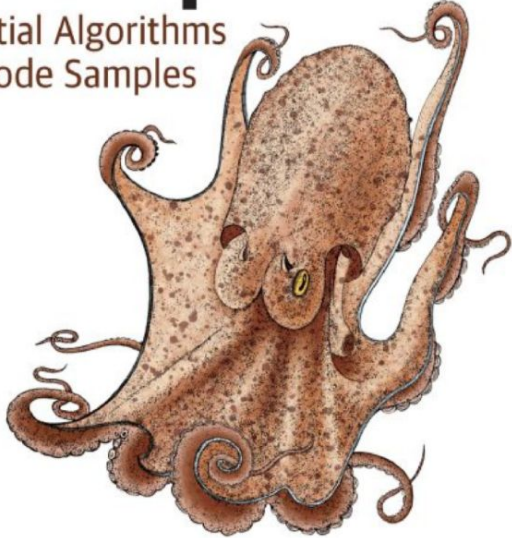


O'REILLY®

Programming Quantum Computers

Essential Algorithms
and Code Samples



Eric R. Johnston, Nic Harrigan
& Mercedes Gimeno-Segovia

Copyrighted material

Bookclub Wk1 - ch1

Logistics



Format











- Weekly 60 minute live meeting
 - recorded
 - posted to YouTube channel
- Slack
 - #intros
 - **#general**
 - channels for book sections
- One person is group lead each week
 - 30-45 min presenting

Goals



Why we are here

- Interest in this topic
- Complexity of topic
- Want to learn together

-  Slackbot
-  Lynn Langit you
-  Dan Friedman
-  Danika Hannon
-  Gerard Beaubrun
-  Jeremy Clark
-  Kelly Kermode
-  kensykora
-  Mike Coon
-  Nathaniel Dalbec

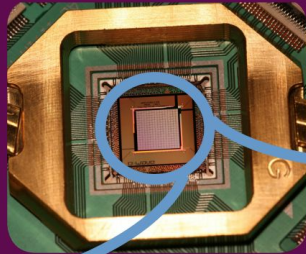
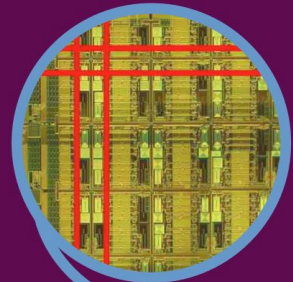
Chapter 1 - Intro



Qubits in red

Quantum processing unit

Inside the D-Wave enclosure

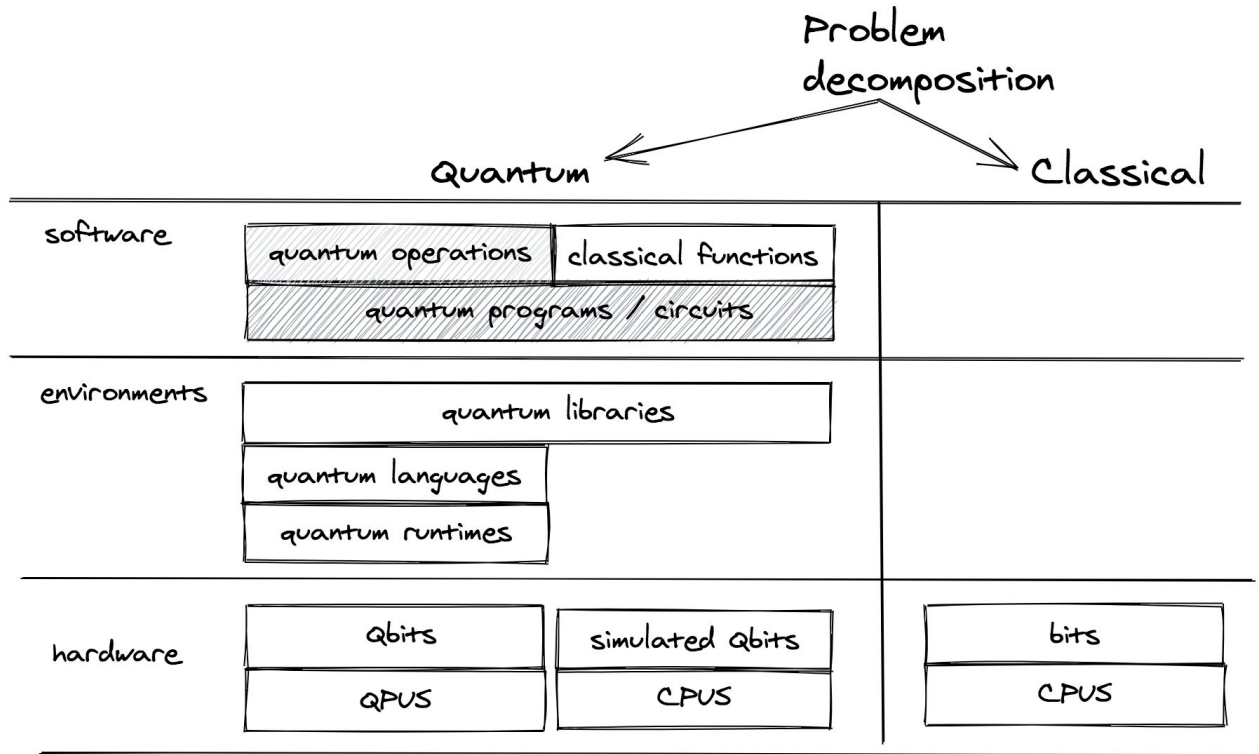


QPU (Quantum Processing Unit) to refer to the device on which our code samples run

The D-Wave QPU is built from a lattice of tiny loops of the metal niobium, each of which is one qubit. Below temperatures of 9.2 kelvin, niobium becomes a superconductor and exhibits quantum mechanical effects.

D-Wave QPU

Coverage of Topic Area



Intro – Info

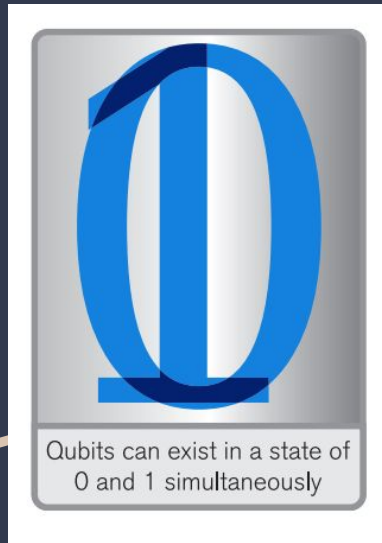
Math background:

- mathematical functions
- trigonometric functions
- converting between binary and decimal representations
- comprehension of complex numbers

Programming background:

- one or more programming languages
- default book examples in JavaScript

Chapter 1 – Info



Qbit - 1 quantum computational unit

Qbyte - 8 qbits

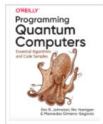
QPU - quantum processing unit

QCEngine - quantum simulator (JavaScript)

Quantum Circuit - program circuit

Circle-notation - program visualization

Running the code



Programming Quantum Computers

Code Samples

Run Program

Ex 2-1: Random bit

QCEngine



```
1 // Programming Quantum Computers
2 // by Eric Johnston, Nic Harrigan and Mercedes Gimeno-Segovia
3 // O'Reilly Media
4
5 // To run this online, go to http://oreilly-qc.github.io?p=2-1
6
7 // This sample generates a single random bit.
8
9 qc.reset(1); // allocate one qubit
10 qc.write(0); // write the value zero
11 qc.had(); // place it into superposition of 0 and 1
12 var result = qc.read(); // read the result as a digital bit
13
```

Source code on GitHub

QCEngine

Qiskit

OpenQASM

Q#

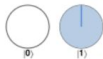
Cirq

Program circuit



0x1 > [H] -

Circle notation



Program output



(output prints here)

<https://oreilly-qc.github.io/>

Options for running examples

Library or Language	Language basis	Env	Other
QCEngine	JavaScript	Browser	Circle and Circuit Viz
Qiskit	Python	Notebook	IBM env
OpenQASM	C-style language	Composer	IBM env
Q#	C#/F# - style language	Notebook	Binder -> GKE
Cirq	Python	Notebook	Colab or GCP



Lab files



New file +

Name	Updated	↓
Untitled.ipynb	2 minutes ago	⋮
qiskit-textbook	3 minutes ago	⋮
qiskit-tutorials	3 minutes ago	⋮
W-state_Apr 05, 2021 ...	8 days ago	⋮

Run Qiskit (python) on IBM Quantum Lab

File Edit View Insert Cell Kernel Widgets Help

Trusted

Kernel jupyter



Code



Memory: 253.6 MB / 8 GB

```
import math
## Uncomment the next line to see diagrams when running in a notebook
%matplotlib inline

## Example 2-1: Random bit
# Set up the program
reg = QuantumRegister(1, name='reg')
reg_c = ClassicalRegister(1, name='regc')
qc = QuantumCircuit(reg, reg_c)

qc.reset(reg)          # write the value 0
qc.h(reg)              # put it into a superposition of 0 and 1
qc.measure(reg, reg_c) # read the result as a digital bit

backend = BasicAer.get_backend('statevector_simulator')
job = execute(qc, backend)
result = job.result()

counts = result.get_counts(qc)
print('counts:', counts)

outputstate = result.get_statevector(qc, decimals=3)
print(outputstate)
qc.draw()              # draw the circuit
```

```
counts: {'0': 1}
[1.+0.j 0.+0.j]
```





Composer files



3 files



New file +



Name	Updated	
Untitled circuit	a few seconds ago	
W-state	8 days ago	
Grover-example	8 days ago	

Run OPENQASM on IBM Quantum Composer

File Edit Inspect View Share

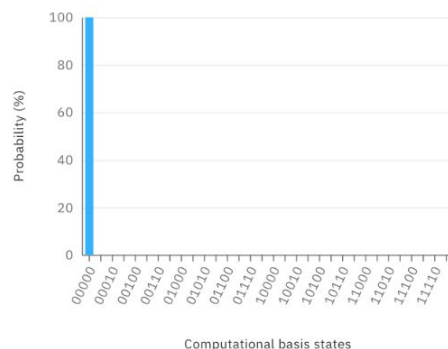
Untitled circuit *Saved*

Visualizations seed

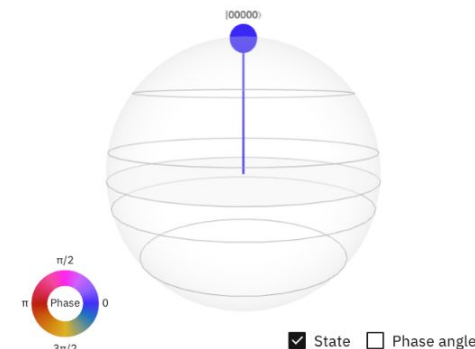
6509



Probabilities



Q-sphere



Setup and run



OpenQASM 2.0

[Open in Quantum Lab](#)

```
1 // Programming Quantum
  Computers
2 // by Eric Johnston, Nic
  Harrigan and Mercedes Gimeno-
  Segovia
3 // O'Reilly Media
4 //
5 // More samples like this can
  be found at http://oreilly-
  qc.github.io
6
7 // Run this sample in the IBM Q
  Experience Circuit Composer
8 // at https://quantum-
  computing.ibm.com
9
10 // This sample generates a
   single random bit.
11
12 OPENQASM 2.0;
13 include "qelib1.inc";
14
15 qreg q[5];
16 creg c[5];
17
18 h q[0];
19 measure q[0] -> c[0];
20
```



Programming Quantum Computers by O'Reilly Media - [book info](#) - [all code samples](#)

Code samples for Chapter 2

These code samples were written by Andres Paz and Mariia Mykhailova.

```
In [1]: // Example 2-1: Random bit

operation RandomBit () : Unit {
    // allocate one qubit
    using (q = Qubit()) {
        // put it into superposition of 0 and 1
        H(q);

        // measure the qubit and store the result
        let bit = M(q);

        // make sure the qubit is back to the 0 state
        Reset(q);

        Message($"{bit}");
    }
}
```

Out[1]: • RandomBit

```
In [2]: %simulate RandomBit
```

Zero

Run Q# on
GKE Jupyter Notebooks via
Binder



×

✓ RAM ▼ | Editing |

▼ Copyright 2020 The Cirq Developers

▶ Licensed under the Apache License, Version 2.0 (the "License");

- ▼ Qubits

[View on QuantumAI](#)
[Run in Google Colab](#)
[View source on GitHub](#)
[Download notebook](#)

```
1 try:
2     import cirq
3 except ImportError:
4     print("installing cirq...")
5     !pip install --quiet cirq
6     print("installed cirq.")
```

A qubit is the basic unit of quantum information, a quantum bit: a two level system that can exist in superposition of those two possible states. Cirq also supports higher dimensional systems, so called [qudits](#) that we won't cover here.

In Cirq, a `Qubit` is nothing else than an abstract object that has an identifier, a `cirq.Qid` and some other potential metadata to represent device specific properties that can be used to validate a circuit. In contrast to real qubits, the Cirq qubit does not have any state. The reason for

Quantum Hardware - public cloud examples

[Home](#) >

Create Quantum Workspace ...

Quantum Workspace

[Basics](#) [Providers](#) [Tags](#) [Review + create](#)

Run your jobs on quantum computers, quantum simulators or using quantum inspired optimizations. [See the terminology sheet.](#)

Available providers



1Qloud Optimization Platform

Optimization

1QBit 1Qloud Optimization Platform
with Quantum Inspired Solutions

[+ Add](#)

Honeywell Quantum Solutions

Quantum Computing

Access to Honeywell Quantum
Solutions' trapped-ion systems

[+ Add](#)

IonQ

Quantum Computing

IonQ's trapped ion quantum computers
perform calculations by manipulating
charged atoms of Ytterbium held in a
vacuum with lasers.

[+ Add](#)

Microsoft QIO

Optimization

Ground-breaking optimization
algorithms inspired by decades of
quantum research.

Added

Providers added to this workspace

Name ↑↓

Provider type ↑↓

SKU ↑↓



Microsoft QIO
Microsoft

Optimization

Learn & Develop

[Modify](#) [Remove](#)

Amazon Braket ×

Devices

Notebooks

Tasks

Announcements

Quantum Processing Units (QPUs)

D-Wave — Advantage_system1.1

Quantum Annealer based on superconducting qubits

Qubits
5760Status
✔️ ONLINERegion
us-west-2Next available
✔️ AVAILABLE NOW

D-Wave — DW_2000Q_6

Quantum Annealer based on superconducting qubits

Qubits
2048Status
✔️ ONLINERegion
us-west-2Next available
✔️ AVAILABLE NOW

IonQ

Universal gate-model QPU based on trapped ions

Qubits
11Status
✔️ ONLINERegion
us-east-1Next available
09:38:28

Rigetti — Aspen-8

Universal gate-model QPU based on superconducting qubits

Qubits
31Status
❌ OFFLINERegion
us-west-1Next available
🕒 UNAVAILABLE

Rigetti — Aspen-9

Universal gate-model QPU based on superconducting qubits

Qubits
31Status
✔️ ONLINERegion
us-west-1Next available
11:38:28

Simulators

Amazon Web Services — SV1

Amazon Braket state vector simulator

Qubits
34Status
✔️ ONLINERegion
us-east-1, us-west-1, us-west-2Next available
✔️ AVAILABLE NOW

Amazon Web Services — TN1

Amazon Braket tensor network simulator

Qubits
50Status
✔️ ONLINERegion
us-east-1, us-west-2Next available
✔️ AVAILABLE NOW

Resources

- Book "**Programming Quantum Computers**" - <https://learning.oreilly.com/library/view/programming-quantum-computers/9781492039679/>
- Lynn's '**learning-quantum**' GitHub Repo - <https://github.com/lynnlangit/learning-quantum>
- Jeremy's '**programming quantum experiments**' GitHub Repo - <https://github.com/jeremybytes/quantum-programming-experiments>
- more ???