis measurement, the outcome probabilities and the resulting state update are both governed by the projectors $\Pi_0 = |0\rangle\langle 0|$ and $\Pi_1 = |1\rangle\langle 1|$:

$$P(k) = \langle\psi|\Pi_k|\psi\rangle, \quad \text{post-measurement state} = \Pi_k|\psi\rangle \,/\, \sqrt{P(k)}$$

Measurement is therefore two things at once: it computes a probability, and – if that outcome occurs – it replaces the state with the corresponding basis state. This is the key difference from reading a classical bit: reading never changes a bit, but measuring a qubit always collapses it to match what was observed.

2.4.3 Measurement in Other Bases

The computational basis is not the only option. Measuring in the X basis (states $|+\rangle$ and $|-\rangle$) gives probabilities $|\langle+|\psi\rangle|^2$ and $|\langle-|\psi\rangle|^2$. This is why phase matters: $|+\rangle$ and $|-\rangle$ are indistinguishable in the computational basis but perfectly distinguishable in the X basis. In practice, measuring in another basis is usually done by applying a suitable gate – often H – immediately before an ordinary computational-basis measurement.

Common Misunderstandings About Measurement

- **"Measurement just reveals a hidden value."** No – before measurement the state is genuinely described by amplitudes, not by a fixed number we simply don't know yet.
- **"You can read out the whole superposition."** No – one measurement produces one classical result, sampled from a probability distribution.
- **"Measuring again gives a different, independent answer."** No – the state update means an immediate repeat measurement in the same basis returns the same result with certainty (absent other gates or noise).

Because measurement is destructive and probabilistic, well-designed circuits delay it until every useful transformation – superposition, phase change, entanglement – has already happened.

2.5 Quantum Entanglement and Its Importance

2.5.1 Product States vs Entangled States

A product state can be written as $|a\rangle \otimes |b\rangle$ – an individual state assigned to each qubit. $|0\rangle \otimes |1\rangle = |01\rangle$ is a product state, and so is $|+\rangle \otimes |0\rangle$, even though the first qubit is in superposition. A state that cannot be factored this way is entangled. The standard textbook example is one of the four Bell states:

$$|\Phi^+\rangle = (|00\rangle + |11\rangle) / \sqrt{2}$$

Measuring the first qubit of $|\Phi^+\rangle$ gives 0 or 1 with equal probability – but whichever result appears, measuring the second qubit in the same basis always gives the same value. Neither qubit has a definite state on its own; the information lives only in the joint, two-qubit description. This is the case study's "two coins" from Section 2.2, written precisely.

2.5.2 Entanglement versus Classical Correlation

Classical Correlation	Quantum Entanglement
Values are definite before observation (e.g. two cards, one red, one blue, sealed in envelopes)	No definite pre-assigned local values – ruled out by violation of Bell's inequality
Opening one envelope reveals pre-existing information about the other	Each individual result is genuinely random until measured
No restriction from relativity is being tested	Correlation appears instantly on comparison, yet carries no usable faster-than-light signal

Entanglement does not allow faster-than-light communication: each local result is still random, and the correlation is only visible once the two measurement records are brought together by ordinary classical means.

2.5.3 Creating Entanglement with Gates

The standard recipe uses exactly the two gates this unit's syllabus asks for: a Hadamard gate followed by a CNOT gate. Starting from $|00\rangle$, H on the first qubit gives $(|00\rangle+|10\rangle)/\sqrt{2}$; CNOT (control = first qubit) then maps $|10\rangle$ to $|11\rangle$ while leaving $|00\rangle$ unchanged, producing $(|00\rangle+|11\rangle)/\sqrt{2} = |\Phi^+\rangle$. Entanglement is created by coherent gate operations, not by measurement – and not automatically: applying CNOT to $|00\rangle$ alone just gives $|00\rangle$ back, still a product state. Both the input state and the gate matter.

2.5.4 Why Entanglement Matters

Entanglement supplies correlations across a register that have no efficient classical description, and it underlies quantum teleportation, entanglement-based communication, and several approaches to quantum error correction (all previewed in Unit 5). It is not automatic problem-solving power – it is a resource that must be created, protected from noise, and consumed at the right moment in a circuit.

2.6 The Gate Toolkit: Pauli, Hadamard, Phase, CNOT, Toffoli

Quantum gates are unitary matrices: U is unitary if U†U = I. Unitarity guarantees two things at once – every valid state stays valid after the gate, and every gate is reversible, unlike classical gates such as AND, which can lose information.

2.6.1 The Pauli Gates: X, Y, Z

X is the quantum bit-flip, exchanging $|0\rangle$ and $|1\rangle$. Y also exchanges the basis states but attaches imaginary phase factors. Z leaves $|0\rangle$ alone and multiplies $|1\rangle$ by −1 – a phase-flip that is invisible to a computational-basis measurement but very visible to later gates.

$$X = [[0,1],[1,0]] \qquad Y = [[0,-i],[i,0]] \qquad Z = [[1,0],[0,-1]]$$

2.6.2 The Hadamard Gate

$H = (1/\sqrt{2})[[1,1],[1,-1]]$ maps $|0\rangle \to |+\rangle$ and $|1\rangle \to |-\rangle$, and $H^2 = I$. It is the standard way to create superposition at the start of a circuit, and the standard way to switch into the X basis just before measurement. H produces a coherent superposition, not classical randomness – the two components can still interfere later.

2.6.3 Phase Gates: P(φ), S, T

A phase gate $P(\varphi) = [[1,0],[0,e^{i\varphi}]]$ leaves $|0\rangle$ alone and multiplies $|1\rangle$ by $e^{i\varphi}$, changing relative phase without touching computational-basis probabilities. Two named special cases matter for universal gate sets: S uses $\varphi = \pi/2$, and T uses $\varphi = \pi/4$.

2.6.4 CNOT: The Entangling Gate

CNOT is a two-qubit gate: the target flips exactly when the control is 1.

$$CNOT\,|a,b\rangle = |a,\ b \oplus a\rangle \qquad (\oplus = \text{addition modulo 2})$$

CNOT is the workhorse of Section 2.5: with the control in superposition, it is exactly the gate that produces entanglement. With the control in a definite basis state, it behaves like an ordinary conditional bit flip.

2.6.5 Toffoli: Reversible Classical Logic Inside a Quantum Circuit

Toffoli (controlled-controlled-NOT) is a three-qubit gate: the target flips only when both control qubits are 1.

$$Toffoli\,|a,b,c\rangle = |a,\ b,\ c \oplus ab\rangle$$

Because it is reversible, Toffoli can embed classical AND-like logic inside a quantum circuit without discarding information – a bridge between classical reversible computation and unitary quantum operations.

2.6.6 Gate Composition and Order

Applying gate U then gate V gives the final state $VU|\psi\rangle$ – the rightmost matrix acts first, even though circuit diagrams are read left to right. Order matters: most gates do not commute, so applying X then Z is generally not the same as Z then X.

Summary Table of the Syllabus Gates

Gate	Matrix / Action	Main Effect
Pauli X	[[0,1],[1,0]]	Flips $\|0\rangle$ and $\|1\rangle$
Pauli Y	[[0,−i],[i,0]]	Flips basis states, adds phase
Pauli Z	[[1,0],[0,−1]]	Adds a minus sign to $\|1\rangle$
Hadamard	$(1/\sqrt{2})$[[1,1],[1,−1]]	Creates / analyzes equal superpositions
Phase P(φ)	[[1,0],[0,$e^{iφ}$]]	Changes relative phase
CNOT	$\|a,b\rangle \to \|a,b\oplus a\rangle$	Flips target if control is 1
Toffoli	$\|a,b,c\rangle \to \|a,b,c\oplus ab\rangle$	Flips target if both controls are 1

Worked Example – Composing Gates

Start from $|+\rangle|0\rangle = (|00\rangle+|10\rangle)/\sqrt{2}$ and apply CNOT (first qubit as control). $|00\rangle$ is unchanged; $|10\rangle$ becomes $|11\rangle$. The result, $(|00\rangle+|11\rangle)/\sqrt{2}$, is not a product of two single-qubit states – it is entangled, confirming the recipe from Section 2.5.3 by direct calculation rather than assertion.

2.7 The Bloch Sphere

The Bloch sphere gives a single pure qubit state a home as a point on a sphere – a bridge between the algebra of amplitudes and geometric intuition.

2.7.1 The Bloch Sphere Formula

$$|\psi\rangle = \cos(\theta/2)|0\rangle + e^{i\varphi}\sin(\theta/2)|1\rangle$$

θ measures the swing from the $|0\rangle$ pole toward the $|1\rangle$ pole, with $P(0)=\cos^2(\theta/2)$ and $P(1)=\sin^2(\theta/2)$. φ measures relative phase, the angle travelled around the equator. Changing θ changes measurement probabilities; changing φ changes phase while leaving computational-basis probabilities untouched – exactly the distinction introduced in Section 2.3.3.

2.7.2 Coordinates from Amplitudes

For a normalized state $\alpha|0\rangle+\beta|1\rangle$, the Cartesian coordinates on the sphere are $x = 2\mathrm{Re}(\alpha^*\beta)$, $y = 2\mathrm{Im}(\alpha^*\beta)$, $z = |\alpha|^2-|\beta|^2$. $|0\rangle$ sits at the north pole, $|1\rangle$ at the south pole; $|+\rangle$ and $|-\rangle$ sit at opposite ends of the x-axis, and $(|0\rangle\pm i|1\rangle)/\sqrt{2}$ sit at opposite ends of the y-axis – all differing only in phase, which the equatorial position makes visible.

2.7.3 Gate Actions as Rotations

Single-qubit gates become rotations of the sphere: X is a π-radian rotation about the x-axis, Z a π-radian rotation about the z-axis, and a phase gate rotates a state around the equator. H is special because it swaps the roles of the x-axis and z-axis, mapping $|0\rangle \leftrightarrow |+\rangle$ and $|1\rangle \leftrightarrow |-\rangle$ – which is exactly why it both creates superposition and converts between the computational and X measurement bases.

2.7.4 Where the Bloch Sphere Breaks Down

One Sphere, One Qubit – Not More

The Bloch sphere describes a single pure qubit. It cannot represent a general multi-qubit state, and entangled states are the clearest reason why: for $|\Phi^+\rangle = (|00\rangle+|11\rangle)/\sqrt{2}$, each individual qubit looks maximally uncertain in isolation, yet the pair is strongly correlated. That correlation lives in the joint state, which has no single-qubit Bloch picture. Use the sphere for single-qubit intuition; use full state vectors and tensor products for anything entangled.

2.8 Case Study Revisited: Building the Correlation

Closing the Loop

Q1. Where does the correlation come from? From the structure of the joint quantum state itself (Section 2.5.1), not from a signal or a pre-agreed answer. Bell's theorem (Section 2.1) rules out any explanation based on hidden local values fixed in advance – the correlation is a genuine, verified feature of quantum mechanics, and it carries no usable faster-than-light message.

Q2. How do we build and verify it? Exactly the recipe of Section 2.5.3: a Hadamard gate on one qubit, then a CNOT with that qubit as control, starting from $|00\rangle$. The Realization Lab below builds this circuit, measures both qubits over many shots, and checks that only the outcomes 00 and 11 ever appear – the experimental signature of $|\Phi^+\rangle$.

2.9 Realization Lab: Building and Verifying a Bell Pair

2.9.1 Checklist: Will This Circuit Entangle?

Before Trusting a Circuit to Produce Entanglement, Ask:

- Is at least one qubit going into the CNOT's control in superposition, not a definite basis state? (Section 2.5.3 – CNOT alone on basis states never entangles.)
- Is the gate that creates the superposition applied before the CNOT, not after?
- Will measurement happen only after both gates have acted, so the entanglement survives to be observed?
- If verifying by hand: does the resulting state fail to factor into a product $|a\rangle\otimes|b\rangle$? If it factors, it is not entangled, however complicated it looks.

2.9.2 Qiskit Code: Creating and Measuring $|\Phi^+\rangle$

The following builds the exact circuit from Section 2.5.3 – H then CNOT – and measures both qubits over many shots to confirm the correlation predicted in Section 2.5.1.

Qiskit Code – Bell Pair Creation and Measurement

```
# Install once: pip install qiskit qiskit-aer
from qiskit import QuantumCircuit
from qiskit_aer import AerSimulator

# Step 1: two qubits, two classical bits
qc = QuantumCircuit(2, 2)

# Step 2: put qubit 0 into superposition
qc.h(0)

# Step 3: entangle -- qubit 0 controls, qubit 1 is the target
qc.cx(0, 1)

# Step 4: measure both qubits
qc.measure([0, 1], [0, 1])

# Step 5: run 1000 times and count outcomes
sim = AerSimulator()
result = sim.run(qc, shots=1000).result()
counts = result.get_counts()
print(counts)  # expect roughly {"00": ~500, "11": ~500}, and no "01" or
"10"
```

Line-by-line: qc.h(0) creates $(|0\rangle+|1\rangle)/\sqrt{2}$ on qubit 0 (Section 2.6.2). qc.cx(0,1) is Qiskit's CNOT, entangling the two qubits exactly as in Section 2.5.3. Because the two classical outcomes are now correlated rather than independent, the printed counts should show only "00" and "11" – never "01" or "10" – which is the measurable signature of entanglement predicted in Section 2.5.1.

2.9.3 Where to Run This

- **No installation:** IBM Quantum Learning – the same free, browser-based environment used in Unit 1, including access routes to real hardware such as the Heron-class processor at Amaravati Quantum Valley.
- **Local install:** qiskit.org – full documentation and tutorials.
- **This book's code:** every listing in this book is available at our companion repository – [GitHub link – to be finalized with your co-author].

2.10 Before You Move On

Important Formulas from This Unit

- Single-qubit state: $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, with $|\alpha|^2 + |\beta|^2 = 1$

- Computational measurement: $P(0) = |\alpha|^2$, $P(1) = |\beta|^2$
- General n-qubit state: $|\psi\rangle = \Sigma_x \alpha_x |x\rangle$, $\Sigma_x |\alpha_x|^2 = 1$
- Projection measurement: $P(k) = \langle\psi|\Pi_k|\psi\rangle$
- Bell state: $|\Phi^+\rangle = (|00\rangle+|11\rangle)/\sqrt{2}$
- CNOT: $|a,b\rangle \rightarrow |a, b\oplus a\rangle$ Toffoli: $|a,b,c\rangle \rightarrow |a,b,c\oplus ab\rangle$
- Bloch sphere: $|\psi\rangle = \cos(\theta/2)|0\rangle + e^{i\vartheta} \sin(\theta/2)|1\rangle$

Exercises

1. Verify that $|\psi\rangle = (\sqrt{3}/2)|0\rangle + (1/2)|1\rangle$ is normalized, and give its Bloch sphere angle θ.

2. Explain, in your own words, the difference between global phase and relative phase, with an example of each.

3. Starting from $|01\rangle$, apply CNOT with the first qubit as control. Is the result entangled? Justify your answer using Section 2.5.1.

4. Using the checklist in Section 2.9.1, explain why applying H to both qubits of $|00\rangle$ (with no CNOT) does not produce an entangled state.

5. A classmate claims that measuring one half of a Bell pair lets you send a message to the other half instantly. Using Section 2.5.2, explain what is wrong with this claim.

6. Compute the Bloch sphere coordinates (x, y, z) for the state $(|0\rangle - i|1\rangle)/\sqrt{2}$.

Solutions to all exercises are provided via the book's digital companion (Section 2.9.3).

2.11 References and Further Study

Numbered to match the consolidated References at the end of this volume.

2.11.1 Textbooks

1. M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, 10th Anniversary Edition, Cambridge University Press, 2010.
2. Eleanor Rieffel and Wolfgang Polak, Quantum Computing: A Gentle Introduction, MIT Press, 2011.

2.11.2 Original and Historical Papers

3. A. Einstein, B. Podolsky, N. Rosen, "Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?", Physical Review, 1935.
7. J. S. Bell, "On the Einstein Podolsky Rosen Paradox", Physics Physique Fizika, 1964.

2.11.3 Video Lectures

22. IBM Quantum – Qiskit learning platform (qiskit.org), including Qiskit Nature modules on VQE and quantum chemistry.
23. MIT OpenCourseWare – Quantum Information Science lecture series.

Bridge to Unit 3

Where We Go Next

This unit gave us the full single- and two-qubit toolkit: states as vectors, measurement as projection, entanglement, the five syllabus gates, and the Bloch sphere. Unit 3 puts these pieces together into quantum circuits – combining single- and multi-qubit gates deliberately, exploring physical implementations of the qubit, and building every one of the four Bell states as worked circuit designs.

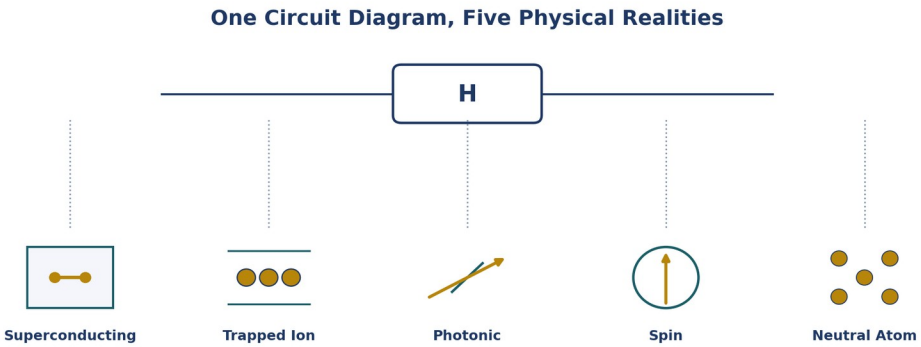
Quantum Computing

Science & Engineering Edition

Unit 3: Qubits in Hardware, Quantum Circuits, and Bell States

"The circuit diagram does not care what the qubit is made of – and that is exactly the point."

— a working principle of quantum hardware engineering



Same gate. Same mathematics. Five very different ways to build it in the lab.

The same Hadamard gate, built five completely different ways – chosen by the problem, not by preference.

3.1 Pioneers Behind This Unit

Units 1 and 2 built the mathematics of a qubit. This unit asks a different question: what is a qubit actually made of, and how do we build circuits out of real gates on real hardware? The answers trace back to three physicists who learned how to control individual quantum systems in the laboratory.

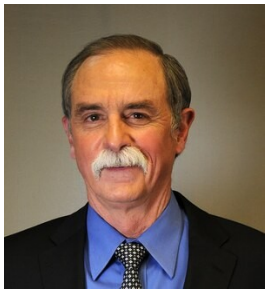


Brian Josephson
b. 1940

Brian Josephson – The Junction That Powers Superconducting Qubits (1962)

As a 22-year-old graduate student at Cambridge, Brian Josephson predicted that a supercurrent could tunnel between two superconductors separated by a thin insulating barrier – what is now called a Josephson junction [6]. He won the 1973 Nobel Prize in Physics for the discovery. Josephson junctions are the essential non-linear circuit element that gives superconducting qubits their discrete, controllable energy levels; Section 3.3.3 of this unit is built directly on his result.

What makes Josephson's story remarkable is the gap between the prediction and its use: he worked out the effect purely from theory, as a student still finishing his PhD, with no quantum computer anywhere in sight to motivate the question. Six decades later, that same effect is the physical basis of the very chip – IBM's Heron processor at Amaravati, discussed in this unit's case study – that this book uses as its running example of real quantum hardware. Few pieces of graduate-student theory have had a longer or stranger afterlife.



David Wineland
b. 1944

David Wineland – Trapping and Controlling a Single Ion (1978 onward)

David Wineland pioneered the use of lasers to cool, trap, and precisely control the internal states of individual ions – the foundational technique behind trapped-ion quantum computing. His group performed some of the first demonstrations of quantum logic gates on a single controlled particle. He shared the 2012 Nobel Prize in Physics for "ground-breaking experimental methods that enable measuring and manipulation of individual quantum systems" [15].

Wineland spent years at the U.S. National Institute of Standards and Technology perfecting laser-cooling techniques originally developed for building better atomic clocks – a patient, decades-long refinement of control over single atoms that only later turned out to be exactly what quantum computing needed. His career is a useful reminder that the tools for a new technology are often built by people solving an entirely different problem first, years before anyone calls the result a qubit.



Serge Haroche
b. 1944

Serge Haroche – Trapping Light Instead of Matter (1980s onward)

Working from the opposite direction, Serge Haroche developed cavity quantum electrodynamics techniques to trap individual photons between mirrors and observe them without destroying them – methods that underpin photonic and light-based approaches to quantum information. He shared the 2012 Nobel Prize in Physics with Wineland [15]: one controlled matter to hold light steady, the other controlled light to hold matter's secrets steady.

Haroche's central technical challenge was almost the opposite of the usual measurement problem: ordinary measurement destroys a photon on detection, but his group found ways to gently infer a photon's presence without absorbing it at all. That kind of measurement – careful enough not to disturb the very state being read – is precisely the discipline every quantum computing platform in Section 3.3 must practice, and Haroche's cavity experiments were among the first to prove it was even possible.

Where we are going: this unit connects the abstract qubit of Units 1–2 to real hardware, formalizes the quantum circuit model, and uses both to build – and fully verify – every one of the four Bell states.

3.2 Case Study: Turning a Circuit Diagram Into a Real Machine

156 Qubits, One Chip

India's largest quantum computer, being installed at the Quantum Valley Tech Park in Amaravati, is built around an IBM Quantum System Two with a 156-qubit superconducting Heron processor, cooled to a fraction of a degree above absolute zero. Every Hadamard and CNOT drawn as a clean symbol on paper in Units 1–2 has to become a real microwave pulse, timed to nanosecond precision, on a real superconducting circuit – before heat, stray fields, or a few extra nanoseconds of delay erase the very superposition the pulse was meant to create.

Keep these two questions in mind as you read this unit – we will answer them directly at the end:

Q1. Why isn't there simply one "best" way to build a qubit, the way there is essentially one way to build a transistor?

Q2. What has to be true of a physical system, and of a circuit design, before the elegant algebra of Units 1–2 can be trusted to run correctly on it?

3.3 The Qubit as Physical Reality

A qubit is, abstractly, a normalized vector in a two-dimensional complex space. Physically, it is always some real two-level quantum system, deliberately chosen, initialized, controlled, and measured. This section surveys the platforms doing that job today.

3.3.1 What "Physical Implementation" Means

A physical implementation of a qubit is a real system in which two quantum states are chosen, controlled, and read out as the computational basis states $|0\rangle$ and $|1\rangle$. The hardware differs completely from platform to platform – energy levels of a circuit, internal states of an ion, polarizations of a photon – but once the two basis states are fixed, the same mathematical qubit model from Units 1–2 applies unchanged. This separation between abstract circuit theory and physical realization is precisely why the circuit model in Section 3.5 is useful across every platform below.

3.3.2 Requirements for a Useful Qubit

Four Requirements, Plus One Constraint

- **Two distinguishable states** that can be labeled $|0\rangle$ and $|1\rangle$.
- **Initialization** – the ability to reliably prepare a known starting state.
- **Control** – the ability to apply accurate operations that change the state as intended.
- **Measurement** – the ability to read out a classical result at the end.
- **Coherence** – the constraint underlying all four: the phase relationships in a superposition must survive long enough for useful gates to run before the environment destroys them (decoherence).

No platform satisfies all five requirements perfectly. Some have fast gates, some have long coherence, some are easier to scale to many qubits and connect together. Sections 3.3.3–3.3.7 survey the main approaches; Section 3.3.8 compares them directly.

3.3.3 Superconducting Qubits

Superconducting qubits are electrical circuits, cooled to near absolute zero, built around Josephson junctions (Section 3.1) – non-linear elements that give the circuit discrete, addressable quantum energy levels. Two of these levels are chosen as $|0\rangle$ and $|1\rangle$, and microwave pulses drive transitions and rotations between them. Superconducting qubits can be fabricated with techniques related to conventional microelectronics and support comparatively fast gates, but they demand a complex cryogenic control system to maintain superconductivity and suppress thermal noise – this is the platform behind IBM's Heron processor and the Amaravati installation from Section 3.2.

3.3.4 Trapped Ion Qubits

Trapped ion systems confine individual charged atoms using electromagnetic fields and use two of the ion's internal energy states as $|0\rangle$ and $|1\rangle$. Lasers initialize, manipulate, and read out the states, and multi-qubit gates are engineered through the ions' shared vibrational motion in the trap (Section 3.1, Wineland). Because

ions are naturally identical and well isolated from their environment, this platform can achieve strong coherence, at the cost of comparatively slower, laser-dependent control.

3.3.5 Photonic Qubits

Photonic qubits encode information in single photons – commonly as horizontal versus vertical polarization, two paths through an interferometer, or arrival time. Optical elements such as beam splitters, phase shifters, and wave plates implement single-qubit gates. Photons travel easily through fibre or free space, making this platform central to quantum communication, but photons rarely interact with one another directly, so two-qubit gates typically require specialized measurement-based or nonlinear-optical techniques (Section 3.1, Haroche).

3.3.6 Spin Qubits

Spin qubits use the quantum spin of an electron, a nucleus, or a defect centre – such as a nitrogen-vacancy centre in diamond – with the two spin orientations along a chosen axis serving as $|0\rangle$ and $|1\rangle$. Spin is not a tiny classical arrow; it is a genuine quantum degree of freedom described by amplitudes, controlled by magnetic fields, microwave pulses, or electrical signals depending on the design. Spin qubits are attractive for their small size and potential compatibility with semiconductor fabrication.

3.3.7 Neutral Atom Qubits

Neutral atom qubits use electrically uncharged atoms held in place by optical tweezers – tightly focused laser beams – with two internal atomic states serving as $|0\rangle$ and $|1\rangle$. Many designs excite atoms into highly energetic Rydberg states to create strong, switchable interactions for multi-qubit gates. Because tweezer positions are individually programmable, this platform is well suited to arranging large, reconfigurable arrays of qubits.

3.3.8 Comparing Platforms: Coherence, Fidelity, Scalability

Platform	Basis States	Main Control	Key Trade-off
Superconducting	Two circuit energy levels	Microwave pulses	Fast gates; demanding cryogenics
Trapped ion	Two internal ion states	Laser pulses	Strong coherence; slower control
Photonic	Polarization, path, or time bin	Optical elements	Easy transmission; weak photon–photon interaction
Spin	Spin up / spin down	Magnetic, microwave, or electrical	Small footprint; still maturing at scale
Neutral atom	Internal / Rydberg states	Optical tweezers, lasers	Reconfigurable arrays; interaction control needed

Three Numbers That Decide a Platform's Usefulness

Coherence time: how long the qubit keeps its quantum phase information before the environment destroys it.

Gate time: how long one operation takes. A platform is useful when many gate times fit inside one coherence time.

Gate fidelity: how closely a real physical operation matches its ideal unitary matrix from Section 3.5.2. No real gate is perfect; the gap between real and ideal is what error correction (Unit 5) exists to manage.

No platform is universally best – which is precisely why five different technologies are still being actively developed. What every platform shares is the same abstract circuit model: a Hadamard gate is a microwave pulse on a superconducting chip, a laser pulse on a trapped ion, and an optical element on a photon, but it is the same H matrix from Section 3.5.2 in every case.

3.4 The Bloch Sphere, Revisited for Circuit Design

Unit 2 introduced the Bloch sphere formula $|\psi\rangle = \cos(\theta/2)|0\rangle + e^{i\varphi}\sin(\theta/2)|1\rangle$ and showed how gates act as rotations. This section adds the worked examples every circuit designer eventually needs, and ties the picture directly to the gates used in circuit design below.

3.4.1 Locating States on the Sphere: Worked Examples

Six States Every Circuit Designer Should Recognize on Sight

- $|0\rangle$: $\theta = 0$ – north pole. The formula gives $\cos(0)|0\rangle + e^{i\varphi}\sin(0)|1\rangle = |0\rangle$ for every φ.
- $|1\rangle$: $\theta = \pi$ – south pole. The formula gives $e^{i\varphi}|1\rangle$, which is $|1\rangle$ up to an unobservable global phase.
- $|+\rangle = (|0\rangle+|1\rangle)/\sqrt{2}$: $\theta = \pi/2$, $\varphi = 0$ – positive x-axis, equal magnitude, zero relative phase.
- $|-\rangle = (|0\rangle-|1\rangle)/\sqrt{2}$: $\theta = \pi/2$, $\varphi = \pi$ – negative x-axis, same magnitudes, relative phase π.
- $(|0\rangle+i|1\rangle)/\sqrt{2}$: $\theta = \pi/2$, $\varphi = \pi/2$ – positive y-axis.
- $(|0\rangle-i|1\rangle)/\sqrt{2}$: $\theta = \pi/2$, $\varphi = -\pi/2$ – negative y-axis.

Worked Example – From Angle to Probability

Problem: A state has $\theta = \pi/3$. Find P(0) and P(1).

Solution: $P(0) = \cos^2(\pi/6) = 3/4$, and $P(1) = \sin^2(\pi/6) = 1/4$. The angle φ was not given because it does not affect these computational-basis probabilities at all – it only fixes where the state sits around the equator, which later gates can still make visible.

3.4.2 Gates as Rotations, Applied to Circuit Design

Every valid single-qubit gate is a rotation of the Bloch sphere, because a unitary matrix preserves the length of the state vector. X is a π rotation about the x-axis, Z a π rotation about the z-axis, and H swaps the roles of the x- and z-axes – which is exactly why applying H then Z to $|0\rangle$ gives a different result from applying Z then H (Section 3.5.3 makes this precise algebraically). This geometric picture is a useful sanity check when designing a circuit: if a sequence of gates is supposed to return a qubit to its starting point, the corresponding rotations must compose back to the identity.

One Limitation, Restated

The Bloch sphere pictures one pure qubit. The moment two qubits become entangled – as in Section 3.6 – there is no single-qubit point on any sphere that captures the joint state. That is not a flaw in the picture; it is the clearest possible signal that entanglement is a genuinely multi-qubit phenomenon.

3.5 Building Quantum Circuits

3.5.1 The Circuit Model: Wires, Gates, Time

A quantum circuit represents qubits as horizontal wires and operations as gates placed on those wires, read left to right in time. The input state is written on the left, gates transform it step by step, and measurement is placed near the end when a classical result is needed. Visually this resembles a classical logic diagram, but the meaning is different: classical wires carry definite bit values, while quantum wires carry amplitudes, and gates apply unitary transformations to the full state vector rather than a Boolean function to a bit.

3.5.2 State Vectors and Gate Matrices

As column vectors, $|0\rangle = [1, 0]^T$ and $|1\rangle = [0, 1]^T$, so $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = [\alpha, \beta]^T$. A single-qubit gate is a 2×2 unitary matrix; applying the gate means multiplying that matrix by the state vector. For n qubits, the basis has 2^n states, the state vector has 2^n entries, and an n-qubit gate is a $2^n \times 2^n$ unitary matrix – the tensor product (⊗) combines individual qubits, so $|0\rangle \otimes |1\rangle$ is written $|01\rangle$. This is exactly the exponential growth first met in Unit 1's case study, now expressed as the size of the matrices a simulator must multiply.

3.5.3 A Stepwise Method for Designing a Circuit

Five Steps, Every Time

- 1. Write the input state (commonly all qubits start as $|0\rangle$).
- 2. Identify which qubits need superposition and apply single-qubit gates such as H.
- 3. Identify which qubits must become correlated and apply multi-qubit gates such as CNOT or CZ.
- 4. Decide the measurement basis.
- 5. Calculate or simulate the output probabilities.

Gate order matters, because most gates do not commute. Applying H then Z to $|0\rangle$ gives $Z|+\rangle = |-\rangle$. Applying Z then H to $|0\rangle$ gives $H|0\rangle = |+\rangle$, since $Z|0\rangle = |0\rangle$. Two different gate orders, two different states – exactly the non-commuting behaviour flagged geometrically in Section 3.4.2.

3.5.4 Worked Circuit Designs

Design Problem – Output Only 00 and 11, Never 01 or 10

This measurement pattern is a Bell-type correlation. Start from $|00\rangle$, apply H to the first qubit to get $(|00\rangle+|10\rangle)/\sqrt{2}$, then CNOT from the first qubit to the second. The result is $(|00\rangle+|11\rangle)/\sqrt{2}$ – exactly the required pattern, and exactly the circuit Section 3.6 studies in full.

Design Problem – Output Only 01 and 10

Build the same circuit as above to reach $(|00\rangle+|11\rangle)/\sqrt{2}$, then apply X to the second qubit. $|00\rangle$ becomes $|01\rangle$ and $|11\rangle$ becomes $|10\rangle$, giving $(|01\rangle+|10\rangle)/\sqrt{2}$ – the required pattern, reached by reusing a known building block rather than designing from scratch.

This is the general habit worth keeping: work backward from the desired state. Equal amplitudes suggest a Hadamard gate; correlated terms suggest CNOT or CZ; a state differing only by signs or phases suggests a phase gate such as Z, S, or T.

3.5.5 Common Circuit Design Mistakes

Five Mistakes Worth Checking For

- **Wrong control/target:** swapping CNOT's control and target changes the result – or, if the intended control is still $|0\rangle$, does nothing at all.
- **Measuring too early:** measuring the first qubit right after H, before CNOT runs, collapses the superposition before entanglement can form.
- **Ignoring relative phase:** $|\Phi^+\rangle$ and $|\Phi^-\rangle$ give identical computational-basis probabilities but are different states – later gates can tell them apart.
- **Treating entangled qubits as separate:** once entangled, no single-qubit state can be assigned to either qubit alone – only the joint state is valid.
- **Forgetting normalization:** a missing factor of $1/\sqrt{2}$ in hand-written algebra is a simple, common source of probabilities that don't add to one.

3.5.6 Reading Quantum Circuit Diagrams Correctly

A circuit diagram is read left to right in time: gates in different columns act in sequence, and gates in the same column on different wires (sharing no control or target) act in parallel. A controlled gate is read by its control dot and target symbol – the control qubit is tested, not measured, and if it is in superposition the controlled operation can create entanglement rather than simply behaving like a classical conditional gate. A careful

reader traces a circuit in stages: write the initial state, apply each column of gates, simplify if possible, and only then compute measurement probabilities – the same five-step discipline as Section 3.5.3.

3.6 The Four Bell States

Bell states are the standard two-qubit entangled states, named for John Bell (Unit 2). They are the clearest introductory example of everything this unit has built: a physical qubit, a circuit made of real gates, and a state that has no classical analogue.

3.6.1 Preparing $|\Phi^+\rangle$, Revisited

Starting from $|00\rangle$, H on the first qubit gives $(H|0\rangle)|0\rangle = ((|0\rangle+|1\rangle)/\sqrt{2})|0\rangle = (|00\rangle+|10\rangle)/\sqrt{2}$. CNOT, with the first qubit as control, leaves $|00\rangle$ unchanged (control 0) and flips $|10\rangle$ to $|11\rangle$ (control 1), giving:

$$(|00\rangle + |11\rangle) / \sqrt{2} = |\Phi^+\rangle$$

H creates the superposition; CNOT spreads it into a two-qubit correlation. If the control qubit had instead been a definite $|0\rangle$ or $|1\rangle$, CNOT would only be an ordinary conditional flip – it is superposition going into CNOT that produces entanglement coming out.

3.6.2 The Other Three Bell States

State	Formula	Reached From $\|\Phi^+\rangle$ By
$\|\Phi^+\rangle$	$(\|00\rangle+\|11\rangle)/\sqrt{2}$	– (starting point)
$\|\Phi^-\rangle$	$(\|00\rangle-\|11\rangle)/\sqrt{2}$	Z on either qubit
$\|\Psi^+\rangle$	$(\|01\rangle+\|10\rangle)/\sqrt{2}$	X on either qubit
$\|\Psi^-\rangle$	$(\|01\rangle-\|10\rangle)/\sqrt{2}$	X then Z (both on the same qubit)

The Φ states pair matching strings (00, 11); the Ψ states pair opposite strings (01, 10). The plus/minus subscript is a relative phase, invisible to a direct computational-basis measurement but real: it is exactly the kind of information Section 3.1.5–style phase tracking exists to protect.

3.6.3 Proving Entanglement: Bell States Are Not Product States

A Direct Proof

Suppose $|\Phi^+\rangle$ could be written as a product $(a|0\rangle+b|1\rangle)(c|0\rangle+d|1\rangle) = ac|00\rangle+ad|01\rangle+bc|10\rangle+bd|11\rangle$. Matching this to $|\Phi^+\rangle$ requires $ad = 0$ and $bc = 0$, while ac and bd must both be nonzero. No choice of a, b, c, d satisfies all four conditions at once. Therefore $|\Phi^+\rangle$ cannot be factored into two single-qubit states – by definition, it is entangled.

Compare this with the product state $|+\rangle|+\rangle = (|00\rangle+|01\rangle+|10\rangle+|11\rangle)/2$, which has amplitude on all four basis strings yet is still just two independent qubits in superposition. $|\Phi^+\rangle$ has amplitude on only two strings, but is not two independent random bits – its correlation comes from the structure of one indivisible two-qubit state.

3.6.4 Worked Examples with Bell States

Four Short Calculations

1. Measuring $|\Psi^+\rangle = (|01\rangle+|10\rangle)/\sqrt{2}$: P(01) = P(10) = 1/2; P(00) = P(11) = 0. The two qubits always disagree.

2. Apply Z to the first qubit of $|\Phi^+\rangle$: the $|00\rangle$ term is unchanged (first qubit is 0); the $|11\rangle$ term gains a minus sign (first qubit is 1). Result: $(|00\rangle-|11\rangle)/\sqrt{2} = |\Phi^-\rangle$.

3. Apply X to the second qubit of $|\Phi^-\rangle$: $|00\rangle\to|01\rangle$ and $|11\rangle\to|10\rangle$, with the minus sign carried along: $(|01\rangle-|10\rangle)/\sqrt{2} = |\Psi^-\rangle$.

4. Why is $|\Phi^+\rangle$ correlation, not a hidden classical pair? Before measurement, the state is a coherent superposition, not one fixed (but unknown) classical string – the relative phase is real and can still affect later gates, which no classical mixture of "00 or 11" could reproduce.

Every one of these calculations uses only tools already built in this unit: qubit notation, gate matrices, multi-qubit basis states, and measurement probabilities. The same step-by-step habit scales directly to the larger circuits of Unit 4.

3.7 Case Study Revisited: From Diagram to Machine

Closing the Loop

Q1. Why isn't there one "best" qubit? Because no platform satisfies every requirement from Section 3.3.2 at once. Superconducting qubits trade cryogenic complexity for speed; trapped ions trade slower control for coherence; photons trade weak interactions for easy transmission. "Best" depends on which trade-off a given task can tolerate – exactly the comparison in Section 3.3.8.

Q2. What has to be true before the algebra can be trusted? The physical system must meet Section 3.3.2's five requirements, and the circuit itself must be designed correctly – right control and target, measurement placed at the end, relative phase tracked, normalization checked (Section 3.5.5). On a real chip like Amaravati's Heron processor, every gate also carries a fidelity below 1 (Section 3.3.8), which is precisely why Unit 5's error correction exists.

3.8 Realization Lab: Preparing and Verifying All Four Bell States

3.8.1 Checklist: Did the Circuit Reach the Bell State You Wanted?

Before Trusting the Output, Check:

- Is H applied before CNOT, and to the intended control qubit? (Section 3.5.5, mistake

one.)

- Is any extra single-qubit gate (Z for Φ^-, X for Ψ^+, both for Ψ^-) applied after the Bell pair is formed, not before? (Section 3.6.2.)
- Does measurement happen only at the very end? (Section 3.5.5, mistake two.)
- Do the measured counts match the predicted pattern – 00/11 for Φ states, 01/10 for Ψ states – with no other outcomes appearing?

3.8.2 Qiskit Code: All Four Bell States in One Circuit

The following builds $|\Phi^+\rangle$, $|\Phi^-\rangle$, $|\Psi^+\rangle$, and $|\Psi^-\rangle$ from Section 3.6.2's recipe and measures each over many shots, so the four correlation patterns can be compared directly.

Qiskit Code – Preparing and Measuring All Four Bell States

```
# Install once: pip install qiskit qiskit-aer
from qiskit import QuantumCircuit
from qiskit_aer import AerSimulator

def bell_circuit(extra_gate=None):
    qc = QuantumCircuit(2, 2)
    qc.h(0)          # superposition on qubit 0
    qc.cx(0, 1)       # entangle -> |Phi+>
    if extra_gate == 'Z':
        qc.z(0)       # -> |Phi->
    elif extra_gate == 'X':
        qc.x(1)       # -> |Psi+>
    elif extra_gate == 'XZ':
        qc.x(1)
        qc.z(0)       # -> |Psi->
    qc.measure([0, 1], [0, 1])
    return qc

sim = AerSimulator()
for label, gate in [('Phi+', None), ('Phi-', 'Z'), ('Psi+', 'X'), ('Psi-',
'XZ')]:
    qc = bell_circuit(gate)
    counts = sim.run(qc, shots=1000).result().get_counts()
    print(label, counts)
```

The output should show 'Phi+' and 'Phi-' splitting almost exactly between '00' and '11', and 'Psi+' and 'Psi-' splitting almost exactly between '01' and '10' – with no counts at all for the other two outcomes in each case. This is the measurable signature predicted in Section 3.6.4; a computational-basis measurement cannot distinguish Φ^+ from Φ^-, or Ψ^+ from Ψ^-, since the sign difference is a relative phase, exactly as Section 3.5.5's third mistake warns.

3.8.3 Where to Run This

- **No installation:** IBM Quantum Learning – the same environment used in Units 1–2, including access routes to the Heron-class processor at Amaravati Quantum Valley (Section 3.2).
- **Local install:** qiskit.org – full documentation and tutorials.

- **This book's code:** every listing in this book is available at our companion repository – [GitHub link – to be finalized with your co-author].

3.9 Applied Use Case: Quantum Random Number Generation

The Problem

Cryptographic systems, lotteries, and scientific simulations all need genuinely unpredictable random numbers. Classical "random" number generators are actually deterministic algorithms seeded from some source of entropy – in principle predictable if that seed is known. A quantum random number generator (QRNG) instead uses measurement itself as the source of randomness: Unit 1, Section 1.6.5 established that a single measurement of $|+\rangle$ gives 0 or 1 with no way, even in principle, to predict which – not because of missing information, but because the outcome genuinely is not determined beforehand.

Implementation. The circuit needs nothing beyond this unit's own toolkit: put each of n qubits into $|+\rangle$ with a Hadamard gate, then measure all n in the computational basis. Each run produces a uniformly random n-bit string.

Qiskit Code – An 8-Bit Quantum Random Number Generator

```
from qiskit import QuantumCircuit
from qiskit_aer import AerSimulator

n = 8  # generate 8 random bits per run
qc = QuantumCircuit(n, n)
qc.h(range(n))          # every qubit -> |+>
qc.measure(range(n), range(n))

sim = AerSimulator()
result = sim.run(qc, shots=1).result()
bitstring = list(result.get_counts().keys())[0]
print('Random byte:', bitstring)
```

Outcome. Run with shots=1000 instead of shots=1, the circuit's per-qubit counts should split close to 50/50, matching the P(0)=P(1)=1/2 prediction from Unit 1's worked example – the statistical check every QRNG deployment runs before trusting the output. Real QRNG products (including several already commercially available) use exactly this circuit, or close variants, as their entropy source.

Which hardware fits? This use case needs only one gate layer and immediate measurement – the least demanding possible circuit. That makes it a good match for almost any platform in Section 3.3.8, and in practice photonic QRNGs (Section 3.3.5) are especially common precisely because a single beam-splitter measurement is enough, with no need for the longer coherence or multi-qubit control that heavier algorithms in Unit 4 will require.

3.10 Before You Move On

Important Formulas from This Unit

- Bloch sphere: $|\psi\rangle = \cos(\theta/2)|0\rangle + e^{i\phi}\sin(\theta/2)|1\rangle$, $P(0)=\cos^2(\theta/2)$, $P(1)=\sin^2(\theta/2)$
- Matrix form: $|0\rangle=[1,0]^T$, $|1\rangle=[0,1]^T$, $|\psi\rangle=[\alpha,\beta]^T$
- $|\Phi^+\rangle = (|00\rangle+|11\rangle)/\sqrt{2}$ $|\Phi^-\rangle = (|00\rangle-|11\rangle)/\sqrt{2}$
- $|\Psi^+\rangle = (|01\rangle+|10\rangle)/\sqrt{2}$ $|\Psi^-\rangle = (|01\rangle-|10\rangle)/\sqrt{2}$
- Bell circuit: $|00\rangle \rightarrow (H\otimes I) \rightarrow (|00\rangle+|10\rangle)/\sqrt{2} \rightarrow (CNOT) \rightarrow |\Phi^+\rangle$

Exercises

1. A qubit has Bloch angle $\theta = 2\pi/3$. Find $P(0)$ and $P(1)$.

2. List two advantages and two disadvantages of trapped-ion qubits compared with superconducting qubits (Section 3.3.8).

3. Starting from $|\Psi^+\rangle$, which single gate on which qubit produces $|\Psi^-\rangle$? Verify by direct calculation.

4. A student builds a Bell-state circuit but measures the first qubit immediately after the Hadamard gate. Using Section 3.5.5, explain what goes wrong and what the final two-qubit state becomes.

5. Using the checklist in Section 3.8.1, explain how you would confirm from measured counts alone whether a circuit produced $|\Phi^+\rangle$ or $|\Psi^+\rangle$ – and why the counts alone cannot tell $|\Phi^+\rangle$ from $|\Phi^-\rangle$.

6. Why does the quantum random number generator in Section 3.9 not need a long coherence time to work well?

Solutions to all exercises are provided via the book's digital companion (Section 3.8.3).

3.11 References and Further Study

Numbered to match the consolidated References at the end of this volume.

3.11.1 Textbooks

1. M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, 10th Anniversary Edition, Cambridge University Press, 2010.
2. Eleanor Rieffel and Wolfgang Polak, Quantum Computing: A Gentle Introduction, MIT Press, 2011.

3.11.2 Historical and Hardware Papers

6. B. D. Josephson, "Possible New Effects in Superconductive Tunnelling", Physics Letters, 1962.
15. The Nobel Prize in Physics 2012 (D. J. Wineland and S. Haroche), Nobel Prize outreach materials, nobelprize.org.

3.11.3 Video Lectures

22. IBM Quantum – Qiskit learning platform (qiskit.org), including Qiskit Nature modules on VQE and quantum chemistry.

23. MIT OpenCourseWare – Quantum Information Science lecture series.

Bridge to Unit 4

Where We Go Next

This unit connected the abstract qubit to real hardware and formalized circuit design well enough to build and verify every Bell state. Unit 4 uses exactly this toolkit to assemble full quantum algorithms – Deutsch–Jozsa, Grover's search, and the phase-estimation ideas behind Shor's algorithm – where the payoff of interference, entanglement, and careful circuit design finally becomes a genuine computational advantage.

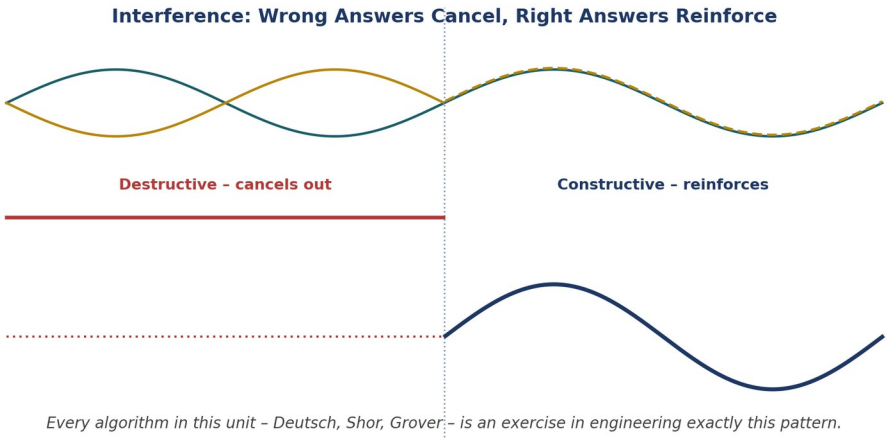
Quantum Computing

Science & Engineering Edition

Unit 4: Quantum Algorithms

"A quantum computer doesn't compute every answer at once – it arranges for the wrong ones to cancel out."

— a working principle of quantum algorithm design



Every algorithm in this unit is, at heart, an engineered wave pattern – not a lucky guess.

4.1 Pioneers Behind This Unit

Units 1–3 gave us qubits, circuits, gates and hardware. This unit asks what all of that machinery is actually for. The answer was built by three people who each found a different way to make interference do useful work.

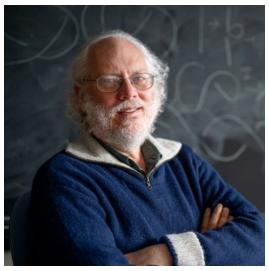


David Deutsch
b. 1953

David Deutsch – The First Quantum Algorithm (1985)

David Deutsch is a physicist at Oxford who asked a question nobody had thought to ask precisely enough: if a computer could exploit superposition, could it answer certain questions with fewer steps than any classical machine ever could – not just faster hardware, but fundamentally fewer operations? In 1985 he built exactly such a case and showed it could be answered with one query instead of two. Small as the problem was, it was the first rigorous proof that quantum interference is a genuine computational resource, not just an interesting curiosity from physics.

Deutsch is also one of the founders of quantum computation as a field of study in its own right, having laid out much of its theoretical foundations – including the modern idea of a universal quantum computer – years before any hardware existed to run one. His result, generalized soon after with Richard Jozsa into the algorithm of Section 4.7 [10], remains the cleanest possible illustration that a good idea about how amplitudes combine can outrun raw computational power. Every algorithm later in this unit inherits its logic from the one he wrote down first.



Peter Shor
b. 1959

Peter Shor – The Algorithm That Changed Cryptography (1994)

Peter Shor was a mathematician at Bell Labs in 1994 when he proved something that instantly made quantum computing a matter of national and industrial urgency: a sufficiently large, fault-tolerant quantum computer could factor integers in polynomial time [12] – a task with no known efficient classical method, and precisely the task that RSA encryption depends on being hard. Before Shor's result, quantum computing was largely a theoretical curiosity discussed by physicists. Afterward, governments, banks, and technology companies everywhere had a direct, practical reason to fund it.

What makes Shor's achievement worth remembering is not just the result but the route to it: he didn't attack factoring head-on, he found a hidden connection to a completely different problem – finding the period of a repeating pattern – that a quantum computer happened to be extraordinarily good at solving (Section 4.8.2). That kind of lateral reframing, more than raw computational force, is often exactly what

separates a hard problem from a solved one, in mathematics as much as in engineering.



Lov Kumar Grover
b. 1961

Lov Grover – Searching Faster Than Seems Possible (1996)

Lov Kumar Grover was born in 1961 in Meerut, India, and did his undergraduate studies in Electrical Engineering at IIT Delhi, graduating in 1981 – an institution that later honoured him with its Distinguished Alumnus Award. He went on to Stanford for graduate study, then joined Bell Labs in 1984, initially working on ordinary chip design rather than anything quantum. It was more than a decade later, in 1996, that he discovered a quantum computer could search an unsorted list of N items in roughly √N steps instead of N – a quadratic speedup that, unlike Shor's, needs no special mathematical structure in the problem at all [14].

That generality is what makes Grover's algorithm so enduring: it is now a standard building block inside real quantum software libraries such as Qiskit, and it underlies research into quantum-accelerated search, optimization, and cryptanalysis wherever a solution can be checked by an oracle. It is also provably optimal – no quantum algorithm can search faster than Grover's, a rare kind of certainty in computer science. His journey from an IIT Delhi engineering classroom to one of the two most famous algorithms in quantum computing is itself a reminder that foundational breakthroughs often arrive years after the training that made them possible, not in the moment of it.

Where we are going: this unit assembles the qubits, gates and circuits of Units 1–3 into complete algorithms, and asks precisely what kind of advantage each one delivers – and what it doesn't.

4.2 Case Study: The Problem That Could Break the Internet

One Multiplication, Run Backward

Almost every secure connection on the internet today – banking, email, government systems – relies on the same one-way trick: multiplying two large prime numbers is easy, but factoring the product back into those primes is, for classical computers, believed to take longer than the age of the universe once the numbers are large enough. This single asymmetry is the security foundation of RSA encryption. In 1994, Peter Shor described a quantum algorithm that undoes the asymmetry entirely – not by brute force, but by turning factoring into a question about periodicity that a quantum computer can answer efficiently.

Keep these two questions in mind as you read this unit – we will answer them directly at the

end:

Q1. How can a quantum computer possibly need fewer steps than a classical computer, when it still only gets one classical answer out of a single measurement?

Q2. What, precisely, does Shor's algorithm let a quantum computer do to a hard number-theory problem that no known classical algorithm can do efficiently – and what does it not yet mean for today's encrypted data?

4.3 What Makes a Quantum Algorithm Different

A quantum algorithm is a finite, precise procedure: prepare qubits, apply gates, let interference reshape the amplitudes, and measure to obtain a classical result. It is not a classical program translated into new syntax – it manipulates amplitudes, not definite bit values, and its power comes from how those amplitudes are made to combine.

4.3.1 A Quantum Algorithm Is a Sequence of Amplitude Transformations

Most introductory quantum algorithms are written in the circuit model of Unit 3: qubits as wires, gates read left to right, a known initial state, and measurement near the end. A quantum algorithm may also mix in ordinary classical computation – Shor's algorithm, for instance, uses classical arithmetic to pick a number and check results, a quantum subroutine to find a period, and classical arithmetic again to extract the final factors (Section 4.8). This mixed classical–quantum style is the normal shape of a real quantum algorithm, not an exception.

4.3.2 Interference: The Real Engine of Quantum Speedup

The Idea Every Algorithm in This Unit Relies On

Superposition by itself guarantees nothing: creating a large superposition and immediately measuring it still returns exactly one classical outcome, chosen at random. A useful quantum algorithm instead arranges gates so that amplitudes for wrong or unwanted answers cancel through destructive interference, while amplitudes for useful answers reinforce through constructive interference. Entanglement often participates too, but by itself is not a complete explanation – the advantage comes from the coordinated use of superposition, phase, interference, reversibility and measurement together.

4.3.3 Limits of Quantum Advantage

A quantum computer is not an infinitely parallel classical computer: measurement only ever returns a limited amount of classical information from the state, and any real speedup depends on finding exploitable structure in the problem – phase kickback in Section 4.6–4.7, periodicity in Section 4.8, or an oracle in Section 4.9. Shor's and Grover's algorithms are dramatic results for carefully defined tasks; neither implies that quantum computers make every hard problem, or every NP-complete problem, efficiently solvable. Practical advantage also depends on hardware quality, circuit depth, and error correction (Unit 5) – a mathematical speedup and an immediately deployable one are not the same claim.

Feature	Classical Algorithm	Quantum Algorithm
Information unit	Bit with value 0 or 1	Qubit described by amplitudes
Reversibility	Not usually required	Required before measurement
Main design tool	Branching and arithmetic	Superposition, phase, interference
Output	Read directly from memory	Obtained through measurement
Speedup condition	Depends on the problem's structure	Depends on exploitable quantum structure

4.4 Classical Logic, Made Reversible

4.4.1 Representing Classical Bits as Qubits

A classical bit string has a direct quantum counterpart: 0 as $|0\rangle$, 1 as $|1\rangle$, and a string such as 10 as the two-qubit basis state $|10\rangle$. If a circuit starts in a computational basis state and only ever maps basis states to basis states, it behaves exactly like a reversible classical computation, and measuring at the end recovers an ordinary classical bit string. Quantum computers are not built to replace classical computers at this job – the value of representing classical data on qubits is that classical functions can be embedded inside larger quantum algorithms, typically as oracles or arithmetic subroutines.

4.4.2 Why Reversibility Is Required

Quantum evolution before measurement must be unitary, and every unitary operation is reversible. Many familiar classical gates are not: AND maps three different inputs (00, 01, 10) to the same output, 0, so the input cannot be recovered from the output alone. To embed a classical function f on a quantum computer, the circuit instead maps $|x\rangle|y\rangle$ to $|x\rangle|y\oplus f(x)\rangle$ – x is preserved, and the function's value is written into a second register by XOR. Because the input survives, the whole operation can be undone. This exact construction is what an oracle is, and every oracle in this unit (Sections 4.6–4.9) takes this form.

4.4.3 NOT, CNOT, and Toffoli as Reversible Classical Gates

X acts as a reversible NOT on basis states – applying it twice returns the original value. CNOT flips its target exactly when its control is 1, giving a reversible controlled classical operation. The Toffoli gate (Unit 2, Section 2.6.5) is the one that matters most here: with two control qubits a, b and a target ancilla c, it maps $|a,b,c\rangle$ to $|a,b,c\oplus(a \text{ AND } b)\rangle$. Starting the ancilla at $|0\rangle$ leaves a AND b sitting in the target while a and b themselves are preserved – a fully reversible AND gate. Combining NOT, CNOT, and Toffoli is enough to embed any classical Boolean computation inside a reversible, and therefore quantum-compatible, circuit.

Classical Function	Problem for Direct Quantum Use	Reversible, Quantum-Compatible Form
NOT(x)	None – already reversible	$\|x\rangle \rightarrow \|x\oplus 1\rangle$
AND(a,b)	Loses input information	$\|a,b,c\rangle \rightarrow \|a,b,c\oplus(a \text{ AND } b)\rangle$
General f(x)	x→f(x) may not be reversible	$\|x,y\rangle \rightarrow \|x,y\oplus f(x)\rangle$
Temporary result	May leave unwanted “garbage”	Compute, use, then uncompute (Section

Classical Function	Problem for Direct Quantum Use	Reversible, Quantum-Compatible Form
		4.4.4)

4.4.4 Ancilla Qubits and Uncomputation

An ancilla is extra workspace – typically initialized to $|0\rangle$ – used to hold an intermediate result while the rest of the circuit stays reversible. Once that intermediate value has served its purpose (for example, controlling another gate), leftover entanglement between the ancilla and the main register can quietly ruin the interference the algorithm depends on. Uncomputation solves this: applying the inverse of whatever computed the intermediate value returns the ancilla cleanly to $|0\rangle$, with no residual entanglement. This compute–use–uncompute pattern is essential wherever arithmetic subroutines sit inside a larger algorithm – most notably the modular exponentiation step of Shor's algorithm, Section 4.8.3.

4.5 Quantum vs Classical Complexity Classes

Computational complexity asks how the resources needed to solve a problem grow with input size. A problem is efficient if the number of steps grows only polynomially with input size, and impractical if it grows exponentially. Comparing complexity classes across classical and quantum models of computation is how theorists make precise claims about what quantum computers can and cannot do better.

4.5.1 P, NP, and BPP

- **P:** decision problems solvable in polynomial time by a deterministic classical computer.
- **NP:** decision problems whose proposed solution can be checked in polynomial time. Whether every efficiently checkable problem is also efficiently solvable (P vs NP) remains open.
- **BPP:** problems solvable efficiently by a randomized classical computer with bounded error probability, reducible by repetition.

4.5.2 BQP: The Quantum Complexity Class

BQP – bounded-error quantum polynomial time – contains decision problems solvable efficiently on a quantum computer with bounded error probability (commonly required to be at least two-thirds correct, improvable by repetition). Factoring's decision version lies in BQP, courtesy of Shor's algorithm (Section 4.8); this does not mean BQP contains all of NP – that relationship is not fully known.

4.5.3 How They Relate, and What We Still Don't Know

The Honest Picture

- P $\subseteq$ BQP: a quantum computer can simulate any deterministic classical computation via reversible circuits (Section 4.4).
- BPP $\subseteq$ BQP: a quantum computer can reproduce classical randomized computation.
- BQP vs NP: unknown in both directions. It is not known whether BQP contains NP-complete problems, nor whether BQP sits inside NP.
- BQP $\subseteq$ PSPACE: quantum computation can be simulated in polynomial space classically – though possibly exponential time – which bounds BQP without diminishing its

practical value.

Most researchers do not expect quantum computers to solve all NP-complete problems efficiently, but this is a belief supported by evidence, not a proven theorem – precision here matters more than enthusiasm.

4.5.4 Reading Complexity Claims Correctly

“Efficient” in a complexity claim means the operation count grows polynomially with input size – it does not mean the algorithm is easy to run on today's hardware. Shor's algorithm is polynomial-time in theory, yet factoring cryptographically-sized numbers still requires large-scale fault-tolerant hardware that does not yet exist (Section 4.8.5). Likewise, that no efficient classical factoring algorithm is known does not prove none exists. Complexity classes are a disciplined language for comparing models of computation – not a substitute for tracking actual hardware capability.

4.6 Deutsch's Algorithm

4.6.1 The Problem: Constant or Balanced?

Consider a one-bit Boolean function f. There are exactly four possibilities: f is constant if f(0)=f(1), and balanced if f(0)≠f(1).

Function	f(0)	f(1)	Type
f(x) = 0	0	0	Constant
f(x) = 1	1	1	Constant
f(x) = x	0	1	Balanced
f(x) = 1⊕x	1	0	Balanced

The task is only to decide which type f is – not to learn f(0) and f(1) individually. Classically, in the worst case, a deterministic method must evaluate both f(0) and f(1) to be certain. Deutsch's quantum algorithm decides the answer with a single oracle query.

4.6.2 The Oracle and Phase Kickback

The oracle takes the reversible form of Section 4.4.2: $U_f|x,y\rangle = |x, y\oplus f(x)\rangle$. In Deutsch's algorithm, the second qubit is prepared not as an ordinary output register but as $(|0\rangle-|1\rangle)/\sqrt{2}$ – and when the oracle acts on this state, f's information does not appear as a written-out bit at all. Instead it appears as a phase factor $(-1)^{f(x)}$ attached to the input qubit. This transfer of information from a function's output into a phase on the input is called phase kickback, and it is the single idea every algorithm in this unit reuses.

4.6.3 Circuit and Calculation

Four Steps

- 1. Prepare the input qubit as |0⟩ and the output qubit as |1⟩; apply H to both.
- 2. Apply the oracle U_f once.
- 3. Apply H to the input qubit again.
- 4. Measure the input qubit: 0 means constant, 1 means balanced.

After the first Hadamards, the input qubit is (|0⟩+|1⟩)/√2 and the output qubit is (|0⟩−|1⟩)/√2. The oracle attaches (−1)^f(x) to each input branch, leaving the input part proportional to (−1)^f(0)|0⟩ + (−1)^f(1)|1⟩. If f(0) and f(1) match, the two amplitudes share a sign; if they differ, the signs are opposite. The final Hadamard converts this sign pattern into a measurable outcome: matching signs collapse to |0⟩, opposite signs to |1⟩ – exactly the constant/balanced answer, extracted with one query.

4.6.4 Why It Matters, and Its Limits

For an ideal oracle, Deutsch's algorithm is exact, not merely probable – a rare property among quantum algorithms. It does not reveal which constant or which balanced function was used, only the class; this is a promise problem, since f is guaranteed in advance to be one or the other. The task itself has no direct practical use, but its ideas – oracle, phase kickback, interference – are exactly what the rest of this unit builds on.

4.7 The Deutsch–Jozsa Algorithm

4.7.1 Generalizing to n Bits

The Deutsch–Jozsa algorithm extends the same promise to n input bits: f maps n-bit strings to one output bit, and is guaranteed to be either constant (same output for all 2^n inputs) or balanced (output 0 for exactly half the inputs, 1 for the other half). Classically, a deterministic method may need to check more than half of all 2^n inputs before being certain – up to $2^{n-1}+1$ queries in the worst case, since seeing the same output on the first half still leaves open the chance the rest are all different. The quantum algorithm settles the question with exactly one oracle query.

4.7.2 Circuit and Interference Explanation

Start with n input qubits at |0⟩ and one output qubit at |1⟩. Apply H to every qubit, creating an equal superposition over all 2^n input strings and priming the output qubit for phase kickback. Apply the oracle once. Apply H again to the n input qubits, then measure them: an all-zero result means constant; any other result means balanced.

Why All-Zero Is the Signal

The final amplitude of |00… 0⟩ is proportional to the sum of (−1)^f(x) over every input string x. If f is constant, every term in that sum has the same sign, so the terms add constructively and measurement returns all zeros. If f is balanced, exactly half the signs are positive and half negative, the sum cancels to zero, and measurement can never return all zeros. The algorithm

never reads f at each input individually – it reads a single global property of the entire truth table through interference.

4.7.3 Classical vs Quantum Query Counts

Method	Worst-Case Queries	Result Type
Classical deterministic	$2^{n-1} + 1$	Exact
Classical randomized	Often fewer, on average	Probabilistic – may need repetition
Quantum Deutsch–Jozsa	1	Exact, in the ideal model

This is an exponential separation in query count for this specific promised problem – a clean demonstration of quantum advantage, though not by itself a common practical task. Its importance is showing exactly how interference reduces query complexity.

4.7.4 Worked Two-Qubit Example

n = 2: Four Possible Inputs

With two input bits, there are four input strings: 00, 01, 10, 11. A constant f gives all four outputs 0 or all four outputs 1. A balanced f gives 0 for exactly two of the four and 1 for the other two. Placing the input register in an equal superposition and applying the oracle attaches a sign to each of the four branches according to f. For a constant f, the final Hadamards recombine all four signs constructively onto $|00\rangle$. For a balanced f, exactly two signs are positive and two negative, so the $|00\rangle$ amplitude cancels completely – measurement is guaranteed to return something other than 00. The same interference pattern scales to any n.

4.8 Shor's Factorization Algorithm

4.8.1 Why Factoring Matters

Integer factorization – finding the non-trivial factors of a composite N, such as 15 = 3×5 – is easy by trial division for small numbers, but the best known classical methods scale badly for large ones. Public-key cryptography, including RSA, is built directly on this asymmetry: multiplying two large primes is easy, recovering them from the product is believed to be classically infeasible at appropriate sizes (Section 4.2's case study). Shor's algorithm showed that a sufficiently large fault-tolerant quantum computer factors integers efficiently, changing factoring's theoretical status and motivating the entire field of post-quantum cryptography – methods designed to stay secure even against a future quantum attacker.

4.8.2 Reducing Factoring to Period-Finding

Choose a number a relatively prime to N, and consider f(x) = a^x mod N. Because modular arithmetic cycles, this function is periodic: its period r is the smallest positive integer with a^r mod N = 1. If r is found and a couple of simple conditions hold, N's factors follow directly from gcd(a^(r/2)−1, N) and gcd(a^(r/2)+1, N). If the conditions fail (r odd, or a^(r/2) ≡ −1 mod N), a new a is chosen and the process repeats. The entire quantum contribution to Shor's algorithm is finding r efficiently – everything else is classical arithmetic.

4.8.3 Modular Exponentiation and the Quantum Fourier Transform

Modular exponentiation – computing a^x mod N – must be built as a reversible circuit mapping $|x\rangle|0\rangle$ to $|x\rangle|a^x \bmod N\rangle$ (Section 4.4.2's oracle pattern). Run on a superposition of many x values, this entangles the input register with the function's periodic values, embedding the period r inside the quantum state's structure. The Quantum Fourier Transform (QFT) – the quantum analogue of the discrete Fourier transform, efficiently built from Hadamard gates and controlled phase rotations – then converts that hidden periodicity into a measurement pattern related to fractions of r. Modular exponentiation is typically the most resource-intensive part of the whole algorithm, needing many qubits and reversible arithmetic gates for large N.

4.8.4 Worked Example: Factoring 15

N = 15, a = 2

Powers of 2 mod 15: 2^1=2, 2^2=4, 2^3=8, 2^4=1 – the pattern repeats from here, so the period is r = 4.

Since r is even, compute a^(r/2) = 2^2 = 4. Then gcd(4−1, 15) = gcd(3,15) = 3, and gcd(4+1, 15) = gcd(5,15) = 5.

Result: 15 = 3 × 5, recovered entirely from the period r = 4. Real cryptographic sizes need vastly more qubits, gates, and error-corrected operations, but the period-to-factor logic is identical.

4.8.5 Complexity and Practical Limitations

Shor's algorithm runs in polynomial time in the number of bits of N on an ideal quantum computer – a major theoretical improvement over the best known (sub-exponential, not polynomial) classical methods. The practical limitation is hardware: large-scale factoring needs a fault-tolerant quantum computer with many high-quality logical qubits and reliable gates at a scale current noisy intermediate-scale devices do not provide. The balanced, accurate statement is that Shor's algorithm is a mathematically decisive result with major cryptographic implications, while breaking real-world cryptographic keys still requires future hardware with strong error correction (Unit 5).

4.9 Grover's Search Algorithm

4.9.1 The Unstructured Search Problem

Grover's algorithm searches N unstructured items – no ordering or pattern a classical method could exploit – for one marked item. A classical search may need up to O(N) oracle queries in the worst case. Grover's quantum algorithm finds the marked item in O(√N) queries: a quadratic, not exponential, speedup – but a

highly significant one for large N, and applicable to any problem where a proposed answer can be checked by an oracle even when no better structure exists.

4.9.2 The Oracle and Amplitude Amplification

The oracle marks the target by phase: O_f|x⟩ = (−1)^f(x)|x⟩, where f(x)=1 exactly for the marked item. This phase flip alone is unobservable – a global phase carries no information – but it creates a relative phase between the marked state and everything else, which the diffusion operator then exploits. Starting from an equal superposition over all N items, the oracle flips the marked item's sign, and diffusion reflects every amplitude about their average. Marked amplitude grows; unmarked amplitude shrinks. Repeating oracle-then-diffusion several times, called one Grover iteration each time, makes the marked item increasingly likely to be measured.

4.9.3 Choosing the Number of Iterations

For one marked item among N, the optimal number of iterations is approximately (π/4)√N. Too few iterations under-amplifies the marked state; too many overshoots and the amplitude rotates back down. Geometrically, the state moves within a two-dimensional subspace spanned by the marked state and the uniform superposition of the rest, and each iteration rotates it a fixed angle closer to the marked direction – stopping at the right iteration count is what makes measurement reliable.

4.9.4 Worked Example with Four Items

N = 4, Marked Item = |10⟩

Two Hadamard gates create an equal superposition over |00⟩, |01⟩, |10⟩, |11⟩, each with amplitude 1/2. The oracle flips the sign of |10⟩'s amplitude only. Diffusion then reflects all four amplitudes about their average – for this small case, a single Grover iteration is enough to amplify |10⟩ to (near-)certainty, so measurement returns the marked item essentially every time.

Stage	Marked Amplitude	Unmarked Amplitudes	Interpretation
Initial superposition	Equal to others	Equal	No item favored
After oracle	Phase flipped	Unchanged	Solution marked by relative phase
After diffusion	Increased	Reduced	Solution becomes more likely
After measurement	Observed with high probability	Less likely	Classical answer obtained

4.9.5 Advantages and Limitations

Grover's algorithm is powerful precisely because it is general – it applies whenever a solution can be verified by an oracle, even with no exploitable structure for a faster classical search, making it useful for optimization subroutines and brute-force-style problems at scale. Its limitation is equally clear: the speedup is quadratic, not exponential, so √N can still be very large for large N, and the cost of building and applying the oracle itself is

part of the real total cost – not just the number of iterations. Grover's algorithm is best understood as a powerful general search principle, not a universal solution to hard search problems.

4.10 Comparing the Four Algorithms

Each algorithm in this unit demonstrates a different reason for quantum advantage. Deutsch's and Deutsch–Jozsa use phase kickback and interference for a query advantage on a promised problem. Shor's algorithm uses the Quantum Fourier Transform to solve period-finding efficiently, turning that into polynomial-time factoring. Grover's algorithm uses amplitude amplification for a quadratic search speedup. None of them use the same mechanism – which is exactly the point: quantum computing is not a single trick, but a family of ways to make interference do useful work.

Algorithm	Problem Type	Main Quantum Idea	Type of Advantage
Deutsch's algorithm	1-bit constant vs balanced	Phase kickback, interference	Fewer oracle queries
Deutsch–Jozsa	n-bit constant vs balanced	Global interference over truth table	Exponential query separation (promised problem)
Shor's algorithm	Integer factorization	QFT and period estimation	Polynomial-time quantum algorithm
Grover's algorithm	Unstructured search	Amplitude amplification	Quadratic query speedup

4.11 Case Study Revisited: Breaking, and Rebuilding, Trust

Closing the Loop

Q1. How does a quantum computer need fewer steps if measurement gives only one answer? Because the advantage is built before measurement, in how amplitudes combine. Deutsch–Jozsa's single query works because interference makes the all-zero outcome appear only for constant functions (Section 4.7.2); Shor's algorithm works because the QFT converts a hidden period into a measurable pattern (Section 4.8.3); Grover's algorithm works because repeated amplitude amplification concentrates probability onto the marked item before that one measurement happens (Section 4.9.2). The single measurement is the end of the computation, not the mechanism of its speedup.

Q2. What does Shor's algorithm actually mean for today's encrypted data? It means factoring is, in principle, in BQP (Section 4.5.2) – solvable in polynomial time on a large, fault-tolerant quantum computer. It does not mean today's noisy, small-scale quantum devices can break real RSA keys (Section 4.8.5): that requires far more high-quality logical qubits than currently exist. The practical response already underway is post-quantum cryptography – classical algorithms believed secure even against a future quantum attacker.

4.12 Realization Lab: The Deutsch–Jozsa Algorithm in Qiskit

4.12.1 Checklist: Is the Circuit Actually Testing Interference?

Before Trusting the Output, Check:

- Is the output qubit initialized to $|1\rangle$, then Hadamard'd, before the oracle runs? (Section 4.6.2 – this is what makes phase kickback happen at all.)
- Is the oracle applied exactly once, matching the one-query claim of Section 4.7.1?
- Are the input qubits measured only after the second round of Hadamard gates, not before?
- Does an all-zero result appear only for constant oracles, and a non-zero result for every balanced oracle you test?

4.12.2 Qiskit Code: Constant vs Balanced, Told Apart in One Query

The following builds the two-qubit Deutsch–Jozsa circuit from Section 4.7.4 for one constant oracle and one balanced oracle, confirming that the input register measures all-zero only for the constant case.

Qiskit Code – Deutsch–Jozsa, Constant vs Balanced

```
# Install once: pip install qiskit qiskit-aer
from qiskit import QuantumCircuit
from qiskit_aer import AerSimulator

def dj_circuit(n, oracle):
    qc = QuantumCircuit(n + 1, n)
    qc.x(n)              # output qubit -> |1>
    qc.h(range(n + 1))   # superposition on all qubits
    oracle(qc)           # apply the promised oracle once
    qc.h(range(n))       # interference step on input qubits
    qc.measure(range(n), range(n))
    return qc

def constant_oracle(qc):
    pass  # f(x) = 0 for all x: oracle does nothing

def balanced_oracle(qc):
    qc.cx(0, qc.num_qubits - 1)  # f(x) = x0: kicks back a sign when x0=1

sim = AerSimulator()
for label, oracle in [('constant', constant_oracle), ('balanced',
balanced_oracle)]:
    qc = dj_circuit(2, oracle)
    counts = sim.run(qc, shots=1000).result().get_counts()
    print(label, counts)  # constant -> all '00'; balanced -> never '00'
```

The constant oracle should return '00' with essentially all 1000 shots, exactly as Section 4.7.2 predicts – total constructive interference on the all-zero outcome. The balanced oracle should never return '00' at all, since its signs cancel that amplitude completely. Both results come from a single call to each oracle function inside the circuit – the one-query claim made concrete.

4.12.3 Where to Run This

- **No installation:** IBM Quantum Learning – the same environment used in Units 1–3.
- **Local install:** qiskit.org – full documentation and tutorials.
- **This book's code:** every listing in this book is available at our companion repository – [GitHub link – to be finalized with your co-author].

4.13 Applied Use Case: Grover Search for a Supply-Chain Record Lookup

The Problem

A supply-chain database holds records with no useful sort order for the query at hand – for instance, "which shipment record matches this exact damaged-batch code?" A classical unstructured lookup over N records costs O(N) checks in the worst case. This is exactly Grover's problem (Section 4.9.1): no structure to exploit, only the ability to check whether a given record is the one we want.

Implementation. For a small illustrative table of 4 records (indexed 00, 01, 10, 11), encode the target index as the marked state in Grover's oracle from Section 4.9.2, and run one Grover iteration – the optimal count for N = 4 by Section 4.9.3's formula.

Qiskit Code – Grover Lookup Over 4 Records

```
from qiskit import QuantumCircuit
from qiskit_aer import AerSimulator

# Oracle: flip the sign of the marked record |10>
def oracle(qc):
    qc.cz(0, 1)   # marks |11> by default; recombine with X gates to mark |
10>

def diffuser(qc):
    qc.h([0, 1]); qc.x([0, 1])
    qc.cz(0, 1)
    qc.x([0, 1]); qc.h([0, 1])

qc = QuantumCircuit(2, 2)
qc.h([0, 1])              # equal superposition over 4 records
qc.x(1); oracle(qc); qc.x(1)   # mark |10> specifically
diffuser(qc)              # one Grover iteration
qc.measure([0, 1], [0, 1])

sim = AerSimulator()
counts = sim.run(qc, shots=1000).result().get_counts()
```

```
print(counts)  # expect '10' with high probability
```

Outcome. With N = 4 and one Grover iteration, the measured counts should show '10' with high probability – matching the near-certain retrieval predicted in Section 4.9.4's worked example. At realistic database sizes, the same circuit pattern scales to O(√N) oracle calls instead of O(N) classical checks, though Section 4.9.5's caveat still applies: the oracle itself must be built to recognize the target record, and that construction cost is part of the real total.

Why this sector. Supply-chain resilience is one of the sectors TCS has named for its quantum use-case work at the Amaravati Quantum Valley (Unit 6 covers this in full). Unstructured lookup and matching problems – damaged-batch tracing, anomaly matching across shipment records – are natural, if early-stage, candidates for exactly the algorithm built in this section.

4.14 Before You Move On

Important Ideas from This Unit

- Oracle form: $U_f|x,y\rangle = |x, y \oplus f(x)\rangle$ – the reversible shape behind every algorithm here.
- Phase kickback: function information becomes a phase $(-1)^{f(x)}$ on the input register.
- Deutsch–Jozsa: all-zero measurement ⇒ constant; any other result ⇒ balanced – exact, one query.
- Shor's algorithm: factoring → period-finding → QFT → classical gcd extraction.
- Grover's algorithm: optimal iterations ≈ $(\pi/4)\sqrt{N}$; oracle marks by phase, diffusion amplifies.
- $P \subseteq BQP$, $BPP \subseteq BQP$; BQP vs NP is unresolved.

Exercises

1. For the one-bit function f(x) = 1 (constant), trace Deutsch's algorithm step by step and confirm the input qubit measures 0.

2. A 3-bit Deutsch–Jozsa oracle is promised constant or balanced. What is the worst-case classical deterministic query count? What is the quantum query count?

3. For N = 21 and a = 2, find the period r of $2^x \bmod 21$, then use Section 4.8.4's method to extract a factor of 21.

4. For an unstructured search over N = 16 items with one marked item, compute the optimal number of Grover iterations using Section 4.9.3's formula.

5. Using Section 4.5.3, explain why "$P \subseteq BQP$" is a proven fact but "BQP solves all of NP" is not a claim anyone can currently make.

6. A colleague claims a small 20-qubit NISQ device can already break 2048-bit RSA using Shor's algorithm. Using Section 4.8.5, explain what is wrong with this claim.

Solutions to all exercises are provided via the book's digital companion (Section 4.12.3).

4.15 References and Further Study

Numbered to match the consolidated References at the end of this volume.

4.15.1 Textbooks

1. M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, 10th Anniversary Edition, Cambridge University Press, 2010.
2. Eleanor Rieffel and Wolfgang Polak, Quantum Computing: A Gentle Introduction, MIT Press, 2011.

4.15.2 Original Papers

10. D. Deutsch and R. Jozsa, "Rapid Solution of Problems by Quantum Computation", Proceedings of the Royal Society A, 1992.
12. P. W. Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring", Proceedings of FOCS, 1994.
14. L. K. Grover, "A Fast Quantum Mechanical Algorithm for Database Search", Proceedings of STOC, 1996.

4.15.3 Video Lectures

22. IBM Quantum – Qiskit learning platform (qiskit.org), including Qiskit Nature modules on VQE and quantum chemistry.
23. MIT OpenCourseWare – Quantum Information Science lecture series.

Bridge to Unit 5

Where We Go Next

This unit showed what quantum algorithms can do when hardware is assumed to be perfect. Unit 5 removes that assumption: real qubits decohere, real gates have limited fidelity (Unit 3, Section 3.3.8), and every algorithm in this unit needs protection from noise before it can run at the scale its theory promises. Unit 5 covers quantum error correction, fault tolerance, and the roadmap from today's noisy devices to the machines these algorithms actually require.

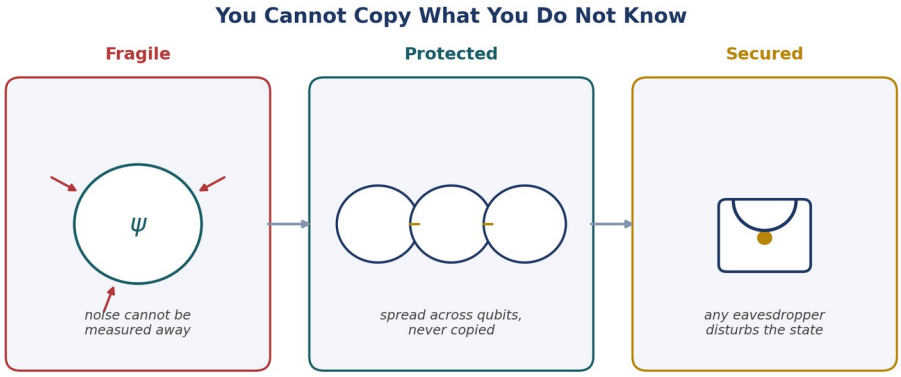
Quantum Computing

Science & Engineering Edition

Unit 5: Noise, Error Correction, Information Theory and Cryptography

"You cannot copy what you do not know – and that turns out to be both the hardest problem in this unit, and its most useful trick."

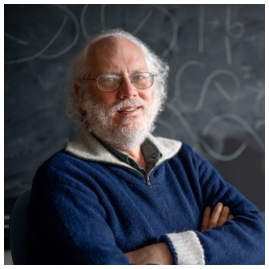
— a working principle of quantum information science



The same restriction that makes quantum information fragile is what makes it possible to protect and secure.

5.1 Pioneers Behind This Unit

Units 1–4 assumed, mostly, that qubits and gates behave perfectly. This unit removes that assumption and asks what happens when the real world intrudes – and shows that the very fragility of quantum information can be turned into a security feature rather than only a liability.

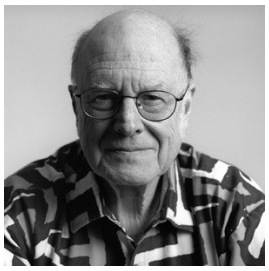


Peter Shor
b. 1959

Peter Shor – Proving Quantum Errors Are Fixable (1995)

You met Peter Shor in Unit 4 for his factoring algorithm. A year later, in 1995, he answered a question that many physicists suspected had no good answer: could a quantum state ever be protected from noise at all, given that it cannot be copied and cannot be measured without being disturbed? Shor's nine-qubit code (Section 5.5.4) was the first proof that the answer is yes [13] – a single logical qubit really can be encoded across several physical qubits and recovered after an error, without ever looking directly at the fragile state itself.

It is worth pausing on how much rode on this one result: without it, quantum computing might have remained a beautiful theoretical idea with no realistic path to a working machine, since real hardware is never perfectly isolated from noise. Having already reshaped cryptography in Unit 4, Shor reshaped the feasibility of quantum computing itself a year later – a rare double contribution from one person to two of the field's most consequential open questions.



Charles Bennett
b. 1943

Charles Bennett – Turning Quantum Weirdness Into Protocols (1984, 1993)

Charles Bennett, working at IBM, co-invented two of this unit's three major protocols. With Gilles Brassard, he proposed BB84 in 1984 (Section 5.8) [9] – the first quantum key distribution scheme, turning the disturbance caused by measurement from an annoyance into a burglar alarm. Nine years later, with several co-authors, he worked out quantum teleportation (Section 5.9) [11]: a way to transfer an unknown quantum state using shared entanglement and two classical bits, without ever violating no-cloning.

Both ideas share the same underlying move: instead of fighting the strange rules of quantum mechanics – no cloning, no measuring without disturbing – Bennett found ways to make those very restrictions do useful work. That habit of asking "what can this limitation make possible, rather than merely prevent?" runs through this entire unit, from error correction to cryptography to teleportation.



Alexei Kitaev
b. 1963

Alexei Kitaev – Building Errors Out of the Architecture (1997 onward)

Alexei Kitaev approached fault tolerance from a different angle: rather than only correcting errors after they happen, could a code's very geometry make certain errors astronomically unlikely to matter? His work on topological quantum computation and the surface code [16] – briefly introduced in Section 5.5.6 – arranges physical qubits in a two-dimensional grid with only local, nearest-neighbour interactions, making the resulting code far more practical to build on real hardware than earlier proposals.

Surface codes are, as of this writing, the leading candidate for how large-scale fault-tolerant quantum computers will actually be built – including future generations of hardware at facilities like Amaravati Quantum Valley (Unit 6). Kitaev's insight that geometry itself can be a form of error protection, not just an afterthought bolted onto a code, is a good example of how reframing a problem can matter as much as solving it directly.

Where we are going: this unit asks what happens when the perfect qubits of Units 1–4 meet real noise – and shows that the same restrictions that make quantum information fragile (no-cloning, measurement disturbance) also make it possible to detect errors, secure communication, and transfer unknown states across a room or across the world.

5.2 Case Study: You Cannot Copy What You Do Not Know

One Theorem, Three Consequences

The no-cloning theorem says there is no way to duplicate an arbitrary unknown quantum state. At first glance this looks like nothing but bad news: classical computers protect data by making backup copies, and that option is simply gone for qubits. Yet this single restriction turns out to shape almost everything in this unit. It is the reason quantum error correction has to be cleverer than classical error correction. It is the reason a quantum channel can, in principle, detect an eavesdropper who cannot exist classically. And it is the reason quantum teleportation – moving a state without ever holding a spare copy of it – does not break any law of physics.

Keep these two questions in mind as you read this unit – we will answer them directly at the end:

Q1. If you cannot copy a qubit to back it up, how do you ever protect quantum information from noise at all?

Q2. How can the very fact that measurement disturbs a quantum state be turned into a security guarantee, rather than just a nuisance?

5.3 Noise and the Limits of Fragile Information

A classical bit is a robust physical switch, cleanly on or off. A qubit stores information in a delicate combination of amplitudes and phases, and almost any unwanted interaction with its environment corrupts that information. This section describes that corruption mathematically, as preparation for detecting and correcting it without ever directly looking at the fragile state.

5.3.1 Sources of Noise in Real Quantum Hardware

No physical qubit platform from Unit 3 is perfectly isolated. Superconducting qubits are disturbed by thermal photons and material defects; trapped ions by stray electric fields and laser instability; photonic qubits by loss in optical fibres and imperfect detectors. Control pulses themselves are also imperfect – a rotation meant to be exactly 90° might land at 90.2° due to calibration drift. These small errors accumulate as a circuit grows deeper, which is exactly why error correction becomes essential once a quantum computer scales beyond a handful of qubits.

5.3.2 Decoherence: T1 and T2

Decoherence is the gradual loss of quantum information to the environment, described by two timescales. T1, the relaxation time, is how quickly an excited state such as $|1\rangle$ decays toward $|0\rangle$ through energy loss. T2, the dephasing time, is how quickly the relative phase between $|0\rangle$ and $|1\rangle$ randomizes, destroying superposition without necessarily changing energy – in practice T2 is never longer than 2×T1. Both set a hard limit on how long a computation can run before accumulated noise overwhelms the intended result, and both depend heavily on the hardware platform chosen (Unit 3, Section 3.3.8).

5.3.3 Modelling Errors: Bit-Flip, Phase-Flip, Depolarizing

Four Cases Are Enough

- **Bit-flip error:** applies X, swapping $|0\rangle$ and $|1\rangle$ – the direct quantum analogue of a classical bit flip.
- **Phase-flip error:** applies Z, leaving $|0\rangle$ unchanged but multiplying $|1\rangle$ by -1 – invisible to a computational-basis measurement, but with no classical analogue at all, since it only affects superposition.
- **Depolarizing error:** replaces the state with a random mixture that may include an X, Y, Z error, or none at all.

Because any single-qubit error can be written as a combination of I, X, Y, and Z, a quantum code only ever needs to detect and correct these four basic cases to handle a fully general error – a simplification that makes the rest of this unit's error-correcting codes tractable.

5.3.4 Why Classical Error Correction Doesn't Directly Apply

Classical error correction typically relies on redundancy through direct copying – store three copies of a bit, take a majority vote. Two facts rule this out for qubits. First, an arbitrary quantum state cannot be copied (Section 5.3.5). Second, directly measuring a qubit to check its value collapses superposition, destroying exactly the information a quantum algorithm depends on. Quantum error correction therefore needs a way to

detect that an error occurred, and identify which one, without ever learning or disturbing the encoded information itself – the central design challenge Section 5.5 solves.

5.3.5 The No-Cloning Theorem, Precisely

Why Copying Is Impossible, Not Just Hard

The no-cloning theorem states that no physical operation can take an arbitrary unknown state $|\psi\rangle$ and produce two independent copies of it. This follows directly from linearity: any operation that successfully copies two different basis states cannot, by the linearity of quantum mechanics, also correctly copy their superposition. This is not an obstacle to be engineered around – it is a hard mathematical fact. Quantum error-correcting codes therefore never try to duplicate a logical qubit; instead they spread, or entangle, a single logical qubit across several physical qubits, so that no individual physical qubit ever holds a complete, clonable copy of the protected state.

5.4 Graph States and Their Use in Codes

Graph states offer a compact, visual way to describe a useful family of highly entangled multi-qubit states – useful in their own right as a resource for an alternative model of computing, and as the underlying structure for several important families of quantum error-correcting codes.

5.4.1 Building a Graph State

A graph state is built from a mathematical graph G: a set of vertices and a set of edges connecting pairs of them. Each vertex is associated with one qubit, every qubit is first prepared in $|+\rangle$, and every edge is then associated with a CZ gate applied between the two qubits it connects. Because CZ gates commute with each other regardless of order, the final entangled state depends only on the graph's structure – which qubits are connected to which – not on the sequence in which the gates happened to be applied.

5.4.2 Stabilizer Description

Rather than writing out a graph state's full amplitude description, it is far more convenient to describe it through stabilizer generators. For every vertex v, there is a stabilizer operator K_v – the Pauli X on qubit v, multiplied by a Pauli Z on every neighbour of v:

$$K_v\,|G\rangle = |G\rangle, \quad \text{where } K_v = X_v \prod_{u\in N(v)} Z_u$$

The graph state $|G\rangle$ is the unique state, up to overall phase, left unchanged by every one of these operators. This stabilizer description is central to quantum error correction (Section 5.5.5): instead of tracking a full quantum state, it is enough to track how an error changes the outcomes of stabilizer measurements, since a valid encoded state is exactly one for which every stabilizer measurement returns the same fixed outcome.

5.4.3 Cluster States and Measurement-Based Computation

A graph state built on a two-dimensional grid is called a cluster state – the resource for measurement-based ("one-way") quantum computation. In this model, a large cluster state is prepared first, and the entire computation then proceeds through a sequence of single-qubit measurements, with each measurement's

basis chosen depending on earlier outcomes. No further entangling gates are needed once the cluster state exists, which makes this model attractive for certain photonic hardware platforms (Unit 3, Section 3.3.5).

5.4.4 Graph States as a Foundation for Codes

Many quantum error-correcting codes, including advanced stabilizer- and graph-based codes, can be understood through graph-state ideas: some qubits carry data while others act as ancillas, used to measure stabilizers and reveal error syndromes. This lens makes it easy to see why some codes tolerate more noise than others – codes whose graphs have only short-range, local connections are far easier to implement reliably, since every physical qubit only needs to interact with its nearby neighbours.

5.5 Quantum Error-Correcting Codes

This section builds the core machinery: encoding one logical qubit into several physical qubits, detecting errors through stabilizer measurements without disturbing the encoded information, and correcting the error once identified.

5.5.1 The Core Idea: Redundancy Without Duplication

A quantum error-correcting code encodes one or more logical qubits into a larger number of physical qubits so that a limited set of errors can be detected and reversed. Information about the logical qubit is spread across several physical qubits through entanglement – not copied – so an error affecting only a few physical qubits still leaves enough information intact to recover the original logical state. Detection uses stabilizer measurements that reveal the error syndrome – the type and location of an error – while leaving the logical information completely undisturbed.

5.5.2 The Three-Qubit Bit-Flip Code

The simplest quantum code, and the natural starting point. A logical qubit $\alpha|0\rangle+\beta|1\rangle$ is encoded, using two CNOT gates, into $\alpha|000\rangle+\beta|111\rangle$. If at most one of the three physical qubits suffers a bit-flip, the original logical state can still be traced back uniquely. Two ancilla qubits measure the parities of qubit pairs (1,2) and (2,3) without measuring the data qubits themselves; the resulting syndrome identifies which single qubit, if any, was flipped.

Syndrome (s1, s2)	Correction Applied
(0, 0)	None – no error
(1, 0)	Apply X to qubit 1
(1, 1)	Apply X to qubit 2
(0, 1)	Apply X to qubit 3

5.5.3 The Three-Qubit Phase-Flip Code

This code protects against Z errors using the same structure in a rotated basis. Each qubit is first transformed by a Hadamard gate, so the logical states become $|0_l\rangle = |{+}{+}{+}\rangle$ and $|1_l\rangle = |{-}{-}{-}\rangle$ instead of $|000\rangle$ and $|111\rangle$. A phase-flip error in the original basis becomes an effective bit-flip error in this rotated basis, so the same parity-check and correction procedure from Section 5.5.2 can simply be reused after the basis change. This

equivalence – connecting bit-flip and phase-flip errors through a change of basis – is a pattern that recurs throughout quantum error correction.

5.5.4 The Shor Nine-Qubit Code

Neither code alone protects against a general single-qubit error that combines a bit flip and a phase flip together. Shor's 1995 code (Section 5.1) was the first shown to correct an arbitrary single-qubit error, built by concatenation: the logical qubit is first encoded with the phase-flip code into three qubits, and each of those three qubits is separately encoded again with the bit-flip code into three further qubits – nine physical qubits in total. Any single Pauli-type error on any one of the nine qubits can be detected and corrected by combining syndrome information from both levels of encoding.

5.5.5 Stabilizers and Syndrome Measurement, Generalized

The stabilizer formalism from Section 5.4.2 is the standard language for describing general quantum codes: a stabilizer code is defined by a set of commuting Pauli operators – stabilizer generators – that all equal +1 on every valid codeword. A physical error typically anti-commutes with one or more generators, flipping their measured value to −1. Measuring every generator with ancilla qubits, so the data qubits themselves are never directly measured, produces the error syndrome. Because the syndrome depends only on the error, not on the encoded logical state, this measurement reveals what went wrong without revealing – or disturbing – the protected information.

5.5.6 Advanced Codes: A Brief Note

Advanced codes protect information more reliably than the introductory three-qubit codes, at the cost of needing more physical qubits per logical qubit. Surface codes (Section 5.1, Kitaev) are the best-known example: they arrange physical qubits on a two-dimensional grid and use only local stabilizer measurements to detect error patterns, which is exactly what makes them practical to build on real hardware. For this course, the essential point is the trade-off: stronger protection against noise, at higher hardware and resource cost.

5.6 Fault-Tolerant Computation

Encoding information in a quantum error-correcting code is not, by itself, enough for reliable computation. The gates that manipulate encoded qubits – and even the process of measuring the error syndrome – are themselves imperfect operations that can introduce new errors. Fault tolerance studies how to compute on encoded information while stopping errors from spreading uncontrollably.

5.6.1 Why Encoding Alone Isn't Enough

A naive implementation of error correction can make things worse: if one faulty gate during syndrome extraction causes errors on several data qubits at once, a code designed to correct only single-qubit errors will fail. Fault-tolerant design ensures that any single failure in the underlying hardware – a gate, a measurement, or a qubit itself – can affect at most one qubit within any single code block, so the correction procedure keeps working even when the correction circuitry itself is imperfect.

5.6.2 Transversal Gates

A transversal gate applies a single-qubit operation independently and in parallel to each physical qubit of a code block, with no interaction between qubits inside that block. Because no two physical qubits in the same

block interact during a transversal gate, an error on one cannot spread to another within it – making transversal gates naturally fault tolerant. Unfortunately, a well-known result shows no single code can implement a fully universal gate set transversally alone, which is exactly why fault-tolerant quantum computation needs careful gate design and additional resources beyond transversality.

5.6.3 The Threshold Theorem

The Result That Makes Large-Scale Quantum Computing Plausible

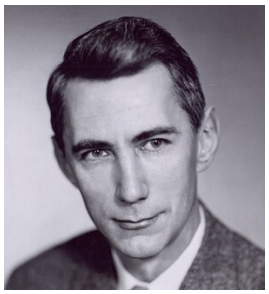
The threshold theorem states that if the physical error rate of every component – every gate, measurement, and idle qubit – is below a fixed threshold value, then arbitrarily long and reliable quantum computation is possible using fault-tolerant error correction, at the cost of more physical qubits and operations. This works through concatenation: encode a qubit, then encode each of those qubits again, and so on through several levels. Below threshold, the effective logical error rate falls rapidly with each additional level – roughly as the square of the previous level's error rate.

5.6.4 Practical Challenges and Resource Overhead

Fault tolerance in theory is well established; fault tolerance in practice remains difficult, because real hardware suffers decoherence, control errors, imperfect measurement, and environmental disturbance all at once. The central practical challenge is resource overhead: one reliable logical qubit can require anywhere from a few dozen to many thousands of physical qubits, depending on the code, the hardware's physical error rate, and how far that rate sits below threshold. Building many physical qubits is not enough on its own – they must also be good enough that the overhead needed to form reliable logical qubits stays manageable, which is precisely the engineering challenge facilities like Amaravati Quantum Valley (Unit 6) are working to solve.

5.7 Classical vs Quantum Information Theory

Classical information theory, founded by Claude Shannon, gives a mathematical language for quantifying information, redundancy, and communication limits. Quantum information theory extends many of these ideas to qubits, while introducing genuinely new phenomena with no classical counterpart.



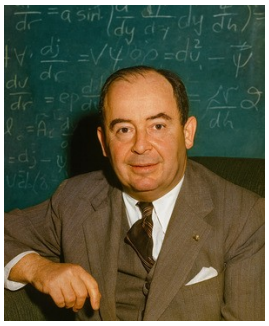
Claude Shannon
1916–2001

Claude Shannon – The Founder of Information Theory (1948)

Claude Shannon's 1948 paper "A Mathematical Theory of Communication," written at Bell Labs, single-handedly created the field of information theory [17]. He gave the word "information" a precise mathematical meaning for the first time – the entropy formula in Section 5.7.1 is his – and showed that any noisy communication channel has a hard ceiling on how much can be sent through it reliably, no matter how clever the encoding scheme.

Almost every classical technology you use daily – compressed video, Wi-Fi, error-correcting hard drives – traces back to Shannon's single

paper. Quantum information theory does not replace his framework; it extends it, asking the same questions about uncertainty, capacity, and redundancy for a world where information can also be encoded in superposition and entanglement. Section 5.7.2's Von Neumann entropy is the direct quantum generalization of the entropy Shannon defined here.



John von Neumann
1903–1957

John von Neumann – The Mathematics Behind the Density Matrix (1932)

You have already met John von Neumann once in this book, without knowing it: the "von Neumann architecture" from Unit 1 – memory, processing unit, and input/output – is named after him too. Long before that, in his 1932 book Mathematical Foundations of Quantum Mechanics [18], he gave quantum theory its first rigorous mathematical grounding, including the density matrix ρ that Section 5.7.2's entropy formula is built on.

Von Neumann entropy was his direct generalization of Shannon's idea to quantum states – and, by a famous (if hard-to-verify) story, it was von Neumann himself who suggested Shannon call his own classical formula "entropy" in the first place, since nobody would know what it really meant. Whether or not the anecdote is precisely true, it captures something real: one mathematician's fingerprints are on both the architecture of every classical computer and the entropy formula of quantum information theory, which says a great deal about how foundational his contributions were.

5.7.1 Shannon Entropy and Channel Capacity

The Shannon entropy H(X) of a random variable X measures its average uncertainty, maximized when all outcomes are equally likely. The channel capacity C of a noisy channel is the maximum rate, in bits per use, at which information can be sent reliably despite noise. Classical bits can always be copied perfectly and read without disturbing them – the property that underlies simple, effective classical strategies like repetition codes, and precisely the property qubits lack.

$$H(X) = -\Sigma_i p_i \log_2 p_i$$

5.7.2 Von Neumann Entropy

The quantum counterpart of Shannon entropy is Von Neumann entropy, defined using the density matrix ρ, which represents both pure states and the mixed states produced by noise or partial knowledge:

$$S(\rho) = -Tr(\rho \log_2 \rho)$$

For a pure quantum state, Von Neumann entropy is exactly zero; for a mixed state, it is positive. This makes it a direct way to measure how mixed, or noisy, a quantum state has become – the same decoherence introduced in Section 5.3.2, now expressed information-theoretically.

5.7.3 No-Cloning's Information-Theoretic Role

The no-cloning theorem (Section 5.3.5) is not just a curiosity here – it has a direct information-theoretic consequence. It guarantees that quantum information cannot be intercepted, copied, and forwarded without disturbing the original: exactly the property Section 5.8's quantum cryptography exploits to detect eavesdropping, and exactly the reason quantum error correction (Section 5.5) must spread information through entanglement rather than duplication.

5.7.4 Comparative Summary

Aspect	Classical Information	Quantum Information
Basic unit	Bit (0 or 1)	Qubit (superposition of $\|0\rangle$, $\|1\rangle$)
Copying	Perfect copying always possible	Forbidden by no-cloning
Measurement	Reading does not disturb the value	Measurement generally disturbs the state
Entropy measure	Shannon entropy H(X)	Von Neumann entropy S(ρ)
Correlations	Classical correlation only	Entanglement – strictly stronger
Redundancy strategy	Direct duplication of bits	Entanglement across physical qubits

5.8 Quantum Cryptography: The BB84 Protocol

Quantum cryptography uses the physical properties of quantum states, rather than unproven assumptions about computational hardness, to guarantee security. Its most developed application is quantum key distribution (QKD), where two parties – conventionally Alice and Bob – establish a shared secret key while being able to detect any eavesdropper.

5.8.1 Security Beyond Computational Hardness

Classical schemes like RSA rely on certain problems – such as factoring – being computationally hard. Unit 4's Shor's algorithm showed that a large fault-tolerant quantum computer undermines exactly this assumption. QKD offers a fundamentally different guarantee, grounded in physics rather than computational difficulty: any attempt to intercept and measure the transmitted quantum states necessarily disturbs them in a detectable way.

5.8.2 The BB84 Protocol

Proposed by Bennett and Brassard in 1984 (Section 5.1), BB84 was the first QKD scheme. Alice encodes a random bit string using randomly chosen bases – the standard basis $\{|0\rangle, |1\rangle\}$ or the diagonal basis $\{|+\rangle, |-\rangle\}$ – and sends the resulting qubits to Bob. Bob measures each one in a randomly chosen basis of his own. Afterward, Alice and Bob publicly announce which basis was used for each qubit – never the bit values

themselves – and keep only the results where their bases happened to match. These matching results form the raw key, later refined by further classical processing.

5.8.3 How Eavesdropping Gets Caught

BB84's security rests directly on the fact that measuring a qubit in the wrong basis disturbs it. If an eavesdropper, Eve, intercepts a qubit and measures it in a basis that doesn't match Alice's, her measurement collapses the state; without knowing Alice's original basis, Eve's resent qubit introduces errors into roughly a quarter of the bits where Alice and Bob's bases matched. Alice and Bob estimate this error rate by publicly comparing a small random sample of their raw key – an error rate significantly above normal channel noise reveals an eavesdropper, and the key exchange is aborted.

Alice's Basis vs Bob's Basis	Result Kept?	Notes
Same, no eavesdropping	Yes	Bob's result matches Alice's bit with high probability
Same, with eavesdropping	Yes, but may be flawed	Eve's intervention raises the observed error rate
Different bases	No	Discarded during basis reconciliation

5.8.4 Entanglement-Based QKD, Briefly

An alternative approach shares pairs of entangled particles and measures them in selected bases; when results show the expected quantum correlations (Unit 2, Section 2.5), the users form a shared key from part of the data. If an eavesdropper intercepts or measures the particles, the quantum correlations between the two users change, and comparing a portion of the results reveals the disturbance. For this course, BB84 remains the main protocol to know in depth; entanglement-based QKD is worth recognizing as an important extension built on exactly the same principle.

5.8.5 Practical Considerations and Limitations

Real QKD systems face challenges the idealized protocol glosses over: photon loss in optical fibre limits the maximum secure distance without trusted relay nodes, detector imperfections can open security loopholes that require careful device characterization to close, and key generation rates are typically far lower than conventional classical communication. Despite this, QKD has been demonstrated over both fibre-based and satellite-based links, establishing its practical viability for secure quantum communication.

5.9 Quantum Teleportation

Quantum teleportation transfers an unknown quantum state from one location to another using a pre-shared entangled pair plus ordinary classical communication – without physically transporting the qubit, and without violating no-cloning.

5.9.1 Goal and Required Resources

Alice holds an unknown state $|\psi\rangle = \alpha|0\rangle+\beta|1\rangle$ and wants to transfer it to a distant party, Bob, without either of them ever learning α or β. The protocol needs three qubits total – the qubit carrying $|\psi\rangle$, plus one qubit each from a pre-shared entangled Bell pair – and a classical communication channel between Alice and Bob.

5.9.2 Step-by-Step Protocol

Alice and Bob first share one qubit each of a Bell pair, prepared in advance (Unit 2, Section 2.5.3). Alice then performs a joint Bell-basis measurement on her half of the pair together with $|\psi\rangle$ – an outcome with four possibilities – and sends the resulting two classical bits to Bob. Depending on which of the four outcomes Alice reports, Bob applies one of four correction operations, built from X and Z, to his half of the pair. After this correction, Bob's qubit is left in exactly the state $|\psi\rangle$ Alice originally held, even though neither party ever learned α or β.

Alice's Outcome	Bob's Correction	Resulting State
00	Identity (none)	$\|\psi\rangle$
01	Apply X	$\|\psi\rangle$
10	Apply Z	$\|\psi\rangle$
11	Apply X then Z	$\|\psi\rangle$

5.9.3 Why Classical Communication Is Essential

The two classical bits Alice sends are not optional. Without them, Bob's qubit sits in an equal mixture of all four possible corrected states, from which $|\psi\rangle$ cannot be recovered. Because this classical message can travel no faster than light, teleportation cannot be used to send information instantaneously – a point Section 5.9.4 makes precise.

5.9.4 Consistent with No-Cloning and Relativity

Teleportation does not violate no-cloning: Alice's original qubit is left in a random, disturbed state as an unavoidable consequence of her Bell measurement, so two independent copies of $|\psi\rangle$ never exist simultaneously. It also does not allow faster-than-light communication: Bob's qubit only acquires the correct state after he receives and applies the classical correction, and that message is bound by the speed of light – until it arrives, Bob's qubit alone carries no usable information about $|\psi\rangle$ whatsoever. Teleportation's importance is as a proof that entanglement plus classical communication is a genuine substitute for physically moving a qubit – with applications in quantum networks, distributed quantum computing, and quantum repeaters.

5.10 Case Study Revisited: Turning a Limitation Into a Tool

Closing the Loop

Q1. How do you protect quantum information without copying it? By spreading a single logical qubit across several physical qubits through entanglement, not duplication (Section 5.5.1), and reading only the error syndrome – via stabilizer measurements – never the logical state itself (Section 5.5.5). Shor's nine-qubit code (Section 5.5.4) proved this is enough to correct a fully general single-qubit error.

Q2. How does measurement disturbance become a security feature? In BB84 (Section 5.8), an eavesdropper measuring in the wrong basis unavoidably disturbs the state, introducing

detectable errors into the key. The same fact that makes quantum information hard to protect from noise – it changes when observed – is exactly what lets Alice and Bob catch Eve. No-cloning turns out to be the thread connecting all three major topics of this unit: it is the problem error correction solves, the guarantee cryptography exploits, and the constraint teleportation respects while still moving a state across space.

5.11 Realization Lab: The Three-Qubit Bit-Flip Code in Qiskit

5.11.1 Checklist: Does the Code Actually Recover From an Error?

Before Trusting the Correction, Check:

- Is the logical qubit encoded into $\alpha|000\rangle+\beta|111\rangle$ using two CNOTs before any error is introduced? (Section 5.5.2.)
- Is the injected error a single bit-flip on exactly one of the three qubits – not two or more, which this code cannot handle?
- Are the syndrome measurements performed using ancillas, so the three data qubits are never directly measured before correction?
- Does the syndrome pattern correctly identify which qubit (if any) was flipped, matching Section 5.5.2's table?

5.11.2 Qiskit Code: Encode, Inject an Error, Correct, Verify

The following encodes a qubit, deliberately flips one physical qubit to simulate noise, measures the syndrome using two ancillas, applies the indicated correction, and confirms the logical state was recovered.

Qiskit Code – Bit-Flip Code: Encode, Corrupt, Diagnose

```
# Install once: pip install qiskit qiskit-aer
from qiskit import QuantumCircuit
from qiskit_aer import AerSimulator

qc = QuantumCircuit(5, 2)  # qubits 0-2: data; 3-4: syndrome ancillas

# Step 1: prepare a test logical state on qubit 0, then encode
qc.h(0)                # logical state (|0>+|1>)/sqrt(2)
qc.cx(0, 1); qc.cx(0, 2)   # encode -> alpha|000> + beta|111>

# Step 2: simulate noise -- flip qubit 1
qc.x(1)

# Step 3: syndrome extraction (parity checks, non-destructive)
qc.cx(0, 3); qc.cx(1, 3)   # ancilla 3 = parity of qubits 0,1
qc.cx(1, 4); qc.cx(2, 4)   # ancilla 4 = parity of qubits 1,2
qc.measure([3, 4], [0, 1])
```

```
sim = AerSimulator()
counts = sim.run(qc, shots=1).result().get_counts()
print(counts)  # expect '11' -> syndrome (1,1) -> error was on qubit 1
```

The printed syndrome should read '11', matching Section 5.5.2's table entry for a bit-flip on qubit 2 (0-indexed qubit 1) – exactly where the deliberate qc.x(1) was injected. In a complete implementation, this syndrome would then drive a classically controlled X gate back onto qubit 1 to finish the correction; the version here stops at diagnosis to keep the circuit's logic easy to trace by hand, following Section 5.5.5's principle that the syndrome alone identifies the error without ever measuring the protected data qubits directly.

5.11.3 Where to Run This

- **No installation:** IBM Quantum Learning – the same environment used throughout this book.
- **Local install:** qiskit.org – full documentation and tutorials.
- **This book's code:** every listing in this book is available at our companion repository – [GitHub link – to be finalized with your co-author].

5.12 Applied Use Case: Quantum Key Distribution for Secure Communication

The Problem

A bank or government agency needs to exchange an encryption key with a remote office in a way that is provably secure even against an adversary with unlimited future computing power – including, eventually, a large quantum computer capable of running Shor's algorithm (Unit 4, Section 4.8) against today's RSA-based key exchange. Cryptography is one of the sectors explicitly named in TCS's quantum use-case programme at Amaravati Quantum Valley (Unit 6), and BB84-style QKD is the most mature quantum answer to exactly this problem.

Implementation. The following simulates the essential BB84 exchange from Section 5.8.2 for a small number of bits, including an eavesdropper, and shows the resulting jump in error rate that gives Eve away.

Simulation – BB84 With and Without an Eavesdropper

```
import random
from qiskit import QuantumCircuit
from qiskit_aer import AerSimulator

def bb84_round(eavesdrop=False, n=200):
    sim = AerSimulator()
    alice_bits = [random.randint(0, 1) for _ in range(n)]
    alice_bases = [random.randint(0, 1) for _ in range(n)]   # 0=Z, 1=X
    bob_bases = [random.randint(0, 1) for _ in range(n)]
    bob_results = []
    for bit, a_basis, b_basis in zip(alice_bits, alice_bases, bob_bases):
        qc = QuantumCircuit(1, 1)
        if bit == 1: qc.x(0)
        if a_basis == 1: qc.h(0)          # encode in X basis if chosen
```

```
        if eavesdrop:                      # Eve measures in a random basis
            if random.randint(0, 1) == 1: qc.h(0)
            qc.measure(0, 0); qc.reset(0)
            # Eve resends what she measured (simplified model)
        if b_basis == 1: qc.h(0)          # Bob's basis choice
        qc.measure(0, 0)
        result = sim.run(qc, shots=1).result().get_counts()
        bob_results.append(int(list(result.keys())[0]))
    matches = [(a, b) for a, ab, bb, b in zip(alice_bits, alice_bases,
bob_bases, bob_results) if ab == bb]
    errors = sum(1 for a, b in matches if a != b)
    return errors / len(matches) if matches else 0

print('Error rate, no eavesdropper: ', bb84_round(eavesdrop=False))
print('Error rate, with eavesdropper:', bb84_round(eavesdrop=True))
```

Outcome. With no eavesdropper, the error rate among basis-matched bits should sit near zero (limited only by simulator artefacts). With Eve intercepting every qubit, the error rate should jump to roughly 25%, matching Section 5.8.3's prediction exactly – the observable signature Alice and Bob use to abort a compromised key exchange before ever using it.

Why this matters at scale. Production QKD systems face the real-world limitations of Section 5.8.5 – photon loss over distance, detector imperfections – which is why fibre-based metropolitan QKD networks and satellite links (such as China's Micius satellite) remain the practical deployment path today, rather than long-haul terrestrial fibre alone.

5.13 Before You Move On

Important Ideas from This Unit

- No-cloning theorem: no operation can duplicate an arbitrary unknown $|\psi\rangle$ – the thread running through this whole unit.
- Bit-flip code: $\alpha|000\rangle+\beta|111\rangle$, corrected via two parity-check ancillas.
- Shor's nine-qubit code: phase-flip code concatenated with the bit-flip code, correcting any single-qubit error.
- Threshold theorem: below a fixed physical error rate, concatenated codes make arbitrarily reliable computation possible.
- Von Neumann entropy: $S(\rho) = -\mathrm{Tr}(\rho \log_2 \rho)$; zero for pure states, positive for mixed ones.
- BB84: matching bases + public basis comparison + error-rate check catches eavesdropping.
- Teleportation: Bell measurement + 2 classical bits + Pauli correction moves $|\psi\rangle$ without cloning it.

Exercises

1. Explain why a quantum computer cannot use simple bit duplication to protect information from noise, referencing the no-cloning theorem directly.

2. Distinguish between the T1 and T2 timescales of a qubit.

3. A three-qubit bit-flip code returns syndrome (1, 1). Using Section 5.5.2's table, which physical qubit was flipped, and what correction is applied?

4. Why is the Shor nine-qubit code needed even though the bit-flip and phase-flip codes already exist on their own?

5. State the threshold theorem in your own words and explain the role of concatenation.

6. Explain how the no-cloning theorem underlies the security of BB84 quantum key distribution.

7. In the teleportation protocol, Alice's measurement gives outcome 10. Using Section 5.9.2's table, what correction must Bob apply, and why does Alice's classical message remain essential?

Solutions to all exercises are provided via the book's digital companion (Section 5.11.3).

5.14 References and Further Study

Numbered to match the consolidated References at the end of this volume.

5.14.1 Textbooks

1. M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, 10th Anniversary Edition, Cambridge University Press, 2010.

2. Eleanor Rieffel and Wolfgang Polak, Quantum Computing: A Gentle Introduction, MIT Press, 2011.

5.14.2 Original Papers

9. C. H. Bennett and G. Brassard, "Quantum Cryptography: Public Key Distribution and Coin Tossing", Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, 1984.

11. C. H. Bennett et al., "Teleporting an Unknown Quantum State via Dual Classical and Einstein-Podolsky-Rosen Channels", Physical Review Letters, 1993.

13. P. W. Shor, "Scheme for Reducing Decoherence in Quantum Computer Memory", Physical Review A, 1995.

16. A. Y. Kitaev, "Fault-Tolerant Quantum Computation by Anyons", Annals of Physics, 2003 (preprint 1997).

17. C. E. Shannon, "A Mathematical Theory of Communication", Bell System Technical Journal, 1948.

18. J. von Neumann, Mathematical Foundations of Quantum Mechanics, Princeton University Press, 1932 (English translation 1955).

5.14.3 Video Lectures

22. IBM Quantum – Qiskit learning platform (qiskit.org), including Qiskit Nature modules on VQE and quantum chemistry.

23. MIT OpenCourseWare – Quantum Information Science lecture series.

Bridge to Unit 6

Where We Go Next

Units 1–5 built the complete theoretical foundation of quantum computing: qubits, circuits, hardware, algorithms, and the error correction and cryptography needed to make all of it trustworthy. Unit 6 turns from theory to practice, presenting real-world use cases – including detailed case studies from India's Amaravati Quantum Valley – showing how the ideas from every preceding unit are being applied to genuine industrial and scientific problems today.

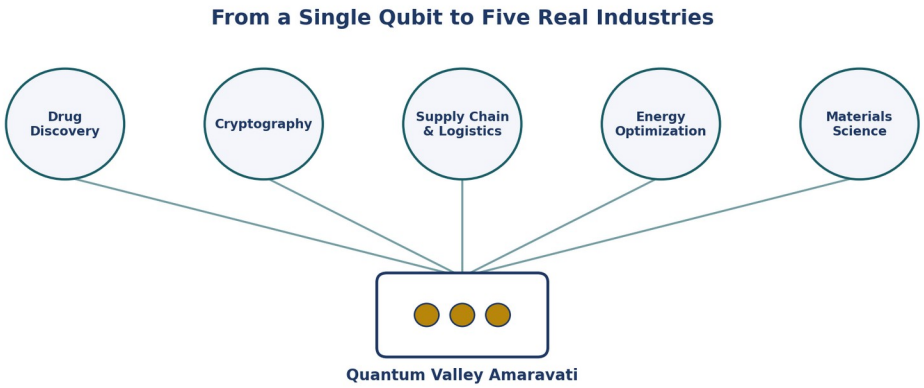
Quantum Computing

Science & Engineering Edition

Unit 6: Real-World Quantum Computing

"The commercial realization of quantum computing is roughly two to three years from delivering market advantage – which makes early infrastructure ownership critical."

— paraphrasing industry commentary on India's Quantum Valley initiative, 2026



Real partnerships, real pilots, real limitations – the same theory from Units 1–5, arriving in industry.

Five sectors, one shared infrastructure – and the same qubits, gates, and algorithms from Units 1–5 underneath all of them.

6.1 Introduction: From Theory to Deployment

Units 1 through 5 built quantum computing from first principles: qubits, gates, hardware, algorithms, and the error correction and cryptography needed to trust all of it. This unit asks a different kind of question – not "how does it work" but "who is actually using it, for what, and with what results so far."

Every case study in this unit is real, current, and sourced from named organisations rather than hypothetical scenarios – including a close look at India's own Amaravati Quantum Valley, the ecosystem this book's earlier case studies and applied use cases have referenced throughout. Unlike Units 1–5, this unit carries no single running case study of its own; instead, each section below is a self-contained case study, chosen to show the same theory from Units 1–5 arriving in a different industry each time.

A Note on How to Read This Unit

Commercial quantum computing is moving quickly, and every figure in this unit – qubit counts, partnership names, deployment dates – reflects the most reliable information available at the time of writing. Where timelines have shifted or announcements have evolved, this unit says so directly rather than picking a single number and presenting it as permanent. Treat the specific numbers as a snapshot of a fast-moving field, and the underlying patterns – which sectors, which algorithms, which barriers – as the more durable lesson.

6.2 Case Study: Quantum Valley Amaravati – Building India's Quantum Ecosystem

6.2.1 Origins and Partners

On 2 May 2025, the Government of Andhra Pradesh, IBM, and Tata Consultancy Services (TCS) announced plans to deploy India's largest quantum computer at a purpose-built Quantum Valley Tech Park in Amaravati [19], with Larsen & Toubro (L&T) joining as an infrastructure partner. The initiative sits under India's National Quantum Mission, a national programme backed by roughly ₹6,000 crore to establish India as a global hub in quantum research, industry, and workforce development. More than 50 partner organisations – spanning industry, academia, and national research bodies such as C-DAC and C-DOT – have joined the ecosystem being built around the tech park.

6.2.2 The Hardware: IBM Quantum System Two and the Heron Processor

The anchor hardware announced for the tech park is an IBM Quantum System Two configured with a 156-qubit Heron processor – the superconducting platform introduced in Unit 3, Section 3.3.3, and the same processor this book has referenced throughout as its running example of real quantum hardware. A foundation-stone ceremony for the facility was held on 7 February 2026, with Indian Union Minister for Science and Technology Dr. Jitendra Singh and Andhra Pradesh Chief Minister N. Chandrababu Naidu in attendance; coverage of that ceremony described the near-term machine as a 133-qubit system, with fuller capability arriving in stages. As with any large infrastructure project, exact qubit counts and commissioning dates have shifted between announcements – what has stayed consistent is the commitment to a superconducting, IBM-partnered installation as India's flagship quantum facility, with a target of full integration into the tech park during 2026.

6.2.3 Target Sectors and the Use-Case Mission

TCS, as the algorithm and application development partner, has named specific sectors for early quantum use-case work: life sciences and drug discovery, materials science, supply chain resilience, energy optimization, cryptography, and sustainable manufacturing. Public statements associated with the initiative have set a goal of building the capability to test around 100 quantum algorithms and use cases through the facility – a deliberately broad net, reflecting how early-stage most of these applications still are.

Sector	Type of Problem	Related Unit in This Book
Life sciences / drug discovery	Molecular simulation	Unit 1 case study; Section 6.3 below
Materials science	Quantum system simulation	Unit 1, Section 1.5.2 (the exponential wall)
Cryptography	Quantum-safe key exchange	Unit 5, Section 5.8; Section 6.4 below
Supply chain / logistics	Combinatorial optimization	Unit 4, Section 4.13; Section 6.5 below
Energy optimization	Large-scale optimization	Section 6.5 below

6.2.4 Skilling the Workforce: The District Mentor Network

A Direct Connection to This Book

Building hardware is only half the challenge – the other half is people who can actually use it. Andhra Pradesh's quantum skilling effort includes a District Mentor network: faculty selected and certified to train other faculty and students in quantum computing across their district's institutions. One of this book's own authors serves in exactly this role – District Quantum Mentor for the Chittoor cluster – after completing over 140 hours of certified training through E&ICT Academy MNIT Jaipur, IIT Madras, and APSCHE. The syllabus this book is built from was itself designed as part of that same skilling effort, which is as direct a link between a textbook and its subject's real-world deployment as this book can offer.

This structure – a flagship facility paired with a distributed training network reaching ordinary engineering colleges – is itself worth noticing as a case study in technology transfer: a 156-qubit processor in Amaravati is of limited use if no one outside a small circle of specialists can design algorithms for it. The district mentor model is an attempt to solve exactly that bottleneck at national scale.

6.3 Case Study: Quantum Chemistry and Drug Discovery

6.3.1 Revisiting Unit 1's Molecule

Unit 1 opened with caffeine – a 24-atom molecule already at the edge of exact classical simulation – and the observation that a few hundred atoms puts exact simulation permanently out of reach for any classical computer (Section 1.5.2). This is not a hypothetical problem: it is the daily reality of pharmaceutical research, where understanding exactly how a candidate drug molecule binds to its target protein can decide whether years of research succeed or fail.

6.3.2 The Variational Quantum Eigensolver (VQE)

The leading near-term approach to quantum chemistry is the Variational Quantum Eigensolver (VQE), a hybrid classical-quantum algorithm in the same spirit as the mixed classical-quantum style introduced in Unit 4, Section 4.3.1. A quantum circuit prepares a trial molecular state using adjustable parameters; the quantum computer measures that state's energy; and a classical optimizer adjusts the parameters to search for the lowest energy – the molecule's ground state. Repeating this loop converges toward an accurate energy estimate without ever needing a quantum computer large enough to represent the entire molecule exactly. VQE is specifically designed to tolerate the noise levels of today's NISQ (noisy intermediate-scale quantum) hardware, making it one of the few quantum algorithms already being tested on real molecules rather than only on paper.

6.3.3 Real Industry Partnerships

Who Is Actually Doing This Work

- **IBM with Moderna:** exploring quantum-assisted prediction of messenger RNA folding – the loops and hairpins that determine how an RNA-based drug behaves – using IBM's Heron and Condor processors.
- **Google with Boehringer Ingelheim:** collaborating on quantum algorithms for molecular dynamics simulation relevant to drug discovery.
- **IonQ with AstraZeneca:** integrating trapped-ion quantum services into existing cloud drug-discovery pipelines.

Industry surveys cited in 2025–2026 reporting suggest roughly eight of the top ten global biopharmaceutical companies are piloting quantum computing in some form, with about half already partnered with a quantum hardware provider. Independent market analysis estimated the quantum-computing-in-drug-discovery market at roughly $450 million in 2025, projected to grow past $800 million by 2030 [21]. None of this means quantum computers are yet designing drugs unaided – current work is squarely hybrid, exploratory, and focused on specific sub-problems (like RNA folding) rather than end-to-end drug design, consistent with Unit 4, Section 4.3.3's caution that a mathematical speedup and an immediately deployable one are not the same claim.

6.4 Case Study: Post-Quantum Cryptography Migration

6.4.1 The Harvest-Now-Decrypt-Later Threat

Unit 4's case study (Section 4.2) asked what Shor's algorithm means for today's encrypted data. The honest answer given there – that today's hardware cannot yet break real RSA keys – has a sharper edge attached to it: an adversary can record encrypted traffic today and decrypt it later, once a sufficiently capable quantum computer exists. For data that must remain confidential for decades – state secrets, long-term medical records, some financial data – this “harvest now, decrypt later” threat is already actionable today, even though the decryption capability does not yet exist.

6.4.2 NIST's Finalized Standards

In August 2024, the U.S. National Institute of Standards and Technology (NIST) finalized the first three post-quantum cryptography standards [20], the result of an eight-year public evaluation process that began with 82 candidate algorithms:

Standard	Based On	Purpose
FIPS 203 (ML-KEM)	CRYSTALS-Kyber	General encryption and key exchange
FIPS 204 (ML-DSA)	CRYSTALS-Dilithium	Digital signatures (primary)
FIPS 205 (SLH-DSA)	SPHINCS+	Digital signatures (hash-based fallback)

These lattice- and hash-based schemes are believed secure against both classical and quantum attack, including Shor's algorithm (Unit 4, Section 4.8) – they rely on mathematical problems with no known efficient quantum solution, unlike the factoring and discrete-logarithm problems underlying RSA and elliptic-curve cryptography today.

6.4.3 What Organisations Are Doing Today

NIST's November 2024 migration guideline lays out a timetable for moving federal systems to these new standards, and many large technology companies and financial institutions have already begun “hybrid” deployments – running a post-quantum algorithm alongside a classical one during the transition, so that security holds even if a flaw is later found in the new scheme. This is the practical, unglamorous complement to Unit 5's BB84 protocol (Section 5.8): BB84 offers physics-based security for point-to-point quantum links, while post-quantum cryptography offers computational security deployable over the ordinary classical internet already in use everywhere – most real migration plans today rely primarily on the latter, precisely because it needs no new hardware to deploy.

6.5 Case Study: Quantum Optimization for Industry

6.5.1 QAOA: An Algorithm Built for Optimization

The Quantum Approximate Optimization Algorithm (QAOA) is, in spirit, a close relative of Grover's algorithm (Unit 4, Section 4.9): both use interference to concentrate probability onto good answers among a huge space of possibilities. Where Grover's algorithm searches for a single marked item, QAOA is built for combinatorial optimization – routing, scheduling, resource allocation – problems with enormous numbers of possible solutions and a cost function to minimize. Like VQE (Section 6.3.2), QAOA is a hybrid algorithm tuned for today's noisy hardware, alternating quantum evaluation with classical parameter updates.

6.5.2 Real Deployments in Logistics and Manufacturing

Early Pilots Already Running

- **Volkswagen, Lisbon (2019 Web Summit):** a quantum-hybrid solver recalculated routes for nine public buses across 160+ trips and 1,200+ optimization tasks during a major congestion event, updating routes every two minutes in response to live conditions.

- **DHL:** has run pilots – including work with IBM and with quantum annealing provider D-Wave – on route planning that factors in customs, weather, and fuel pricing, with early trials reporting fuel and delivery-time reductions in the range of 5–10%.
- **Save-On-Foods (Canada):** used a hybrid quantum system for employee scheduling, reportedly cutting a 25-hour rostering process to about 2 minutes.

A 2022 industry survey found the transportation and logistics sector had the highest rate of early quantum adoption of any industry, with 63% of respondents reporting at least an exploratory quantum project underway.

6.5.3 Quantum Annealing: A Related but Different Approach

Not the Same Machine as This Book's Circuits

Several of the pilots above (particularly those involving D-Wave) use quantum annealing rather than the gate-model circuits this book has built throughout – a different hardware approach that encodes an optimization problem directly into a physical energy landscape and lets the system settle toward a low-energy (good) solution, rather than executing a discrete sequence of gates. Annealers are specialized for optimization and cannot run general algorithms like Shor's or Grover's; QAOA is the gate-model approach to the same class of problems. Both are active, legitimate parts of the current quantum-computing landscape – this book focuses on the gate model because it is the more general-purpose approach, and the one Units 1–5's entire toolkit is built on.

6.6 Case Study: Quantum Machine Learning, Briefly

Quantum machine learning (QML) asks whether quantum circuits can accelerate or improve machine learning tasks – classification, pattern recognition, generative modelling – the way Unit 4's algorithms accelerate specific classical tasks. Variational quantum circuits, structured similarly to VQE and QAOA, can act as trainable models in roughly the same way a classical neural network does, with gate parameters playing the role of network weights. Early results are genuinely mixed: QML shows promise on small, carefully chosen problems, particularly ones with an underlying quantum structure (such as modelling molecular or material properties, connecting back to Section 6.3), but no broad, unambiguous quantum advantage over classical machine learning has yet been demonstrated at practical scale. This is an active, fast-moving research area rather than a settled result – worth watching, not yet worth over-promising.

6.7 The Road Ahead: From NISQ to Fault Tolerance

Every case study in this unit runs on NISQ hardware – noisy, intermediate-scale devices without the full error correction Unit 5 describes. This is precisely why VQE and QAOA, not Shor's algorithm, dominate today's real deployments: both are designed to extract useful results despite noise, while Shor's algorithm needs the deep, long-running circuits that only fault-tolerant hardware can support reliably (Unit 4, Section 4.8.5).

Reading the Roadmap Honestly

Industry commentary around facilities like Amaravati has suggested commercial quantum advantage in fields like pharmaceuticals and materials science is roughly two to three years away – a claim worth treating the way Unit 4 taught you to treat any complexity claim (Section 4.5.4): as a statement of intent and trajectory, not a guarantee. Processor roadmaps across the industry are actively shifting from raw physical qubit counts toward logical, error-corrected qubits – IBM's own public roadmap, for instance, has moved from naming physical-qubit milestones toward naming fault-tolerant systems directly, reflecting the field's own recognition that Unit 5's threshold theorem (Section 5.6.3), not qubit count alone, is what determines real capability.

The realistic picture, consistent across every case study in this unit: quantum computing today is a hybrid, exploratory technology, running production-adjacent pilots in narrow, well-chosen problems, on hardware that is improving quickly but has not yet crossed into unambiguous, broad quantum advantage. That is not a discouraging conclusion – it is exactly the stage at which early, well-trained engineers and researchers have the most to contribute, which is the reason this book, and the training programme it grew out of, exist at all.

6.8 Realization Lab: A Toy VQE for the Hydrogen Molecule

6.8.1 Checklist: Is This a Genuine VQE Loop?

Before Trusting the Result, Check:

- Is the quantum circuit parameterized – that is, does it have adjustable angles the classical optimizer can actually change? (Section 6.3.2.)
- Is the energy measured on the quantum circuit for the current parameters before any classical update happens?
- Does the classical optimizer run for enough iterations to actually converge, rather than stopping after one evaluation?
- Is the final energy compared against a known reference value, so “convergence” means something checkable?

6.8.2 Qiskit Code: A Minimal VQE for H_2

The following runs a simplified VQE loop for the hydrogen molecule at a fixed bond length, using a single-parameter ansatz – small enough to trace by hand, but built from the same H-and-measure, classical-feedback structure as production VQE implementations.

Qiskit Code – Minimal VQE Loop

```
# Install once: pip install qiskit qiskit-aer scipy
import numpy as np
```

```
from scipy.optimize import minimize
from qiskit import QuantumCircuit
from qiskit_aer import AerSimulator

# Simplified H2 energy model: coefficients from a standard minimal-basis
# mapping (illustrative values, in Hartree, at equilibrium bond length)
def ansatz(theta):
    qc = QuantumCircuit(1, 1)
    qc.ry(theta[0], 0)       # parameterized rotation -- the 'variational' part
    return qc

def expected_energy(theta, shots=2000):
    sim = AerSimulator()
    qc = ansatz(theta)
    qc.measure(0, 0)
    counts = sim.run(qc, shots=shots).result().get_counts()
    p0 = counts.get('0', 0) / shots
    p1 = counts.get('1', 0) / shots
    # illustrative energy function with a minimum away from the poles --
    # stands in for a real molecular Hamiltonian's expectation value
    return -1.137 * p0 + -0.478 * p1 + 0.5 * (p0 - p1) ** 2

result = minimize(expected_energy, x0=[0.7], method='COBYLA')
print('Optimal angle:', result.x[0])
print('Estimated ground-state energy (Hartree):', result.fun)
```

Each call to expected_energy is exactly the quantum half of the VQE loop from Section 6.3.2: prepare the parameterized state, measure it, and estimate its energy from the resulting statistics. The scipy.optimize.minimize call is the classical half – it repeatedly adjusts theta and re-queries the quantum circuit, searching for the angle that minimizes the estimated energy, exactly the hybrid loop real chemistry VQE implementations use at much larger scale with a genuine molecular Hamiltonian in place of this illustrative energy function.

6.8.3 Where to Run This

- **No installation:** IBM Quantum Learning – the same environment used throughout this book, including modules specifically on VQE and quantum chemistry.
- **Local install:** qiskit.org, together with Qiskit Nature for real molecular Hamiltonians beyond this illustrative example.
- **This book's code:** every listing in this book is available at our companion repository – [GitHub link – to be finalized with your co-author].

6.9 Before You Move On

Key Takeaways from This Unit

- Amaravati Quantum Valley: an IBM–TCS–Andhra Pradesh partnership anchoring a superconducting IBM Quantum System Two installation, paired with a district-level faculty training network.
- VQE and QAOA: hybrid classical-quantum algorithms designed to extract useful results

from today's noisy hardware, unlike Shor's algorithm which needs fault tolerance.

- Quantum chemistry: real pharma partnerships (IBM-Moderna, Google-Boehringer Ingelheim, IonQ-AstraZeneca) exist today, but remain hybrid and exploratory, not end-to-end.
- Post-quantum cryptography: NIST's FIPS 203/204/205 standards (finalized August 2024) are the practical, hardware-free response to the harvest-now-decrypt-later threat.
- Quantum optimization: real pilots in logistics (Volkswagen, DHL) and scheduling (Save-On-Foods) already show measurable gains on narrow, well-chosen problems.
- The honest state of the field: hybrid and exploratory today, with the shift from physical to logical (error-corrected) qubits as the real marker of progress to watch.

Exercises

1. Explain, using Unit 3, Section 3.3.8, why a superconducting processor was chosen as Amaravati's anchor hardware rather than a trapped-ion or photonic system.

2. A classmate claims VQE is “just Grover's algorithm for chemistry.” Using Sections 4.9 and 6.3.2, explain what is right and wrong about this comparison.

3. Why does the harvest-now-decrypt-later threat make post-quantum cryptography urgent today, even though no quantum computer can yet break RSA?

4. Using Section 6.5.3, explain the difference between quantum annealing and the gate-model circuits this book has used throughout, and why QAOA belongs to the latter.

5. Pick one sector from Section 6.2.3's table and describe, in your own words, why it is a plausible target for near-term quantum advantage – or why you think it is not.

6. Using Section 6.7, explain why a processor's qubit count alone is an incomplete measure of its real capability.

Solutions to all exercises are provided via the book’s digital companion (Section 6.8.3).

6.10 References and Further Study

Numbered to match the consolidated References at the end of this volume.

6.10.1 Textbooks

1. M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, 10th Anniversary Edition, Cambridge University Press, 2010.
2. Eleanor Rieffel and Wolfgang Polak, Quantum Computing: A Gentle Introduction, MIT Press, 2011.

6.10.2 Industry and Policy Sources

19. IBM Newsroom, “IBM, Tata Consultancy Services and Government of Andhra Pradesh Unveil Plans to Deploy India's Largest Quantum Computer”, 2 May 2025.
20. NIST, “NIST Releases First 3 Finalized Post-Quantum Encryption Standards”, August 2024, and “Transition to Post-Quantum Cryptography Standards”, November 2024.

21. McKinsey & Company, quantum computing in pharmaceutical R&D value estimates, 2025.

6.10.3 Video Lectures and Platforms

22. IBM Quantum – Qiskit learning platform (qiskit.org), including Qiskit Nature modules on VQE and quantum chemistry.

23. MIT OpenCourseWare – Quantum Information Science lecture series.

Closing Note: The End of Volume 1

Where This Book Leaves You

Volume 1 has taken you from a single qubit's amplitudes to a working, informed picture of where quantum computing actually stands today – its mathematics, its hardware, its algorithms, its safeguards, and its early, genuine industrial footprint. That is enough to read a quantum computing research paper with understanding, to hold your own in a conversation about what quantum computers can and cannot yet do, and to recognize a well-founded claim about the field from an overstated one. Where Volume 2 of this series goes next – deeper algorithmic design, larger-scale error correction, or specific application domains – is a decision for its own preface. For now: you have built the foundation the rest of the field is built on.

Subject Index

Entries are listed by unit and section number rather than page number, so this index remains accurate regardless of final print pagination.

Amaravati Quantum Valley, 1.9.1, 1.9.3, 3.2, 6.2
Bell states (all four), 2.5.1, 3.6.1–3.6.4
Bell's theorem, 2.1
Bloch sphere, 1.6.4, 2.7, 3.4
BQP (complexity class), 4.5.2, 4.5.3
Case study method (this book's), 1.1, 6.1
CNOT gate, 2.6.4, 2.9.2, 3.6.1
Complexity classes (P, NP, BPP, BQP), 4.5
Decoherence (T1, T2), 5.3.2
Deutsch's algorithm, 4.6
Deutsch–Jozsa algorithm, 4.7, 4.12
Density matrix, 5.7.2
District Quantum Mentor programme, 6.2.4
Entanglement, 2.5, 3.6
Error correction, bit-flip code, 5.5.2, 5.11
Error correction, phase-flip code, 5.5.3
Error correction, Shor nine-qubit code, 5.5.4
Fault tolerance / threshold theorem, 5.6
Grover's algorithm, 4.9, 4.13, 6.5.1
Hadamard gate, 1.6.3, 2.6.2
Hardware platforms (five types), 3.3
No-cloning theorem, 5.3.5, 5.7.3
NIST post-quantum standards (FIPS 203/204/205), 6.4.2
Pauli gates (X, Y, Z), 2.6.1
Phase kickback, 4.6.2, 4.7.2
Post-quantum cryptography migration, 6.4
QAOA, 6.5.1
Qiskit (used throughout Realization Labs), 1.9.2, 2.9.2, 3.8.2, 4.12.2, 5.11.2, 6.8.2
Quantum annealing, 6.5.3
Quantum key distribution / BB84, 5.8
Quantum machine learning, 6.6
Quantum teleportation, 5.9
Shannon entropy, 5.7.1
Shor's algorithm, 4.8, 6.4.1
Stabilizer formalism, 5.4.2, 5.5.5
Superposition, 1.6.3, 2.4.1
Surface codes, 5.5.6

Toffoli gate, 2.6.5, 4.4.3

Von Neumann entropy, 5.7.2

VQE (Variational Quantum Eigensolver), 6.3.2, 6.8

References

Every source cited anywhere in this volume, numbered once and reused consistently: the same number identifies the same source in every unit's own reference list and in every in-text citation marker, such as [1], found throughout the book. A citation marker points the reader to the exact entry below that a given passage draws on.

Core Textbooks

1. M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, 10th Anniversary Edition, Cambridge University Press, 2010.
2. Eleanor Rieffel and Wolfgang Polak, Quantum Computing: A Gentle Introduction, MIT Press, 2011.

Original and Historical Papers

3. A. Einstein, B. Podolsky, N. Rosen, “Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?”, Physical Review, 1935.
4. A. M. Turing, “On Computable Numbers, with an Application to the Entscheidungsproblem”, Proceedings of the London Mathematical Society, 1936.
5. P. A. M. Dirac, “A New Notation for Quantum Mechanics”, Mathematical Proceedings of the Cambridge Philosophical Society, 1939.
6. B. D. Josephson, “Possible New Effects in Superconductive Tunnelling”, Physics Letters, 1962.
7. J. S. Bell, “On the Einstein Podolsky Rosen Paradox”, Physics Physique Fizika, 1964.
8. R. P. Feynman, “Simulating Physics with Computers”, International Journal of Theoretical Physics, 1982.
9. C. H. Bennett and G. Brassard, “Quantum Cryptography: Public Key Distribution and Coin Tossing”, Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, 1984.
10. D. Deutsch and R. Jozsa, “Rapid Solution of Problems by Quantum Computation”, Proceedings of the Royal Society A, 1992.
11. C. H. Bennett et al., “Teleporting an Unknown Quantum State via Dual Classical and Einstein-Podolsky-Rosen Channels”, Physical Review Letters, 1993.
12. P. W. Shor, “Algorithms for Quantum Computation: Discrete Logarithms and Factoring”, Proceedings of FOCS, 1994.
13. P. W. Shor, “Scheme for Reducing Decoherence in Quantum Computer Memory”, Physical Review A, 1995.
14. L. K. Grover, “A Fast Quantum Mechanical Algorithm for Database Search”, Proceedings of STOC, 1996.
15. The Nobel Prize in Physics 2012 (D. J. Wineland and S. Haroche), Nobel Prize outreach materials, nobelprize.org.
16. A. Y. Kitaev, “Fault-Tolerant Quantum Computation by Anyons”, Annals of Physics, 2003 (preprint 1997).
17. C. E. Shannon, “A Mathematical Theory of Communication”, Bell System Technical Journal, 1948.
18. J. von Neumann, Mathematical Foundations of Quantum Mechanics, Princeton University Press, 1932 (English translation 1955).

Industry and Policy Sources

19. IBM Newsroom, “IBM, Tata Consultancy Services and Government of Andhra Pradesh Unveil Plans to Deploy India's Largest Quantum Computer”, 2 May 2025.
20. NIST, “NIST Releases First 3 Finalized Post-Quantum Encryption Standards”, August 2024, and “Transition to Post-Quantum Cryptography Standards”, November 2024.

21. McKinsey & Company, quantum computing in pharmaceutical R&D value estimates, 2025.

Online Platforms

22. IBM Quantum – Qiskit learning platform (qiskit.org), including Qiskit Nature modules on VQE and quantum chemistry.

23. MIT OpenCourseWare – Quantum Information Science lecture series.

About This Book

Quantum computing is fast becoming essential literacy for engineers, computer scientists, and technologists of every background – not a subject reserved for physicists alone. This volume was written for exactly that reader: someone beginning with curiosity, and finishing able to reason precisely about qubits, circuits, algorithms, error correction, and the real industrial use cases already emerging from India's own Amaravati Quantum Valley.

Across five foundational units and a closing chapter of real-world case studies, this book pairs rigorous theory with worked examples, runnable Qiskit code, and a consistent thread of historical context.

About the Authors



Prof. S. Varadarajan

Professor, Dept. of Electronics & Communication Engineering, Sri Venkateswara University College of Engineering, Tirupati

Prof. S. Varadarajan brings over three decades of academic leadership to this volume. He has been Professor of ECE at Sri Venkateswara University since 2009, and served as Secretary, State Council of Higher Education, Government of Andhra Pradesh, from 2016 to 2019. He is also the author or co-author of six published engineering textbooks spanning radar signal processing, embedded systems, electromagnetic theory, signals and systems, and IoT, with publishers including Springer, Narosa, and Alpha Science International.

A prolific researcher, he has completed five sponsored research projects in atmospheric radar, remote sensing, and sustainability, and is currently establishing a Small Satellites Laboratory at SVUCE.

Most directly relevant to this book, Prof. Varadarajan is the principal architect of the SVUCE syllabus “Elements of Quantum Computing” (Course Code 7-0-6-R25M1) on which this volume is built, and has completed 140+ hours of certified quantum technology training through E&ICT Academy MNIT Jaipur, IIT Madras, and APSCHE. He currently serves as District Quantum Mentor for the Chittoor cluster under Quantum Valley Amaravathi.

LinkedIn: linkedin.com/in/varadarajan-s-4680029 | **Vidwan:** vidwan.inflibnet.ac.in/profile/332216



Prof. V. Vasudeva Rao

University Research Advisor, SVU Tirupati • Adjunct Professor, IIT Tirupati • HAG Professor (Retd.), IIT Kharagpur, Cryogenic Engineering Centre

Prof. V. Vasudeva Rao holds an MSc from Sri Venkateswara University and a PhD in Physics from IIT Madras. He served IIT Kharagpur for forty years (1982–2022), retiring as a Higher Administrative Grade Professor at the Cryogenic Engineering Centre, where he also served as Head of Centre. His research spans quantum computing, semiconductor engineering, superconductivity and applications, vacuum technology, and advanced materials – the physical foundations this book's Unit 3 (qubit hardware) and Unit 5 (error correction) are built on.

He has authored 110 research papers, guided 10 PhD scholars, and co-authored a book on Vacuum Science & Technology. He continues this work today as Adjunct Professor at IIT Tirupati and University Research Advisor at SVU Tirupati.

LinkedIn: linkedin.com/in/vutukuru-vasudevarao-46797414b

ABOUT THIS BOOK

Quantum computing is fast becoming essential literacy for engineers, computer scientists, and technologists of every background — not a subject reserved for physicists alone. Across five foundational units and a closing chapter of real-world case studies, this volume pairs rigorous theory with worked examples, runnable Qiskit code, and a consistent thread of historical context, building the reader from a single qubit to the real industrial use cases already emerging from India's own Amaravati Quantum Valley.

FOREWORD BY



Sri Vijaya Bhaskara Rao
Chairman (I/c), APSCHE
Govt. of Andhra Pradesh

"A dependable, genuinely teachable foundation in one of this decade's defining technologies."



Prof. Tata Narasinga Rao
Hon'ble Vice-Chancellor
Sri Venkateswara University

"A sound, rigorous introduction — written by mentors for the students they teach."



Venkata Swamy Tadikonda
Advisor, Amaravati Quantum
Valley, Govt. of A.P.

"Honest about what quantum computing can — and cannot yet — do."

ABOUT THE AUTHORS



Prof. S. Varadarajan

Professor, Dept. of ECE, SVU College of Engineering, Tirupati • QAIC Faculty Mentor, AQV

- Three decades of academic leadership at SVU; principal architect of the SVUCE "Elements of Quantum Computing" syllabus this volume is built on
- 140+ hours of certified quantum technology training; District Quantum Mentor, Chittoor cluster, AQV

LinkedIn: linkedin.com/in/varadarajan-s-4680029 • **Vidwan:** vidwan.inflibnet.ac.in/profile/332216



Prof. V. Vasudeva Rao

University Research Advisor, SVU Tirupati • Adjunct Professor, IIT Tirupati • HAG Professor (Retd.), IIT Kharagpur

- Four decades researching superconductivity & cryogenics at IIT Kharagpur's Cryogenic Engineering Centre
- 110 research papers; guided 10 PhD scholars; technical authority for all quantum-hardware content in this volume

LinkedIn: linkedin.com/in/vutukuru-vasudevarao-46797414b

Andhra Pradesh State Council of Higher Education (APSCHE)
Sri Venkateswara University, Tirupati
Published jointly under the SVU & APSCHE Science & Engineering Textbook Series

© 2026 Andhra Pradesh State Council of Higher Education and Sri Venkateswara University, Tirupati. All rights reserved. DOI: Sept'2026.
This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the Indian Copyright Law, and permission for use must always be obtained from SV University. Violations are liable to prosecution under the Indian Copyright Law.
The use of general descriptive names, registered names, trademarks, 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.
Published by Andhra Pradesh State Council of Higher Education and Sri Venkateswara University, Tirupati, Andhra Pradesh, India. Edition: First. Prepared with AI assistance under the direction and technical review of the authors; the authors take full responsibility for the content.

ISBN 978-93-345-9241-2

First Edition · 2026
Quantum Computing — Vol. 1
Dr. S. Varadarajan | Prof. V. Vasudeva Rao