Quantum Computing (QUC)

Your Lecturer

Stefano Gogioso

Faculty in the Oxford Quantum Group

- Quantum Information
- Quantum Causality
- Foundations of Physics
- Quantum Software

External business interests:
crypto & decentralised finance

HASHBERG

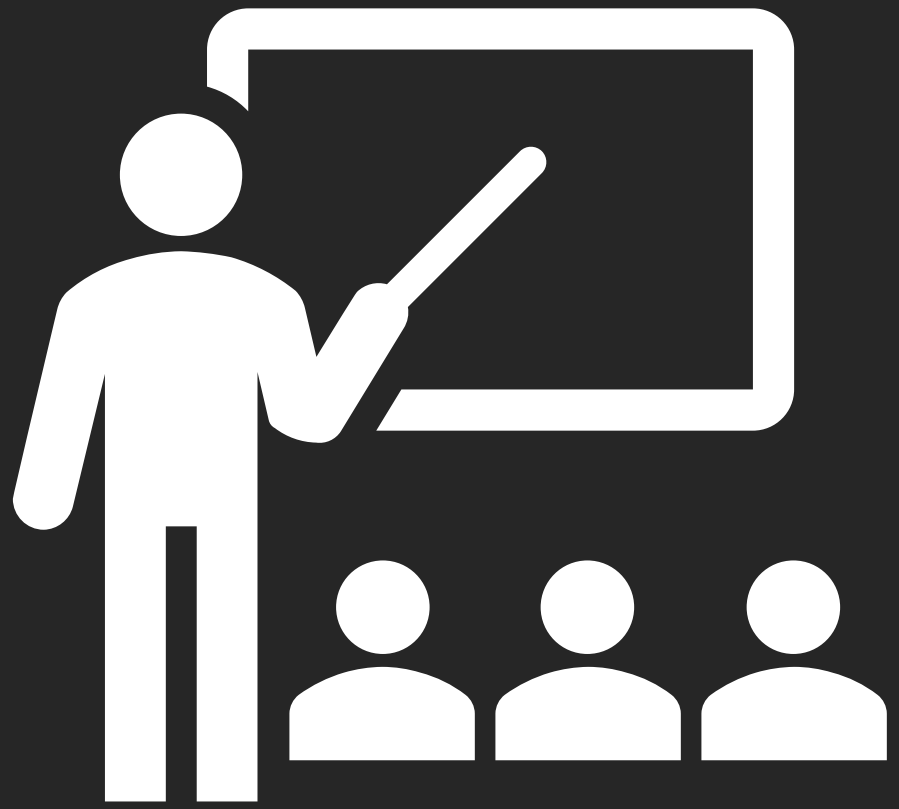
Your TA

Nicola Pinzani

Student in the Oxford Quantum Group

- Quantum Information
- Quantum Causality
- Foundations and Philosophy of Physics

Tell us a little about you!



Plan for this course

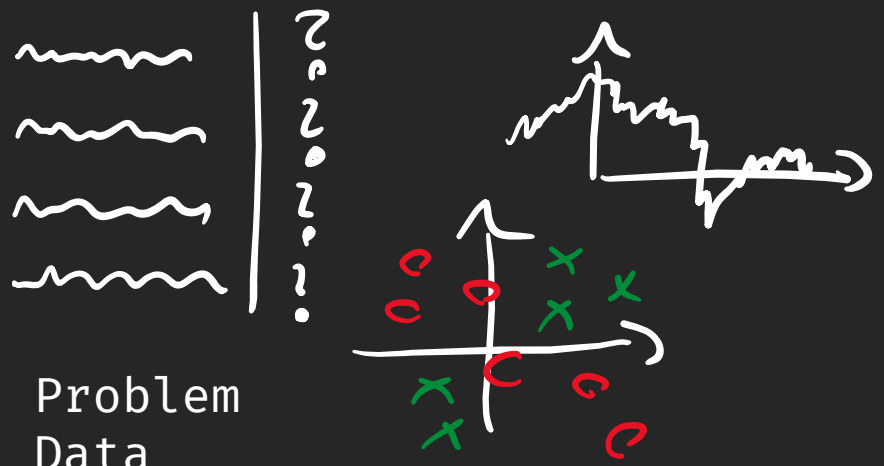
- Lectures Mon-Fri 9am-12:15pm
- Exercises Mon-Thu 2pm-5pm

What is Quantum Computing?

- Quantum Computers as black-box devices used inside classical algorithms
- Quantum Computers as a programmable interface to the quantum world

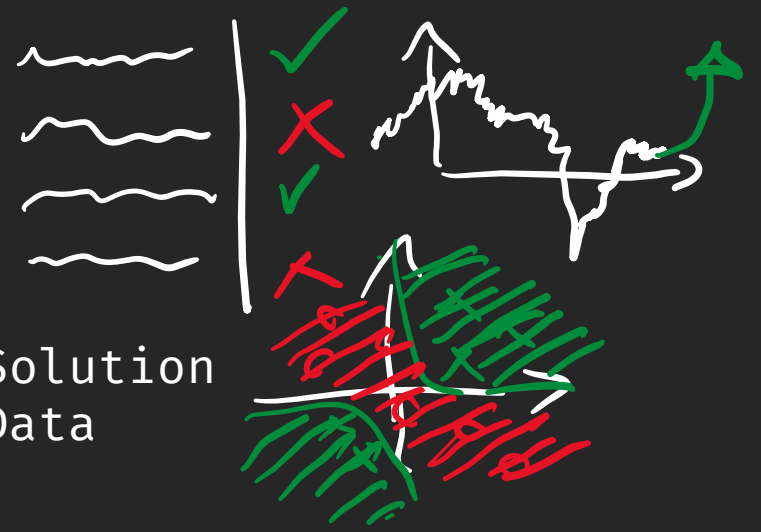
What is Quantum Computing?

- Quantum Computers as black-box devices used inside classical algorithms
  - Breaking your Bitcoin wallet (one day)
  - Combinatorial Optimization
  - Solutions of Linear/non-linear Systems
  - Complex Physics Simulations (e.g. fluid dynamics)
  - Machine Learning
- Quantum Computers as a programmable interface to the quantum world



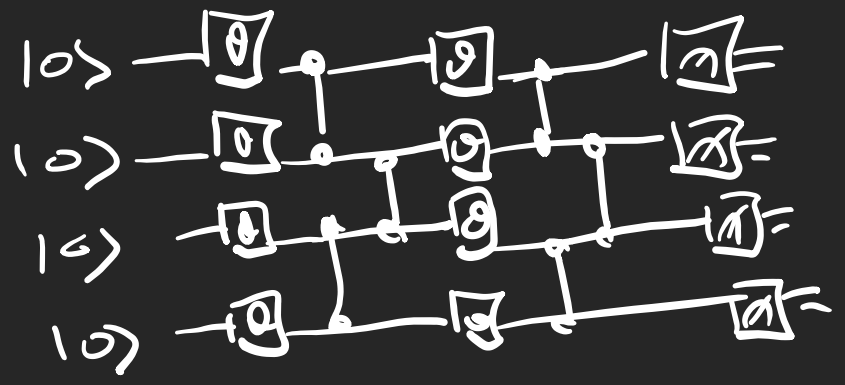
Problem Data

Iterate (?)

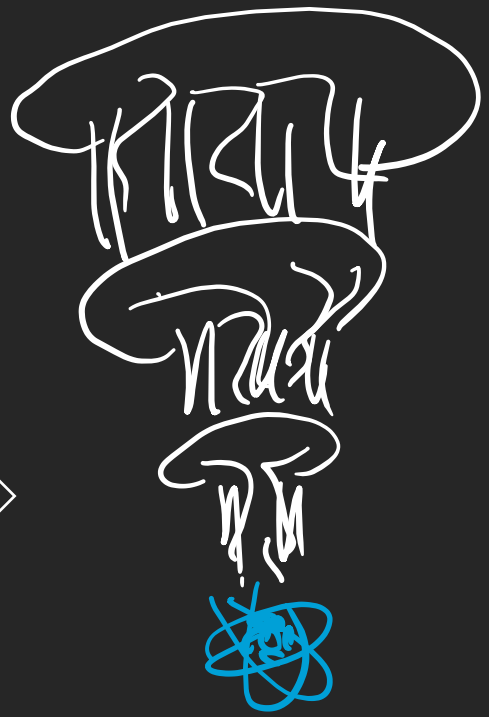


Solution Data

Pre-processing and/or encoding

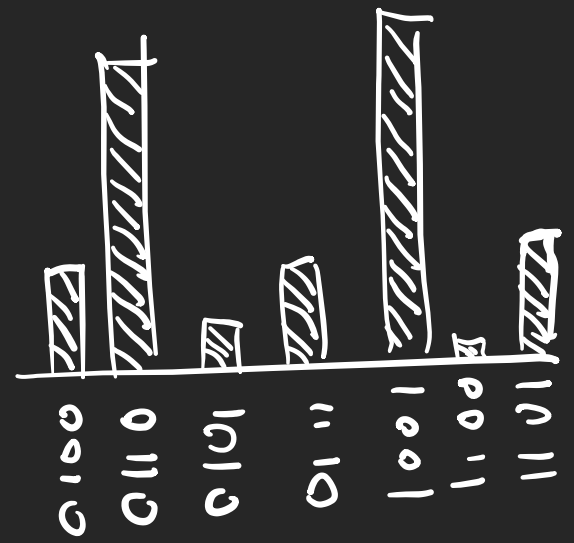


Quantum Circuit(s)



Quantum Processing Unit

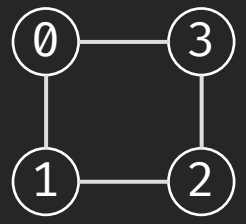
Post-processing and/or decoding



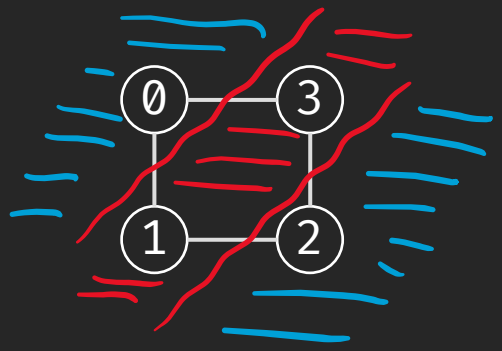
Measurement Samples

Example: max-cut with QAOA

Graph

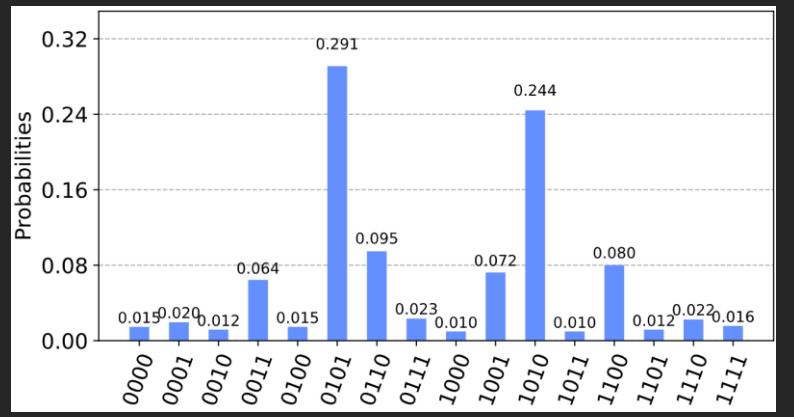


Graph Cut



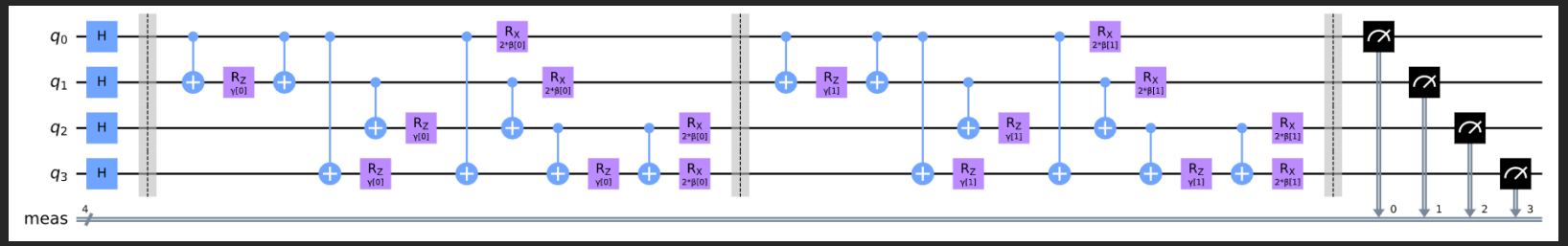
Max count solution

Samples

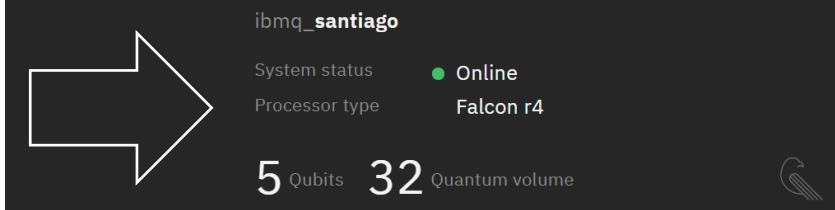


Encoding using 2-qubit gates

Parameter optimisation



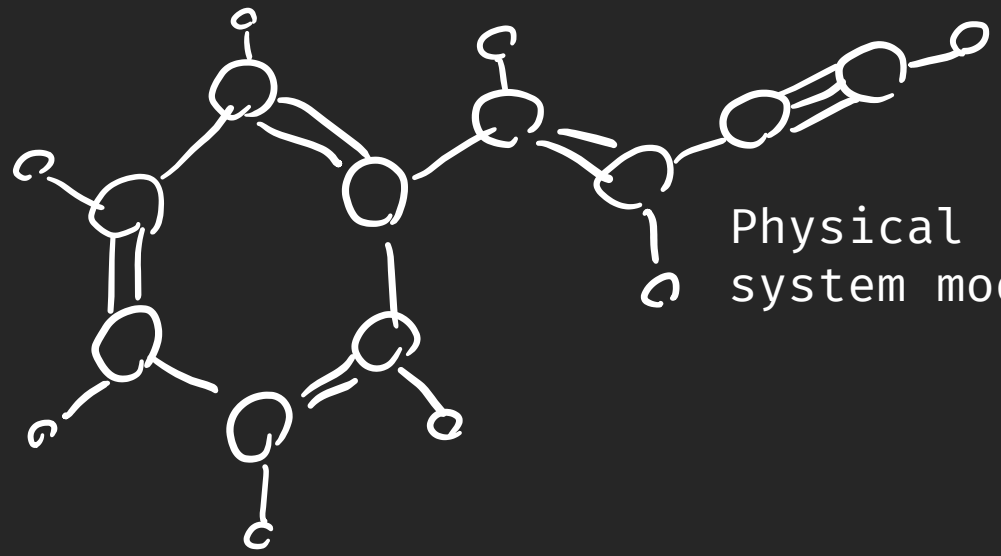
QAOA Circuit (parametric)



QPU

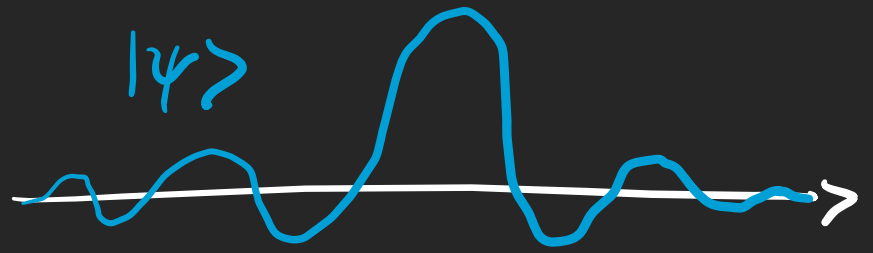
What is Quantum Computing?

- Quantum Computers as black-box devices used inside classical algorithms
- Quantum Computers as a programmable interface to the quantum world
  - Quantum Chemistry Simulations
  - Quantum Physics Simulations
  - Quantum Information/Communication/Cryptography
  - Quantum Metrology
  - Programmable Quantum Experiments



Physical
system model

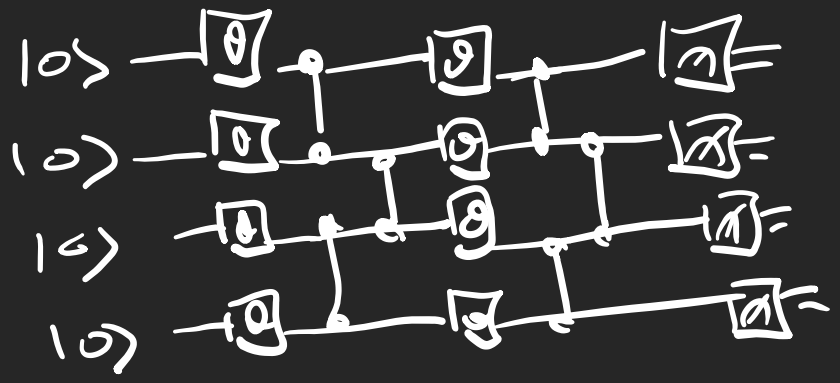
$$\psi^{(\alpha)}(x,t) = \left(\frac{m\omega}{\pi\hbar}\right)^{1/4} \exp\left(-\frac{m\omega}{2\hbar}\left(x - \sqrt{\frac{2\hbar}{m\omega}}\Re[\alpha(t)]\right)^2 + i\sqrt{\frac{2m\omega}{\hbar}}\Im[\alpha(t)]x + i\theta(t)\right)$$

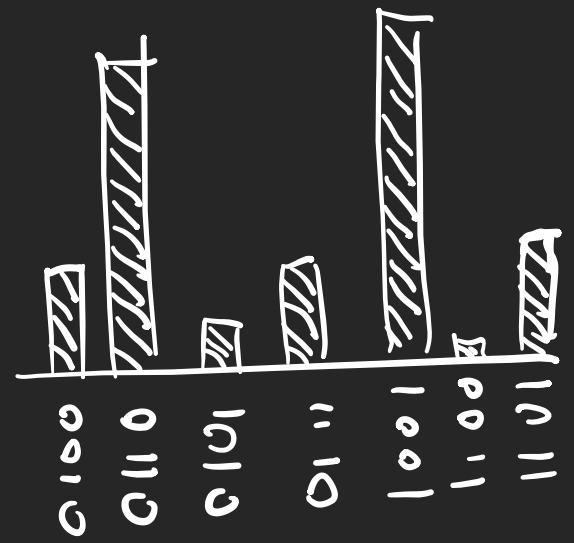


Quantum state

Encoding

Tomography





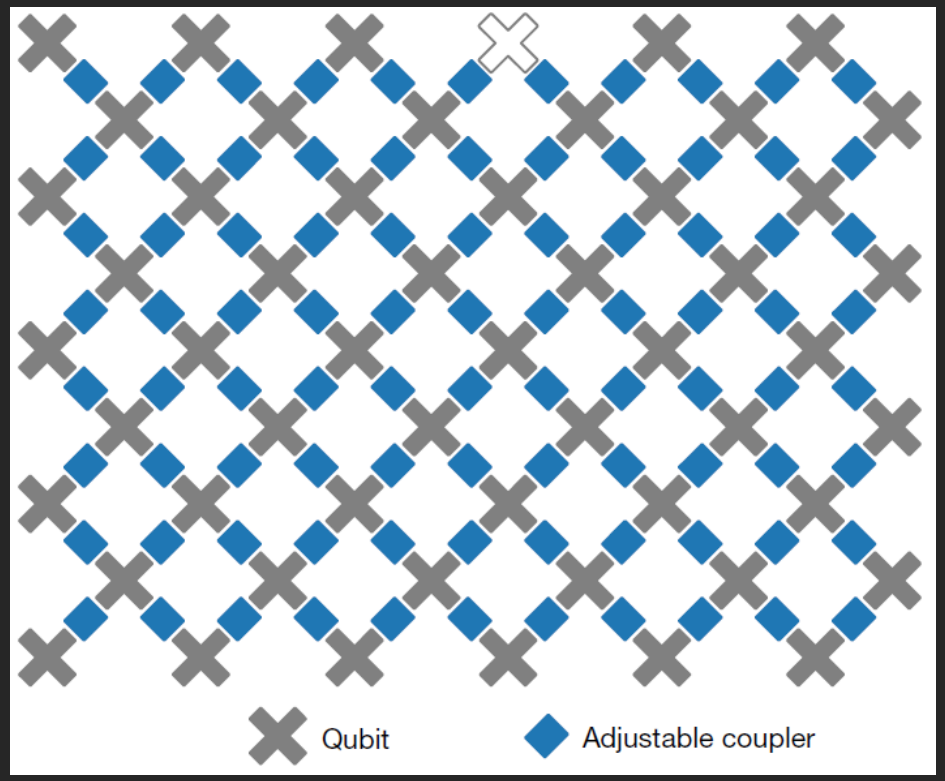
Quantum Circuits

Quantum Processing Unit

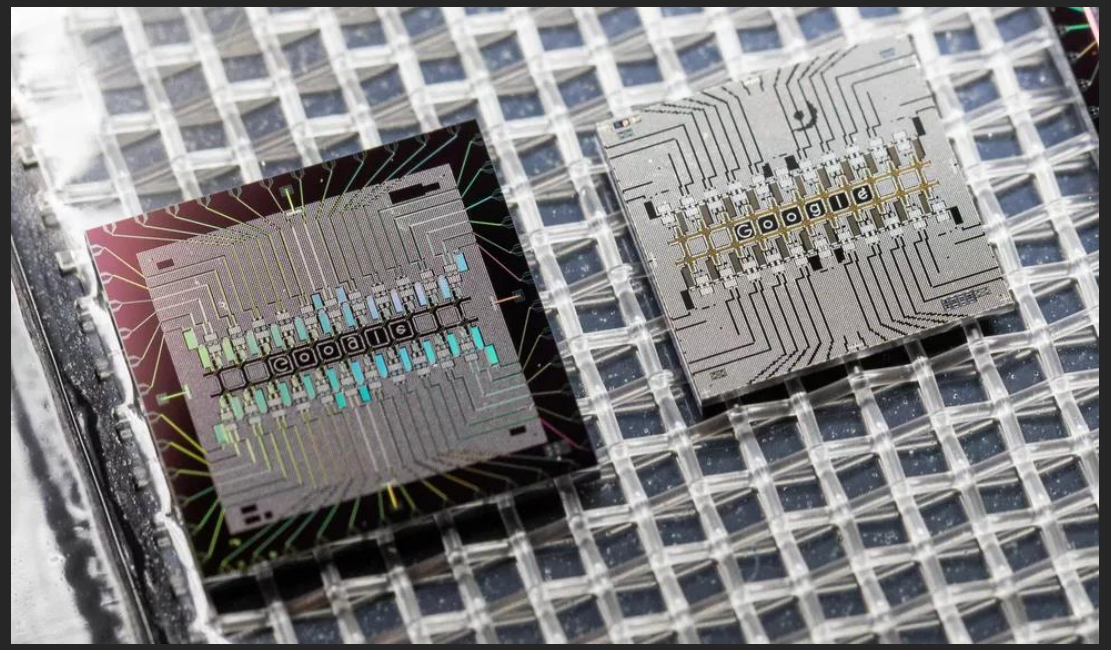
Measurement Samples

Hardware Providers

Google Sycamore (54 qubits)

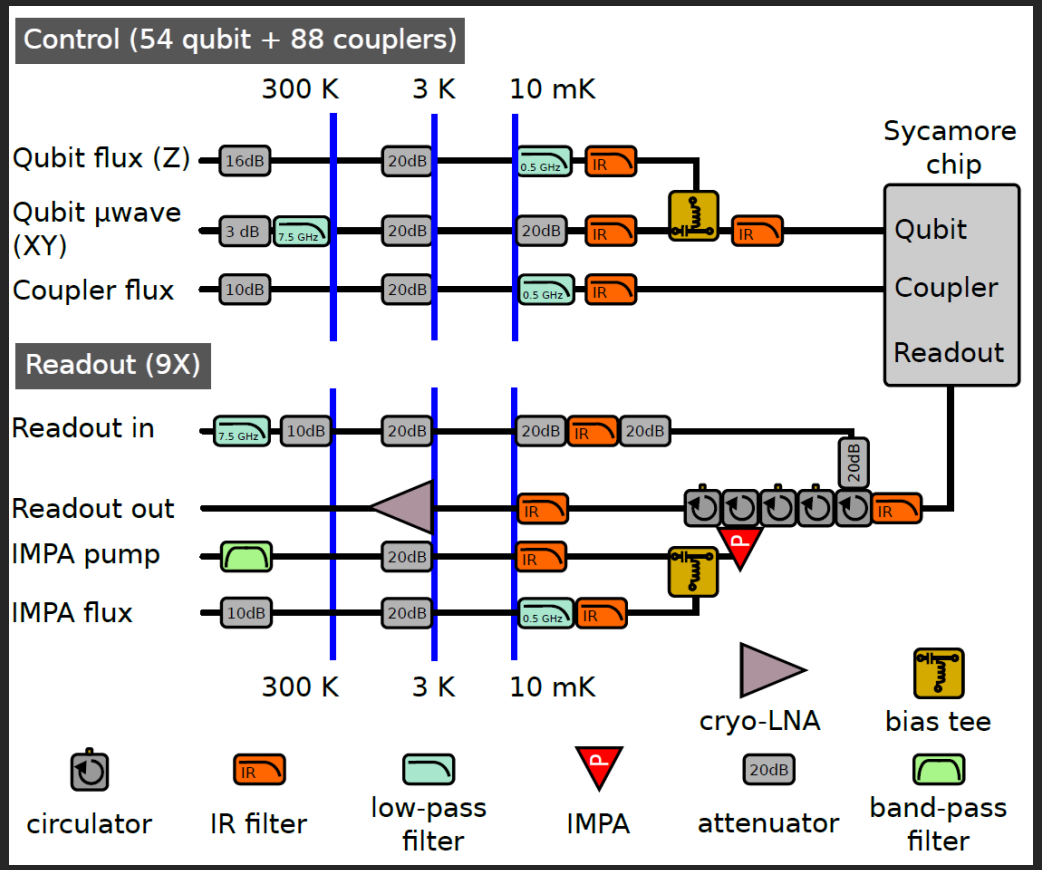


Credit: Nature/Google
https://doi.org/10.1038/s41586-019-1666-5

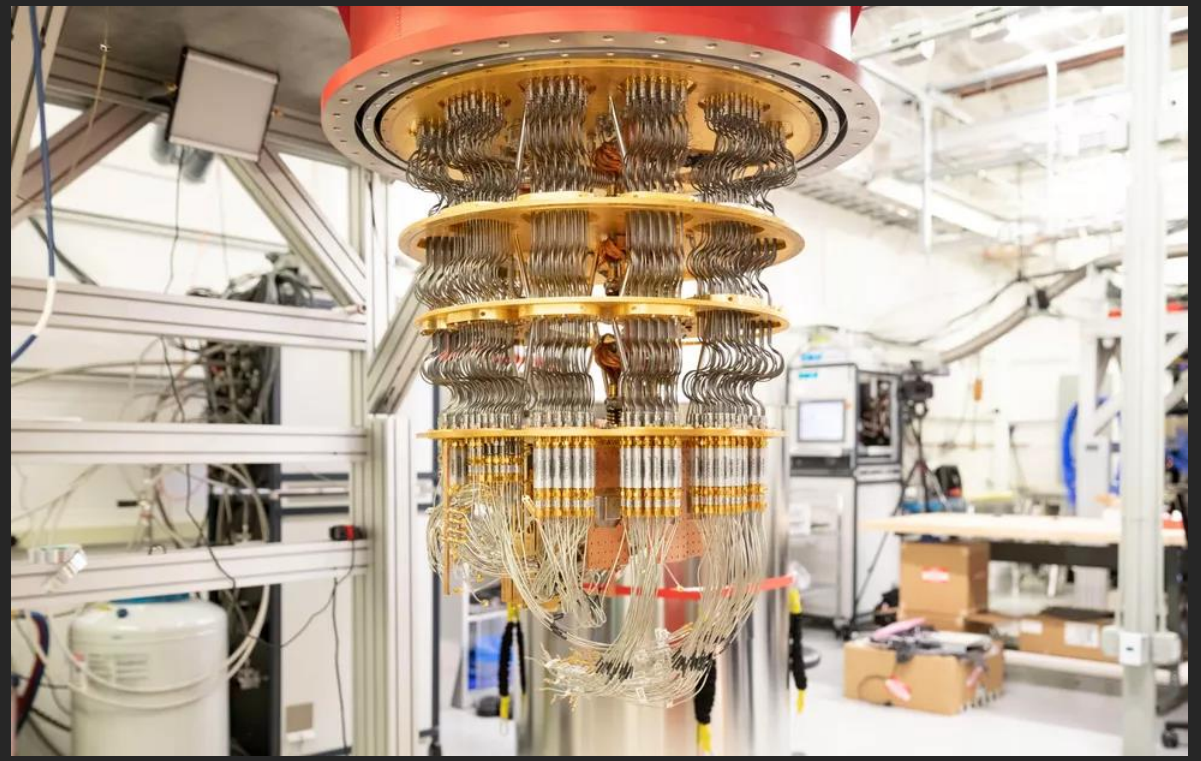


Credit: Stephen Shankland/CNET

Google Sycamore (54 qubits)

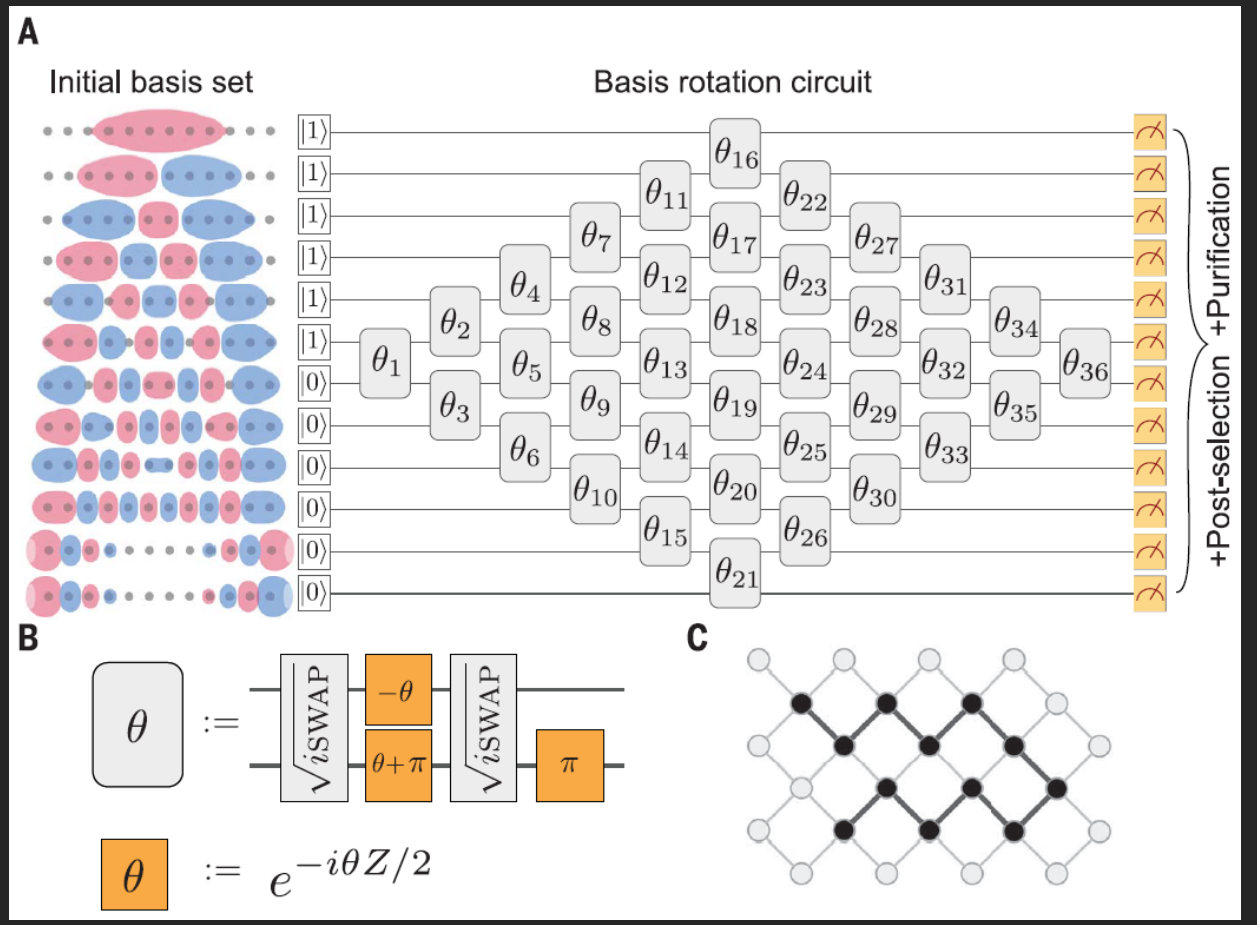


Credit: Nature/Google
https://doi.org/10.1038/s41586-019-1666-5



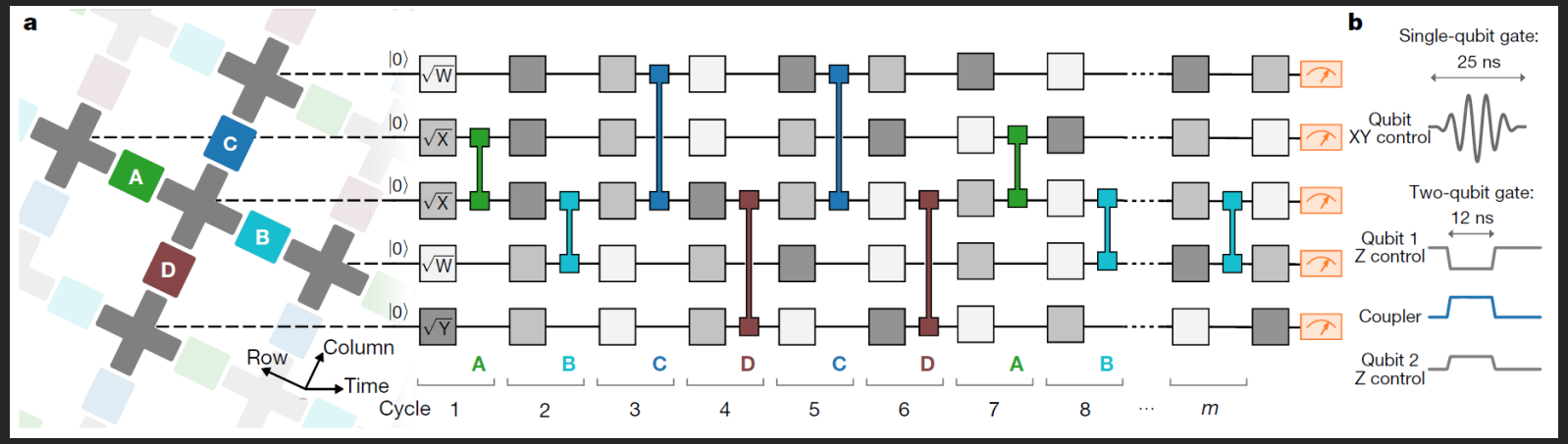
Credit: Stephen Shankland/CNET

Quantum Chemistry Simulations

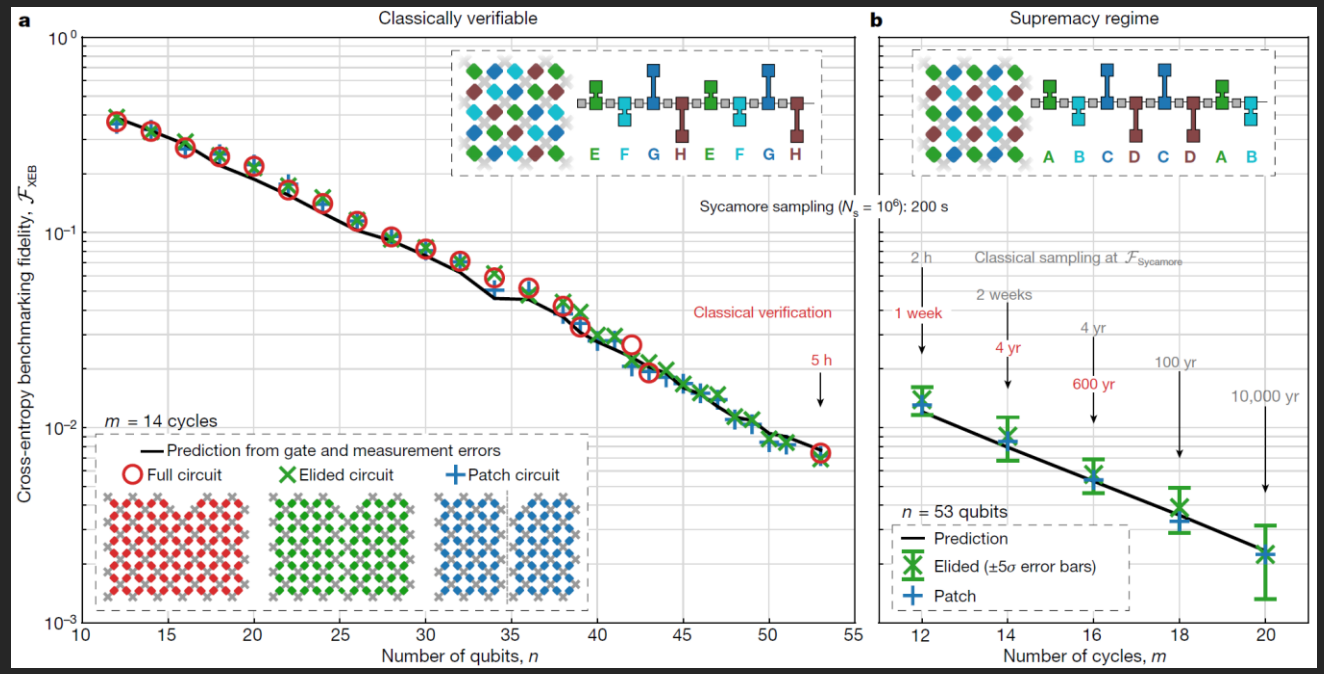


Credit: Science/Google
https://doi.org/10.1126/science.abb9811

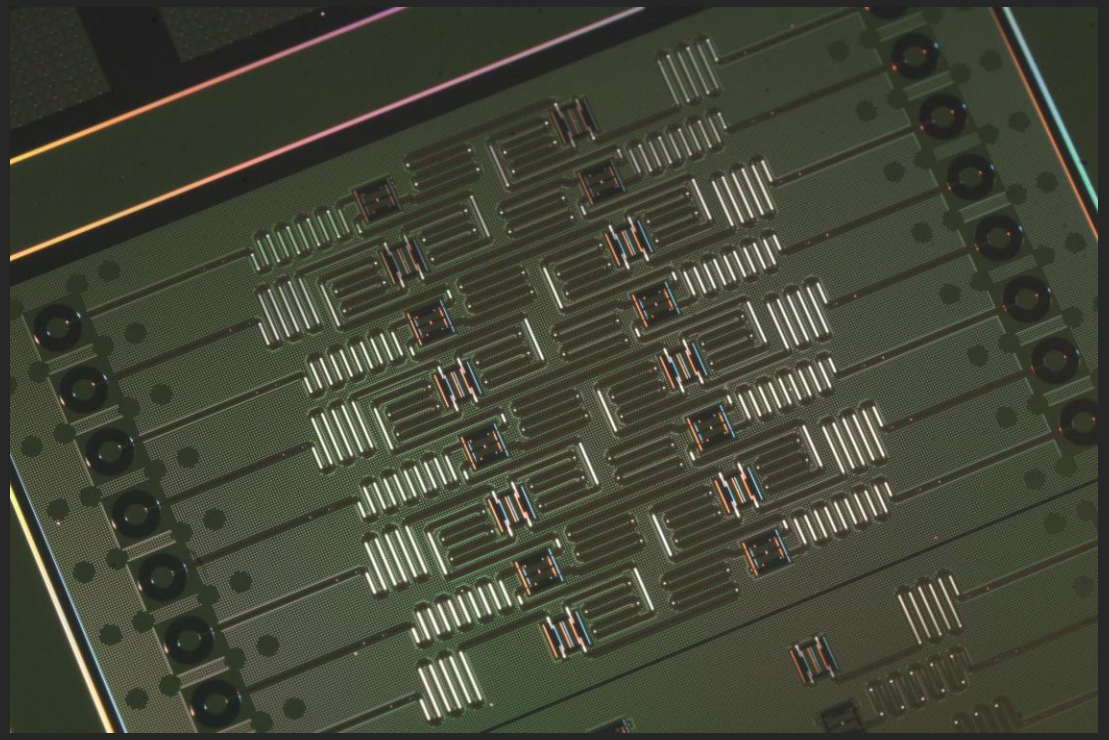
Randomized Benchmarking (XEB)



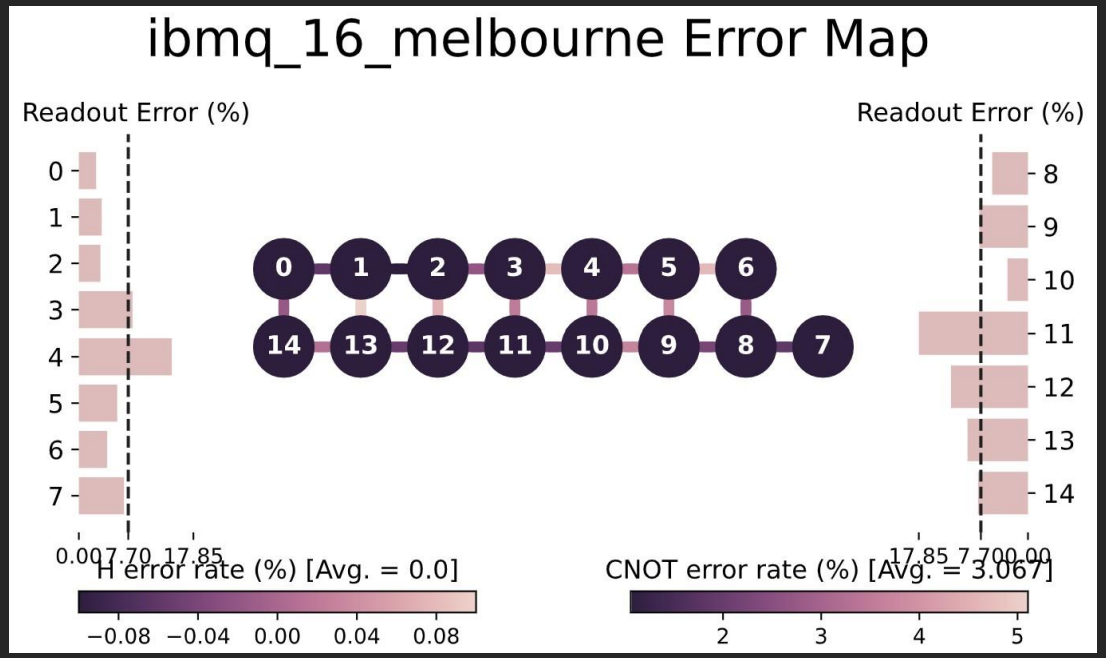
Credit: Nature/Google
https://doi.org/10.1038/s41586-019-1666-5



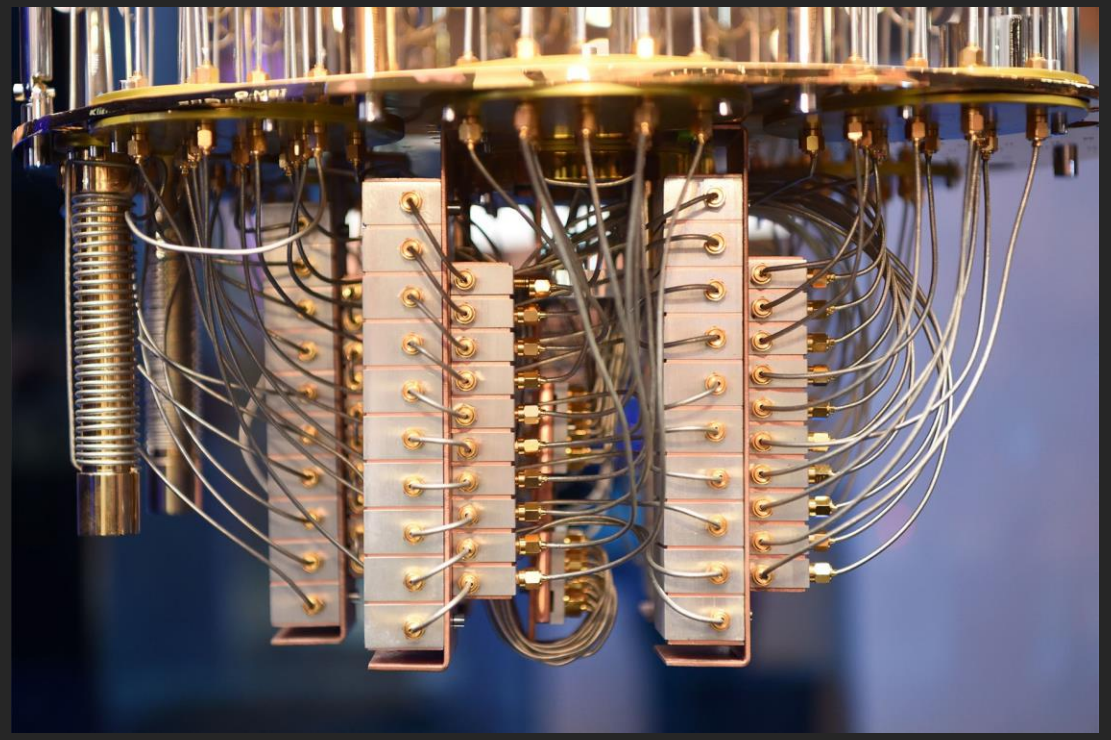
IBMQ Melbourne (16 qubits)



Credit: IBM

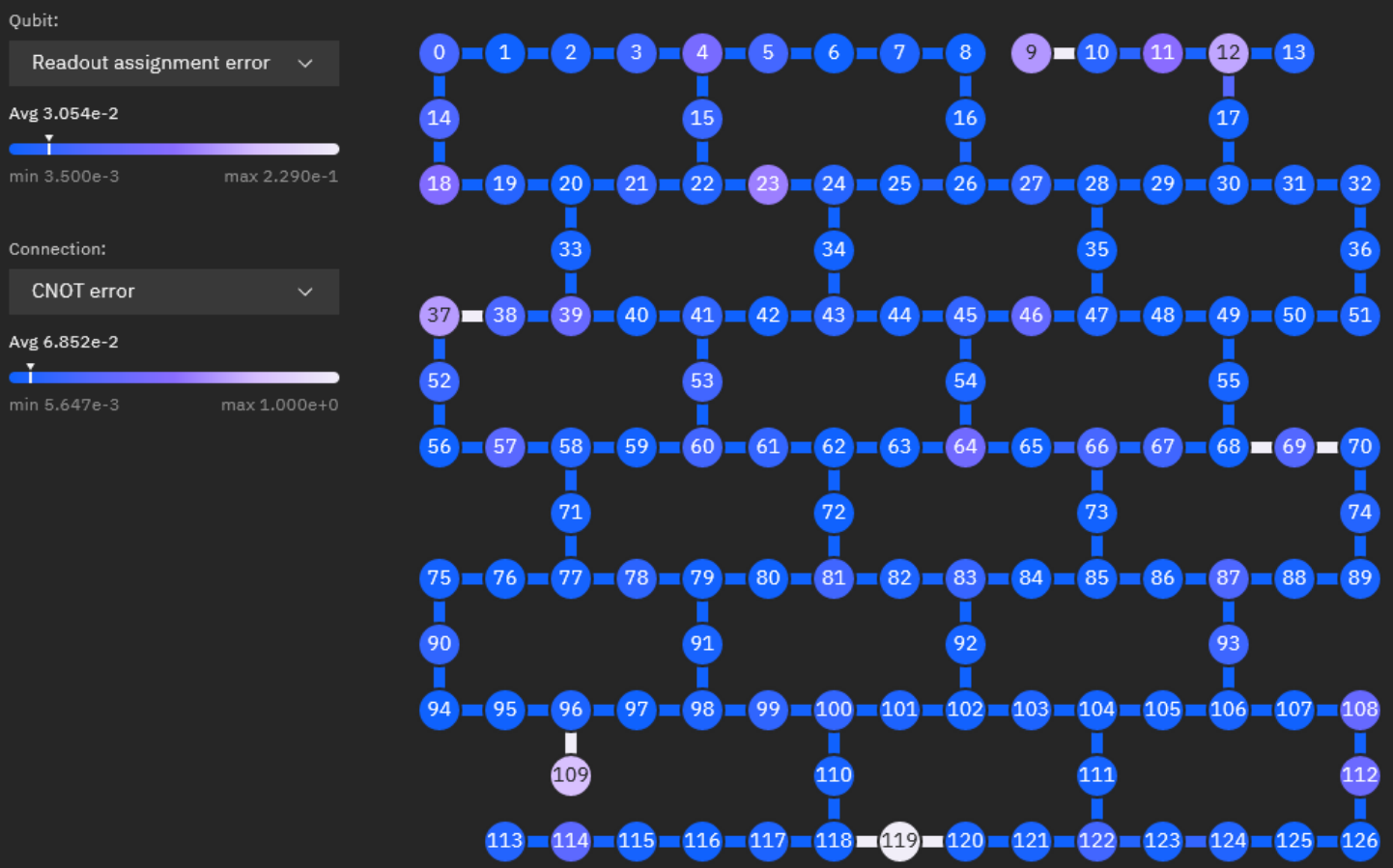


IBMQ System One (20 qubits)

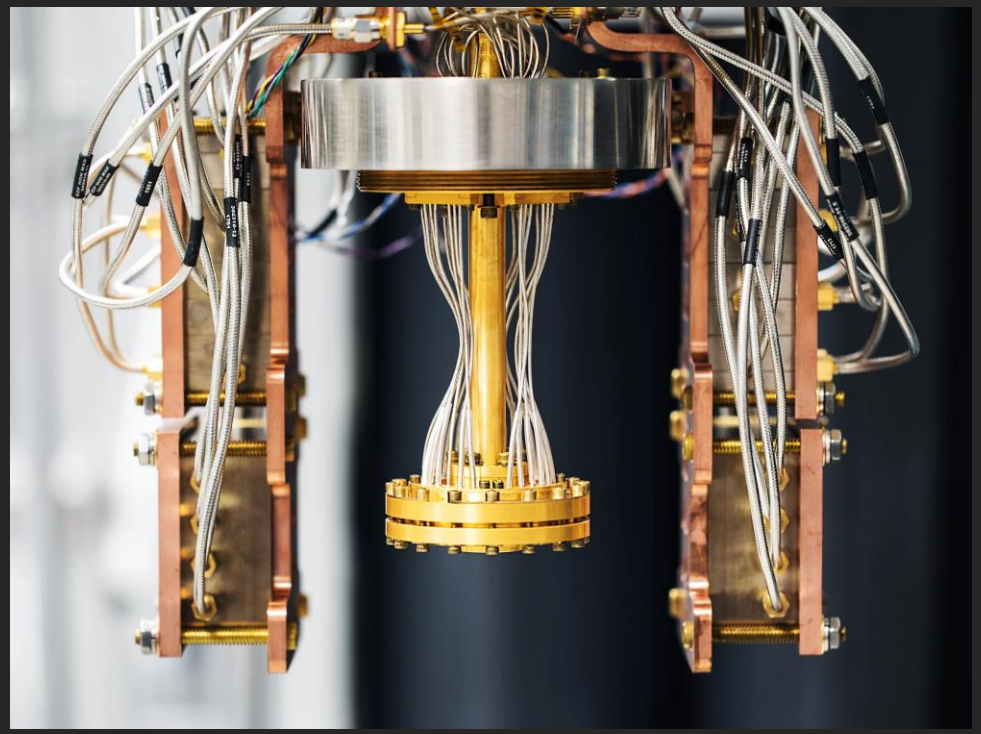


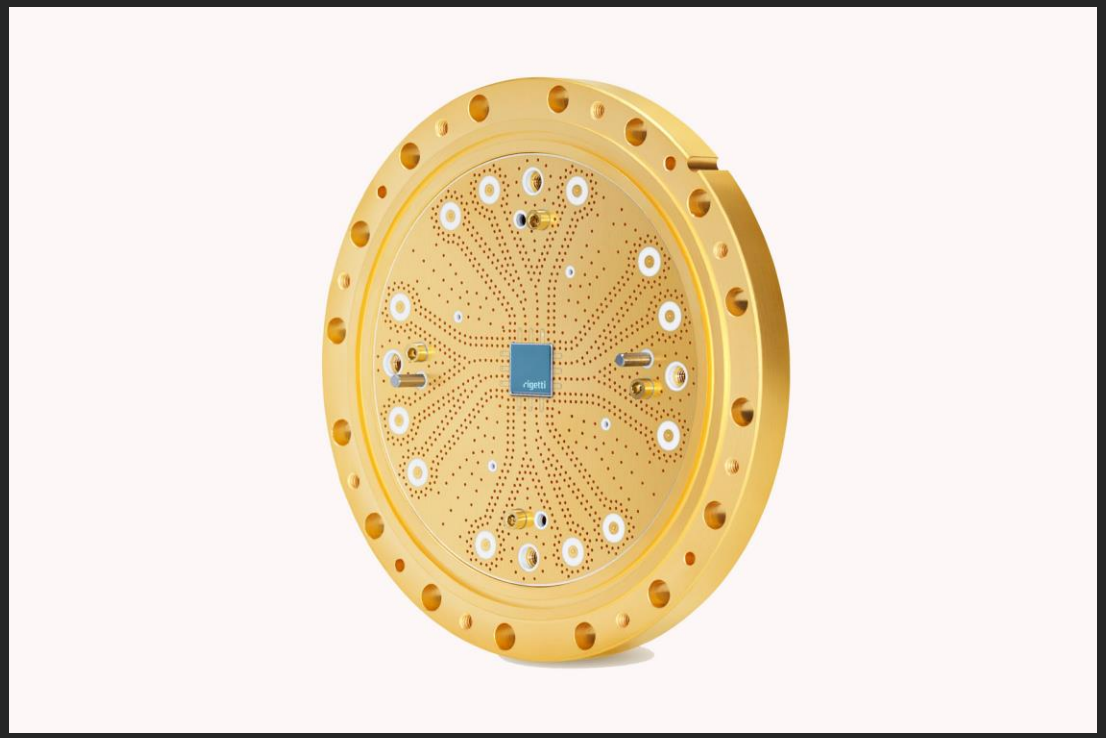
Credit: IBM

IBM Washington (127 qubits)



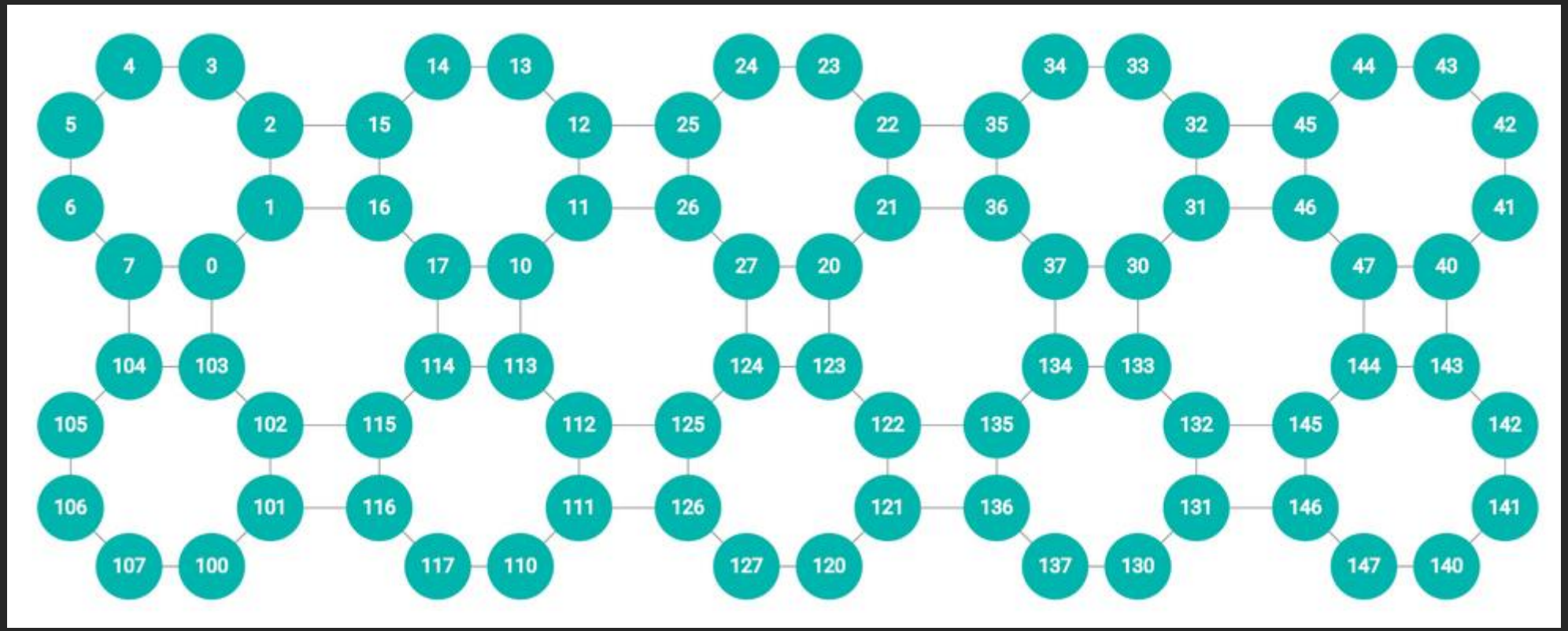
Rigetti Acorn (19 qubits)





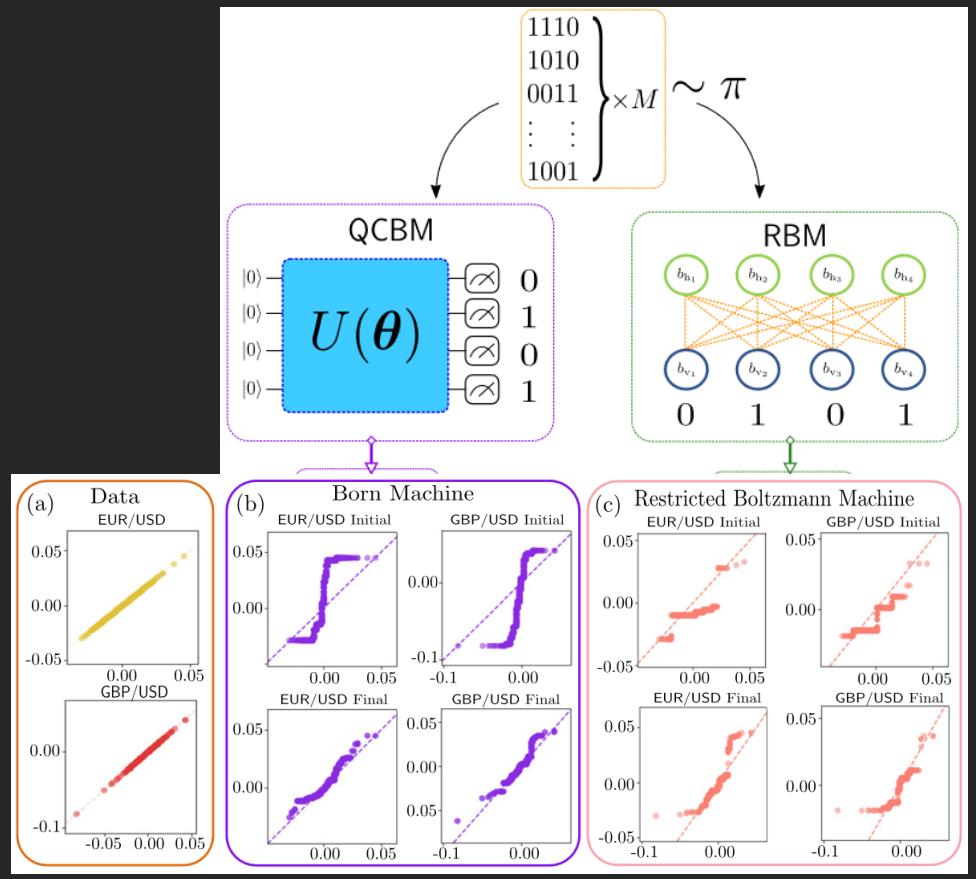
Credit: Rigetti

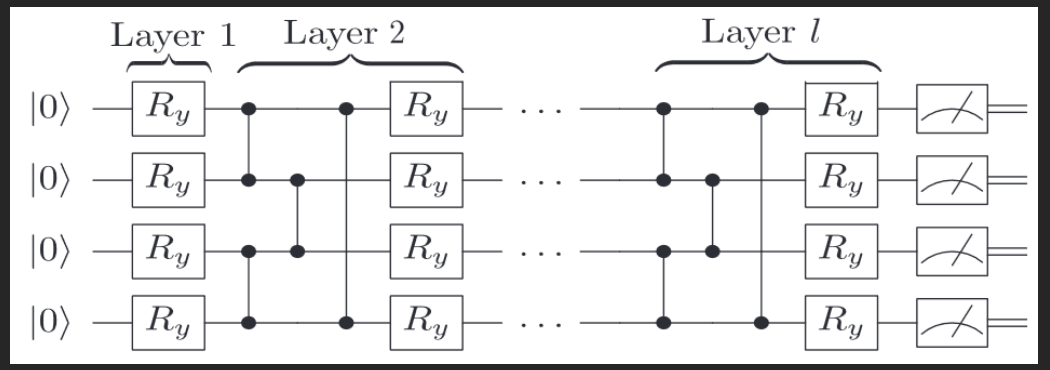
Rigetti Aspen-M (80 qubits)

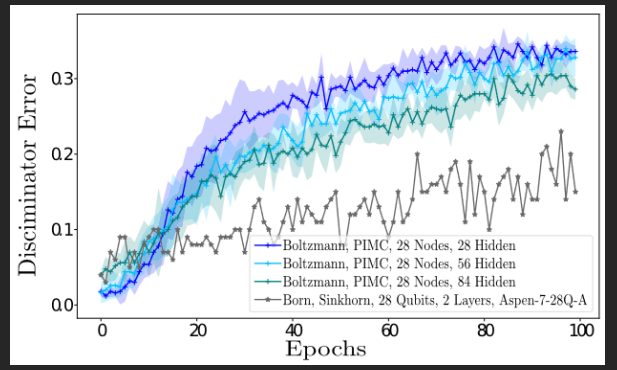


Credit: Amazon AWS/Rigetti

Generative Financial Modelling

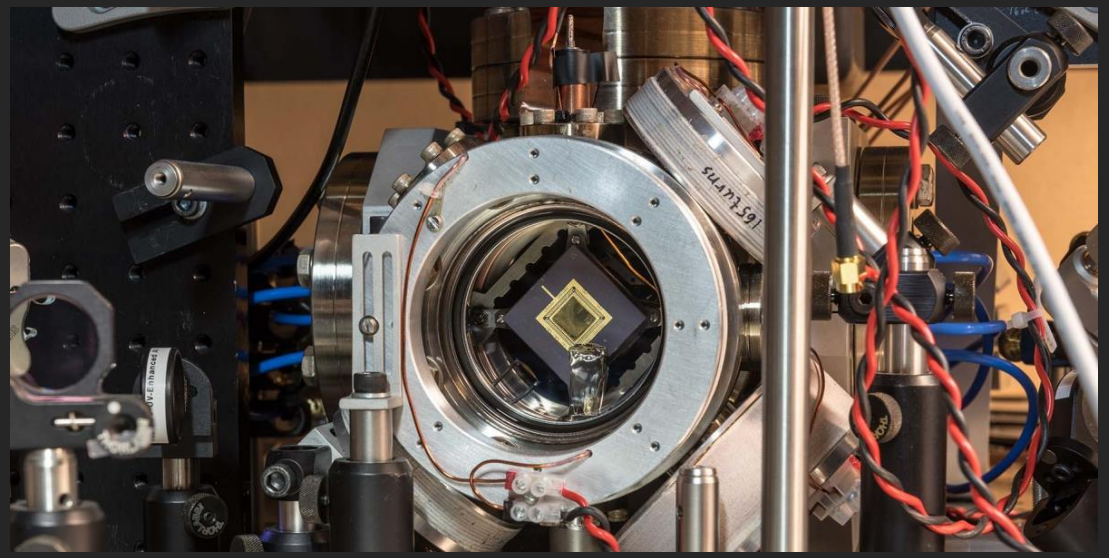




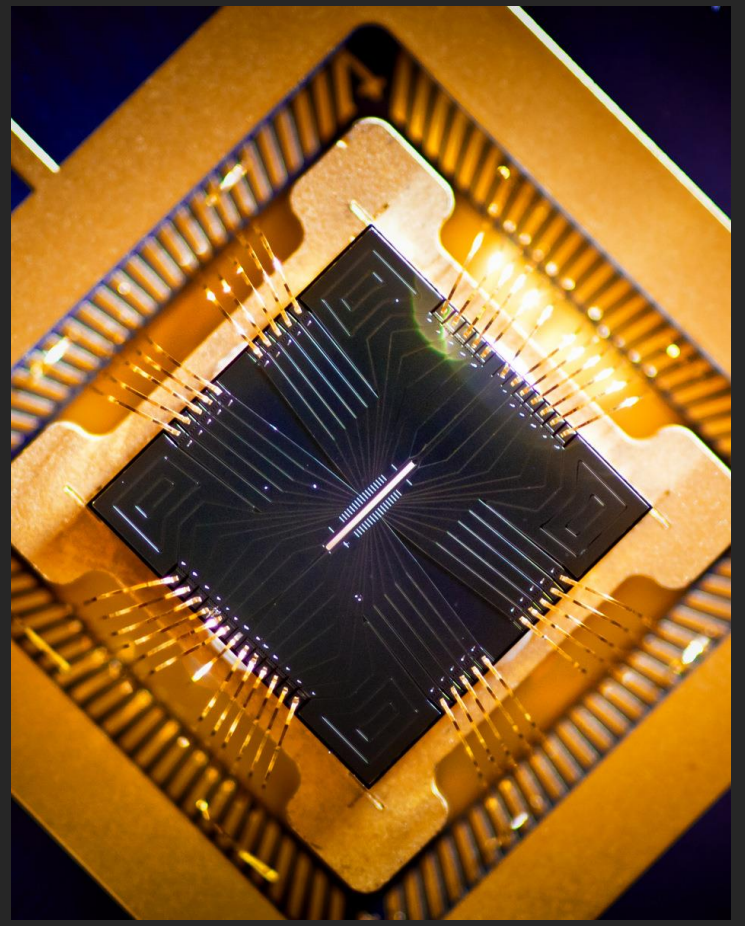


Credit: IOP/various institutions (on Rigetti HW)
https://doi.org/10.1088/2058-9565/abd3db

Ion-trap Quantum Computers

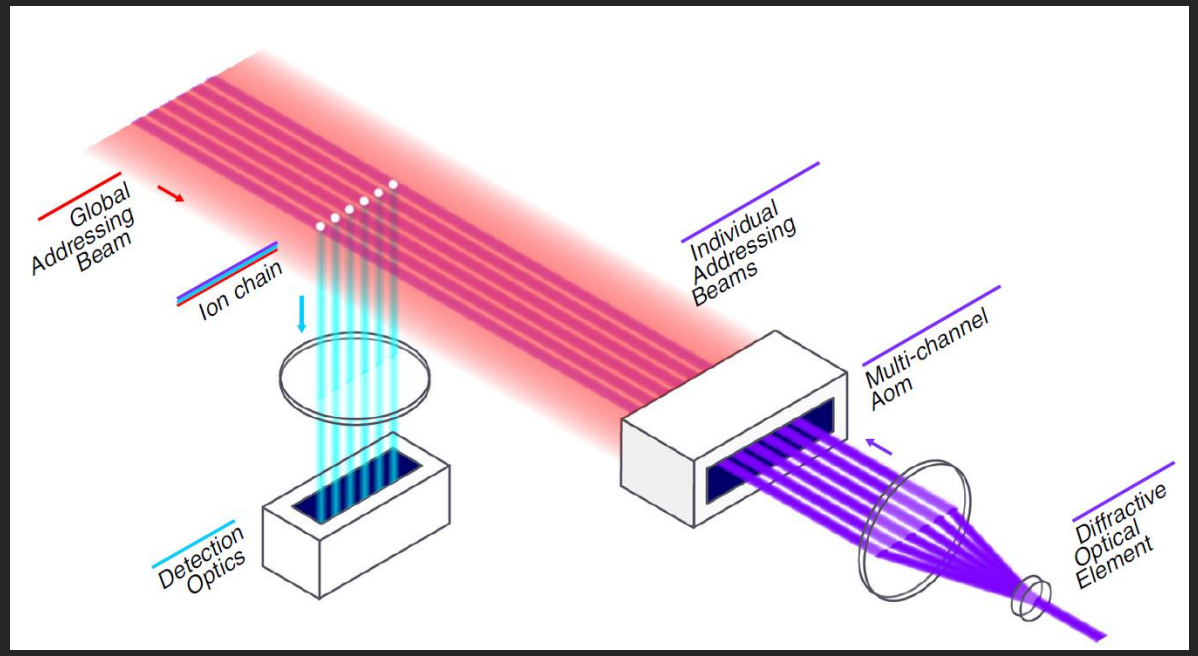


Credit: NQIT/Stuart Bebb

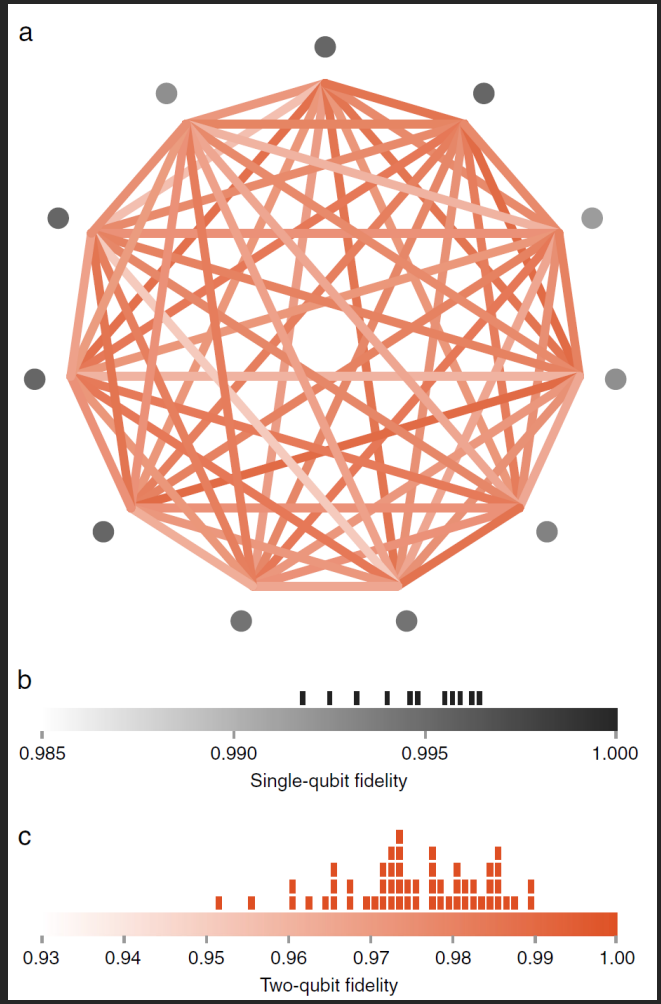


Credit: JQI

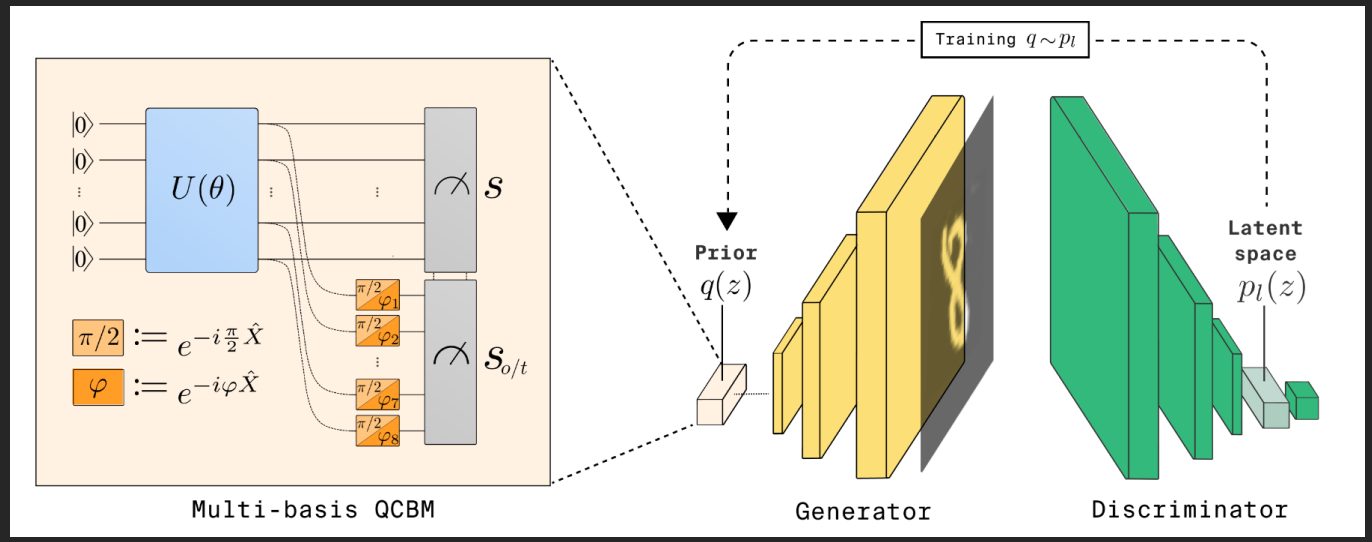
IonQ 171Yb+ (11 qubits)



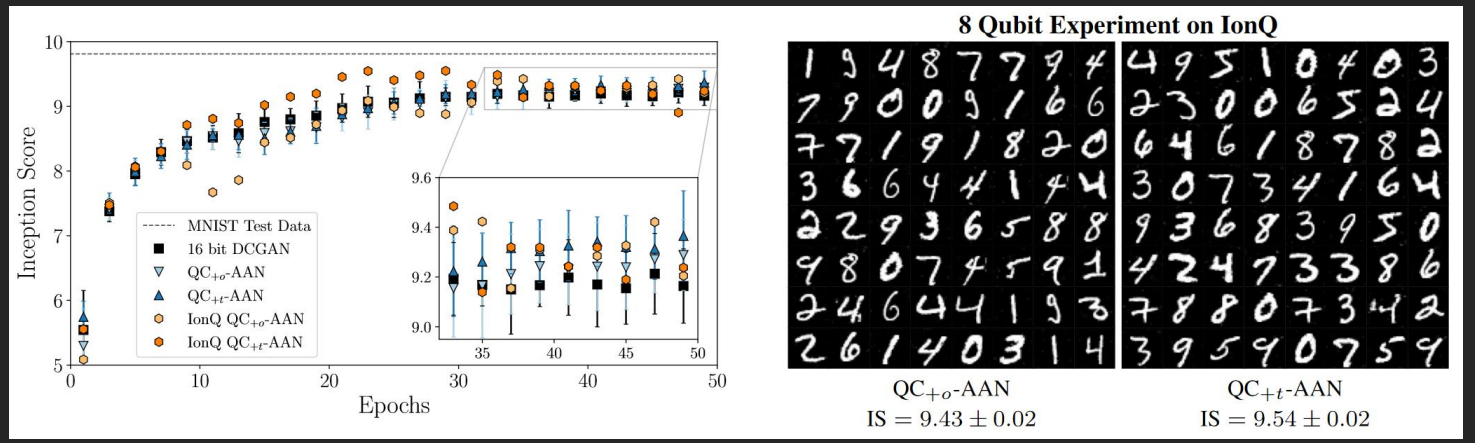
Credit: Nature/IonQ
https://doi.org/10.1038/s41467-019-13534-2



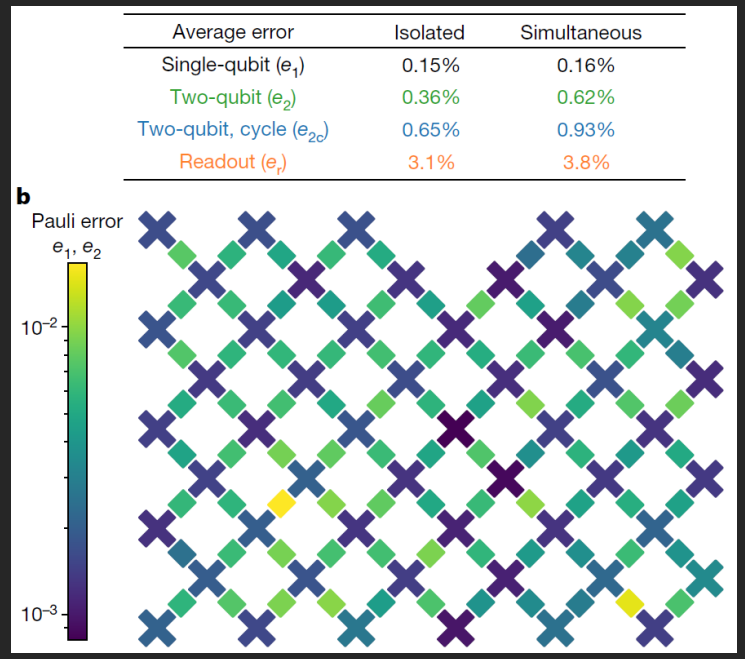
Quantum Generative Adversarial Networks



Credit: Zapata Computing/IonQ
https://arxiv.org/abs/2012.03924

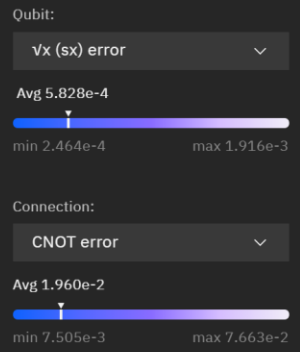


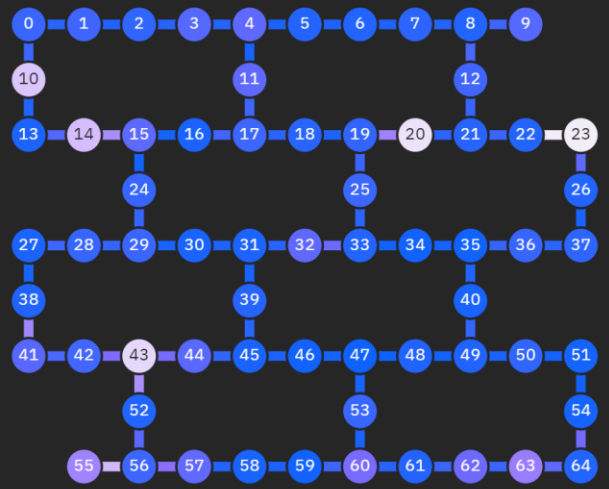
Superconducting vs Ion-trap



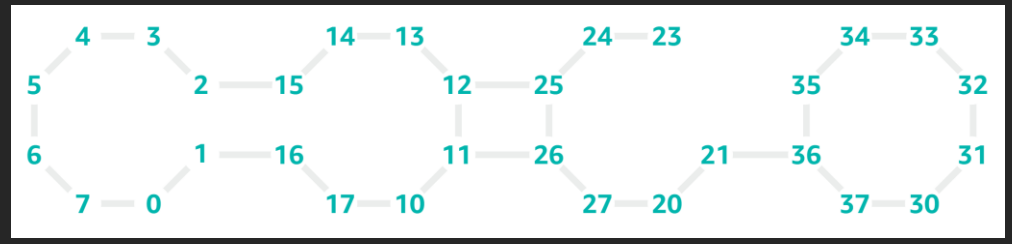
Average error	Isolated	Simultaneous
Single-qubit (e_1)	0.15%	0.16%
Two-qubit (e_2)	0.36%	0.62%
Two-qubit, cycle (e_{2c})	0.65%	0.93%
Readout (e_r)	3.1%	3.8%

Credit: Nature/Google
https://doi.org/10.1038/s41586-019-1666-5

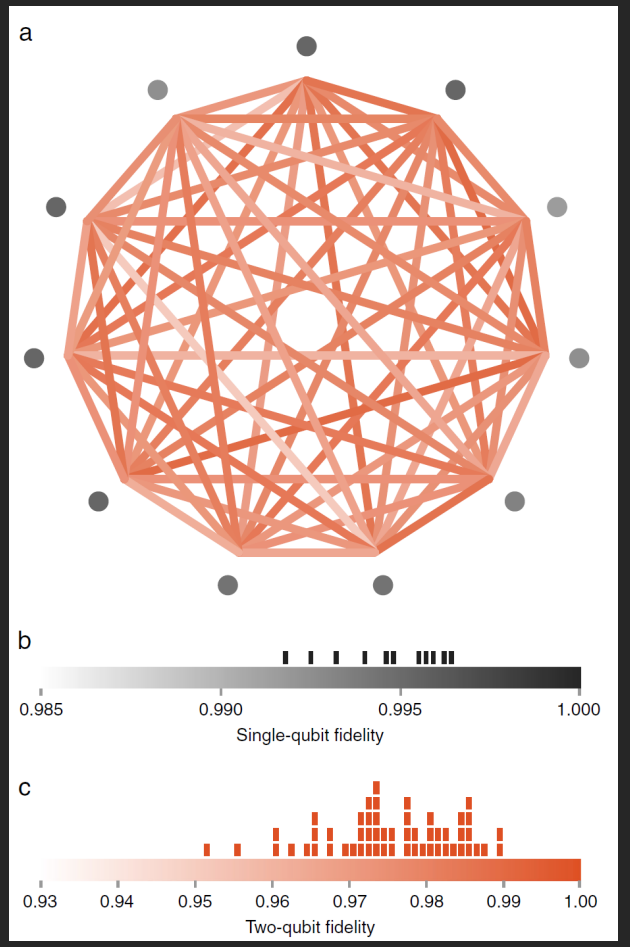




Credit:IBMQ

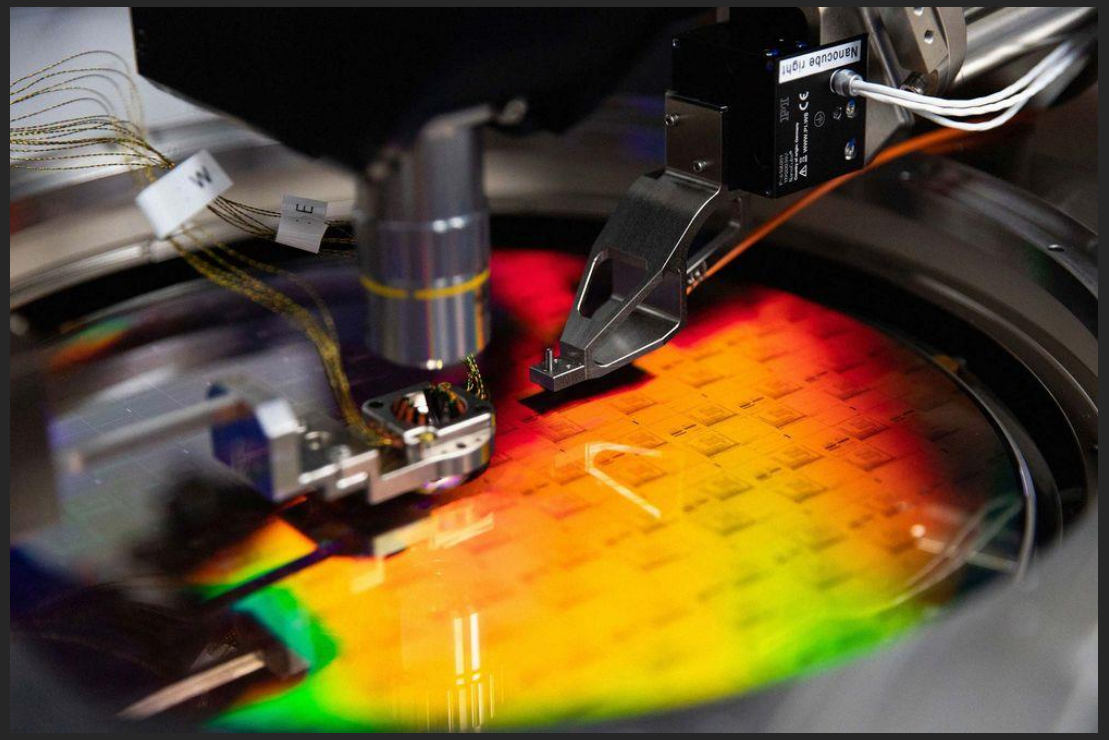


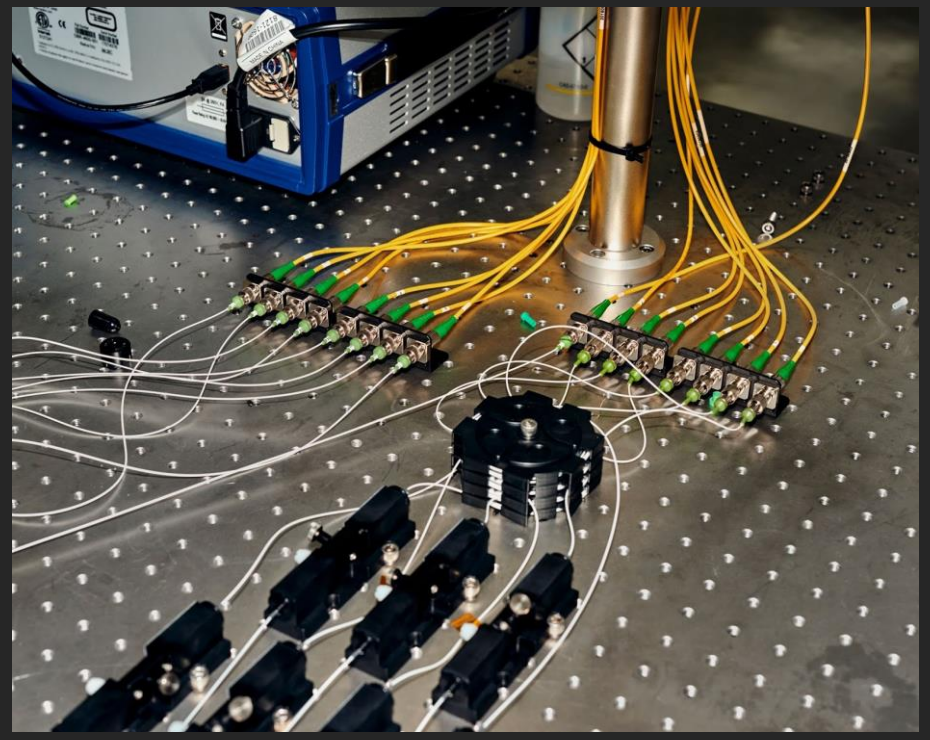
Credit: Amazon AWS/Rigetti



Credit: Nature/IonQ
https://doi.org/10.1038/s41467-019-13534-2

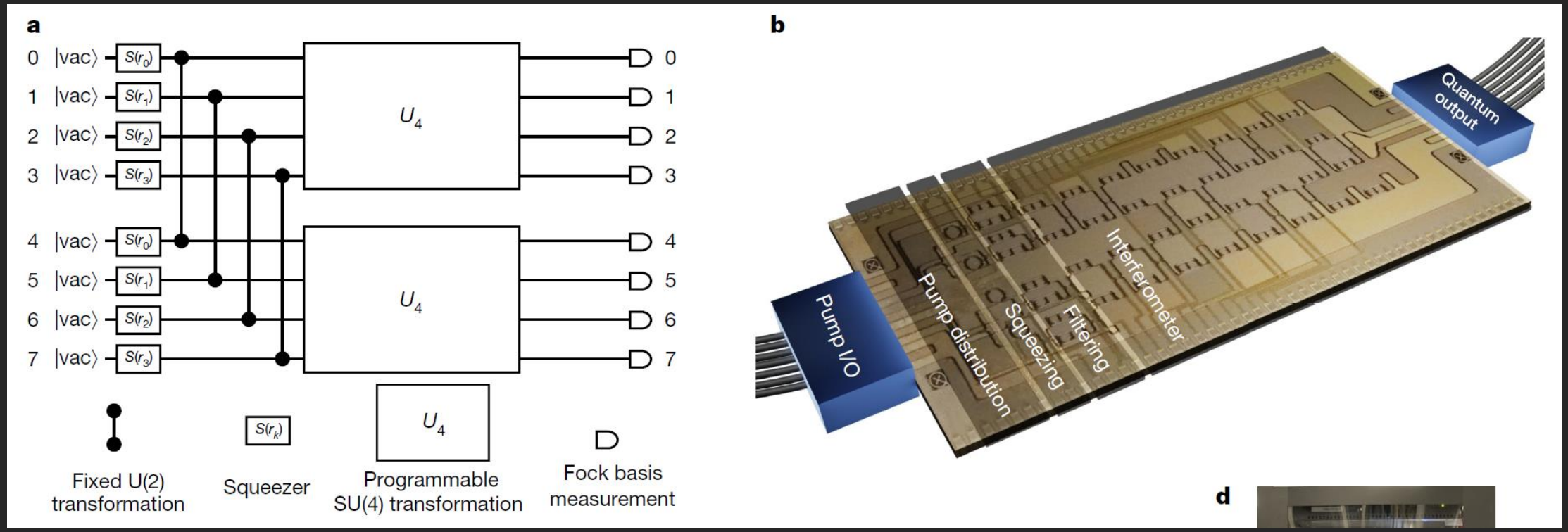
Photonic Quantum Computers





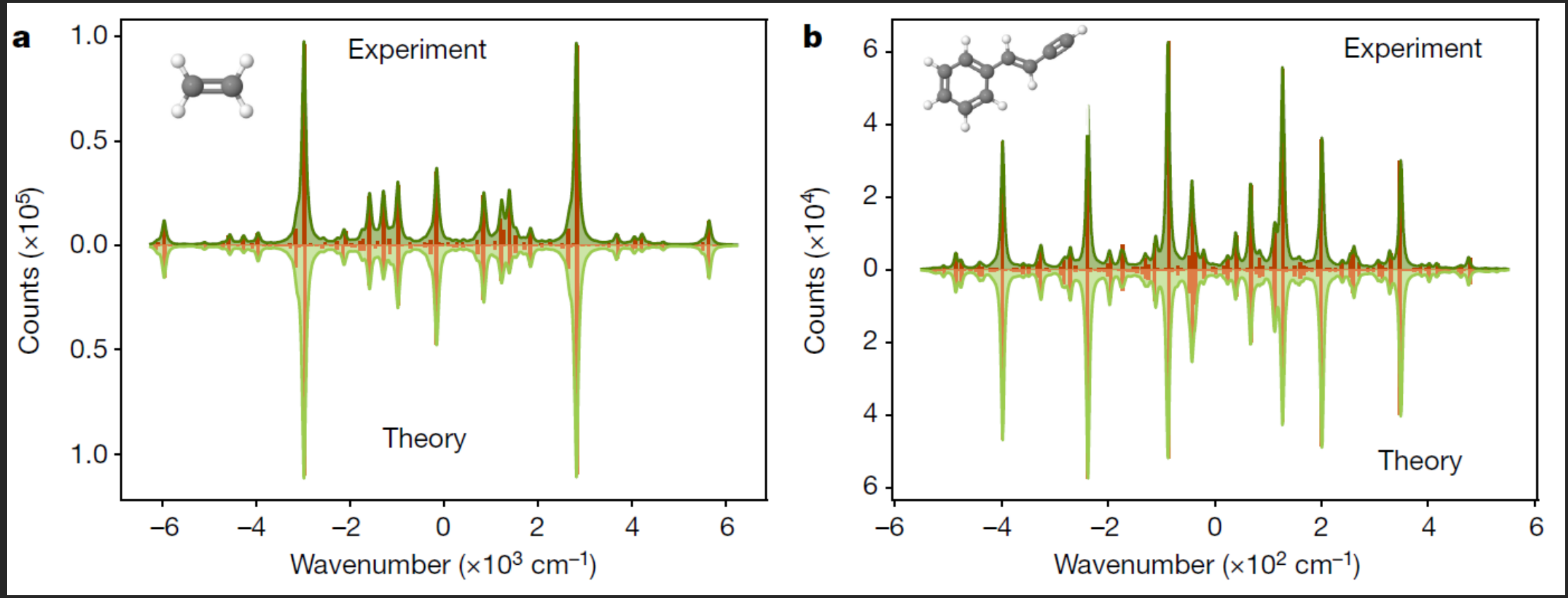
Credit: PsiQuantum

Xanadu 8-mode photonic chip



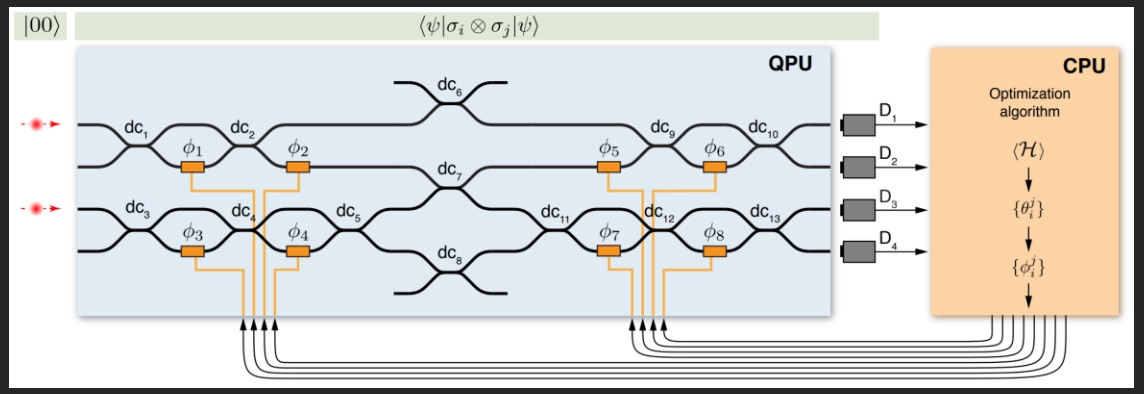
Credit: Nature/Xanadu
https://doi.org/10.1038/s41586-021-03202-1

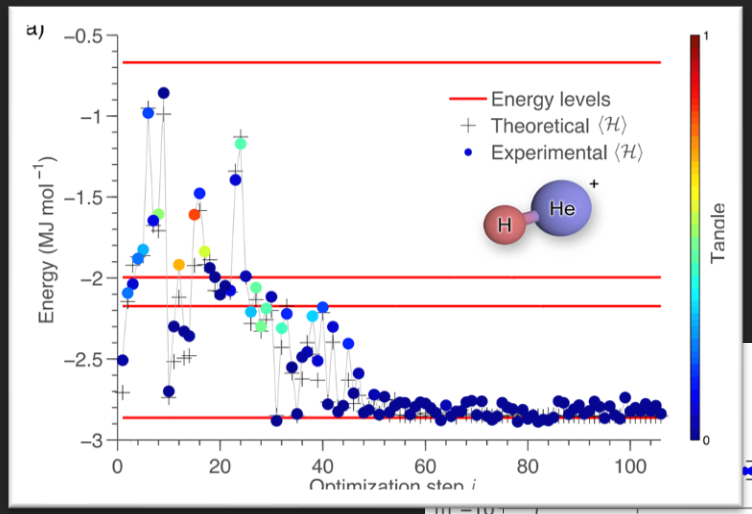
Quantum Chemistry Simulations

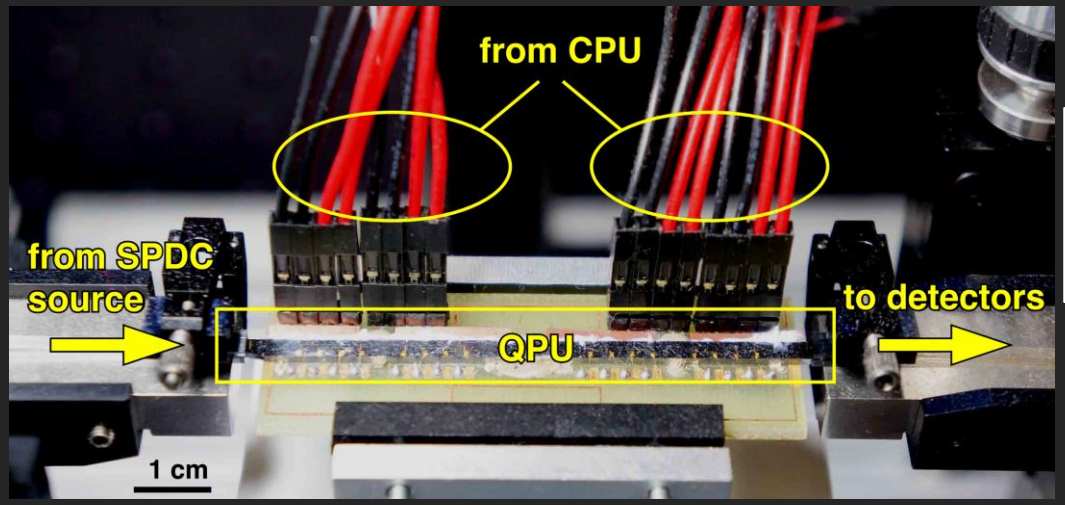


Credit: Nature/Xanadu
https://doi.org/10.1038/s41586-021-03202-1

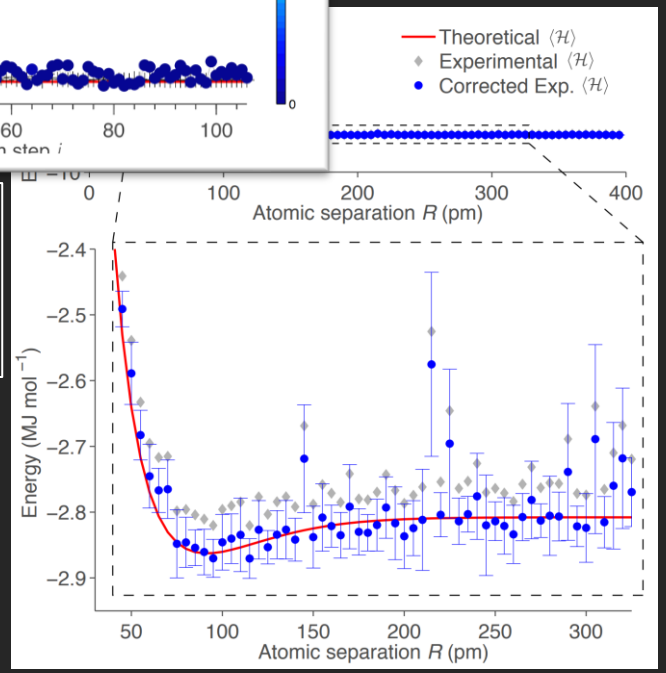
Quantum Chemistry Simulations





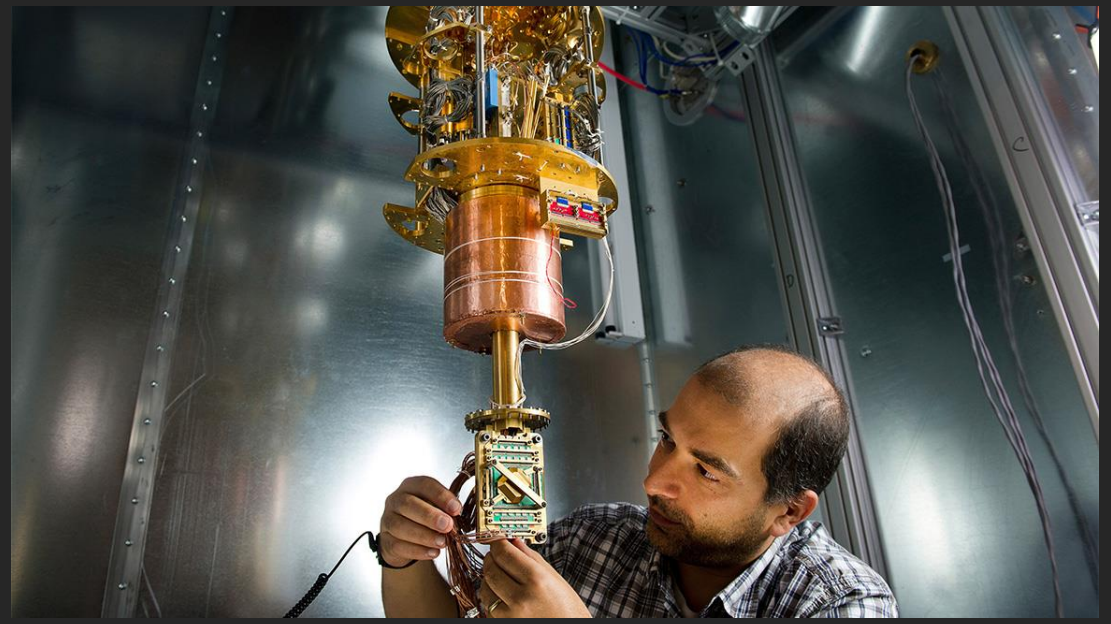


This is a photonic QPU running VQE in 2003… Ancient QC history.



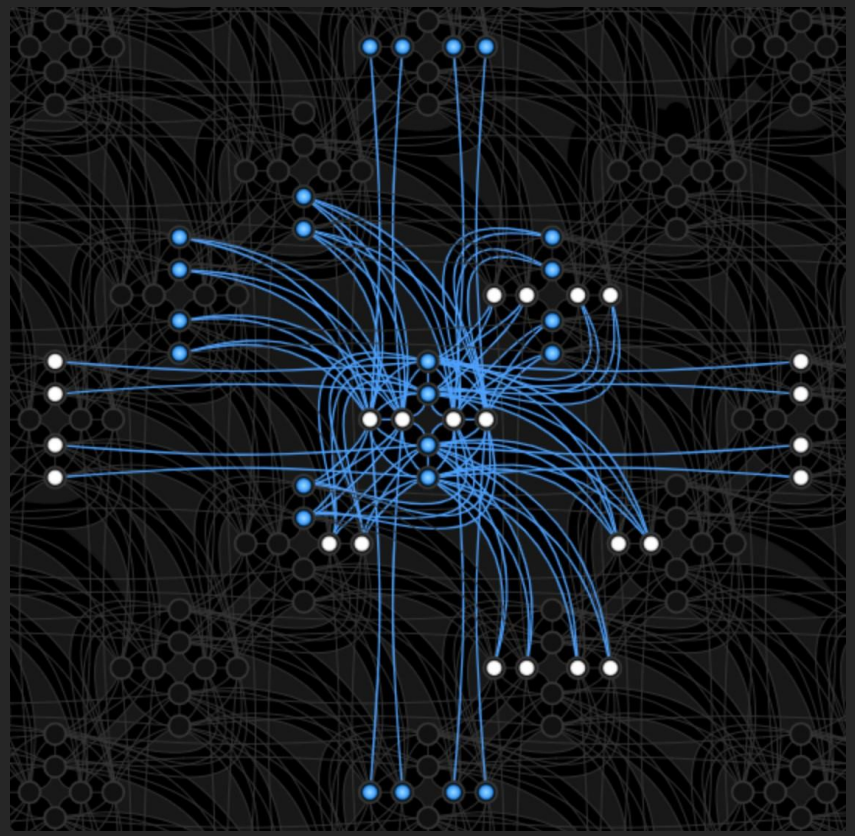
Credit: Nature/various institutions
https://doi.org/10.1038/ncomms5213

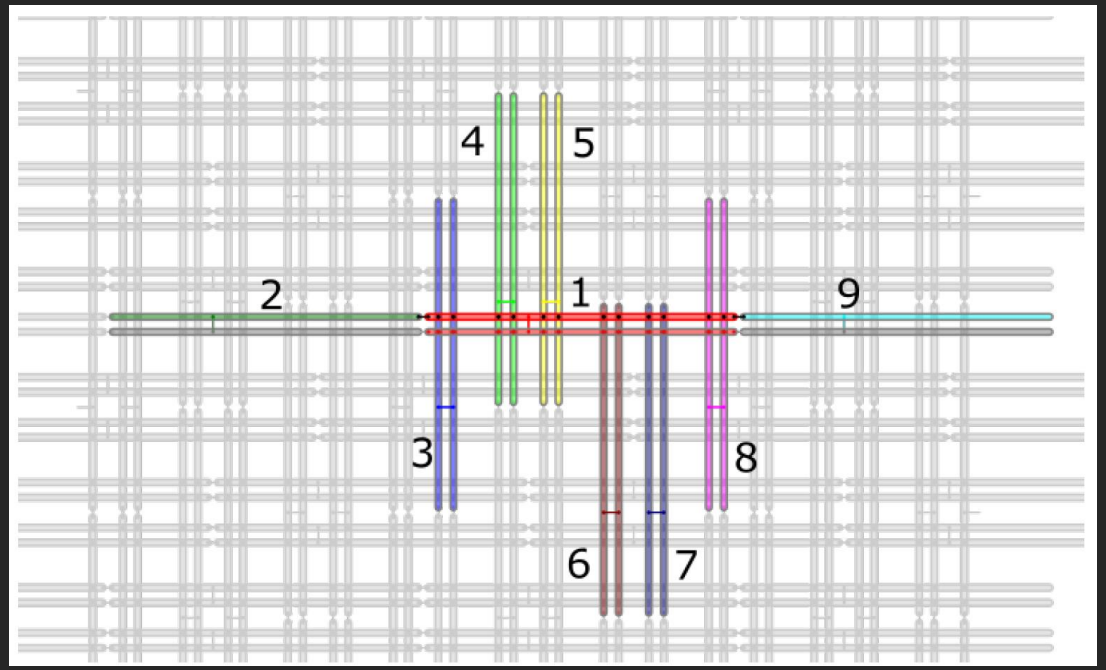
Quantum Annealers



Credit: D-Wave

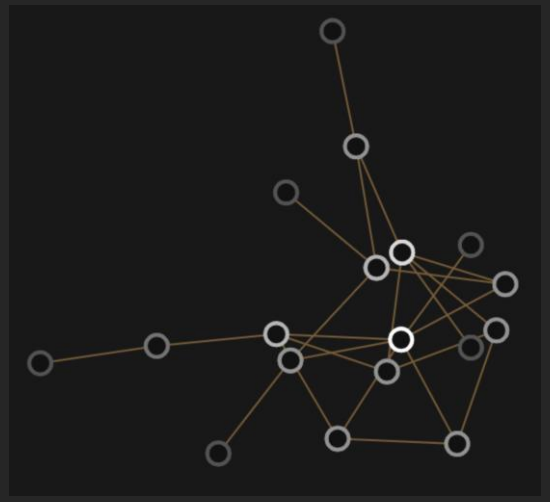
D-Wave Advantage (5760 qubits, Pegasus topology)



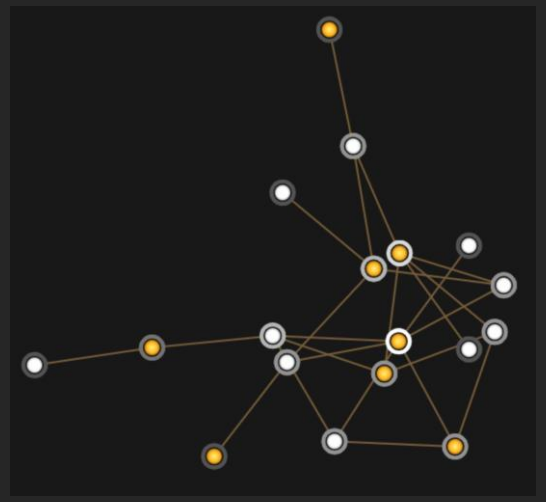


Credit: D-Wave

Example: Max-cut with QA

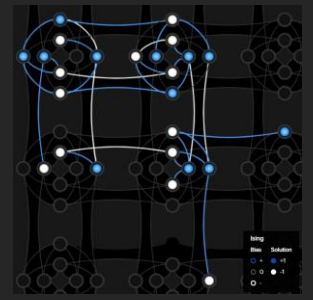


Graph

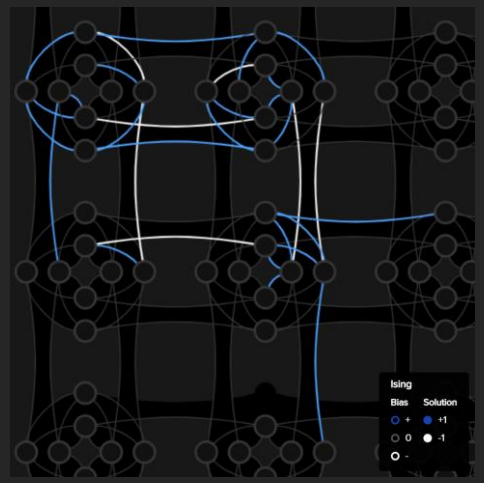


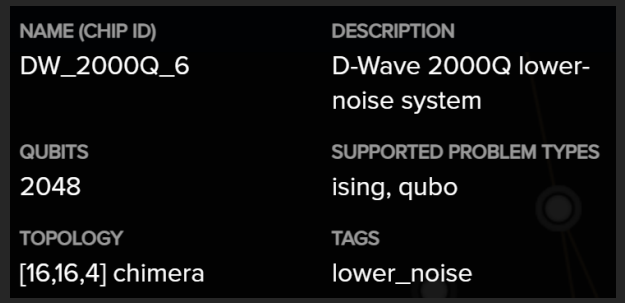
Graph Cut

Min energy solution

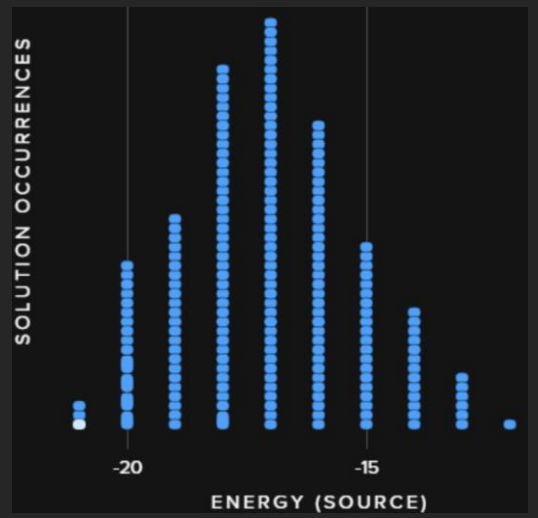


Encoding as Ising model



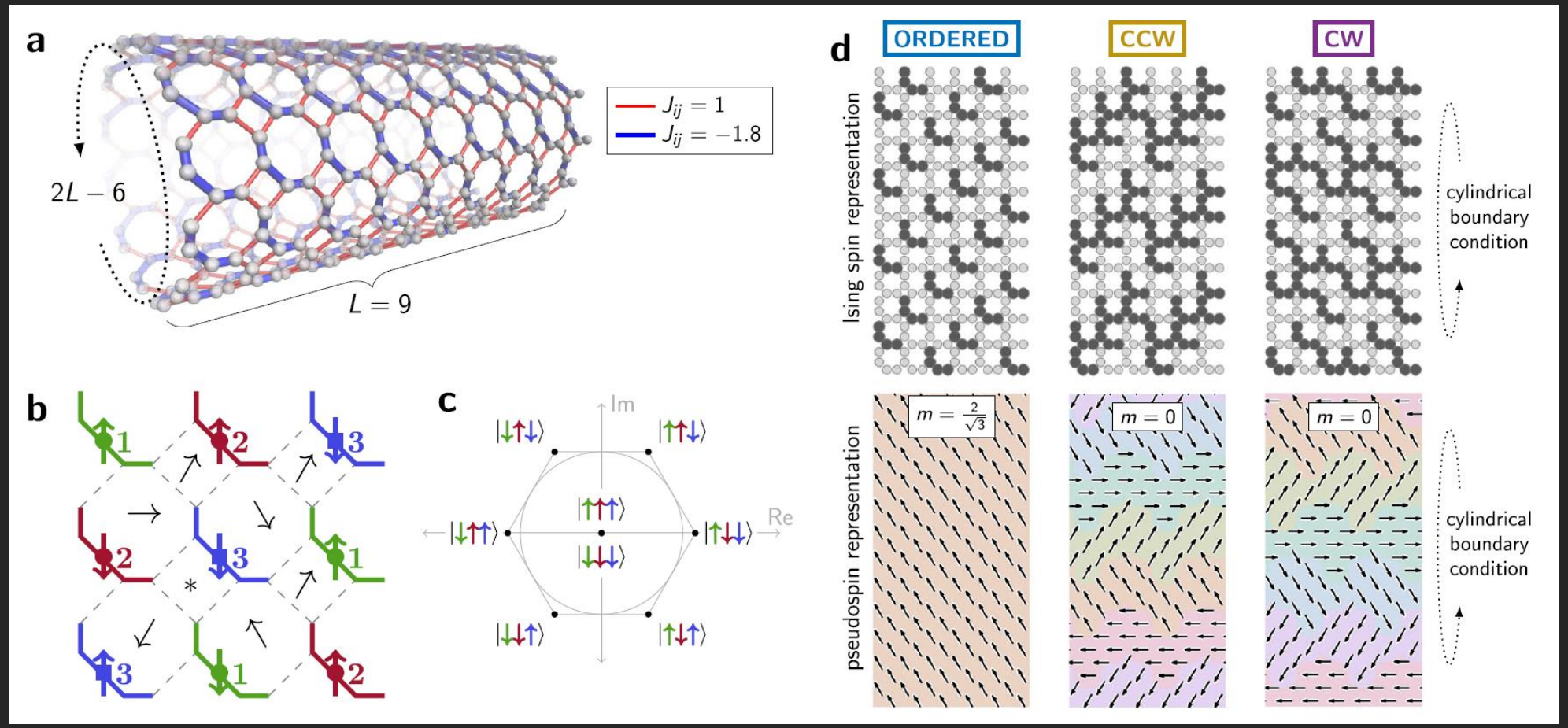


QPU



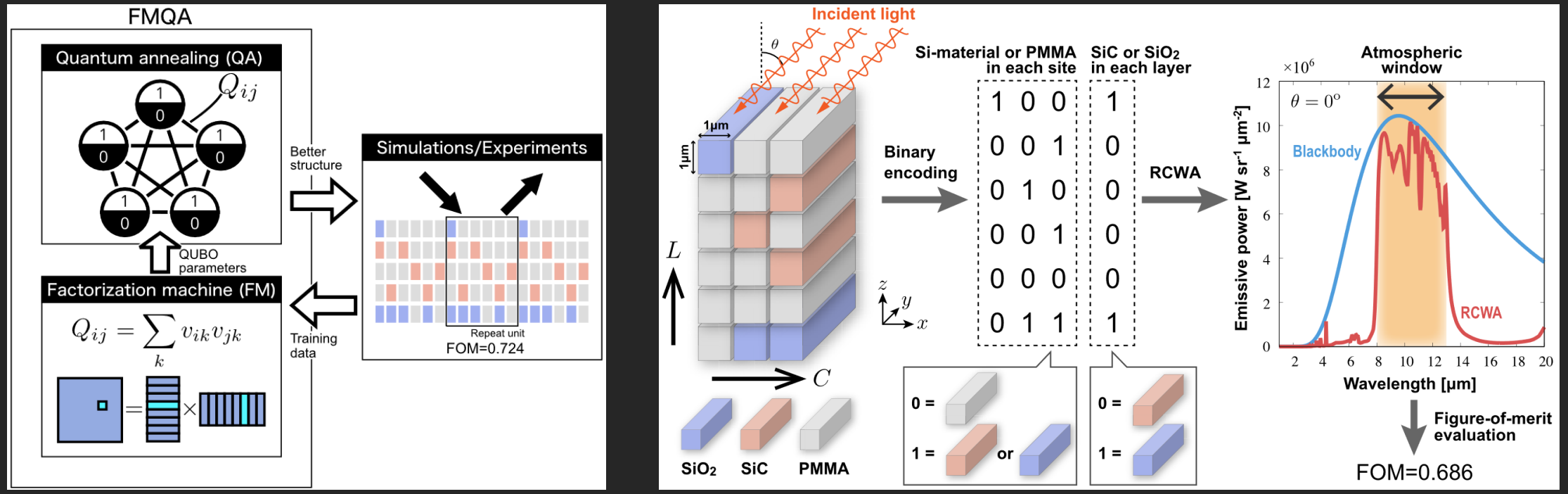
Samples

Spin-glass Simulations



Credit: Nature/D-Wave
https://doi.org/10.1038/s41467-021-20901-5

Metamaterial Design



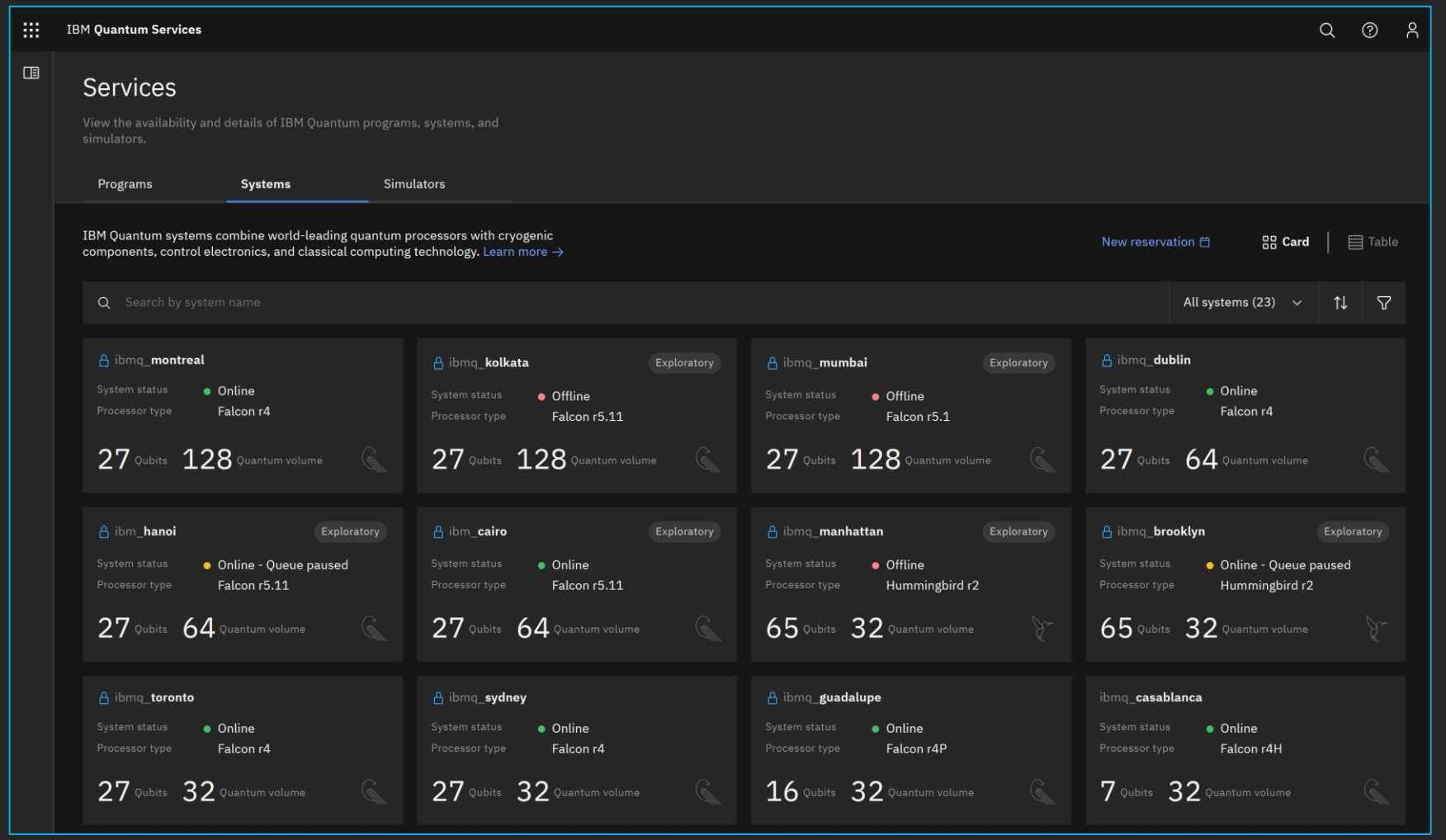
Credit: APS/various institutions (on D-Wave HW)
https://doi.org/10.1103/PhysRevResearch.2.013319

QKD and QRNG (not QC)

Credit: ID Quantique

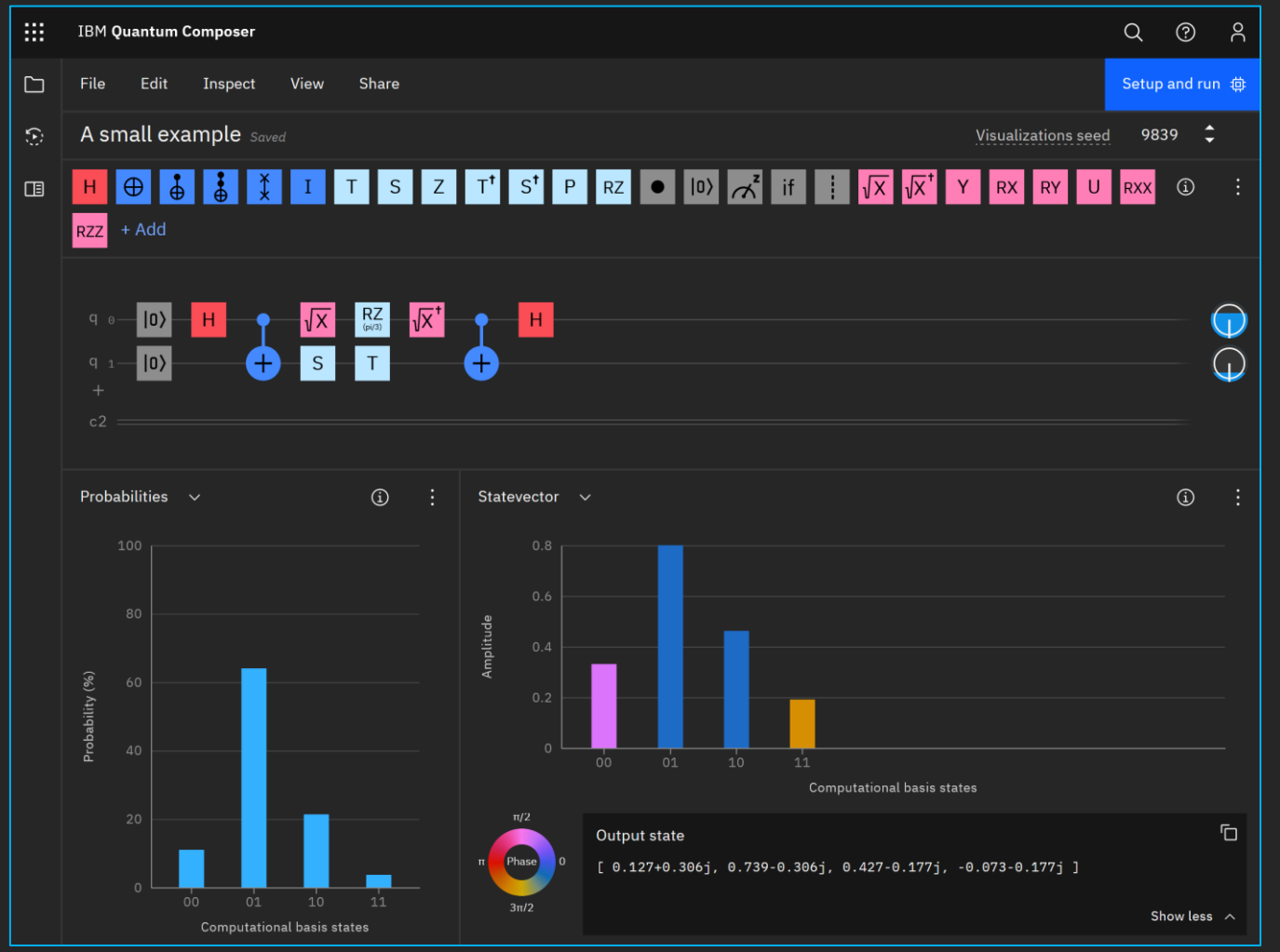
Software Providers

IBM Quantum



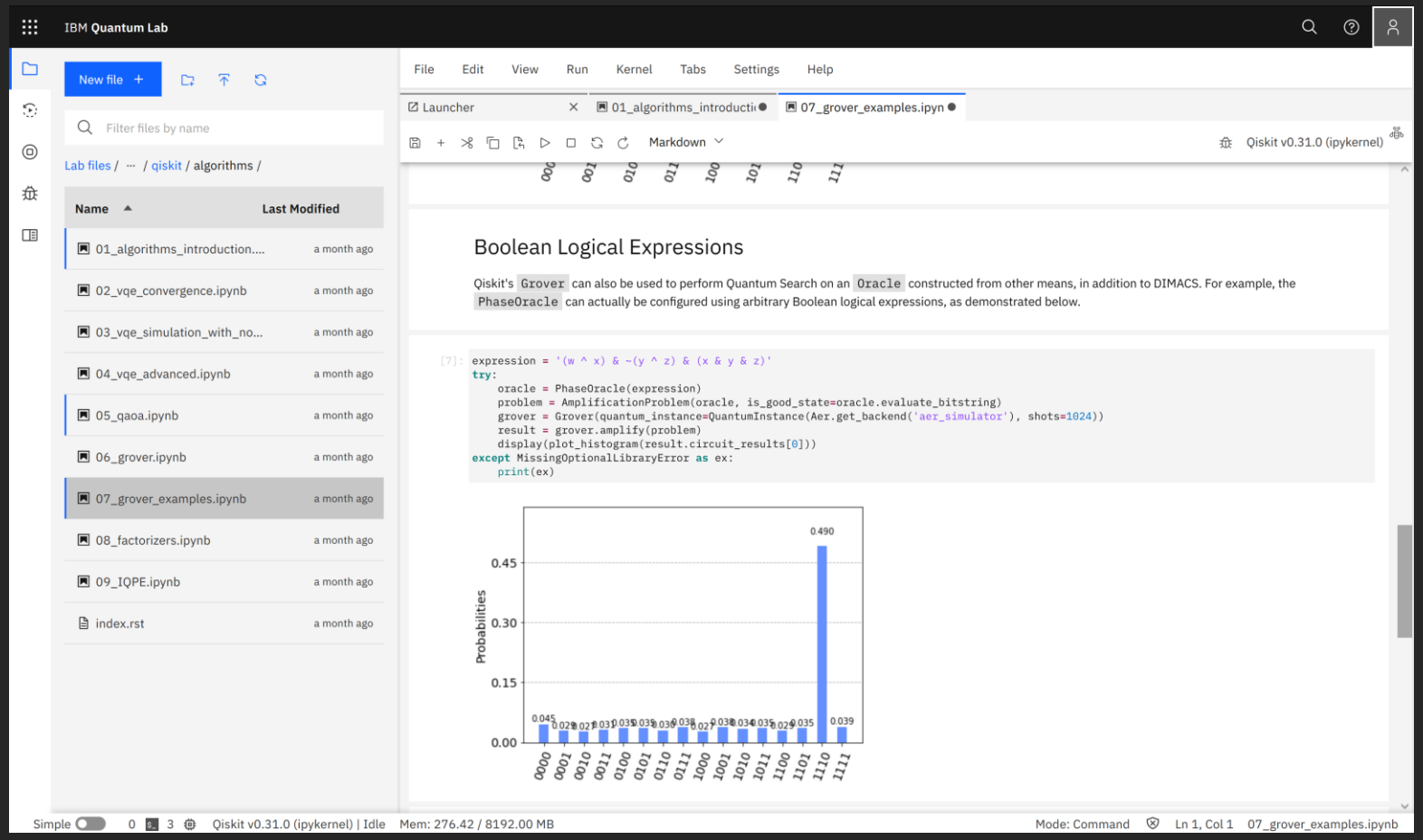
Credit: [IBM – Quantum Services]

IBM Quantum



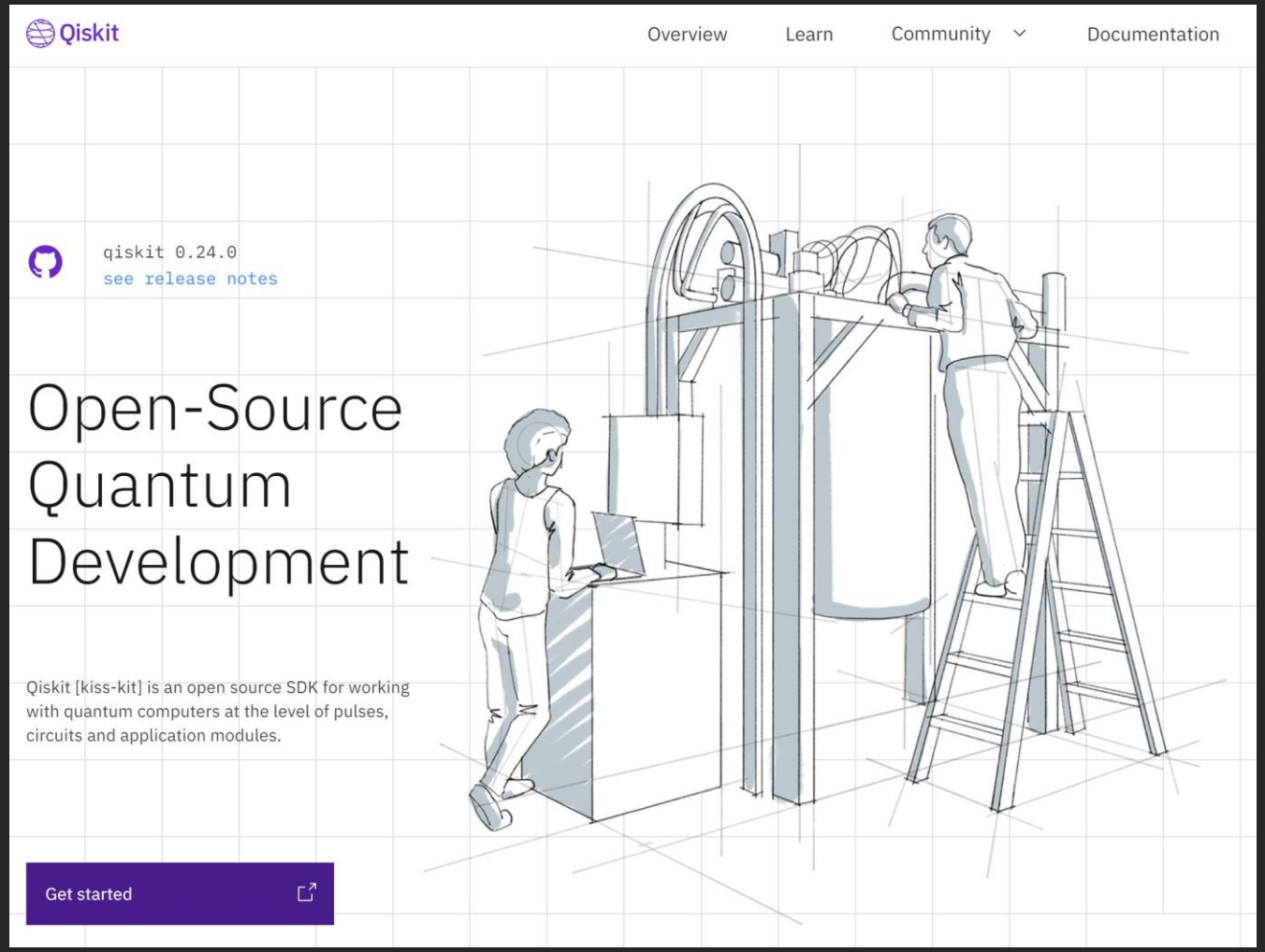
Credit: [IBM – Quantum Composer]

IBM Quantum



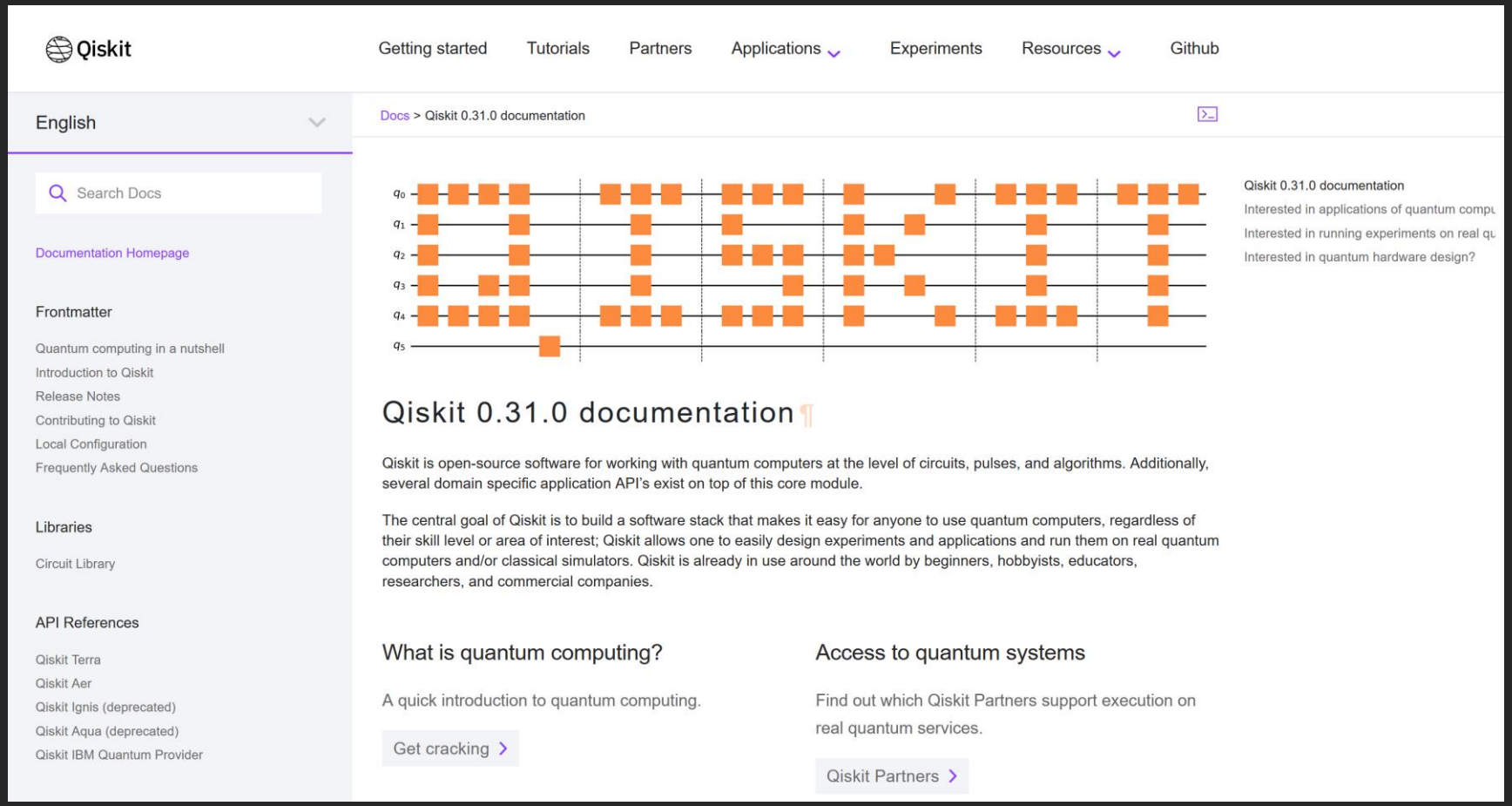
Credit: [IBM – Quantum Lab]

IBM Quantum



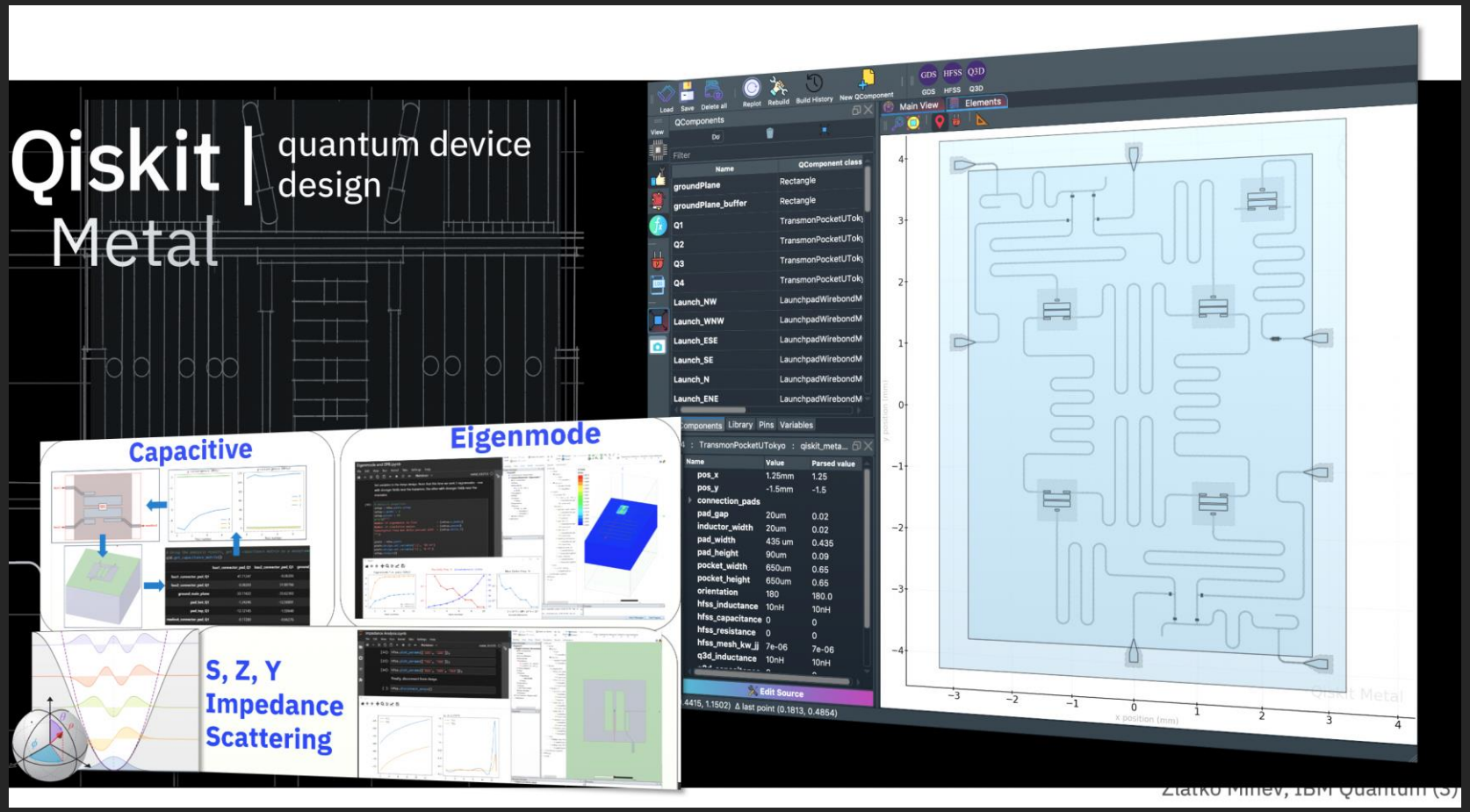
Credit: IBM – Qiskit

IBM Quantum



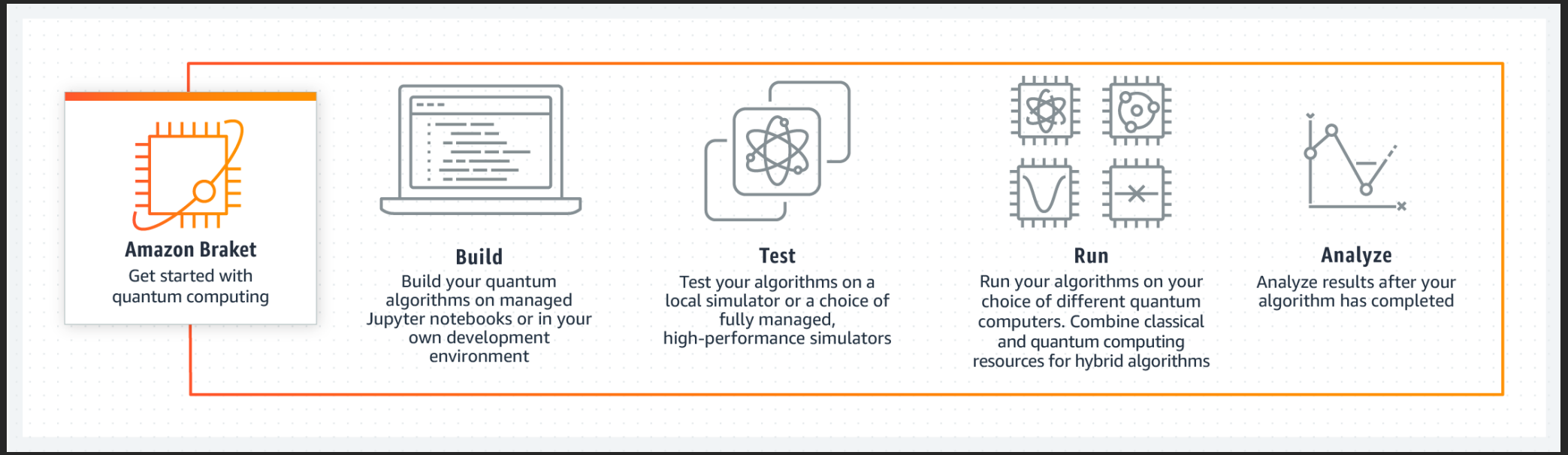
Credit: [IBM – Qiskit documentation]

IBM Quantum



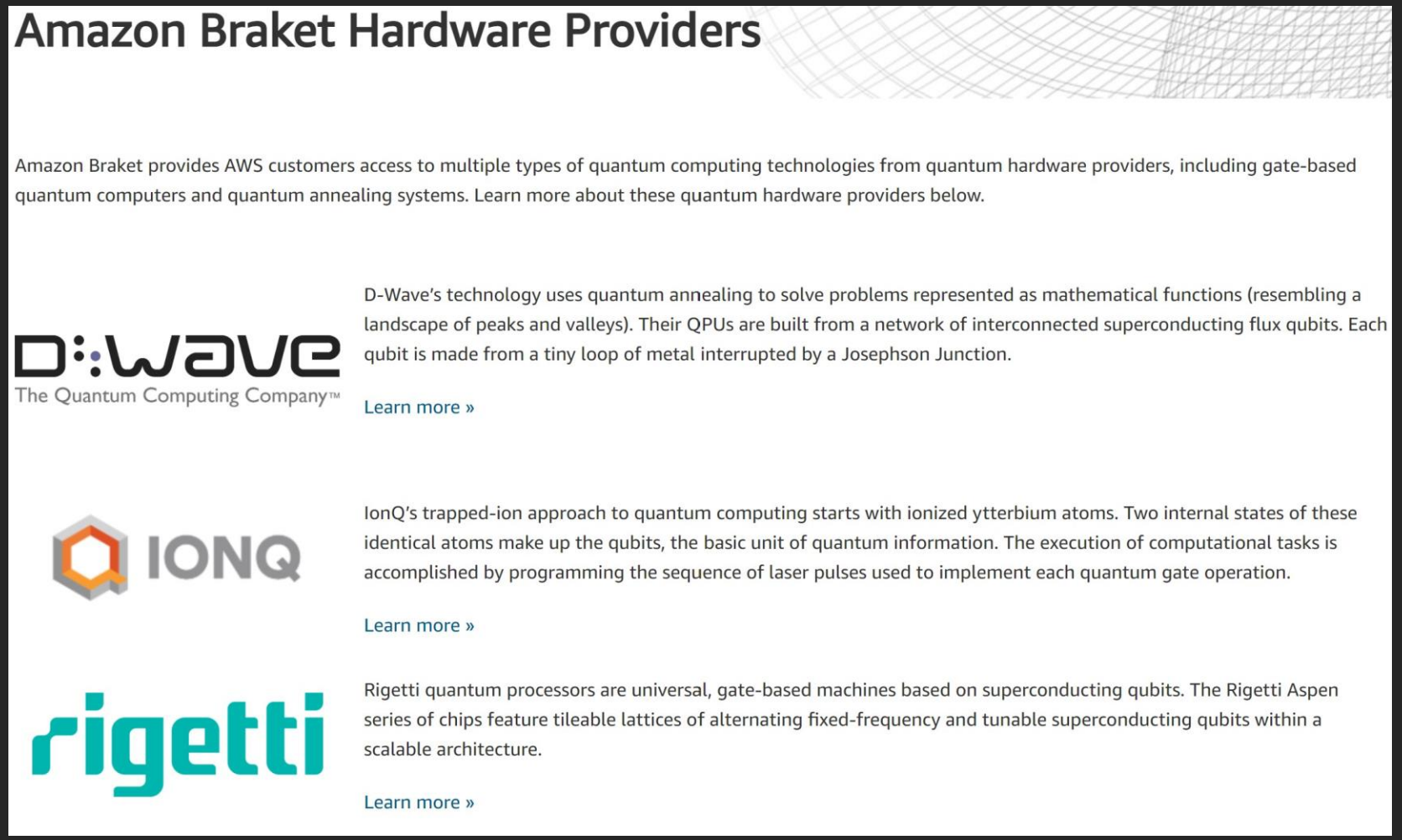
Credit: [IBM - Qiskit Metal]

AWS Braket



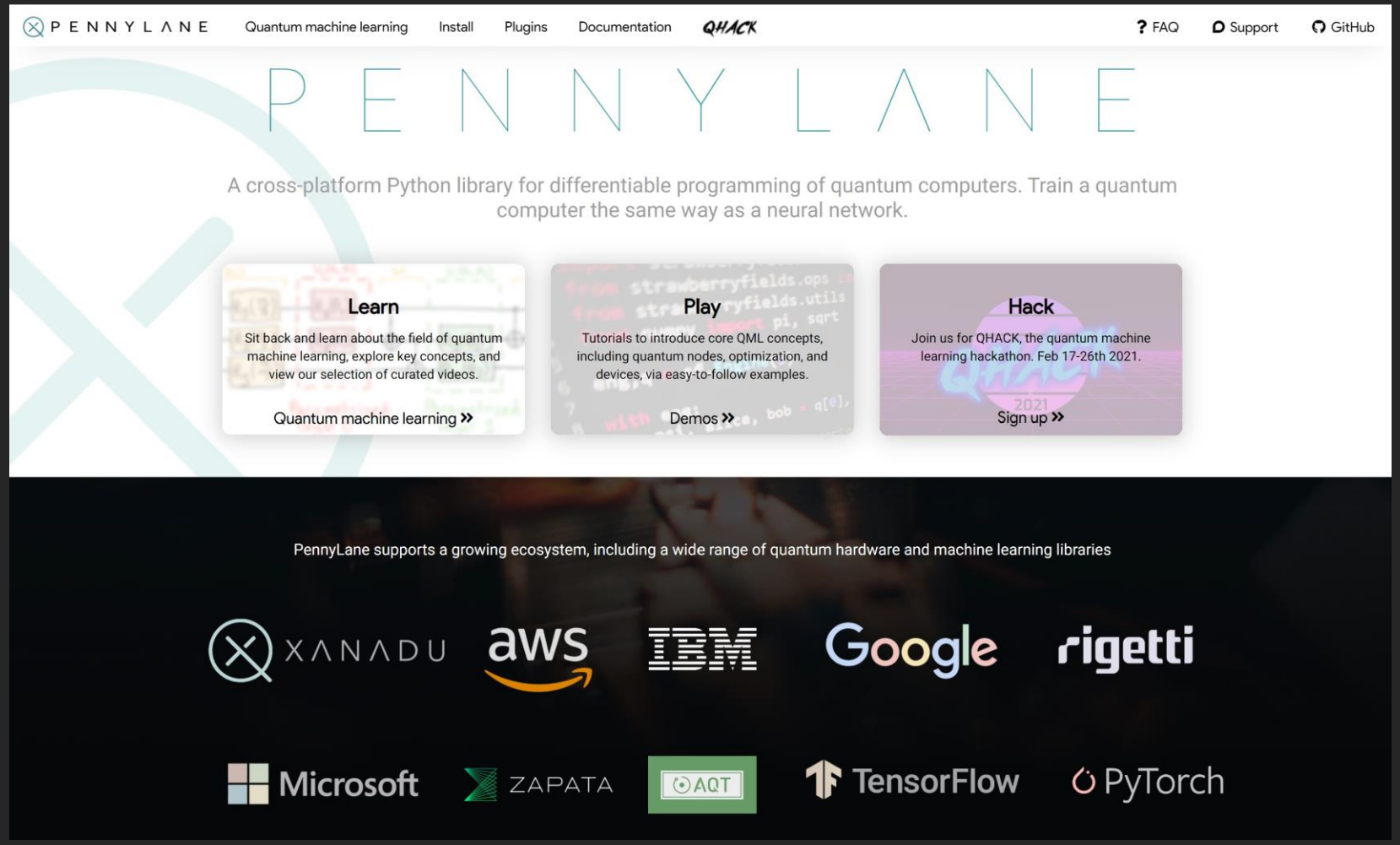
Credit: [Amazon AWS - Braket]

AWS Braket



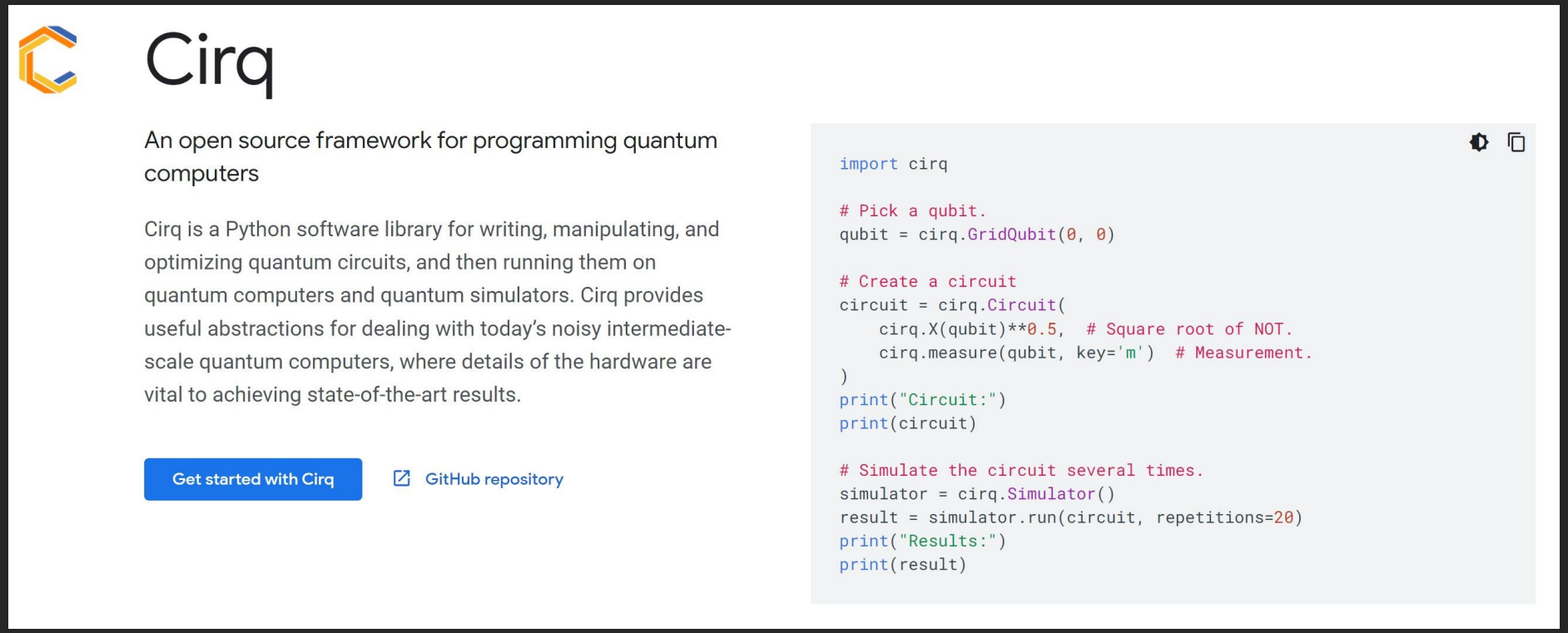
Credit: [Amazon AWS - Braket]

PennyLane



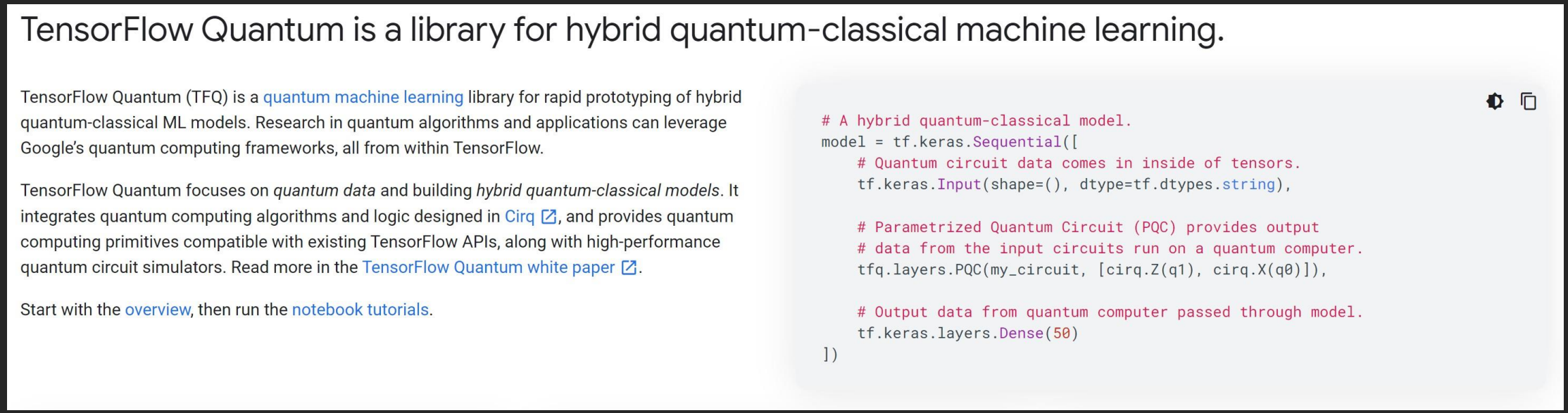
Credit: [Xanadu - PennyLane]

Google Cirq



Credit: [Google Quantum AI - Cirq]

TensorFlow Quantum



TensorFlow Quantum is a library for hybrid quantum-classical machine learning.

TensorFlow Quantum (TFQ) is a quantum machine learning library for rapid prototyping of hybrid quantum-classical ML models. Research in quantum algorithms and applications can leverage Google's quantum computing frameworks, all from within TensorFlow.

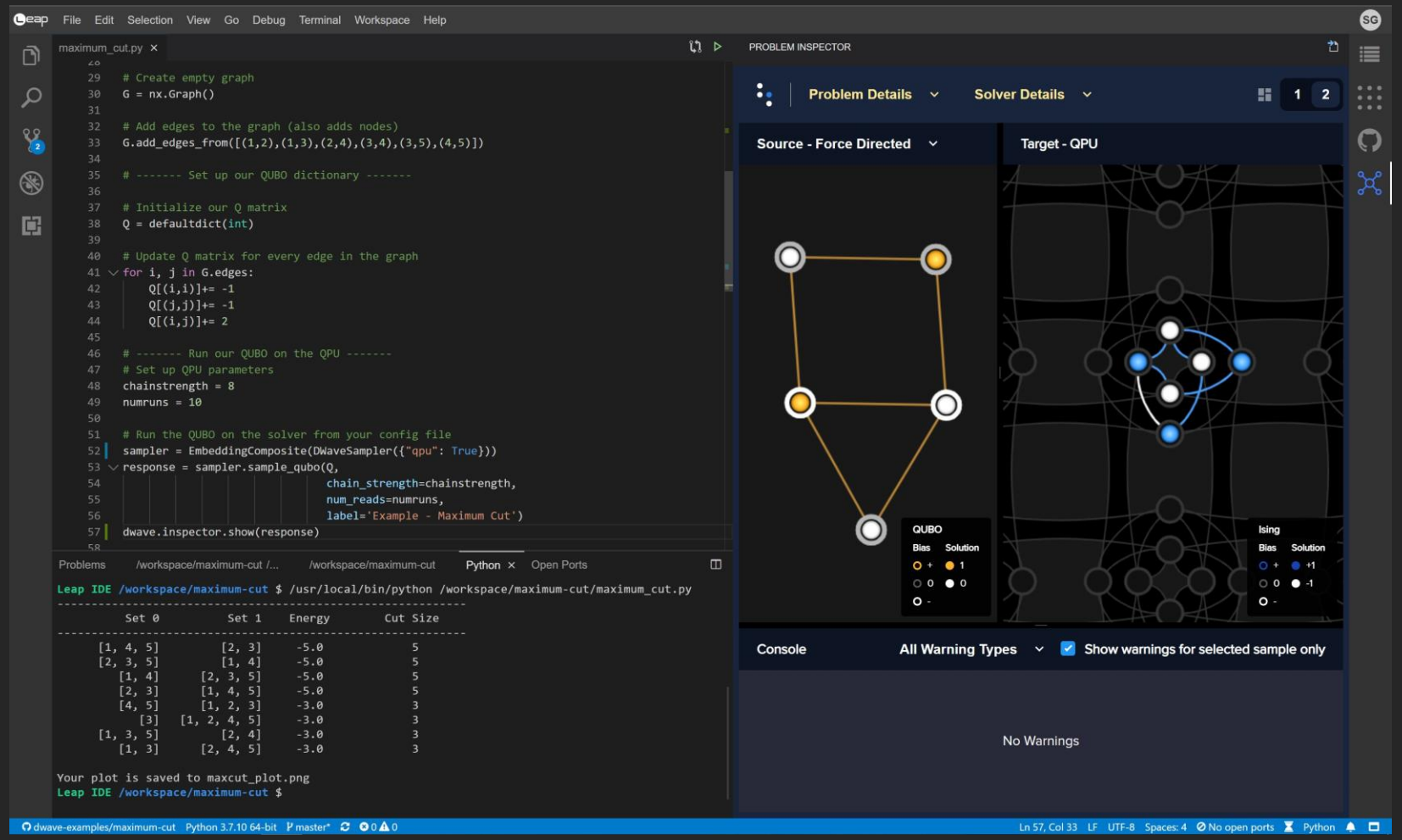
TensorFlow Quantum focuses on *quantum data* and building *hybrid quantum-classical models*. It integrates quantum computing algorithms and logic designed in Cirq, and provides quantum computing primitives compatible with existing TensorFlow APIs, along with high-performance quantum circuit simulators. Read more in the TensorFlow Quantum white paper.

Start with the overview, then run the notebook tutorials.

```
# A hybrid quantum-classical model.
model = tf.keras.Sequential([
    # Quantum circuit data comes in inside of tensors.
    tf.keras.Input(shape=(), dtype=tf.dtypes.string),

    # Parametrized Quantum Circuit (PQC) provides output
    # data from the input circuits run on a quantum computer.
    tfq.layers.PQC(my_circuit, [cirq.Z(q1), cirq.X(q0)]),

    # Output data from quantum computer passed through model.
    tf.keras.layers.Dense(50)
])
```

Credit: Google – TensorFlow Quantum

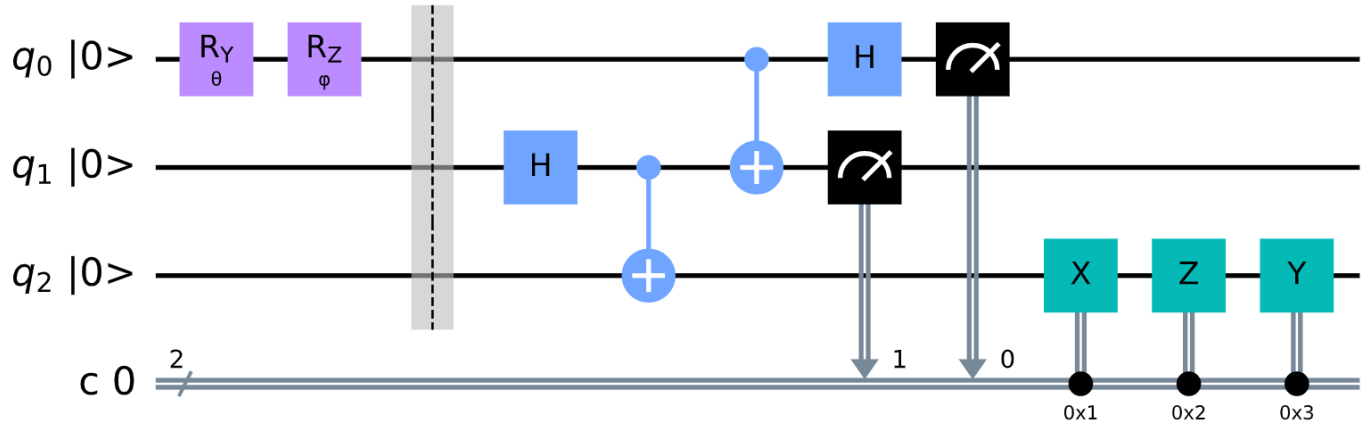
D-Wave Ocean/Leap



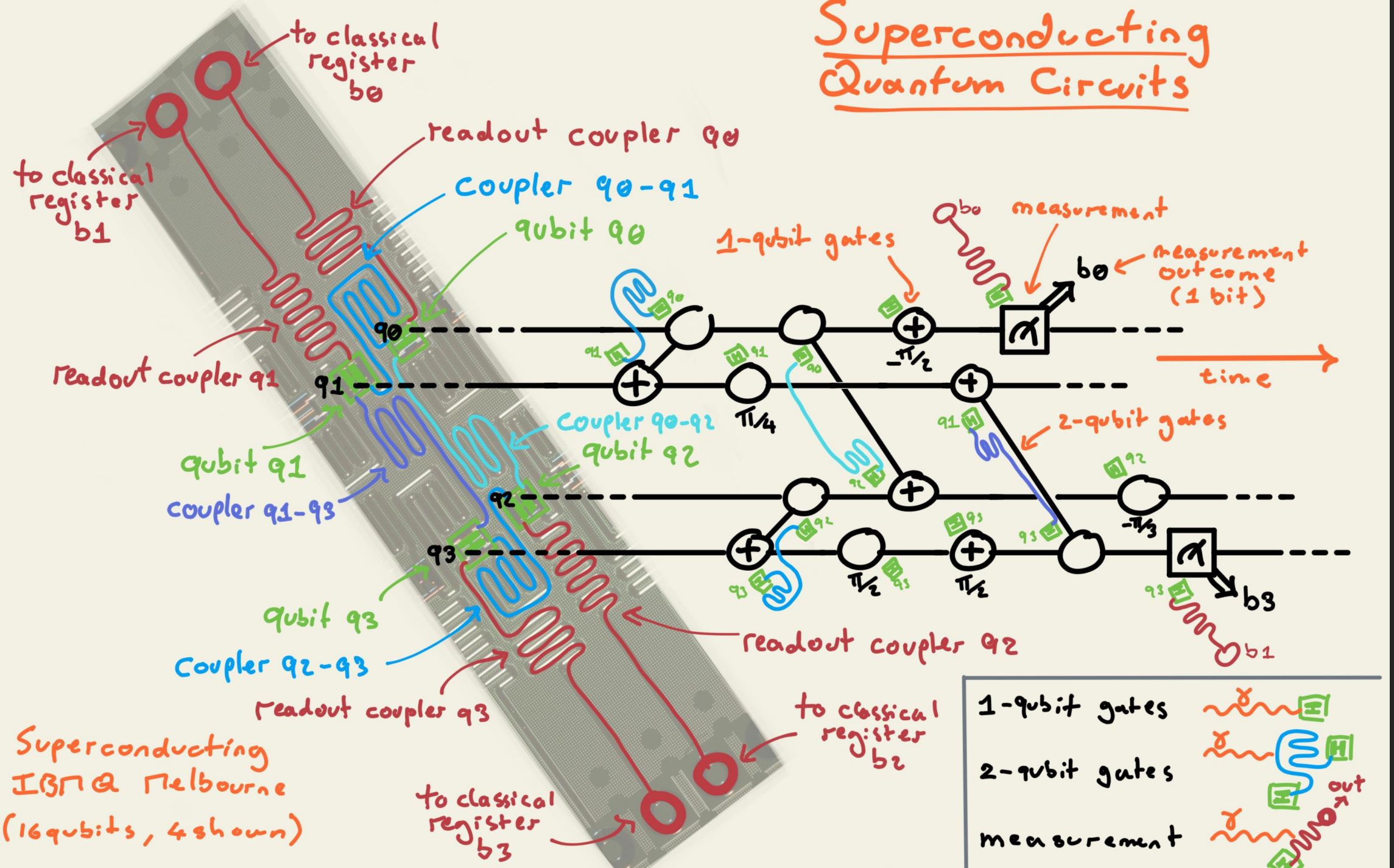
Credit: [D-Wave - Leap]

Quantum Circuits

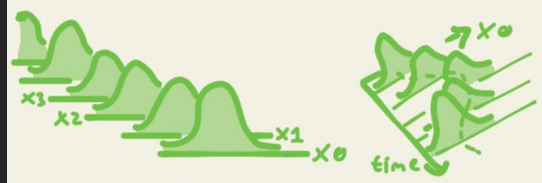
```
from qiskit import QuantumCircuit
from qiskit.circuit import Parameter
# create a new circuit (3 qubits, 2 bits)
circ = QuantumCircuit(3, 2)
# custom input state on q0:
circ.ry(Parameter("θ"), 0)
circ.rz(Parameter("φ"), 0)
circ.barrier()
# quantum teleportation circuit:
circ.h(1)
circ.cx(1, 2)
circ.cx(0, 1)
circ.h(0)
circ.measure([0, 1], [0, 1])
circ.x(2).c_if(circ.cregs[0], 0b01)
circ.z(2).c_if(circ.cregs[0], 0b10)
circ.y(2).c_if(circ.cregs[0], 0b11)
# draw the circuit:
circ.draw("mpl", initial_state=True)
```

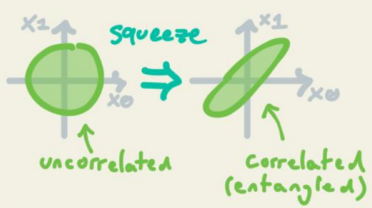


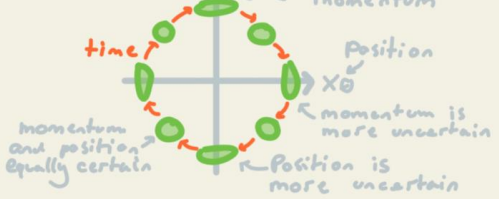
Superconducting Quantum Circuits



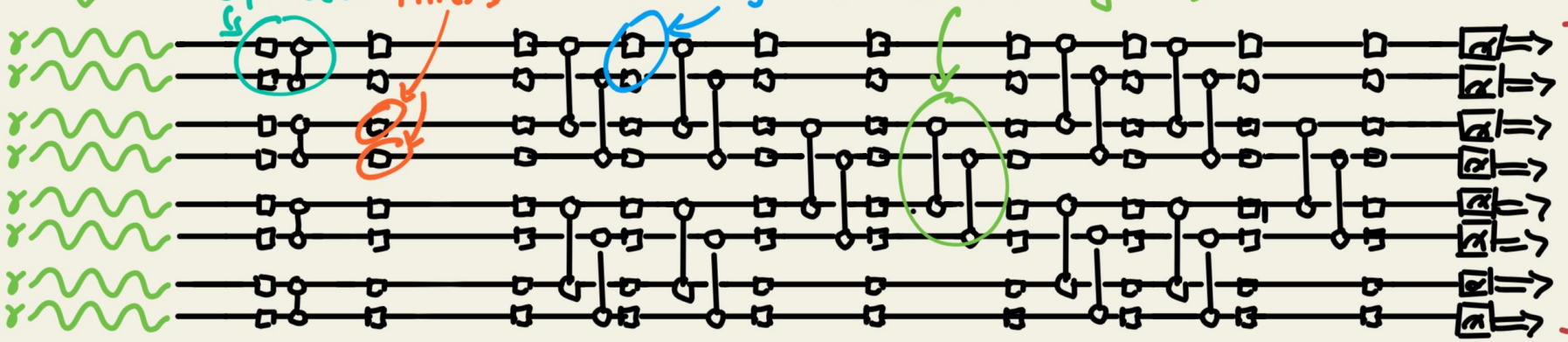
Photonic Quantum Circuits

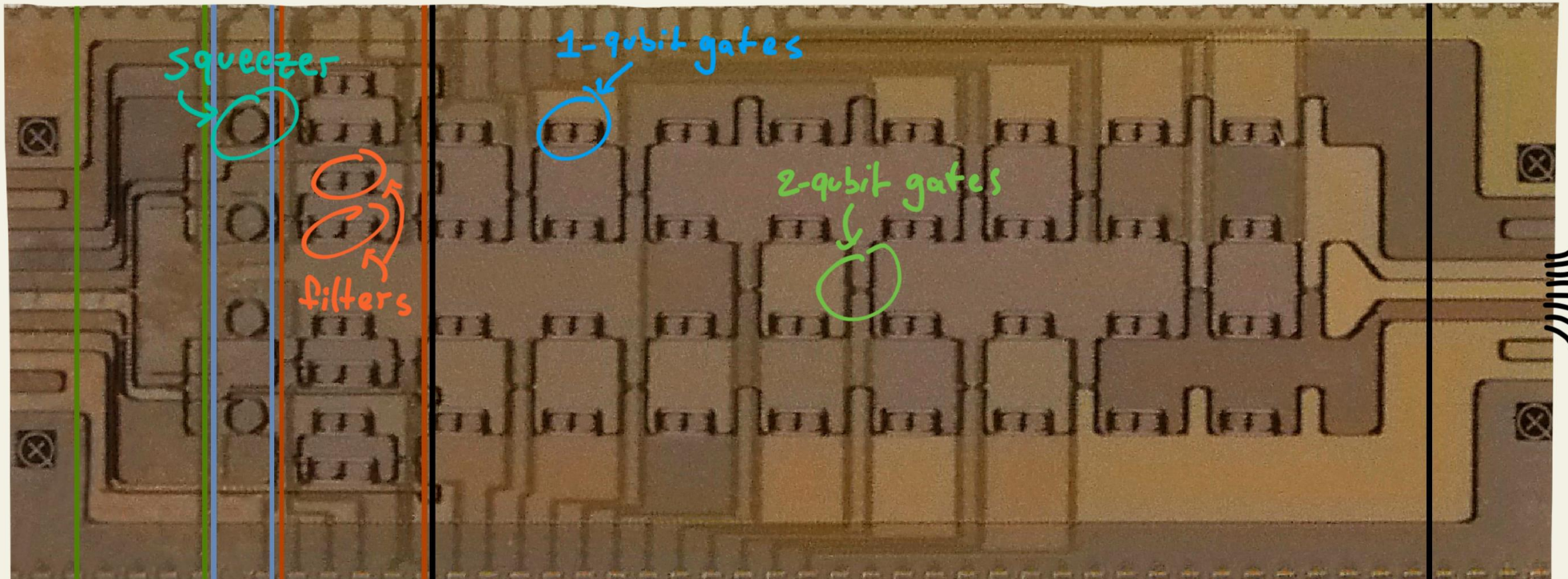


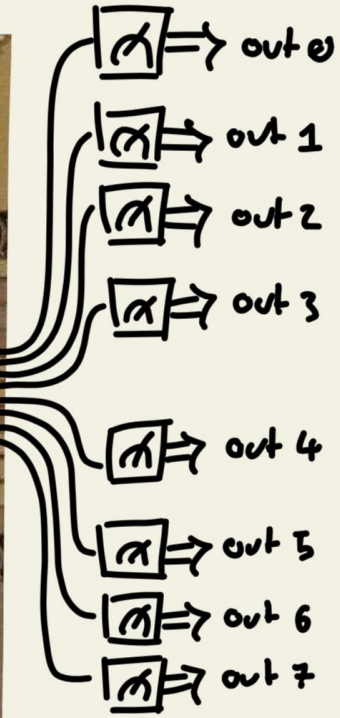




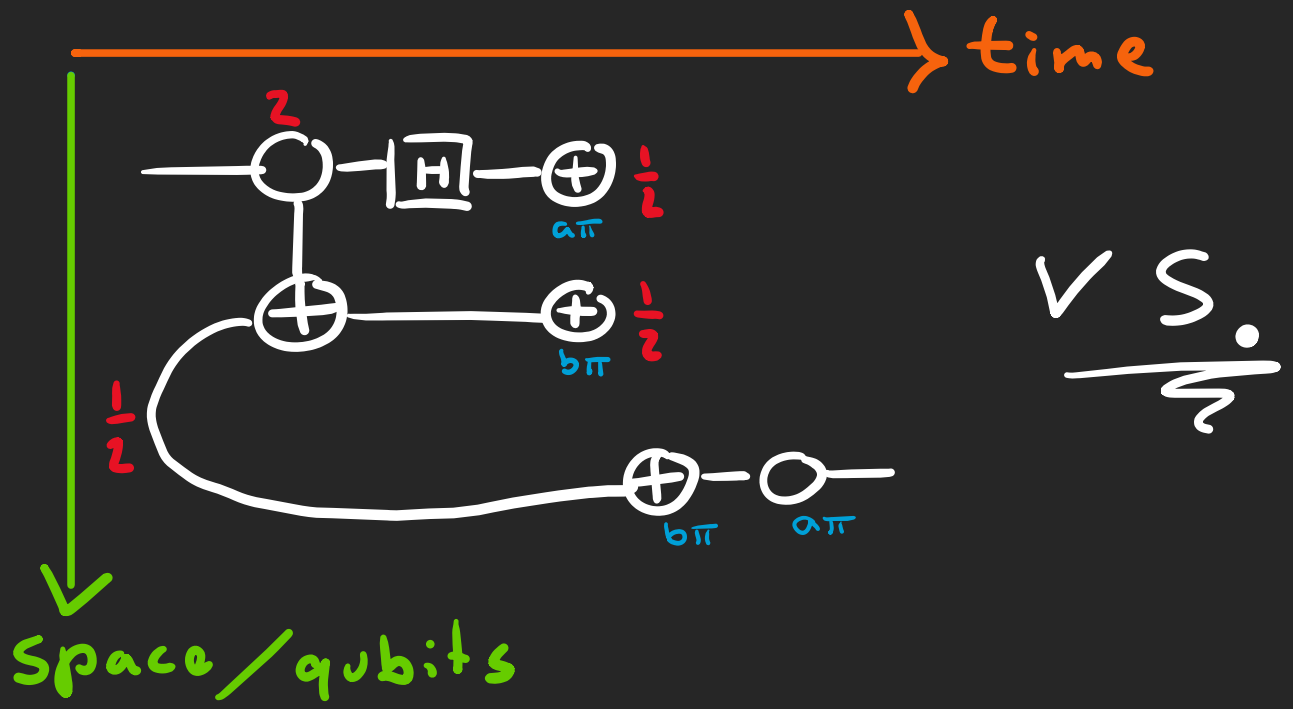
pump laser injects photons







Diagrams as code



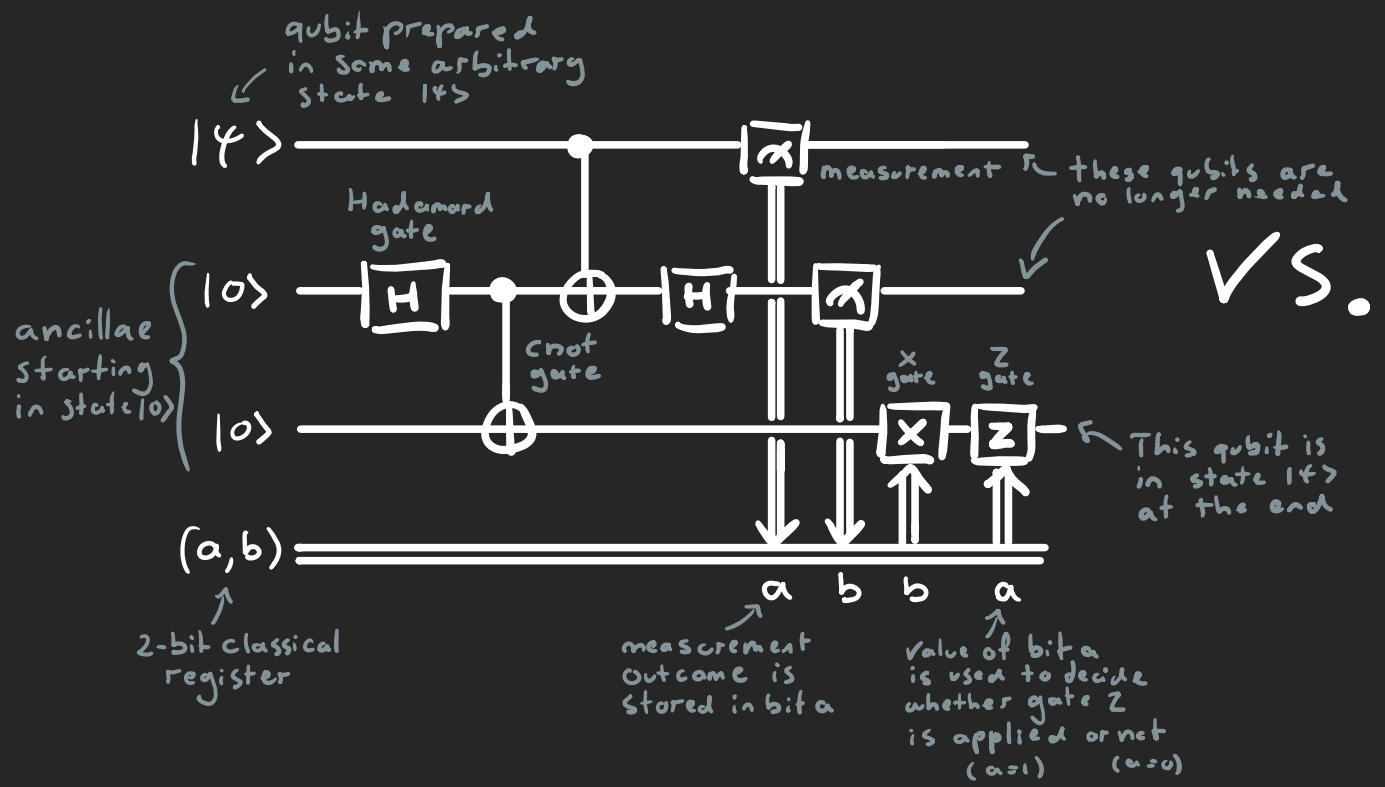
VS.

lines of code

```python
from pyquil.quil import Program
from pyquil import api
from pyquil.gates import X, Z, H, CNOT


def make_bell_pair(q1, q2):
    """Makes a bell pair between qubits q1 and q2
    """
    return Program(H(q1), CNOT(q1, q2))


def teleport(start_index, end_index, ancilla_index):
    """Teleport a qubit from start to end using an ancilla qubit
    """
    program = make_bell_pair(end_index, ancilla_index)

    # do the teleportation
    program.inst(CNOT(start_index, ancilla_index))
    program.inst(H(start_index))

    # measure the results and store them in classical registers [0] and [1]
    program.measure(start_index, 0)
    program.measure(ancilla_index, 1)

    program.if_then(1, X(2))
    program.if_then(0, Z(2))

    program.measure(end_index, 2)

    return program


if __name__ == '__main__':
    qvm = api.QVMConnection()

    # initialize qubit 0 in |1>
    teleport_demo = Program(X(0))
    teleport_demo += teleport(0, 2, 1)
    print("Teleporting |1> state: {}".format(qvm.run(teleport_demo, [2])))

    # initialize qubit 0 in |0>
    teleport_demo = Program()
    teleport_demo += teleport(0, 2, 1)
    print("Teleporting |0> state: {}".format(qvm.run(teleport_demo, [2])))

    # initialize qubit 0 in |+>
    teleport_demo = Program(H(0))
    teleport_demo += teleport(0, 2, 1)
    print("Teleporting |+> state: {}".format(qvm.run(teleport_demo, [2], 10)))

print(Program(X(0)).measure(0, 0).if_then(0, Program(X(1))))
```

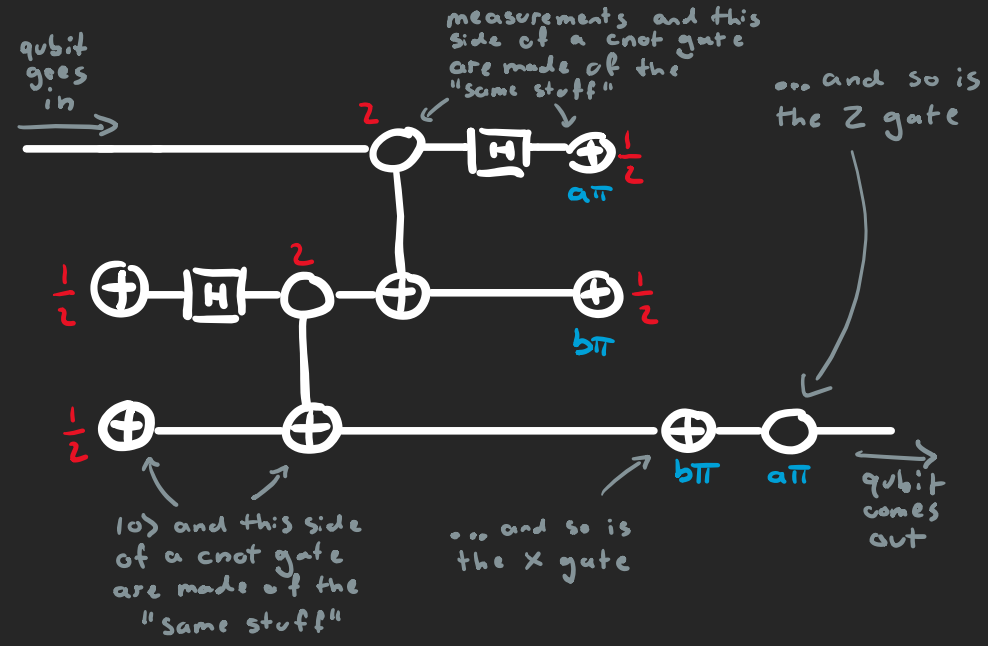
Circuits vs Diagrams

Quantum Circuit



vs.

ZX Diagram

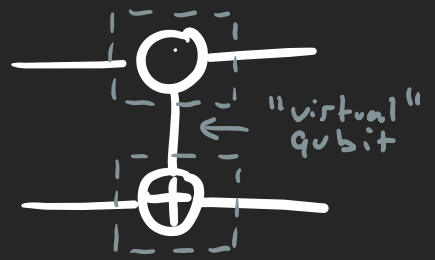


In this course, the two words will be used interchangeably,
always meaning a ZX diagram.

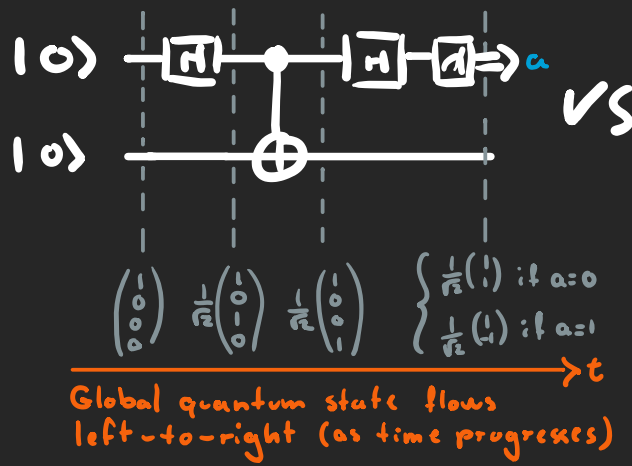
Circuits vs Diagrams

Cnot gate is just one symbol

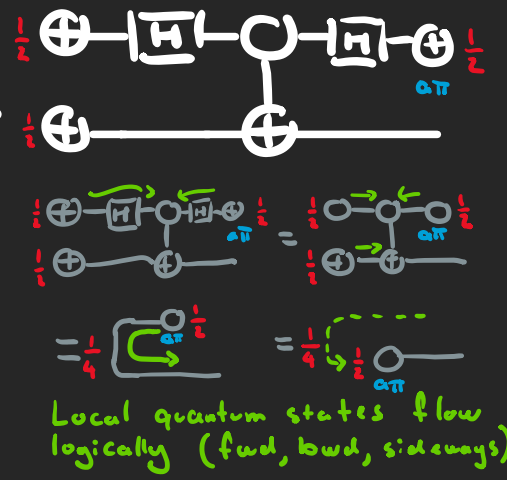
vs.

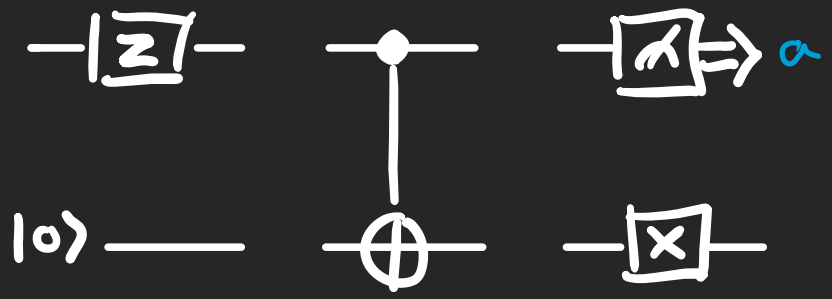


Cnot gate is made of two separate pieces, connected by a "virtual" qubit

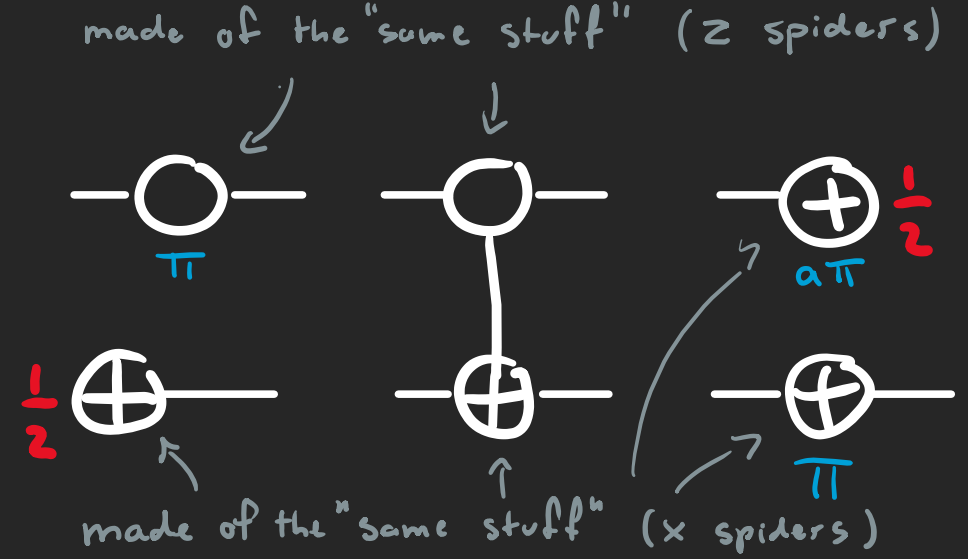


vs.

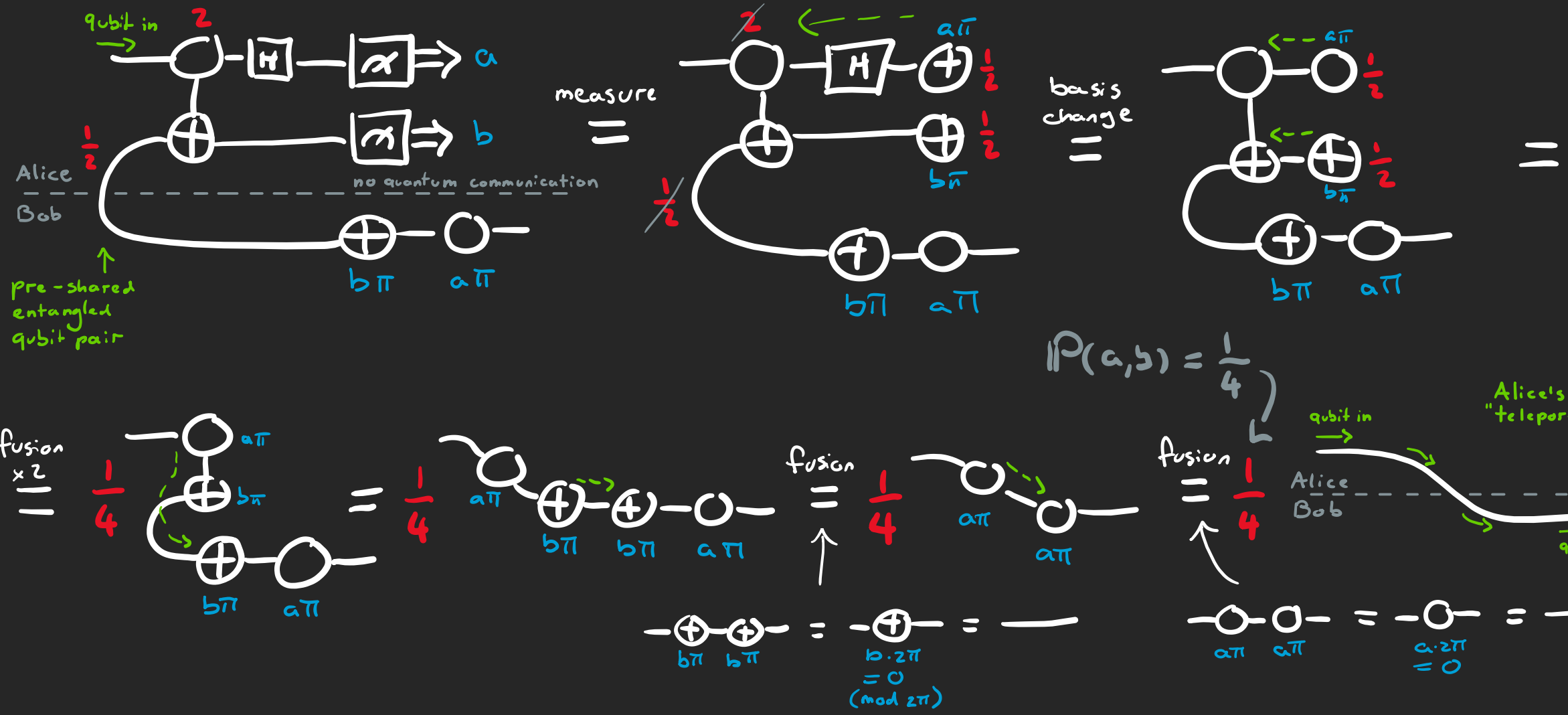




vs.



Diagrammatic Reasoning



ZX Calculus

Bob Coecke
Chief Scientist

Ross Duncan
Head of Quantum Software

Aleks Kissinger
Professor of Quantum Computation

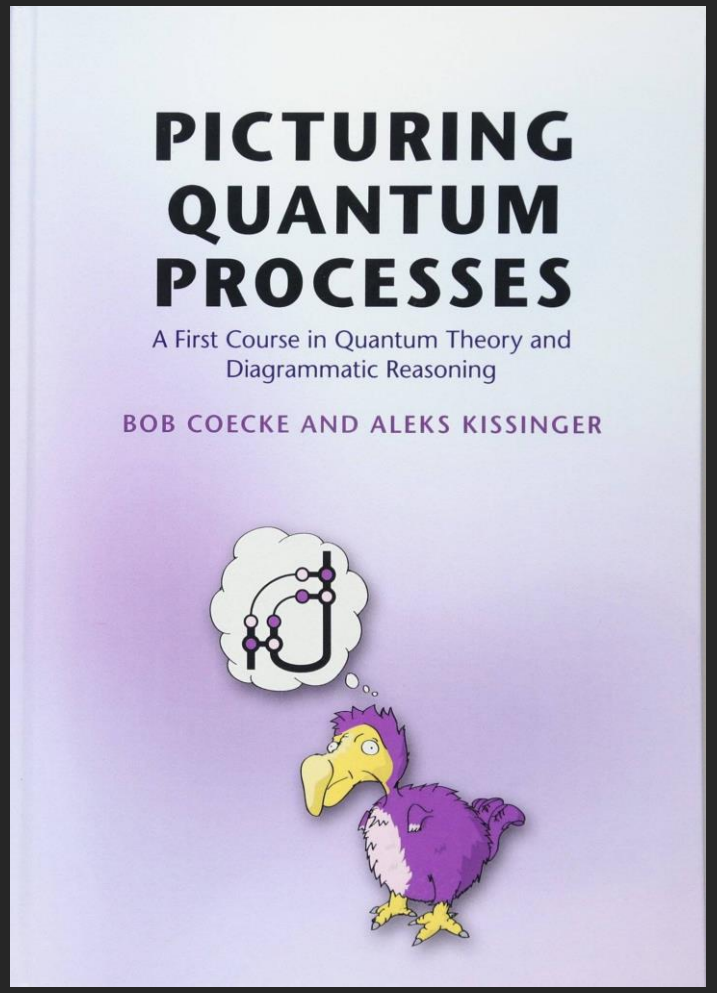
QUANTINUUM

UNIVERSITY OF OXFORD

ZX Calculus

Since its invention in 2007, the ZX calculus has seen more than 150 publications!

You can find them listed here.



ZX-calculus for the working quantum computer scientist

John van de Wetering[1,2]

[1]Radboud Universiteit Nijmegen
[2]Oxford University
December 29, 2020

Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus

Ross Duncan[1,2], Aleks Kissinger[3], Simon Perdrix[4], and John van de Wetering[5]

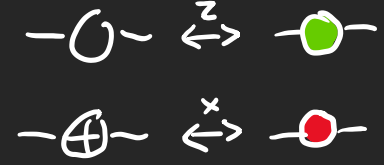
[1]University of Strathclyde, 26 Richmond Street, Glasgow G1 1XH, UK
[2]Cambridge Quantum Computing Ltd, 9a Bridge Street, Cambridge CB2 1UB, UK
[3]Department of Computer Science, University of Oxford
[4]CNRS LORIA, Inria-MOCQUA, Université de Lorraine, F 54000 Nancy, France
[5]Institute for Computing and Information Sciences, Radboud University Nijmegen
May 27, 2020

PyZX: Large Scale Automated Diagrammatic Reasoning

Aleks Kissinger
University of Oxford
aleks.kissinger@cs.ox.ac.uk

John van de Wetering
Radboud University
john@vdwetering.name

Most ZX calculus literature uses a different notation (colour-coded):



PyZX on GitHub

LET’S DIVE IN!