Quantum Software

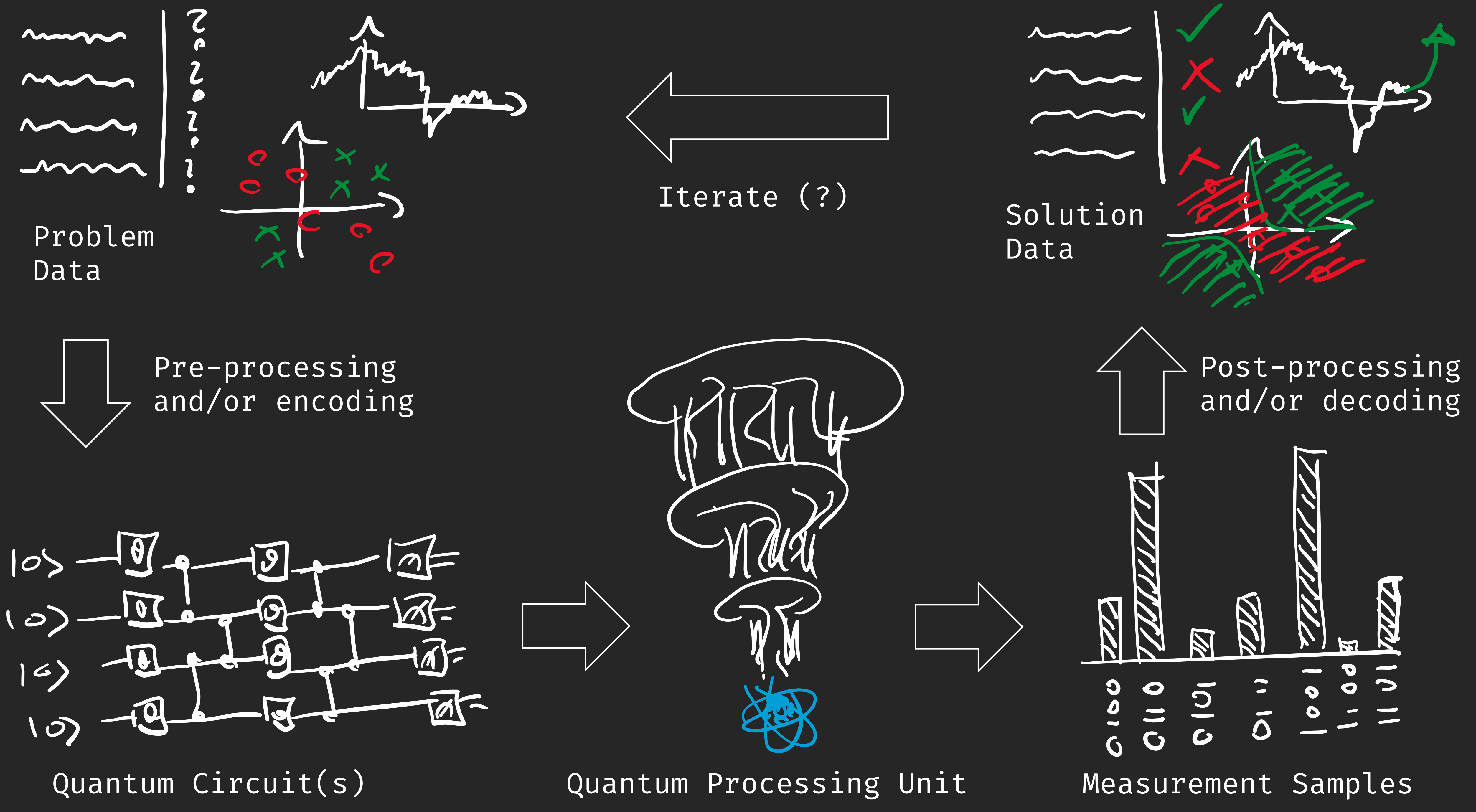
UNIVERSITY OF OXFORD

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?

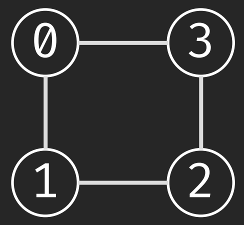
- Quantum Computers as black-box devices used inside classical algorithms
  - Breaking your Bitcoin wallet (one day)
  - Combinatorial Optimization
  - Solutions of Linear/non-linear Systems
  - Complex Physics Simulations (e.g. fluid dynamics)
  - Machine Learning
- Quantum Computers as a programmable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



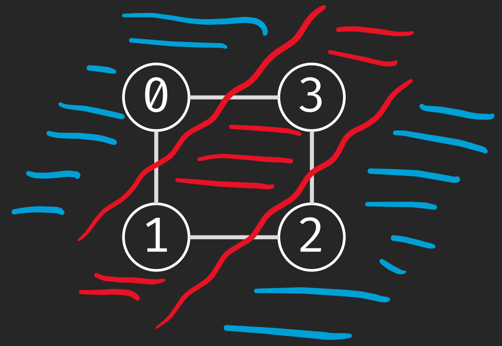
Problem Data
Iterate (?)
Solution Data
Pre-processing and/or encoding
Post-processing and/or decoding
Quantum Circuit(s)
Quantum Processing Unit
Measurement Samples

Example: max-cut with QAOA

Graph

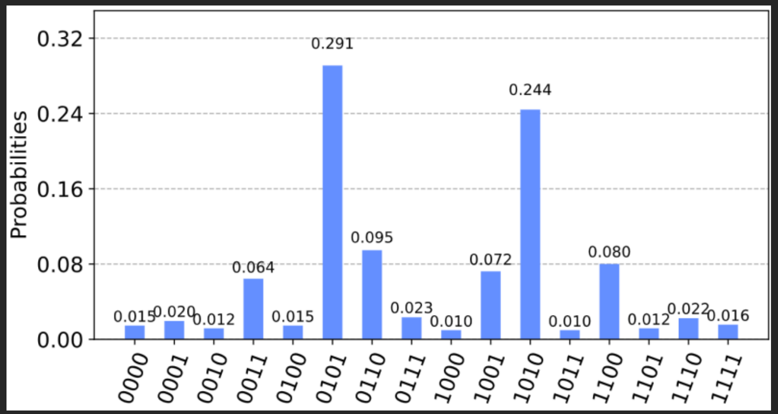


Graph Cut



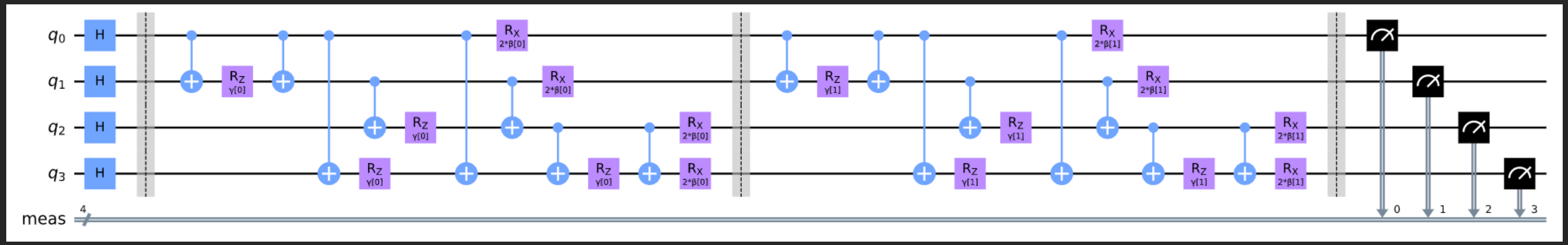
Max count solution

Samples

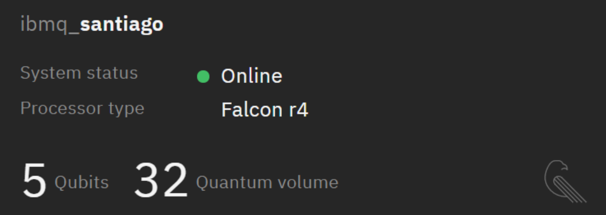


Encoding using 2-qubit gates

Parameter optimisation



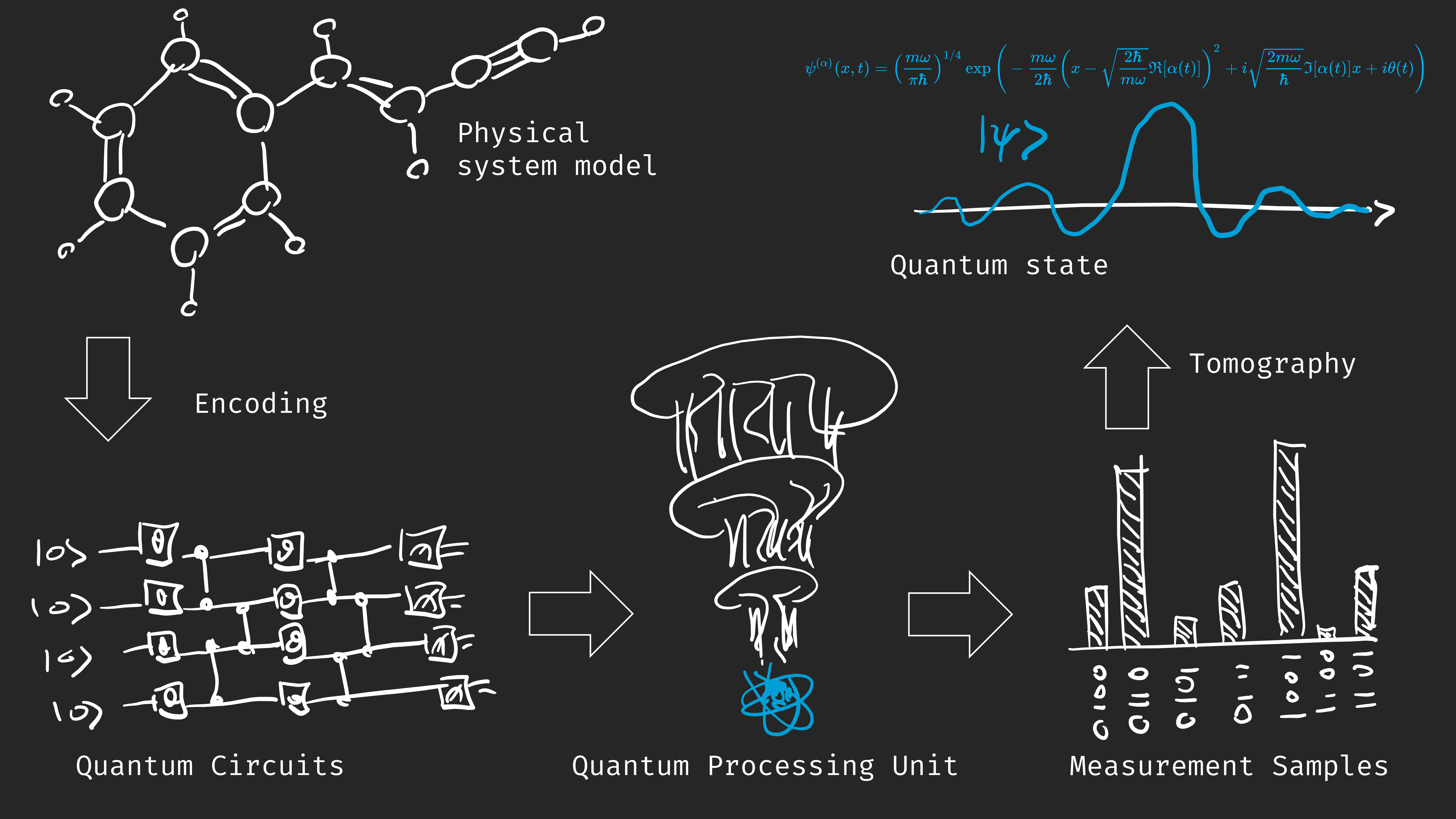
QAOA Circuit (parametric)



QPU

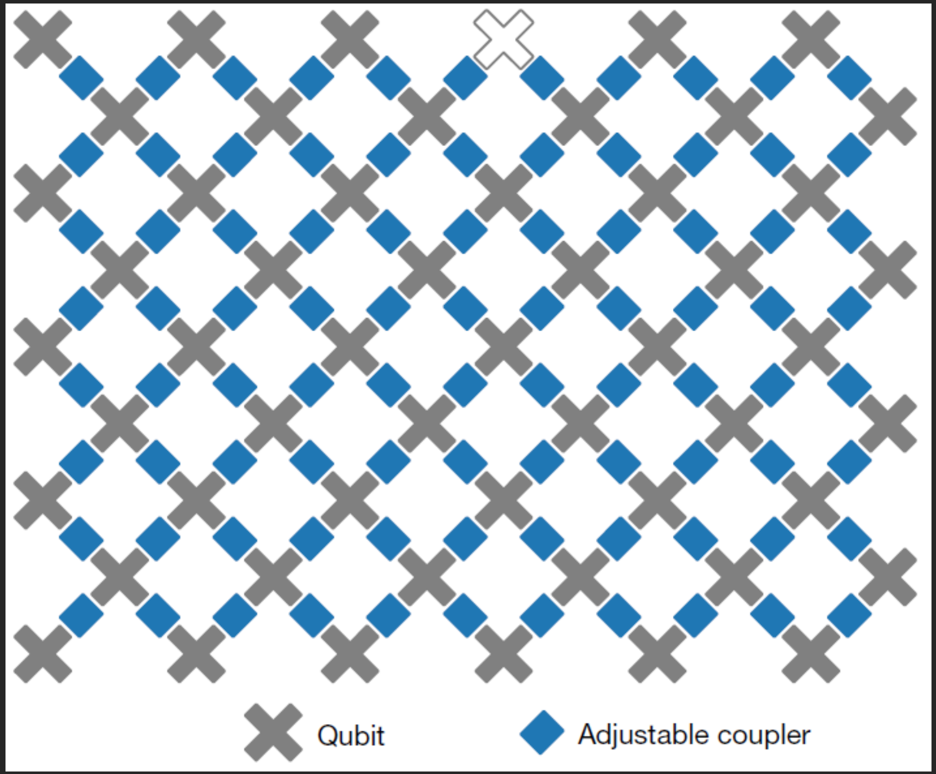
What is Quantum Computing?

- 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

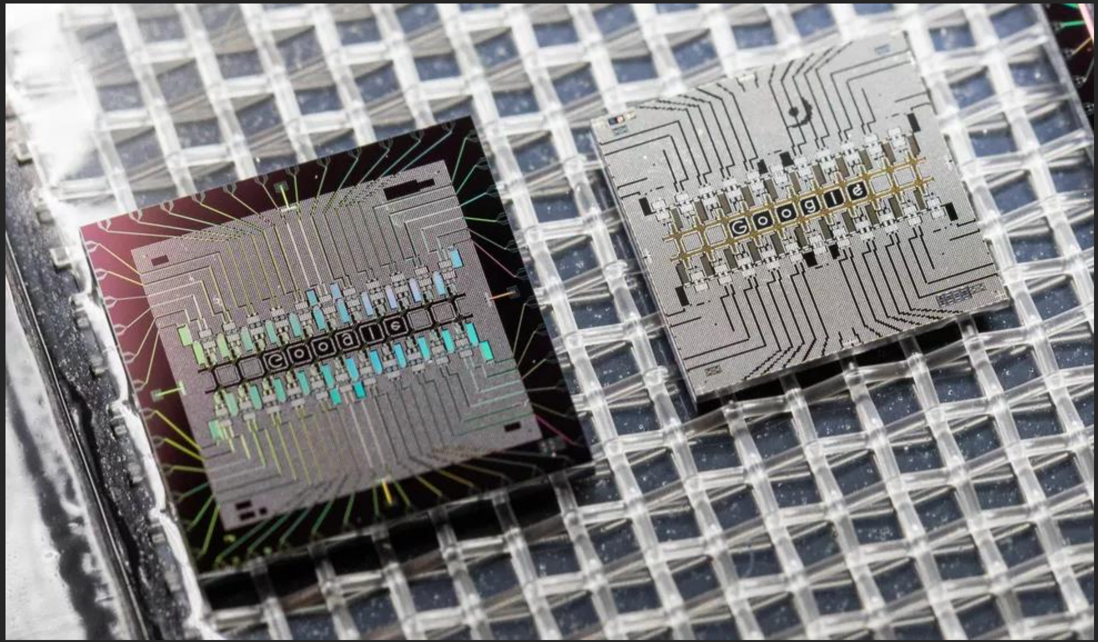

Physical system model
$\psi^{(\alpha)}(x,t) = \left(\frac{m\omega}{\pi\hbar}\right)^{1/4} \exp\left(-\frac{m\omega}{2\hbar}\left(x - \sqrt{\frac{2\hbar}{m\omega}}\Re[\alpha(t)]\right)^2 + i\sqrt{\frac{2m\omega}{\hbar}}\Im[\alpha(t)]x + i\theta(t)\right)$
Quantum state
Encoding
Tomography
Quantum Circuits
Quantum Processing Unit
Measurement Samples

Hardware Providers

Google Sycamore (54 qubits)

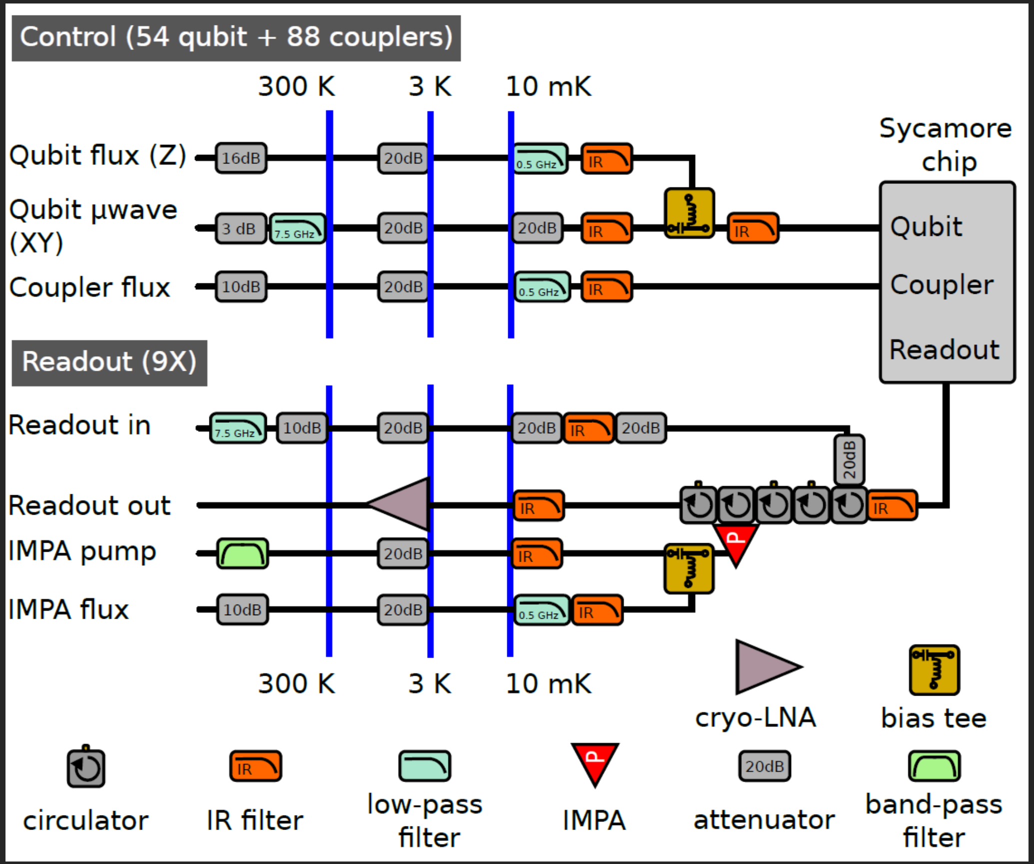


Credit: Nature/Google
https://doi.org/10.1038/s41586-019-1666-5

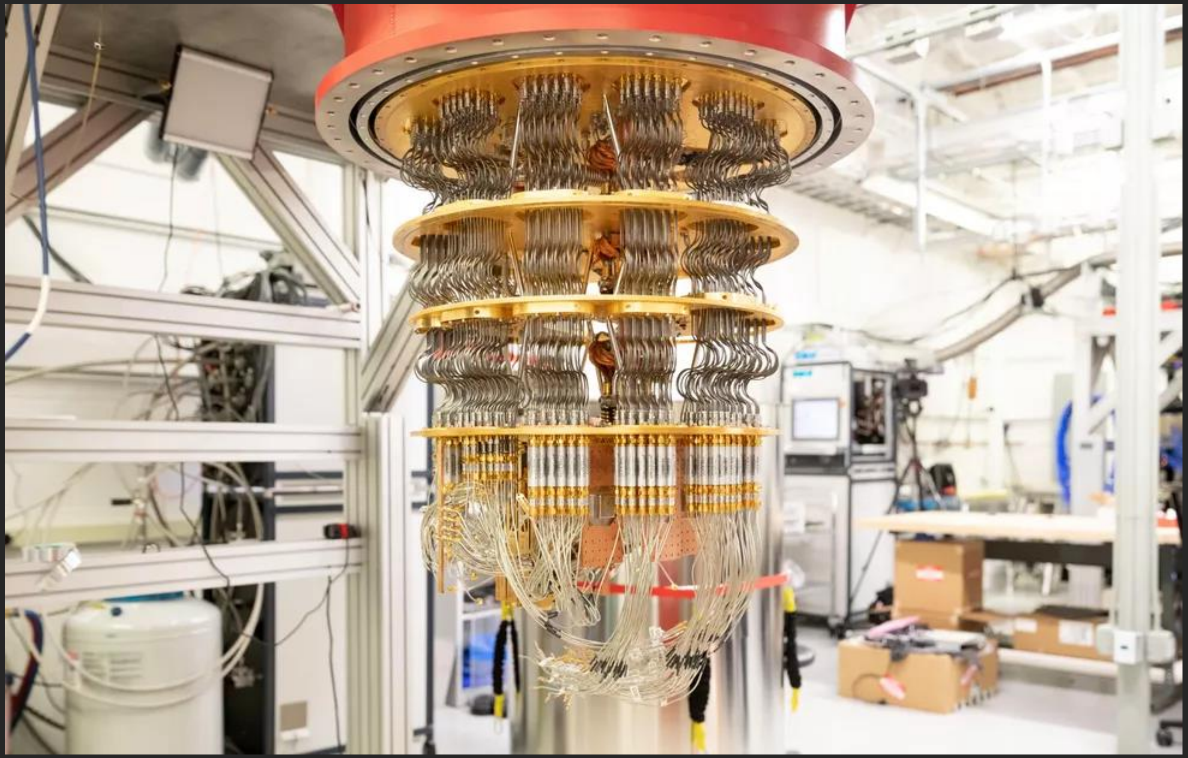


Credit: Stephen Shankland/CNET

Google Sycamore (54 qubits)

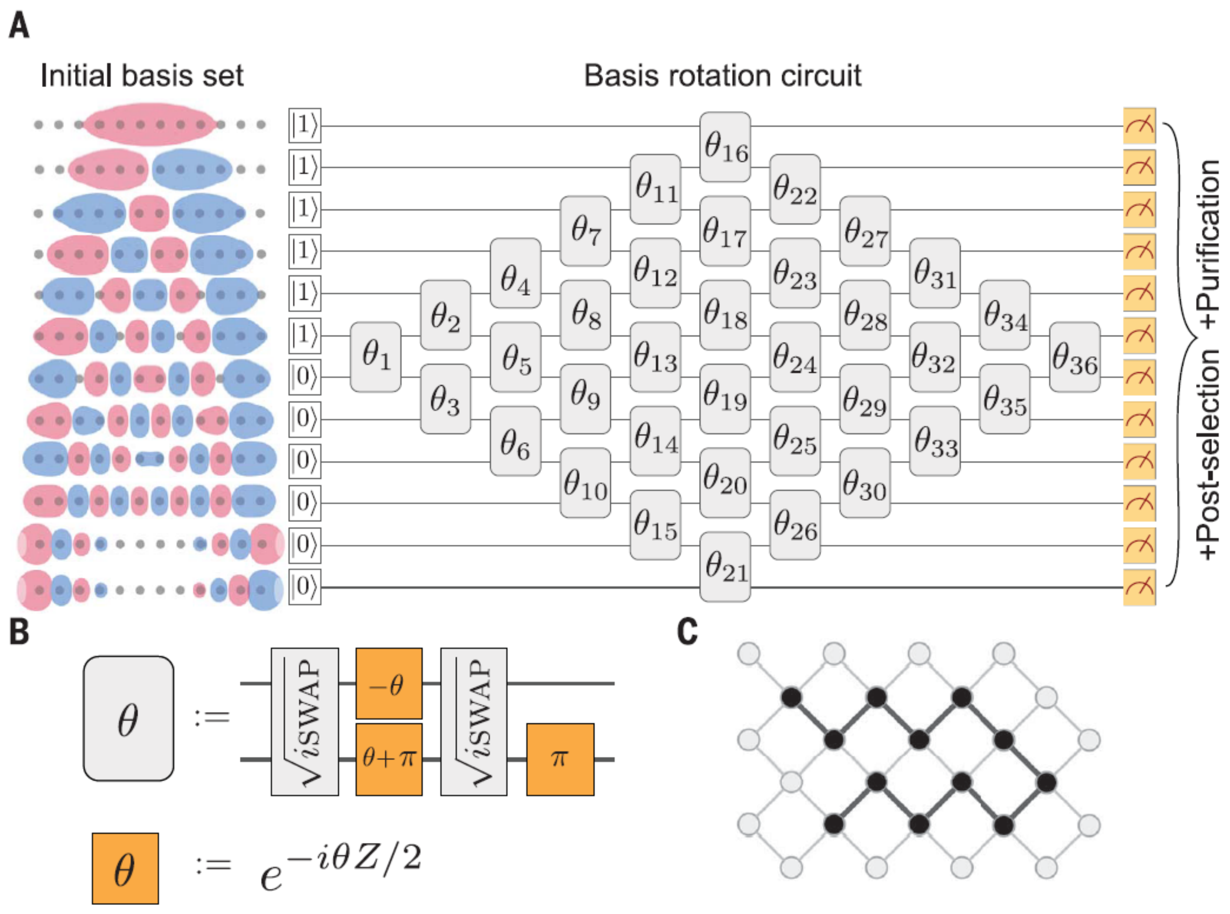


Credit: Nature/Google
https://doi.org/10.1038/s41586-019-1666-5



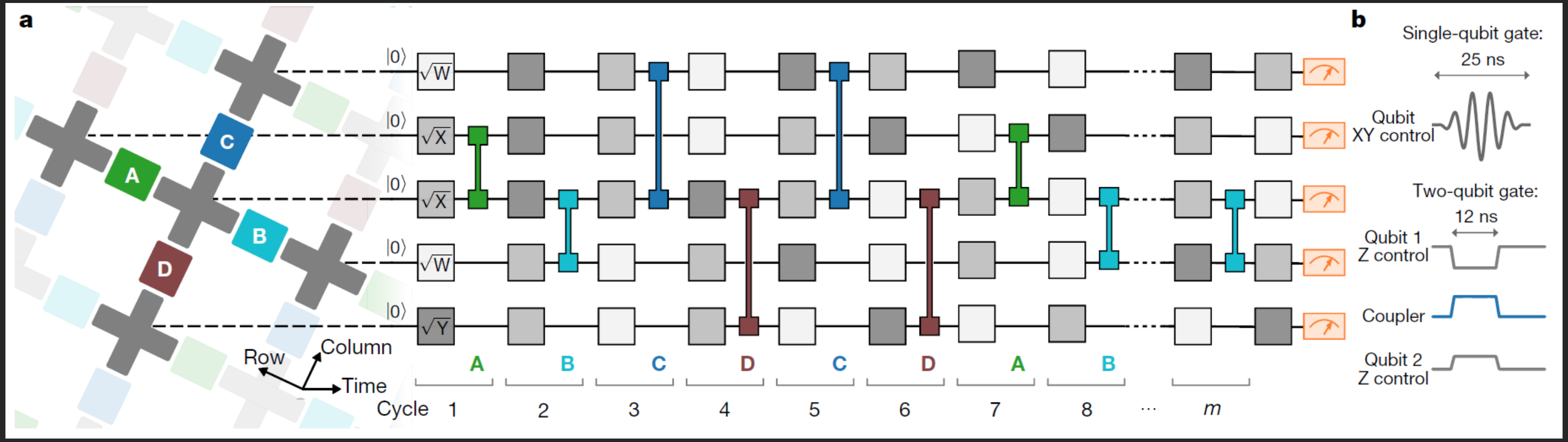
Credit: Stephen Shankland/CNET

Quantum Chemistry Simulations

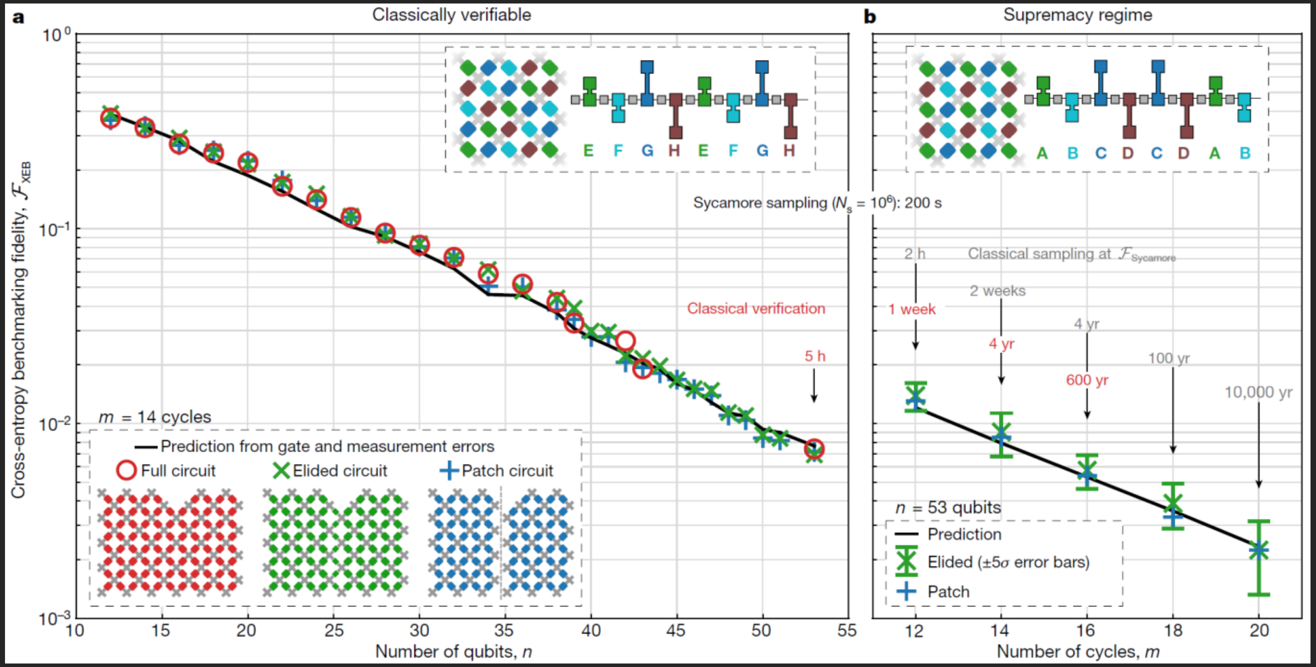


Credit: Science/Google
https://doi.org/10.1126/science.abb9811

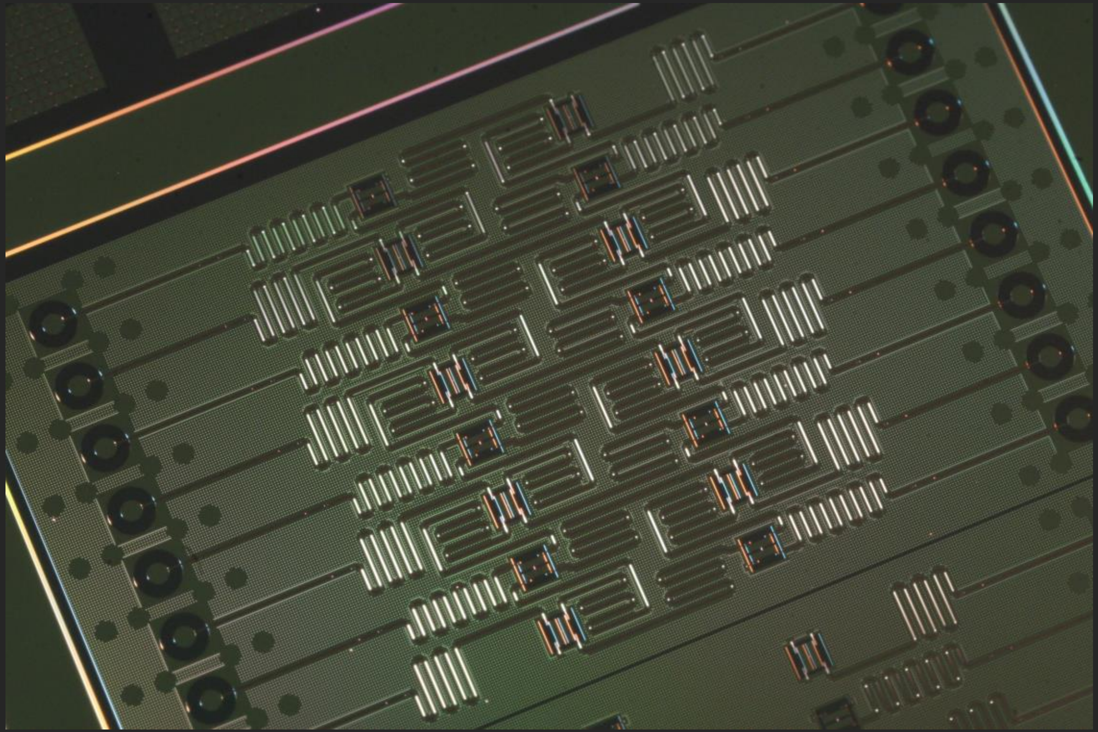
Randomized Benchmarking (XEB)



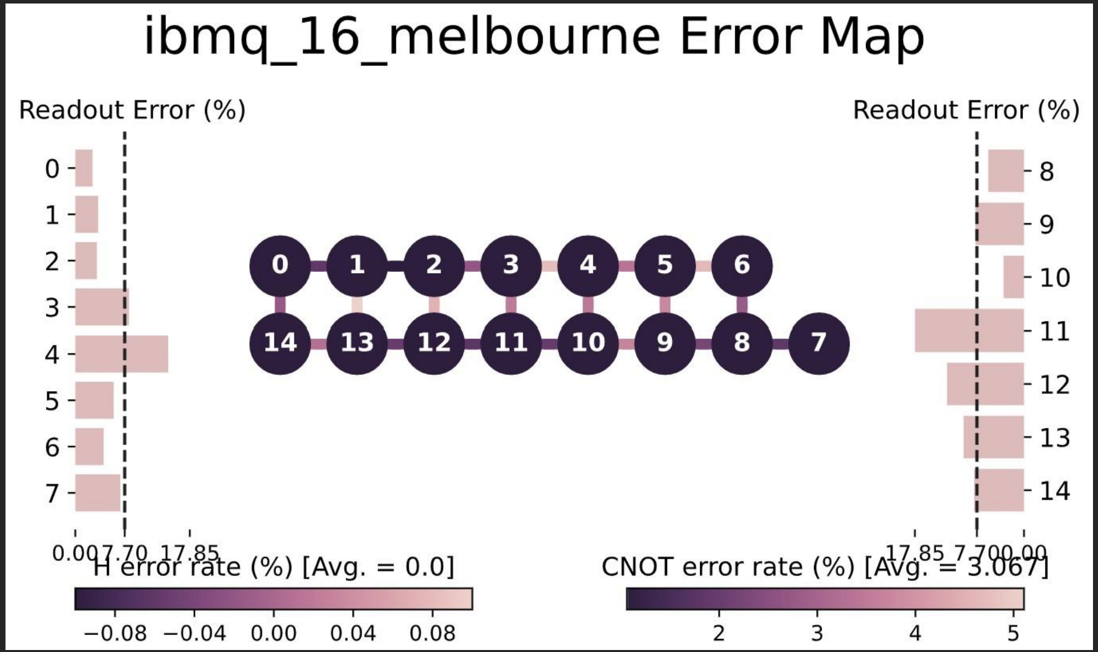
Credit: Nature/Google
https://doi.org/10.1038/s41586-019-1666-5



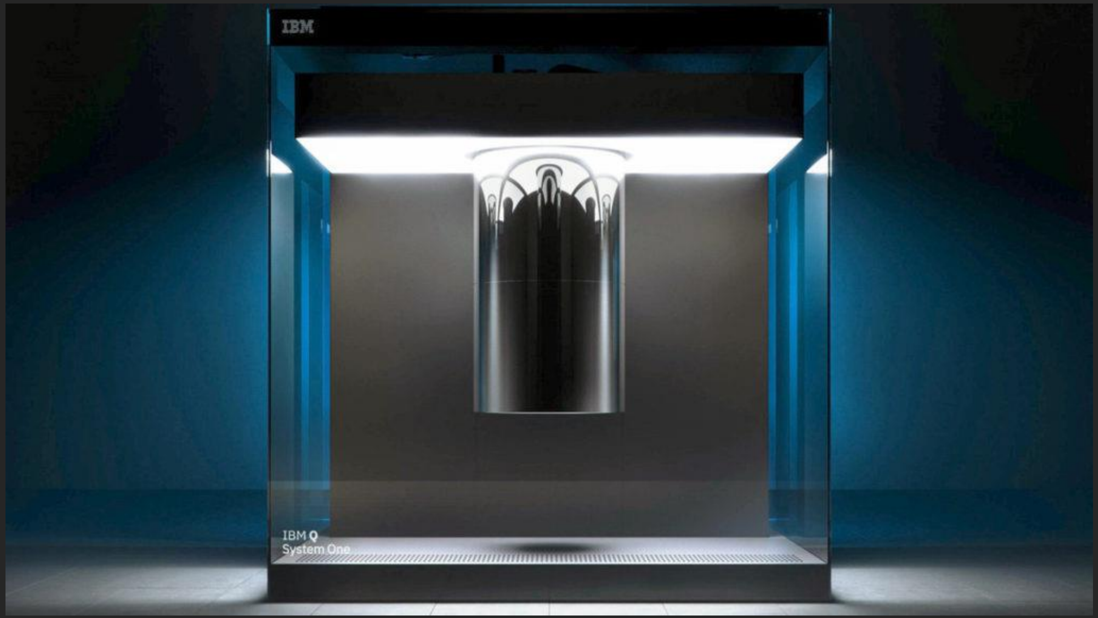
IBMQ Melbourne (16 qubits)



Credit: IBM

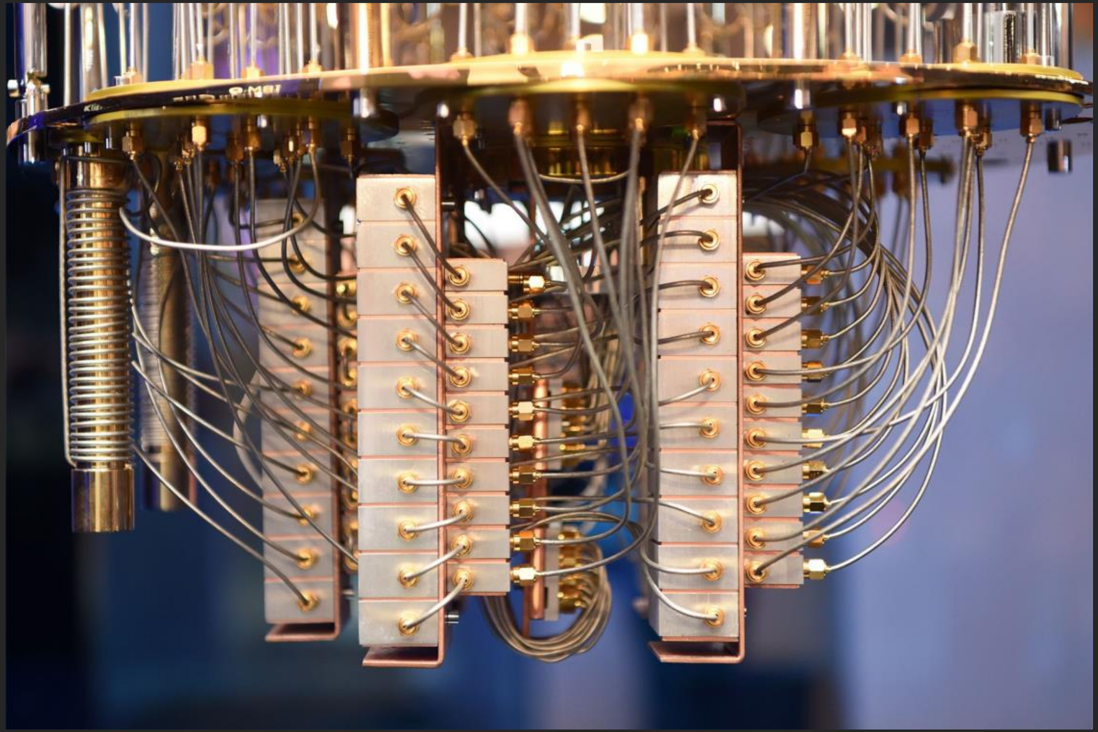


IBMQ System One (20 qubits)



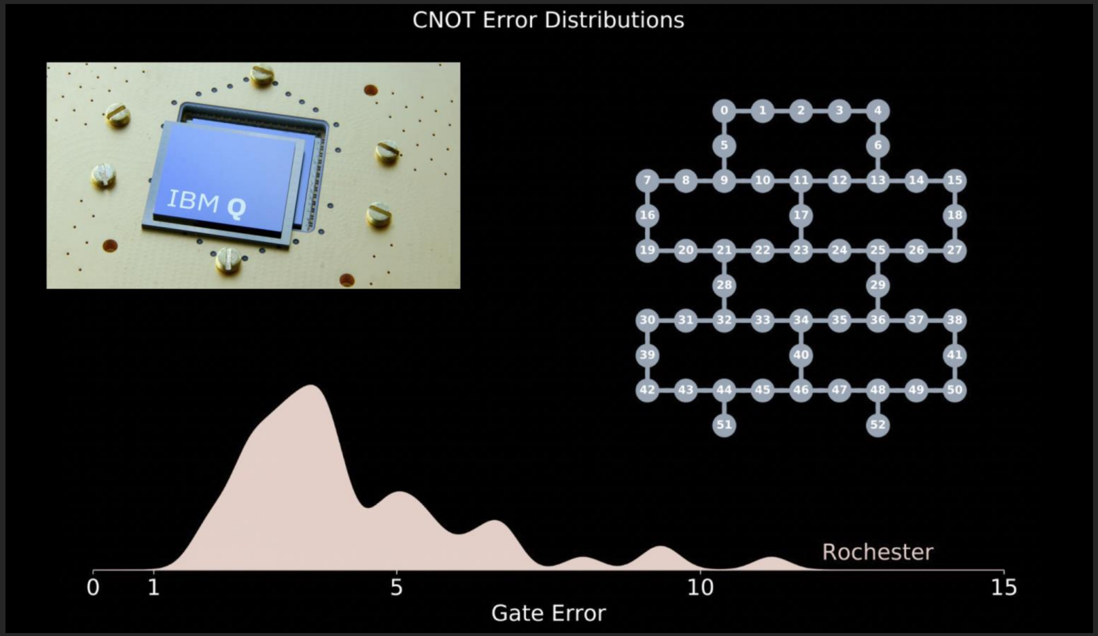
Credit: IBM

IBMQ System One (20 qubits)



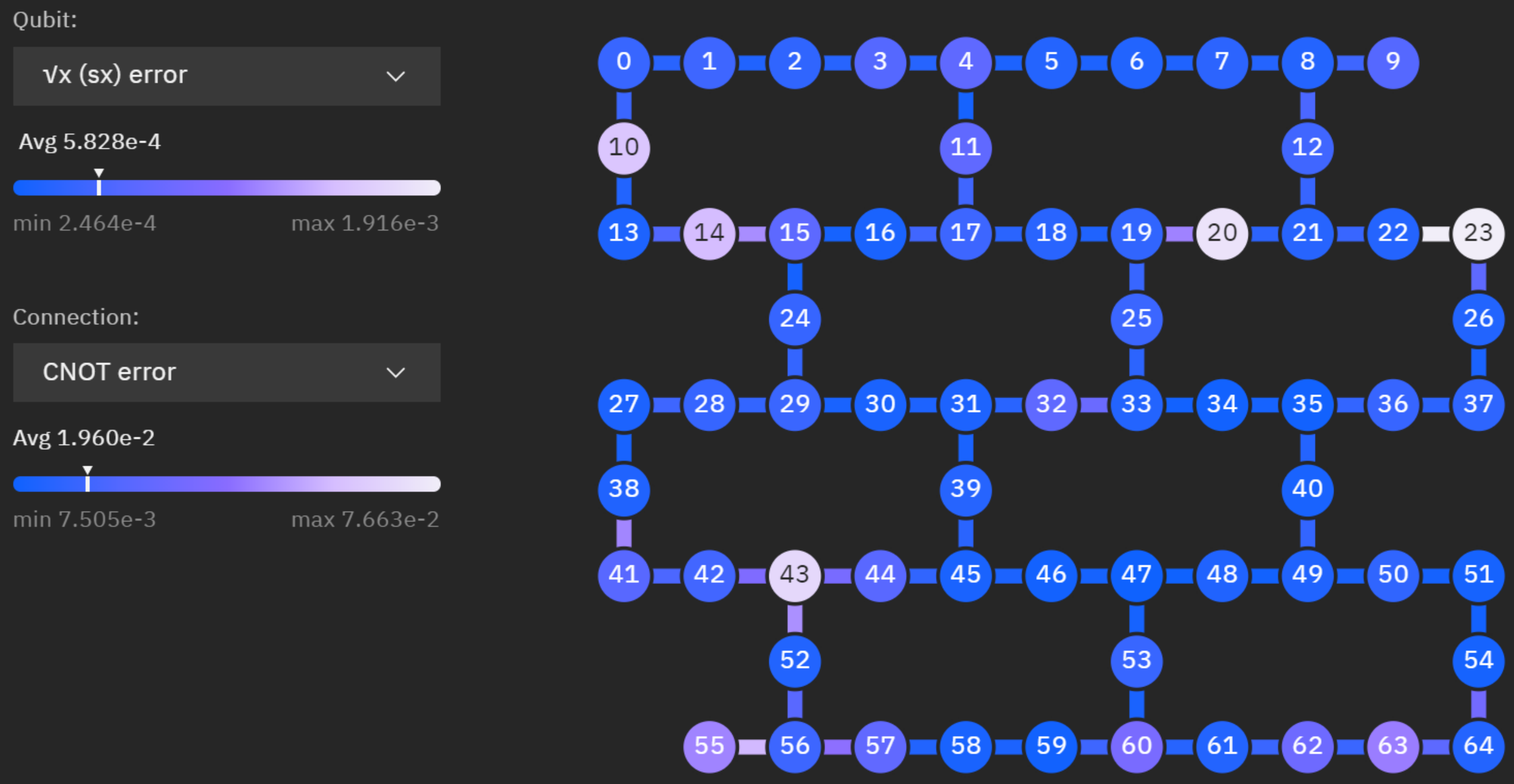
Credit: IBM

IBMQ Rochester (53 qubits)

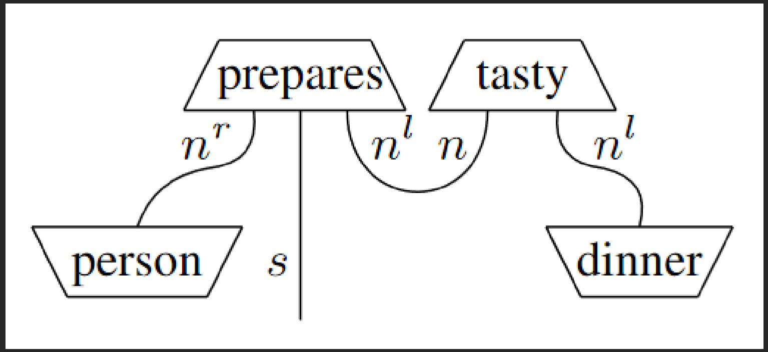


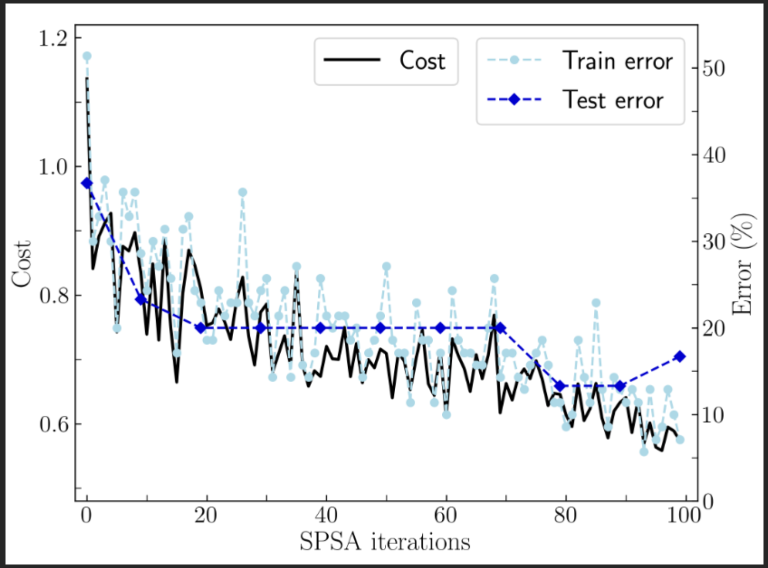
Credit: IBM

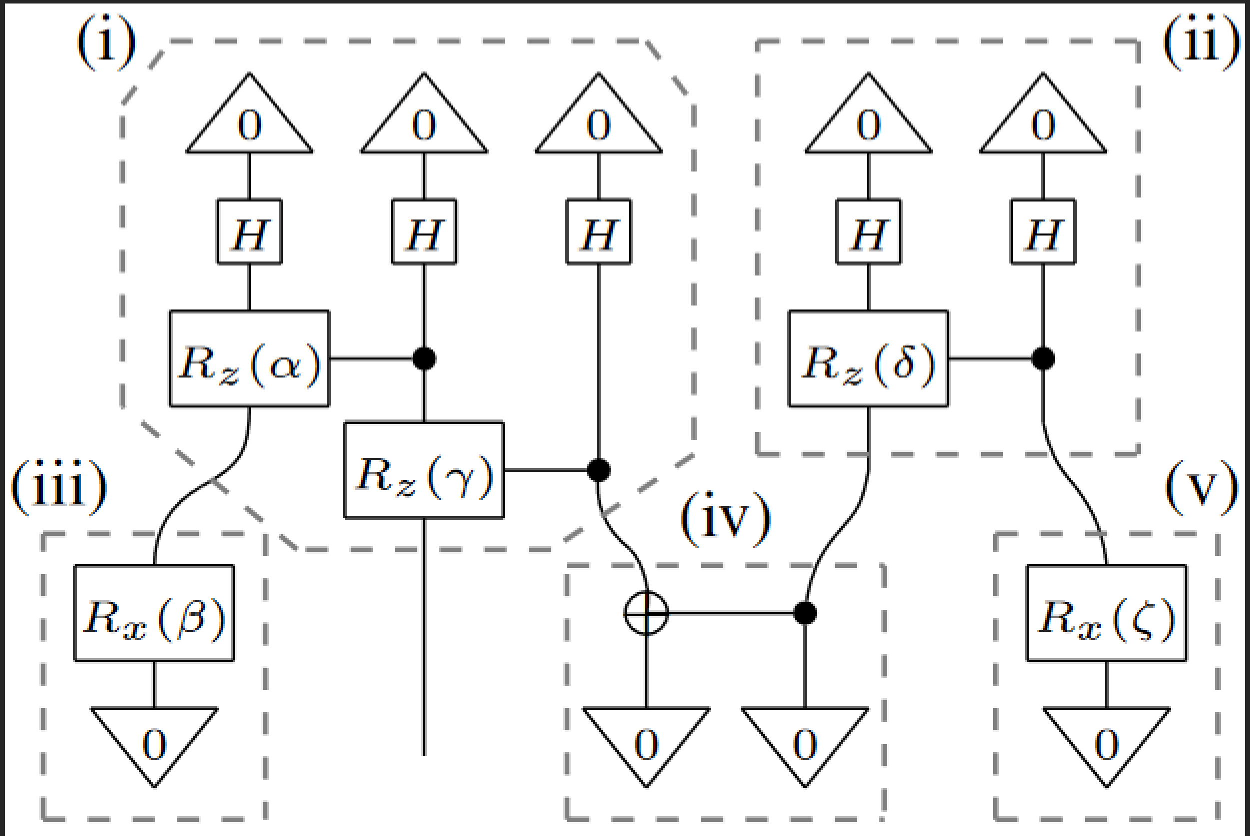
IBMQ Manhattan (65 qubits)



Quantum Natural Language Processing

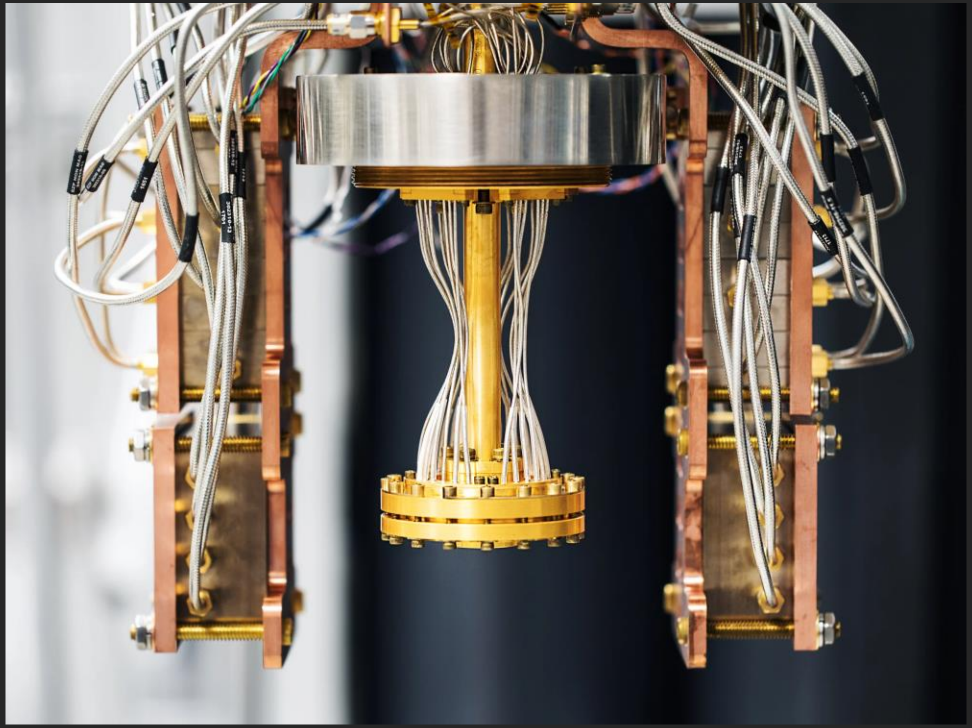


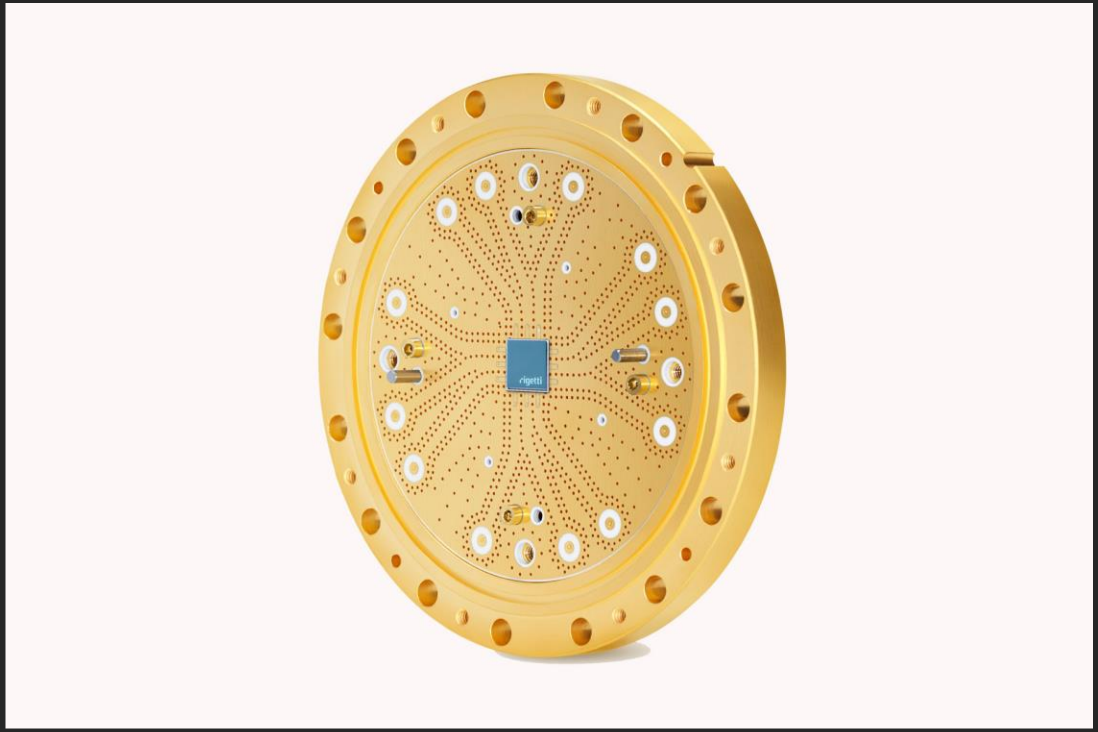




Credit: Cambridge Quantum Computing (on IBMQ)
https://arxiv.org/abs/2102.12846

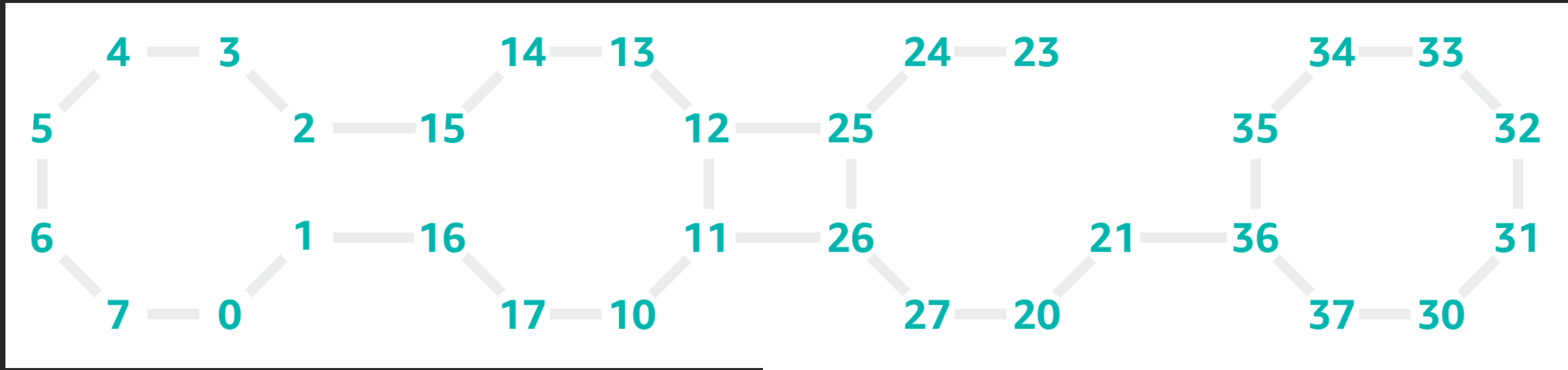
Rigetti Acorn (19 qubits)





Credit: Rigetti

Rigetti Aspen-9 (31 qubits)



Credit: Amazon AWS/Rigetti

Credit: Rigetti

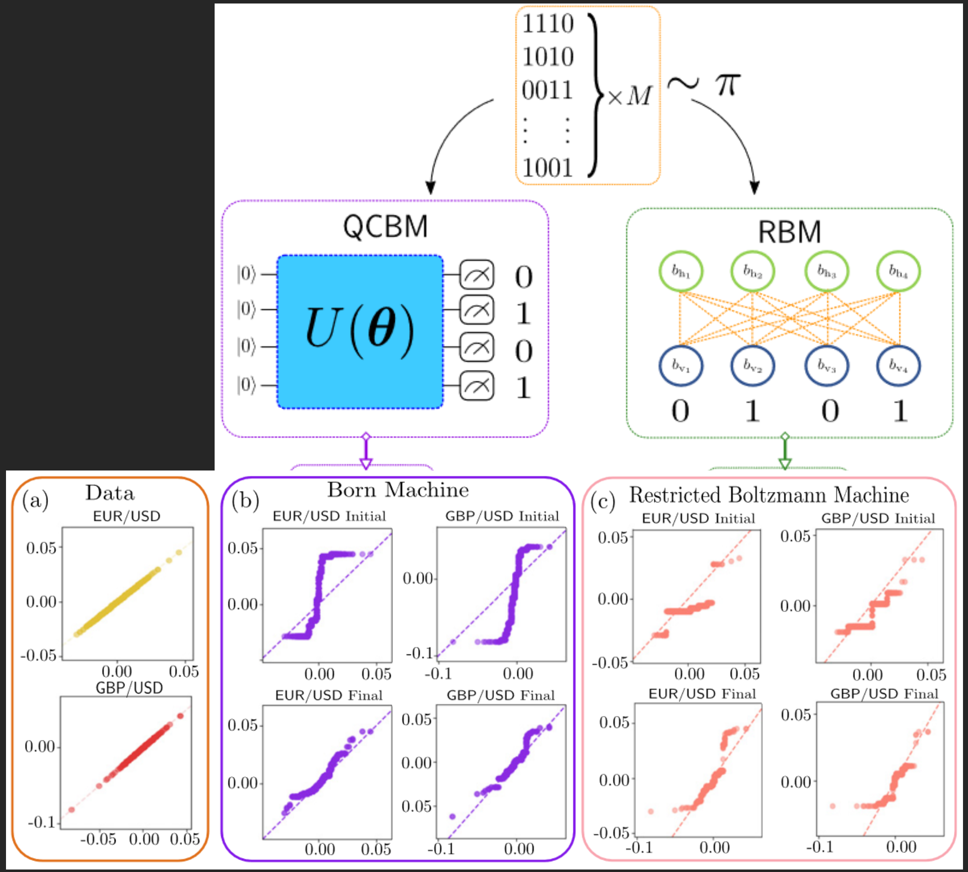
System

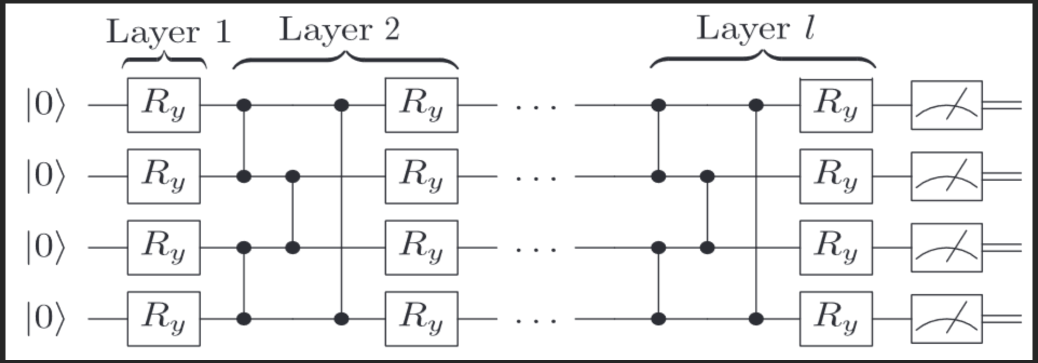
Architecture	Aspen
Qubits on device	32
Rep rate	50 kHz
Date deployed	May 20, 2020

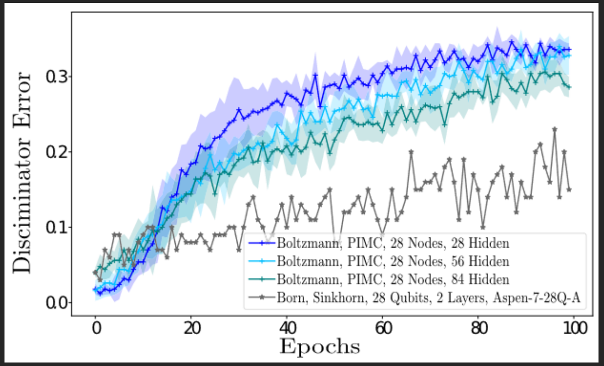
Performance Snapshot

Median T1	30.1 μs
Median T2	17.5 μs
Median Sim 1Q Fidelity	99.3%
Median 2Q XY Fidelity	93.6%
Median 2Q CZ Fidelity	92.6%
Median RO Fidelity	95.3%
Median Active Reset Fidelity	99.4%

Generative Financial Modelling

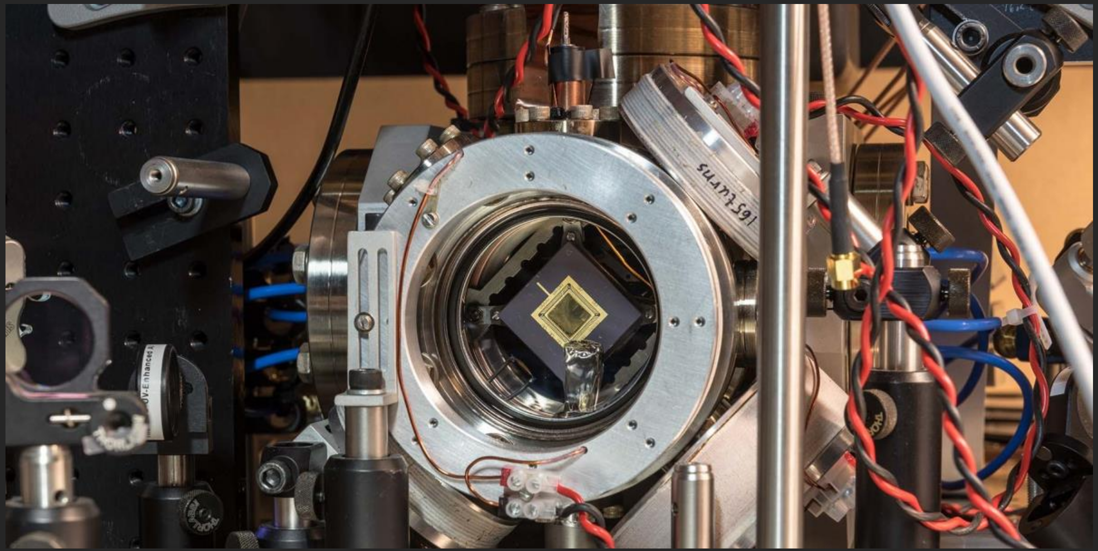




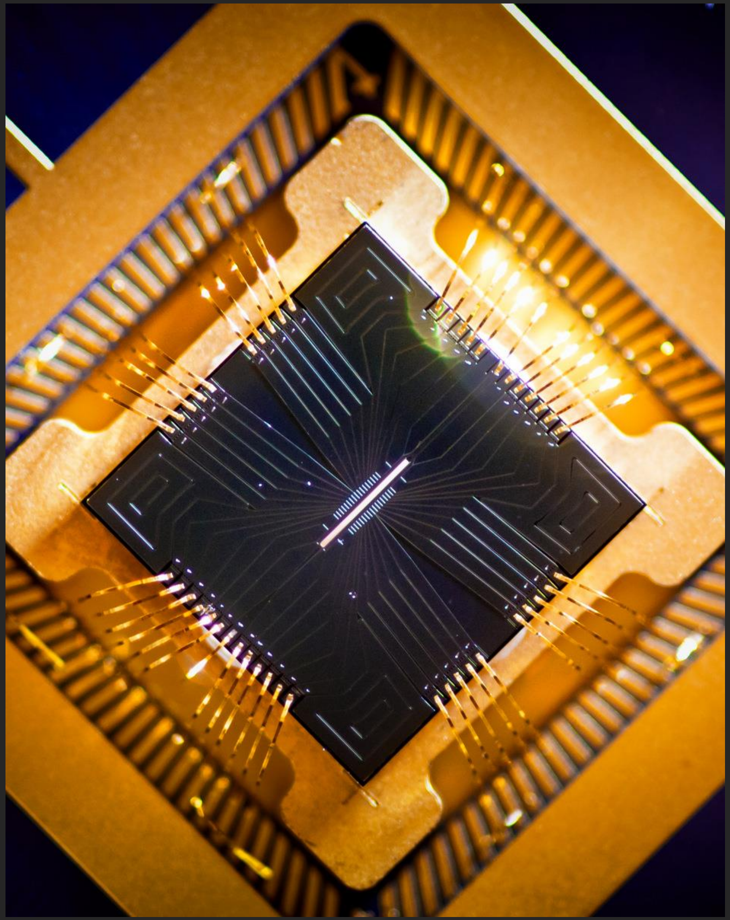


Credit: IOP/various institutions (on Rigetti HW)
https://doi.org/10.1088/2058-9565/abd3db

Ion-trap Quantum Computers

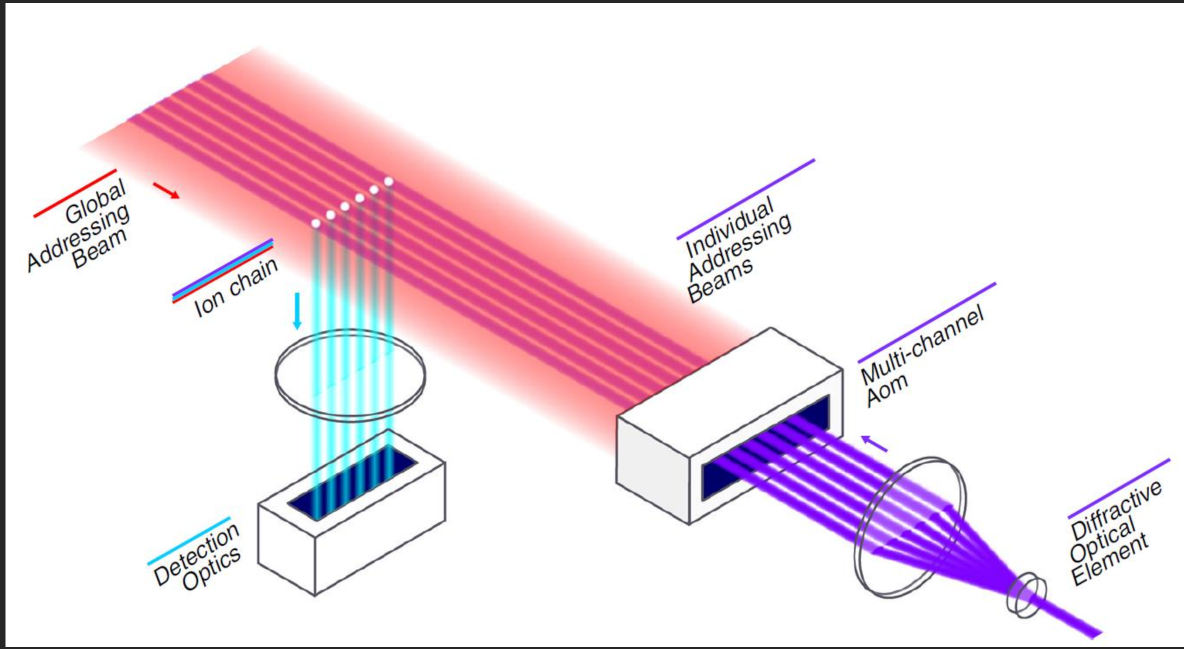


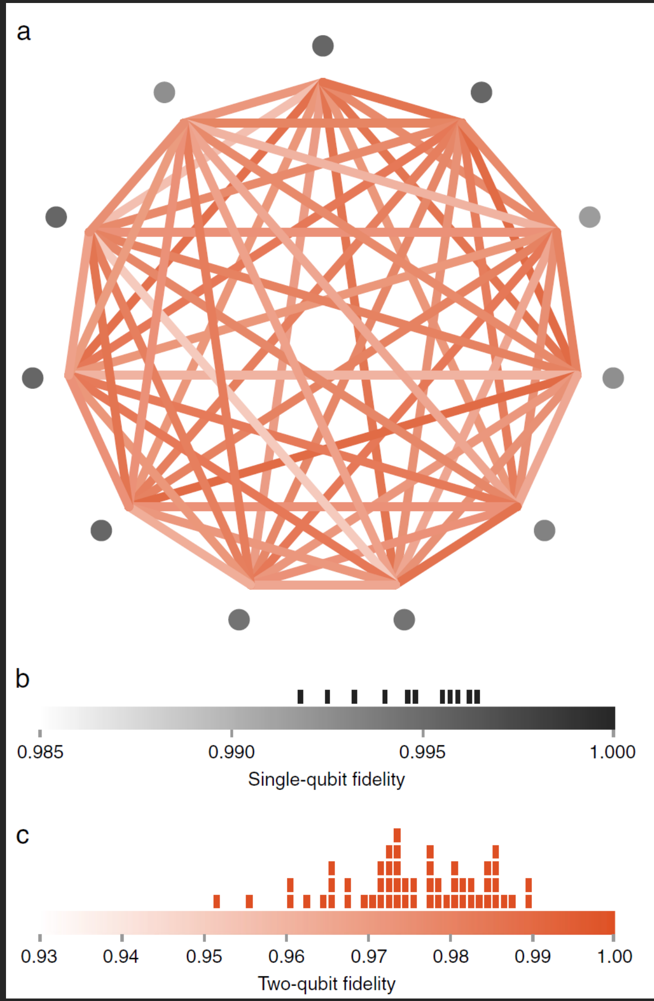
Credit: NQIT/Stuart Bebb



Credit: JQI

IonQ 171Yb+ (11 qubits)

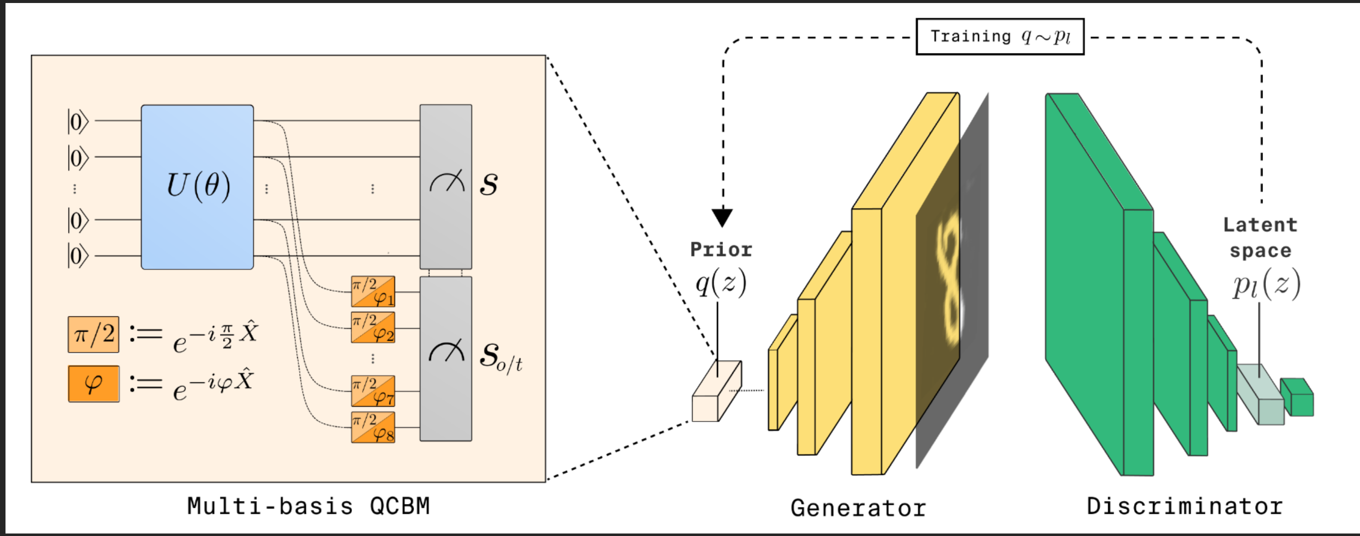




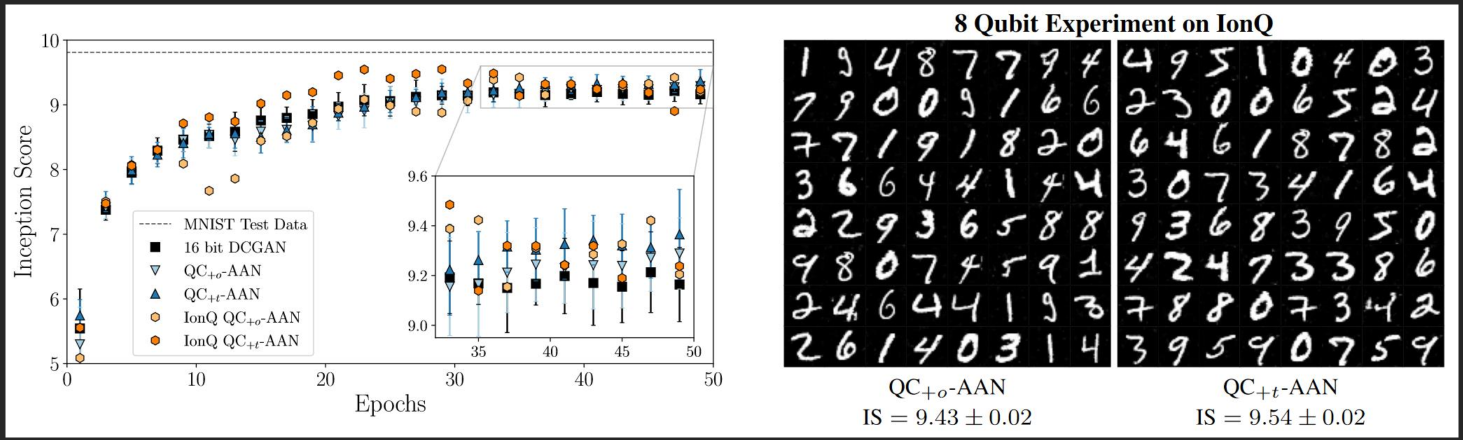
Credit: Nature/IonQ

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

Quantum Generative Adversarial Networks

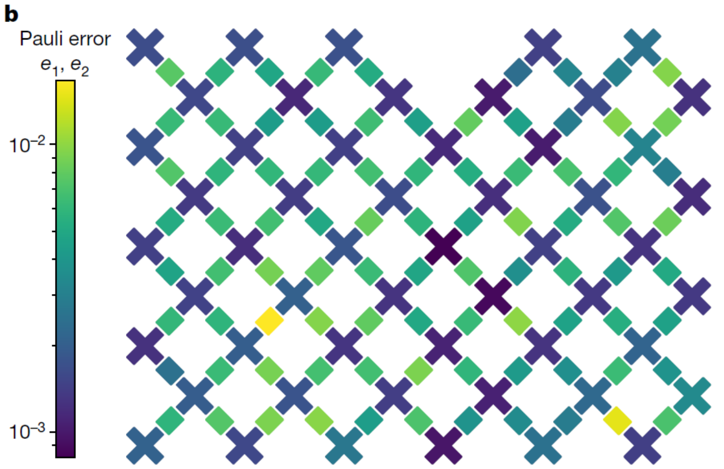


Credit: Zapata Computing/IonQ
https://arxiv.org/abs/2012.03924

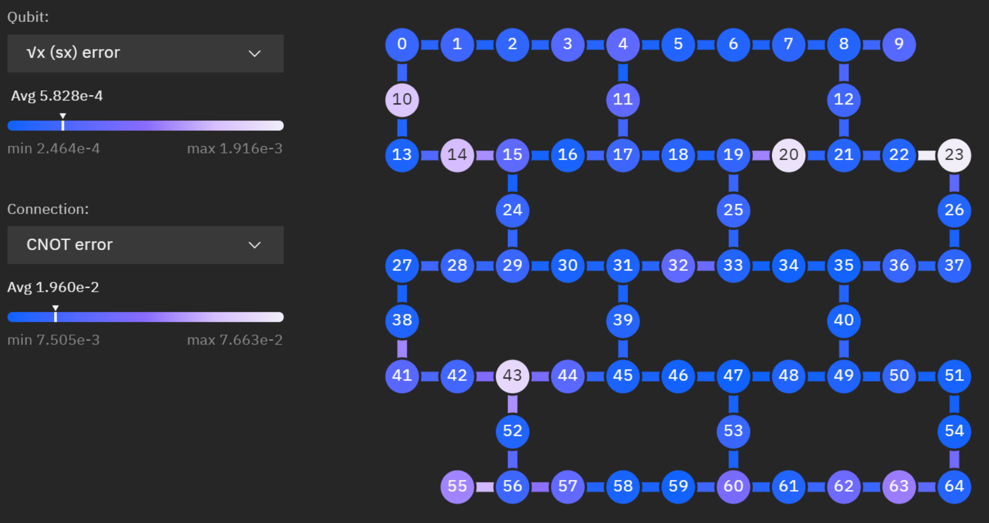


Superconducting vs Ion-trap

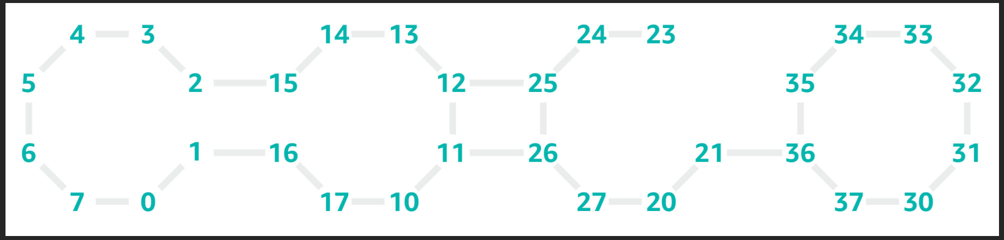
Average error	Isolated	Simultaneous
Single-qubit (e_1)	0.15%	0.16%
Two-qubit (e_2)	0.36%	0.62%
Two-qubit, cycle (e_{2c})	0.65%	0.93%
Readout (e_r)	3.1%	3.8%



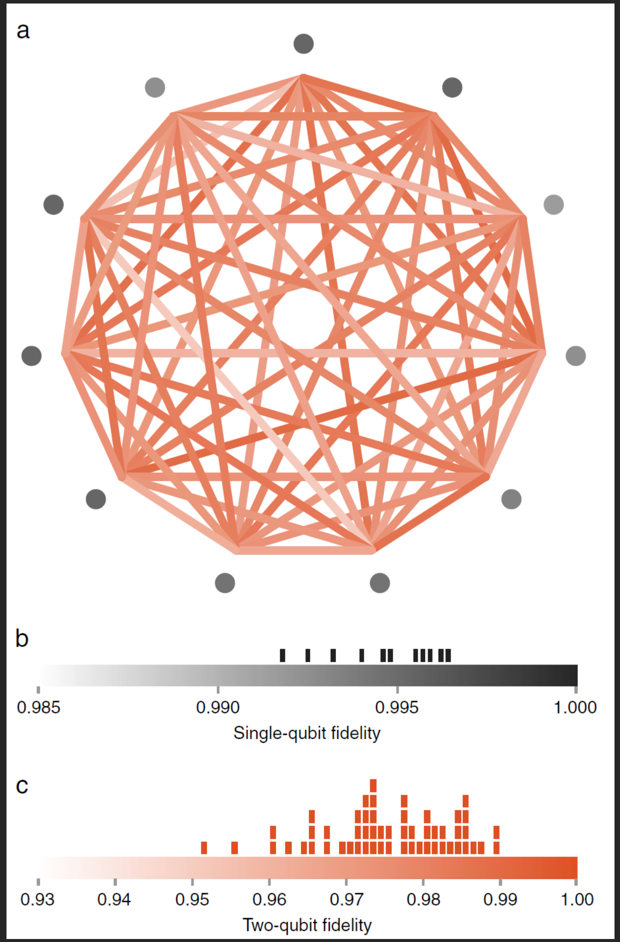
Credit: Nature/Google
https://doi.org/10.1038/s41586-019-1666-5



Credit:IBMQ

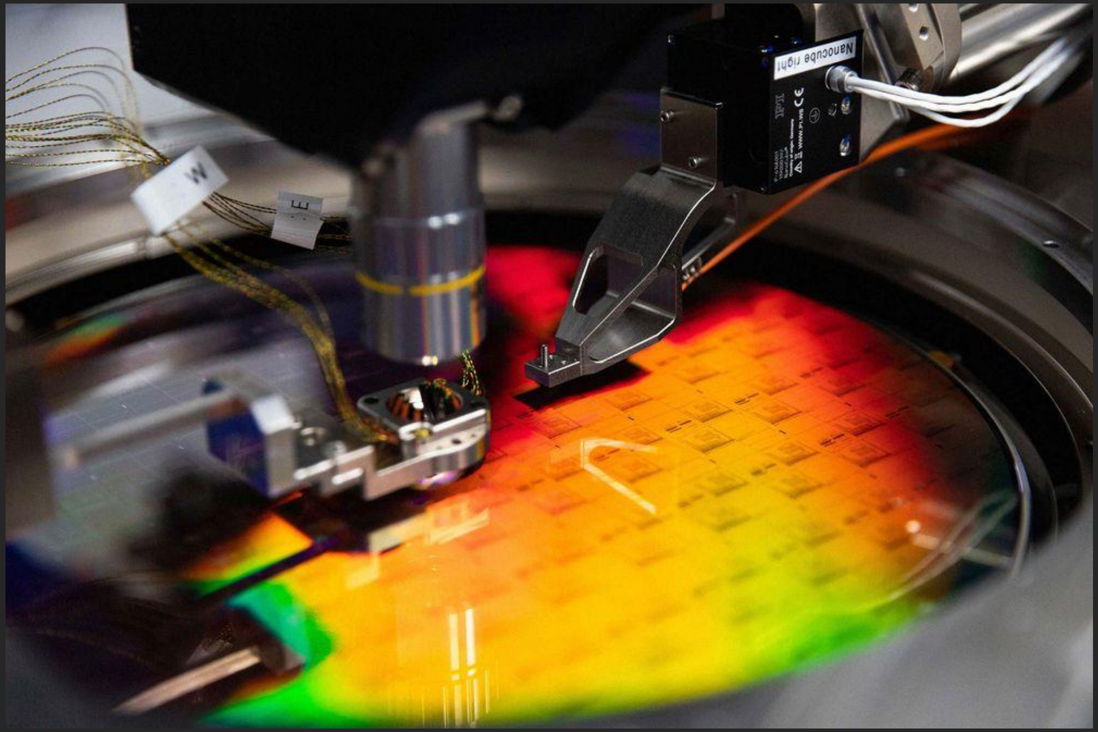


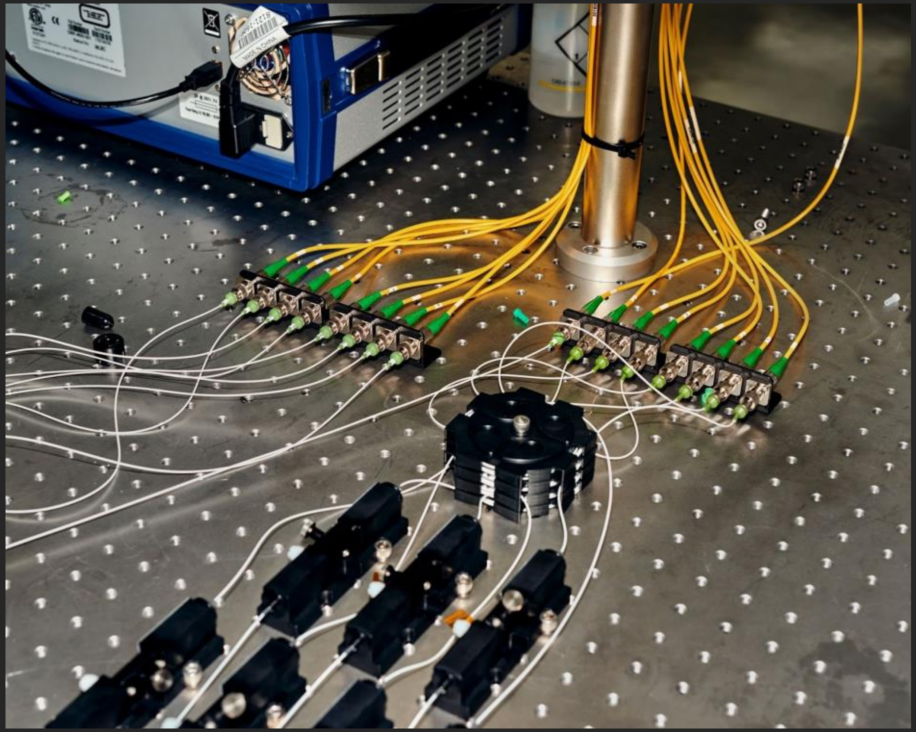
Credit: Amazon AWS/Rigetti



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

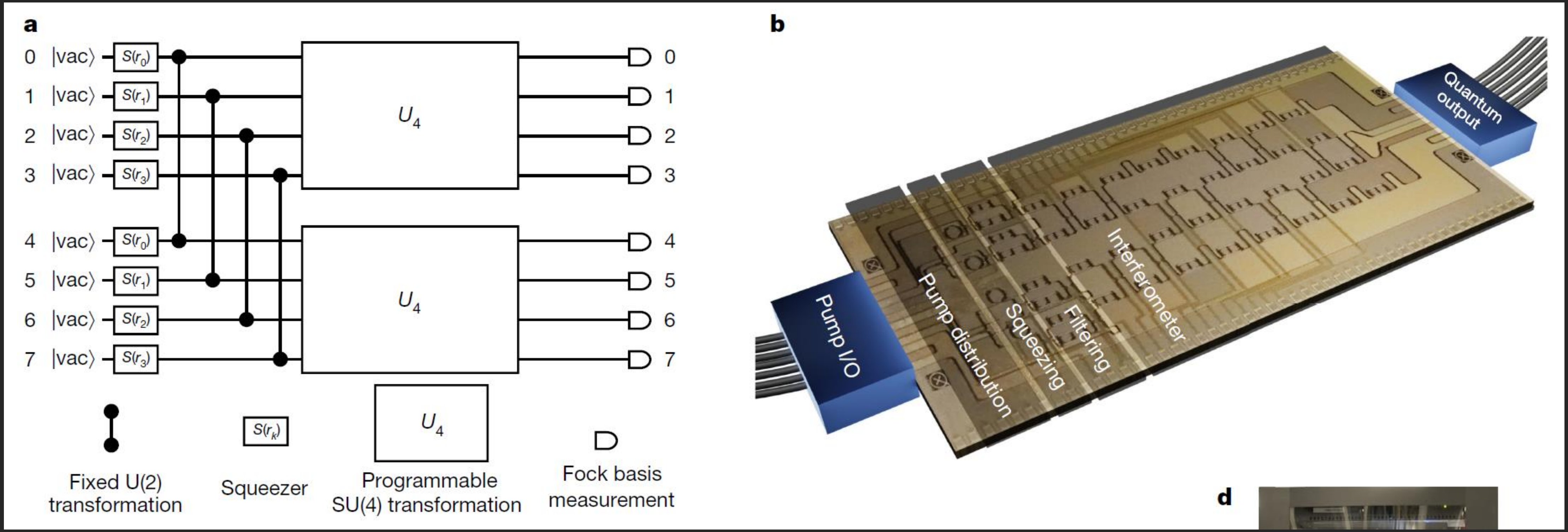
Photonic Quantum Computers





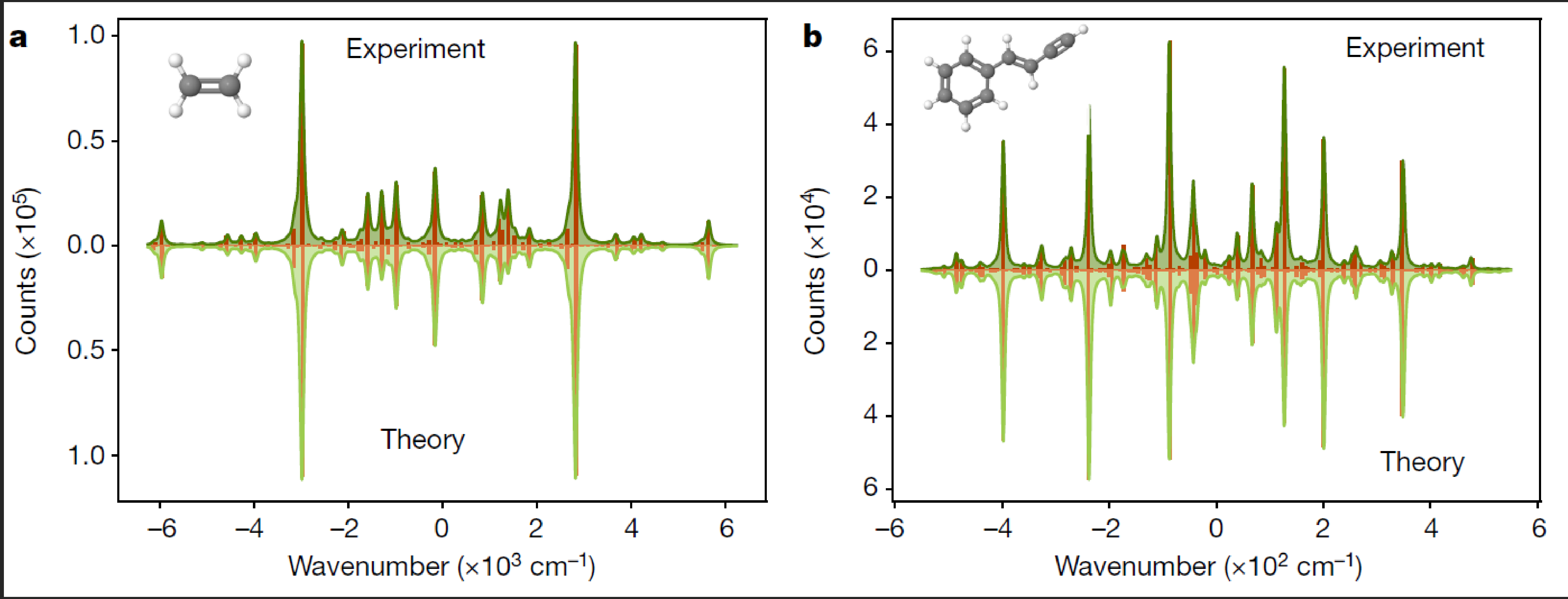
Credit: PsiQuantum

Xanadu 8-mode photonic chip



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

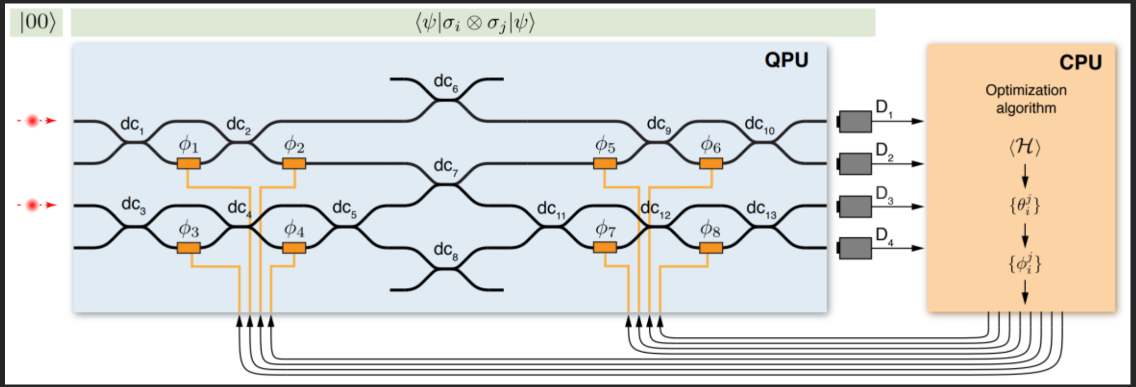
Quantum Chemistry Simulations

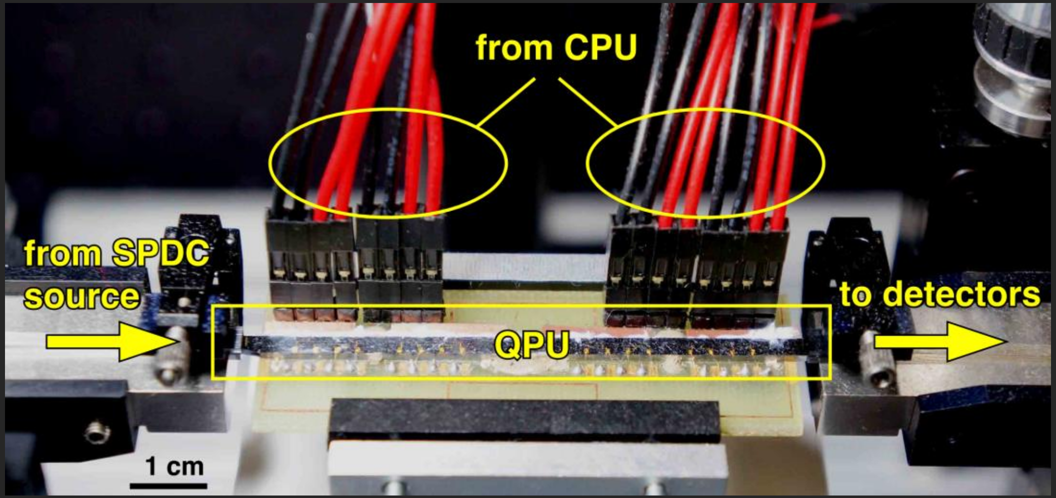


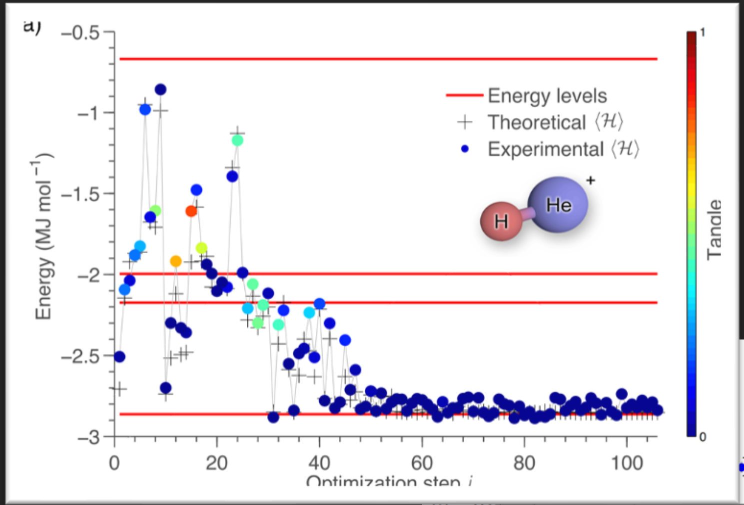
Credit: Nature/Xanadu

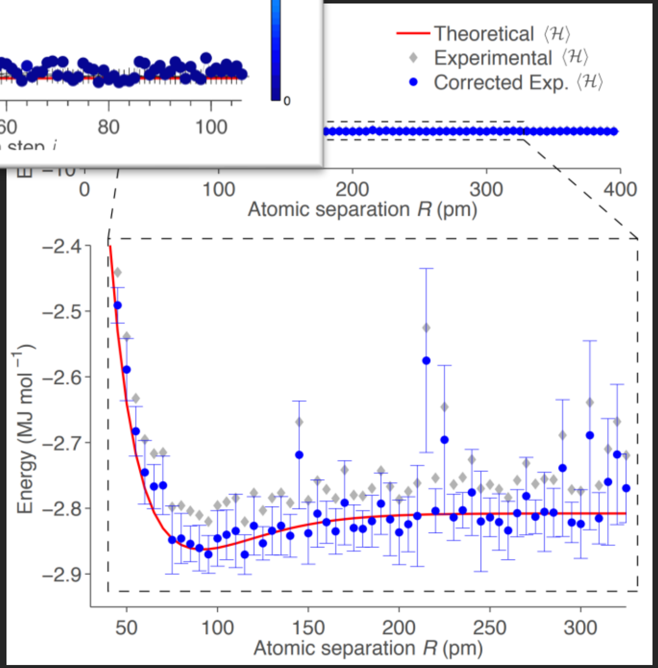
https://doi.org/10.1038/s41586-021-03202-1

Quantum Chemistry Simulations









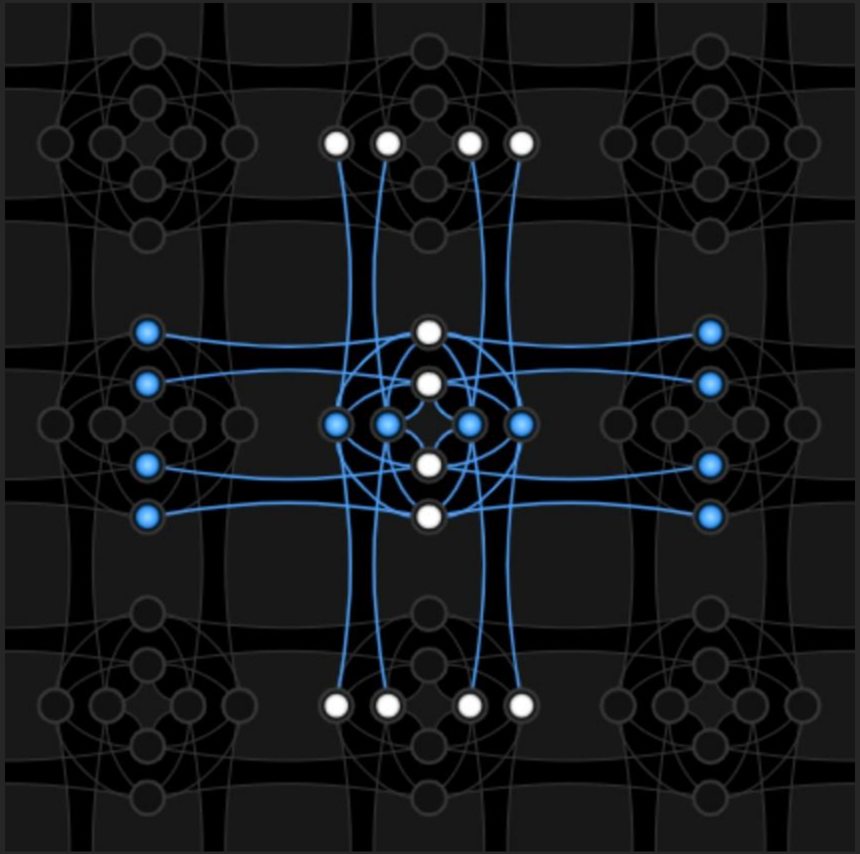
This is a photonic QPU running VQE in 2003… Ancient QC history.

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

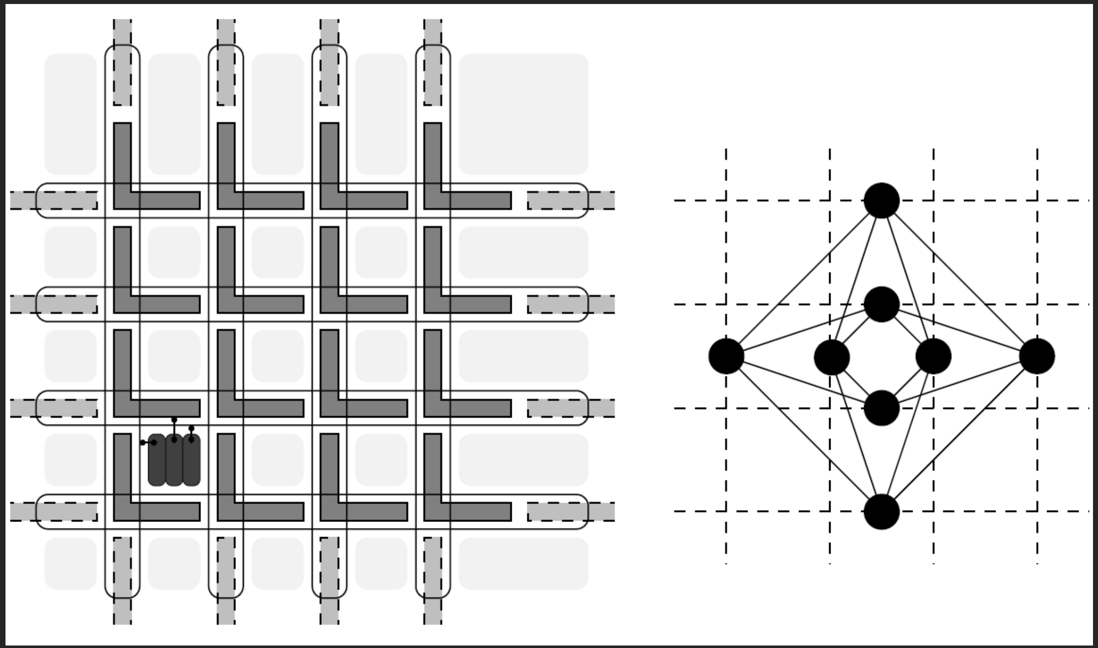
Quantum Annealers

Credit: D-Wave

D-Wave 2000Q (2048 qubits, Chimera topology)

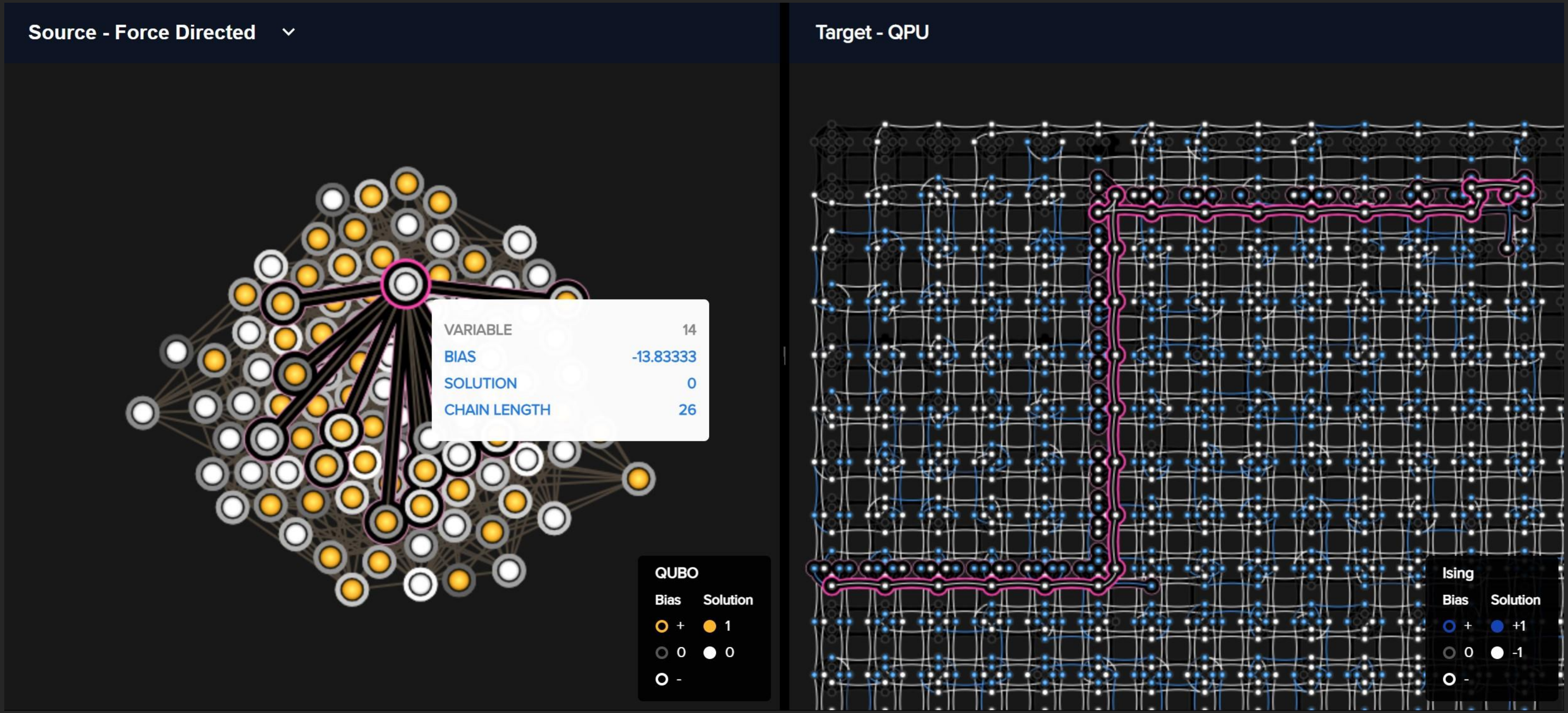


Credit: D-Wave



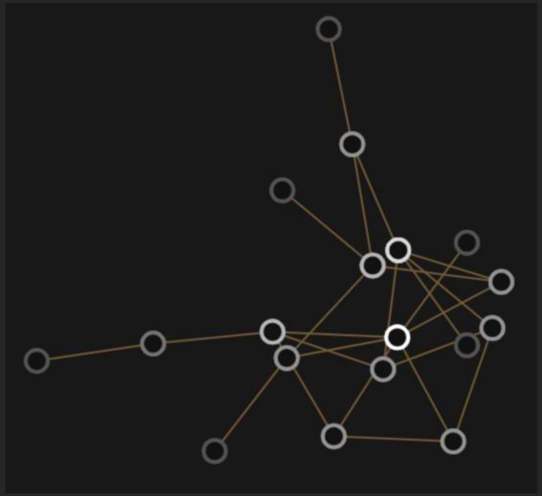
https://doi.org/10.1109/TASC.2014.2318294

Quantum Annealers

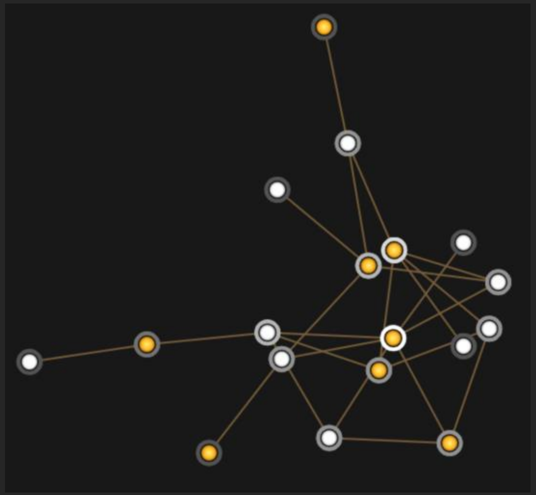


Credit: D-Wave

Example: Max-cut with QA

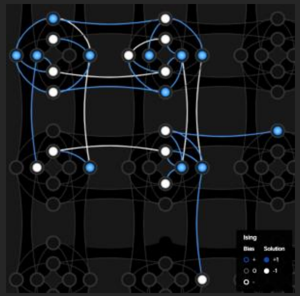


Graph

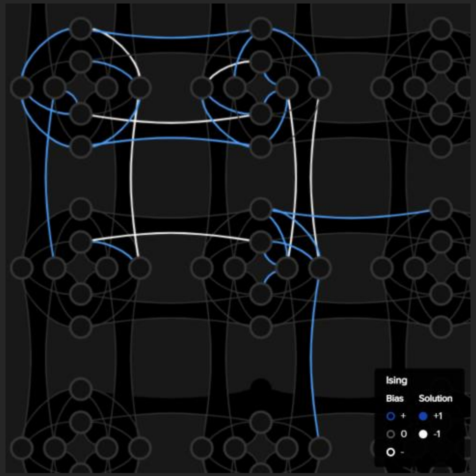


Graph Cut

Min energy solution

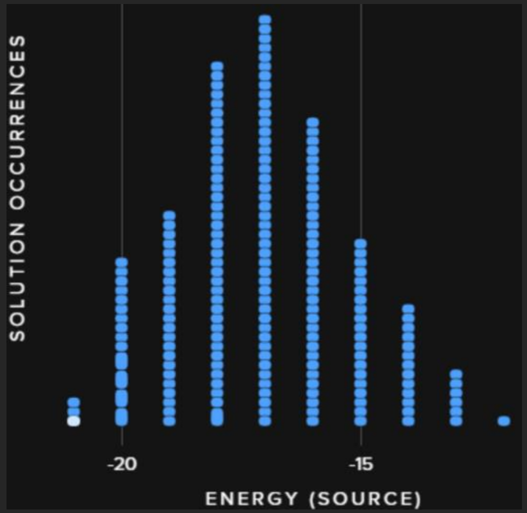


Encoding as Ising model



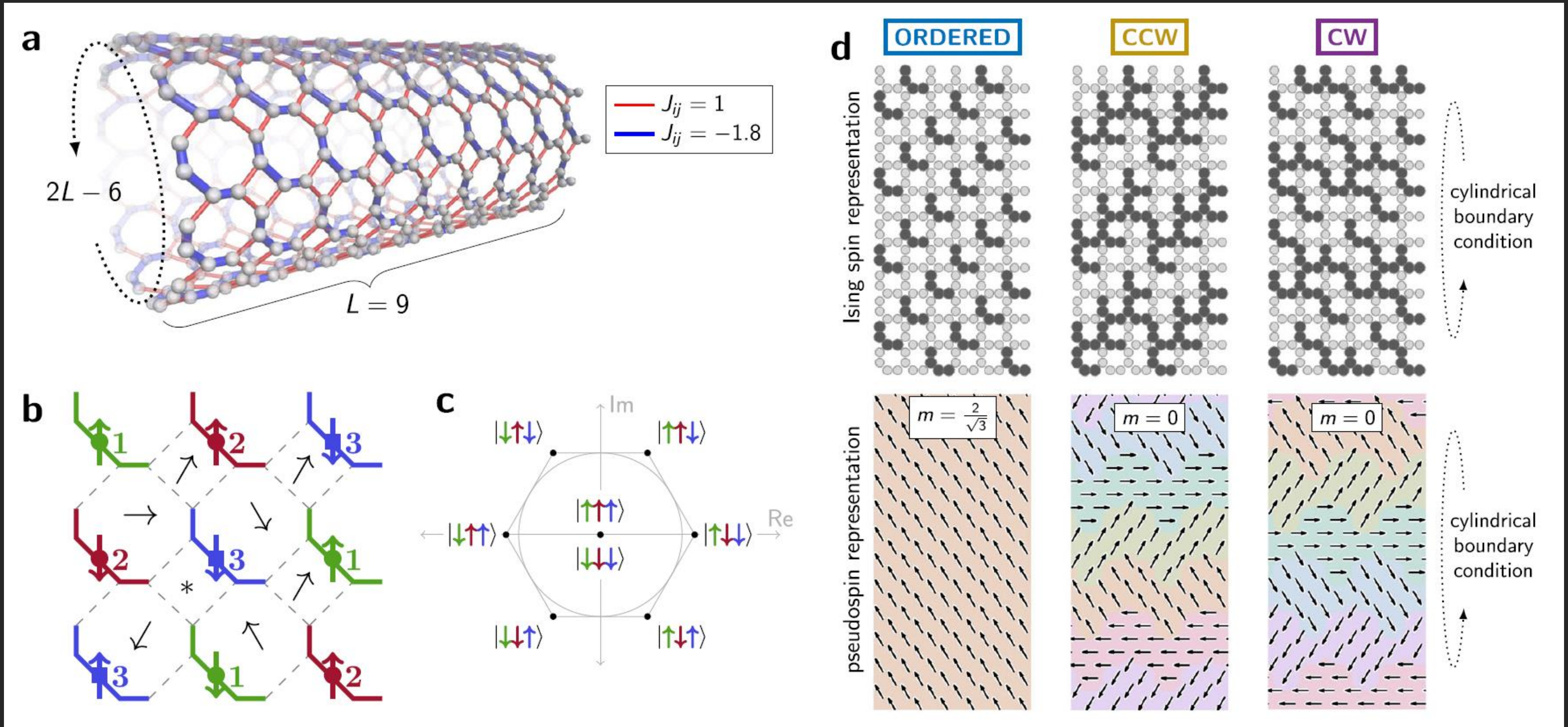
NAME (CHIP ID)	DESCRIPTION
DW_2000Q_6	D-Wave 2000Q lower-noise system
QUBITS	SUPPORTED PROBLEM TYPES
2048	ising, qubo
TOPOLOGY	TAGS
[16,16,4] chimera	lower_noise

QPU



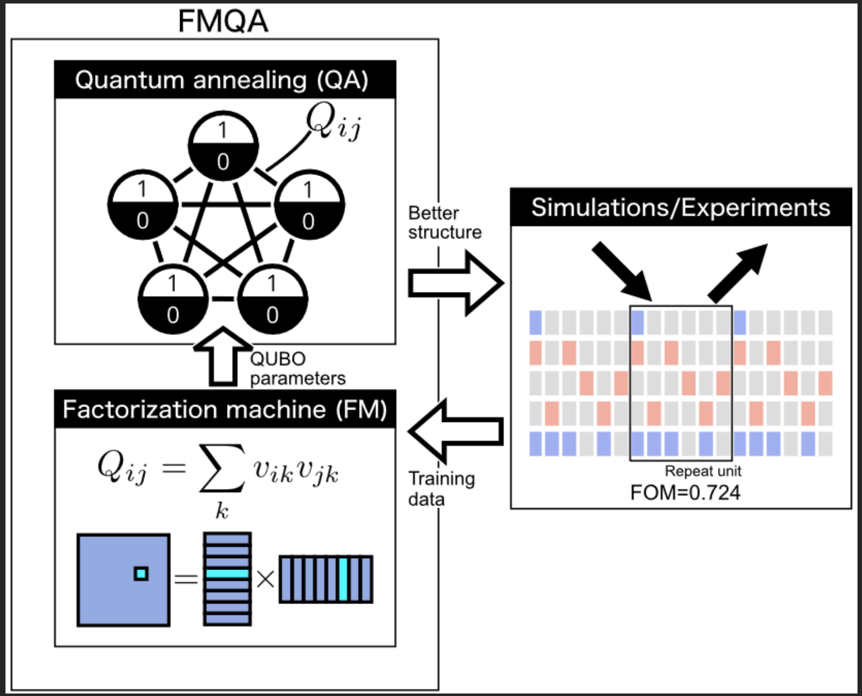
Samples

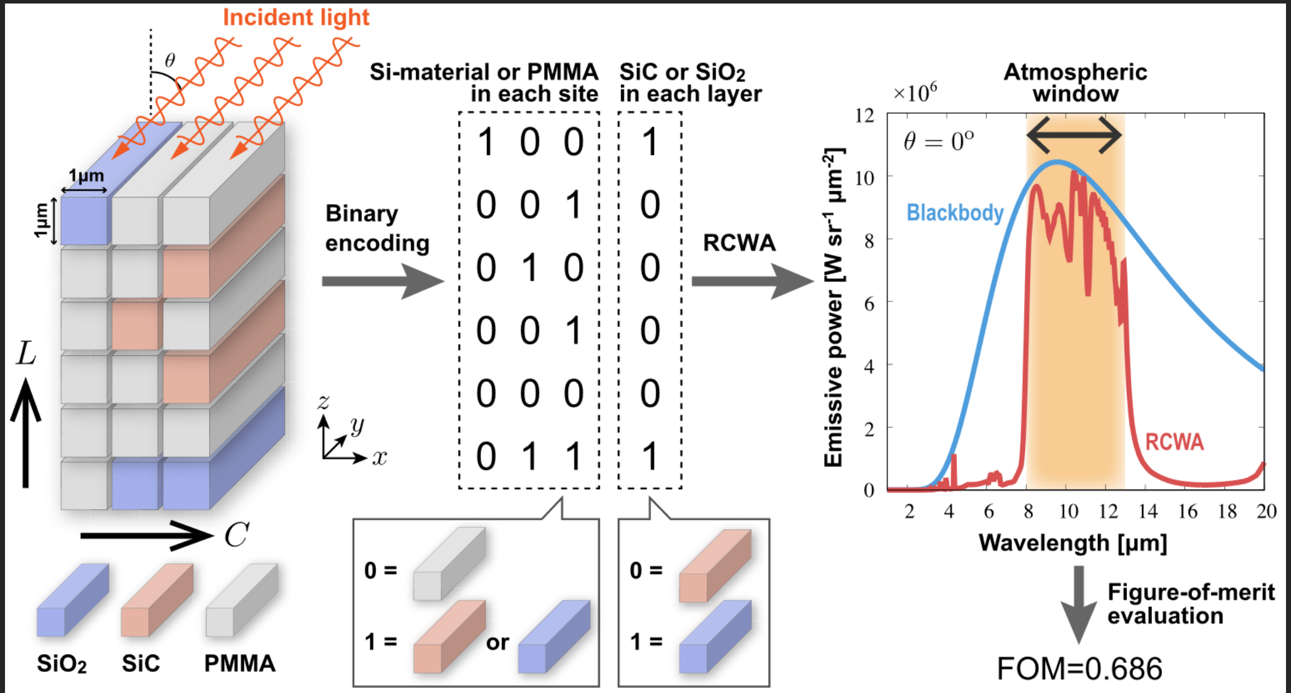
Spin-glass Simulations



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

Metamaterial Design





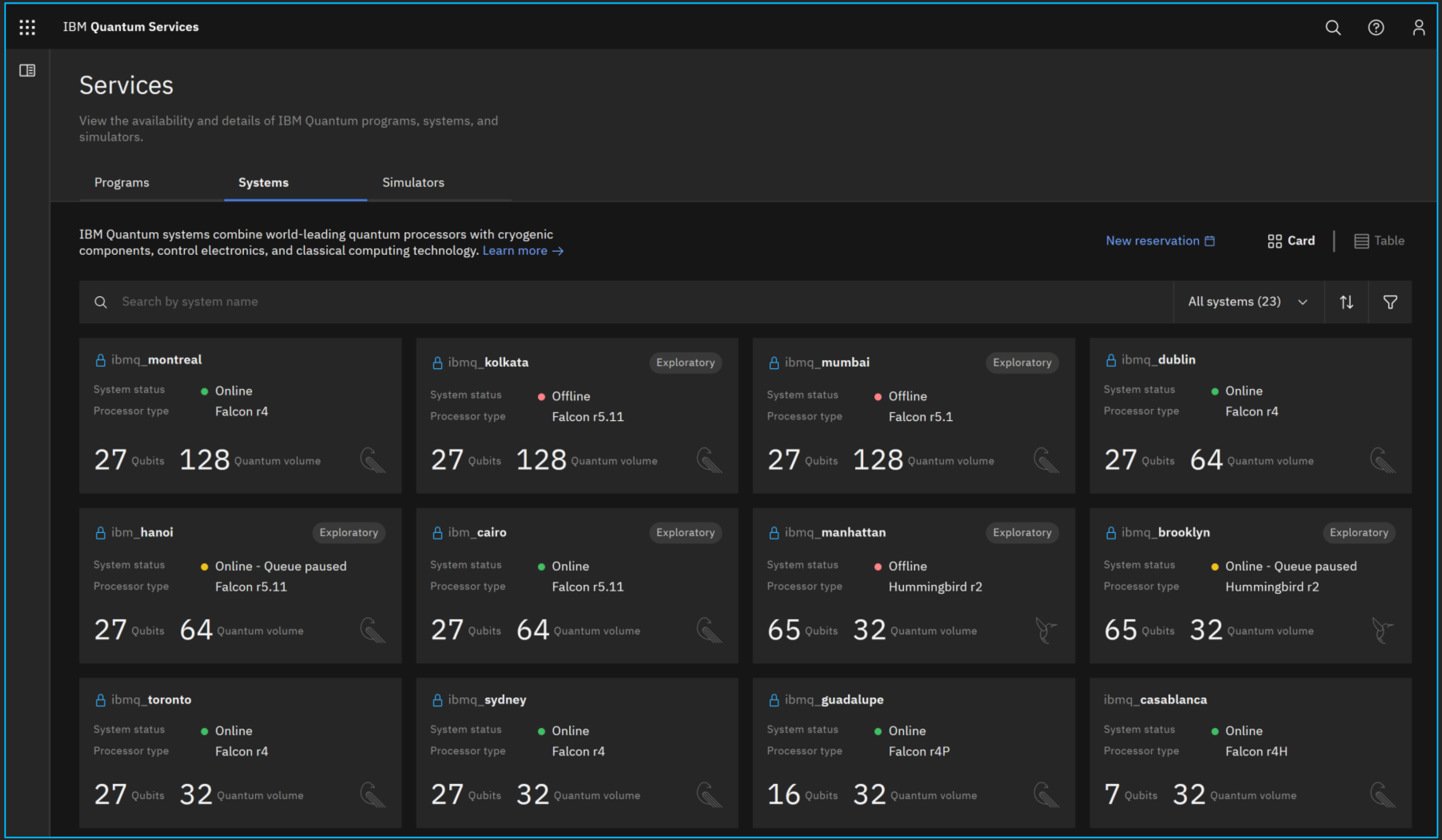
Credit: APS/various institutions (on D-Wave HW)
https://doi.org/10.1103/PhysRevResearch.2.013319

QKD and QRNG (not QC)

Credit: ID Quantique

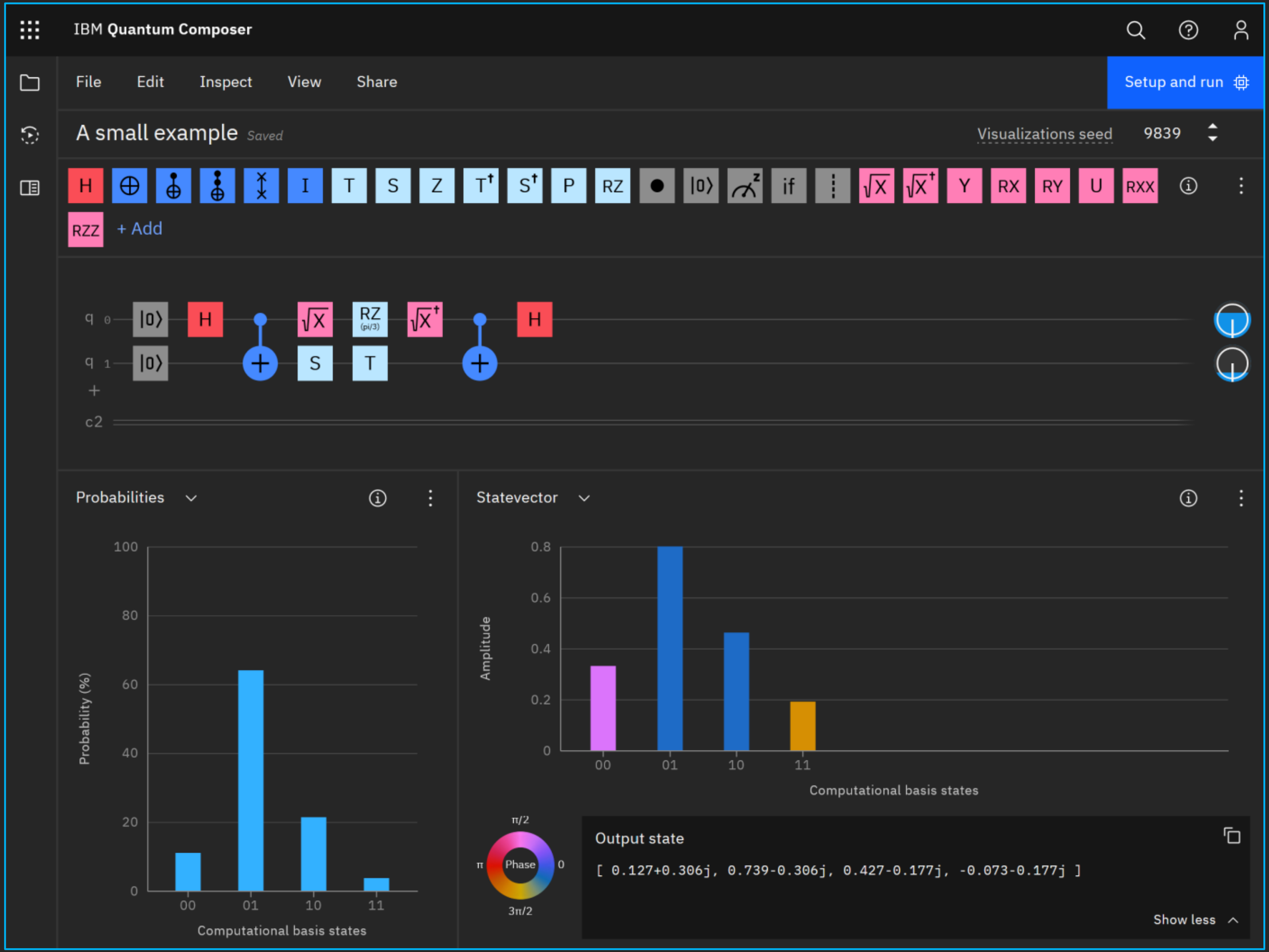
Software Providers

IBM Quantum



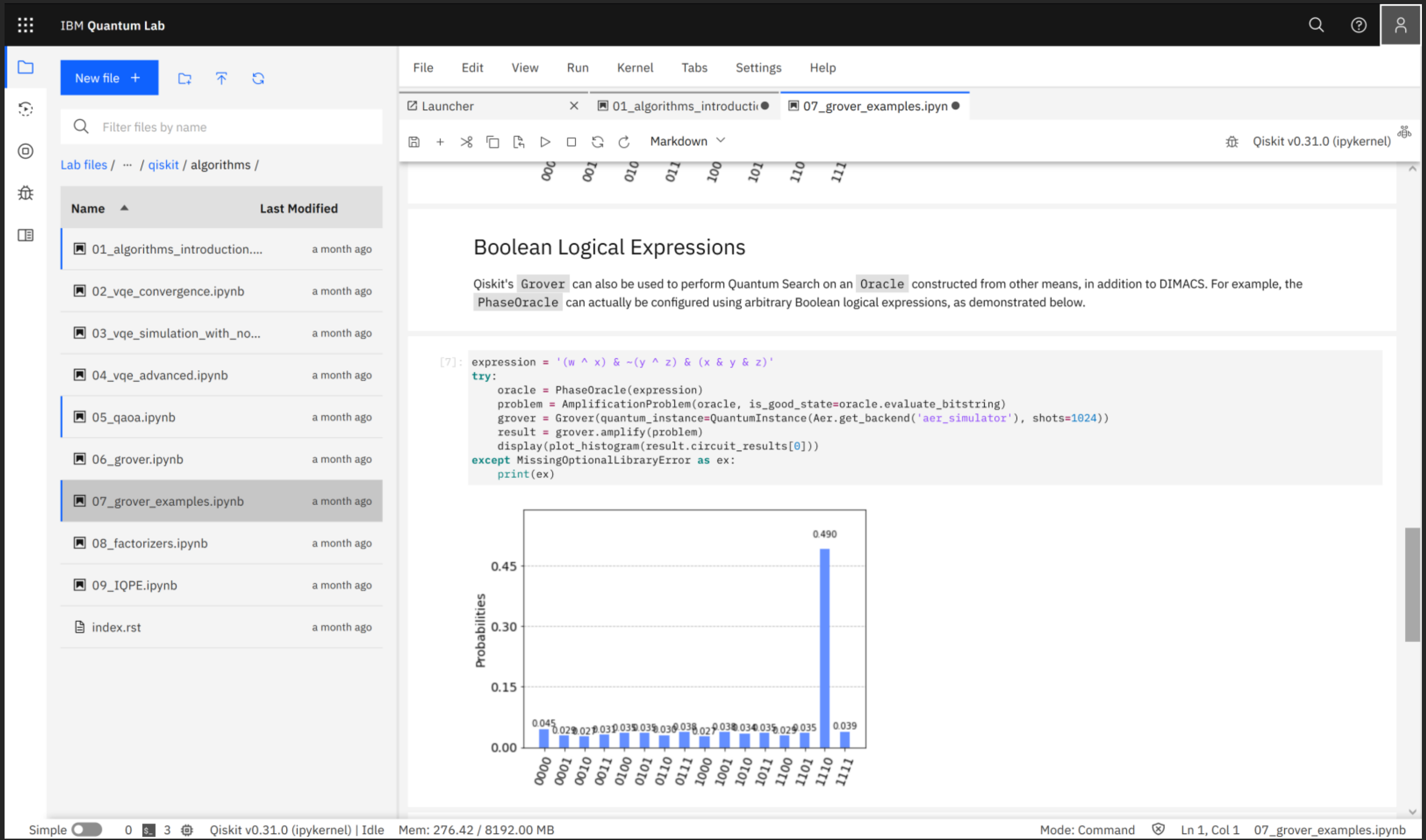
Credit: [IBM - Quantum Services]

IBM Quantum



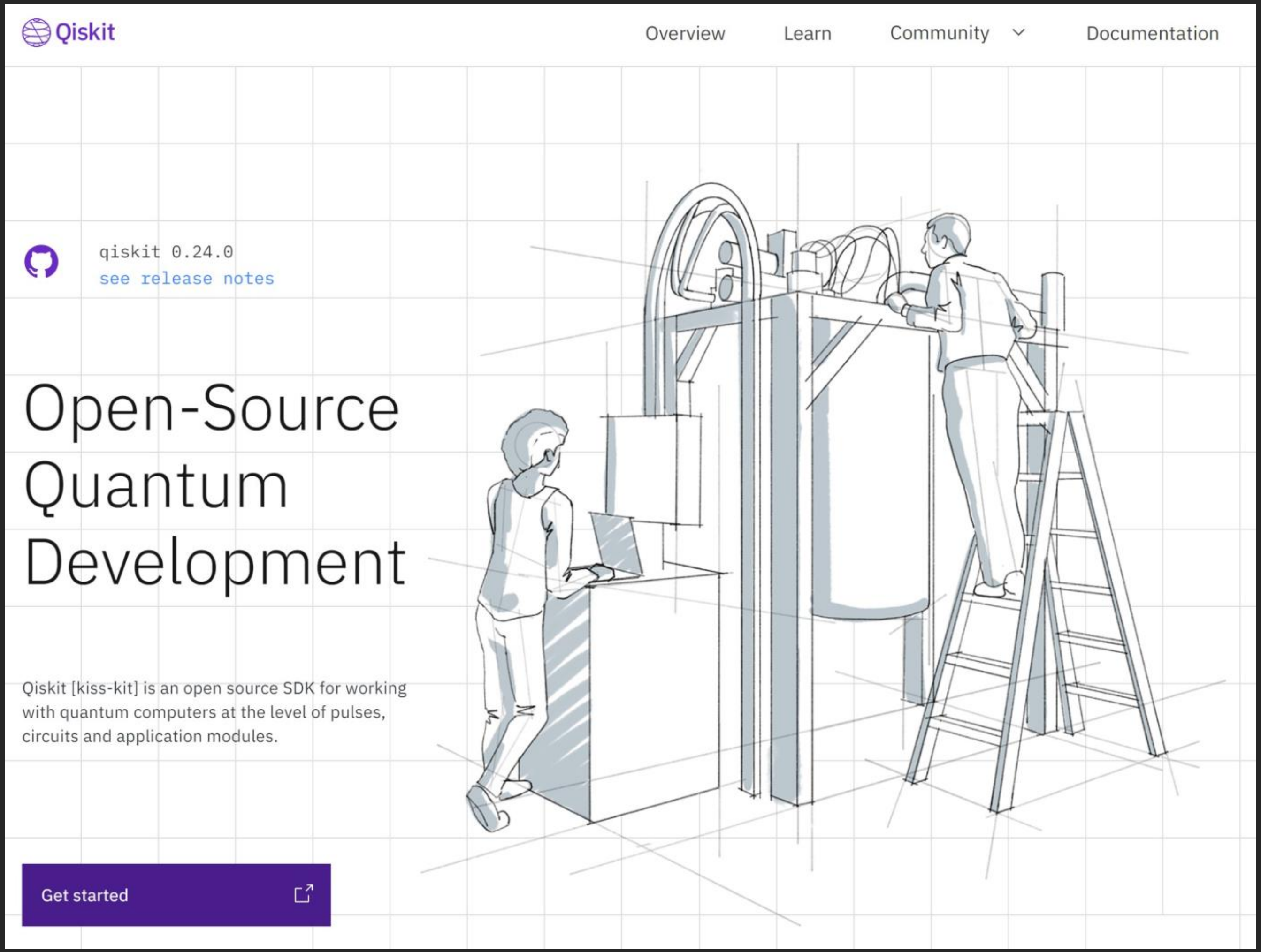
Credit: [IBM - Quantum Composer]

IBM Quantum



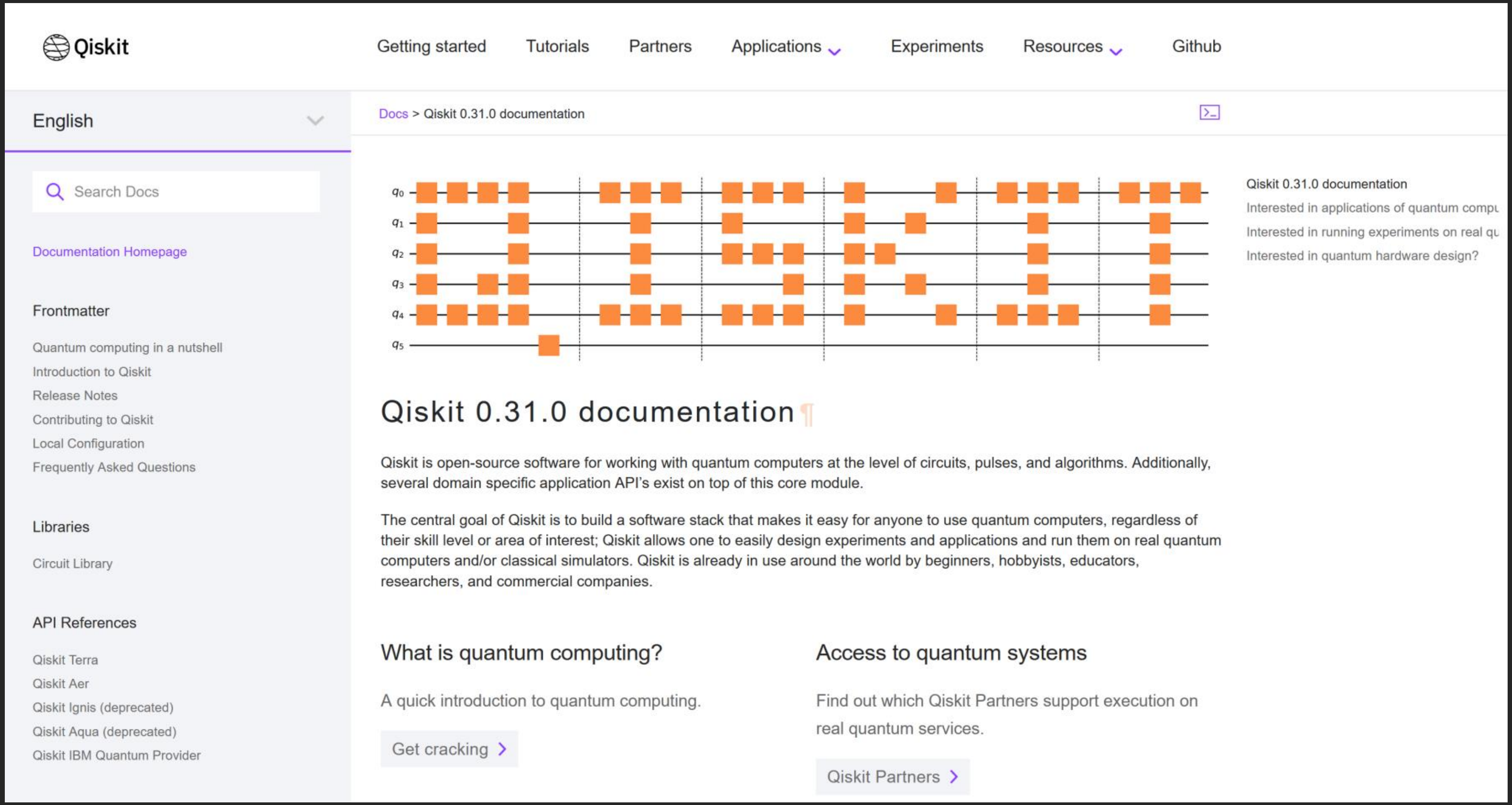
Credit: [IBM – Quantum Lab]

IBM Quantum



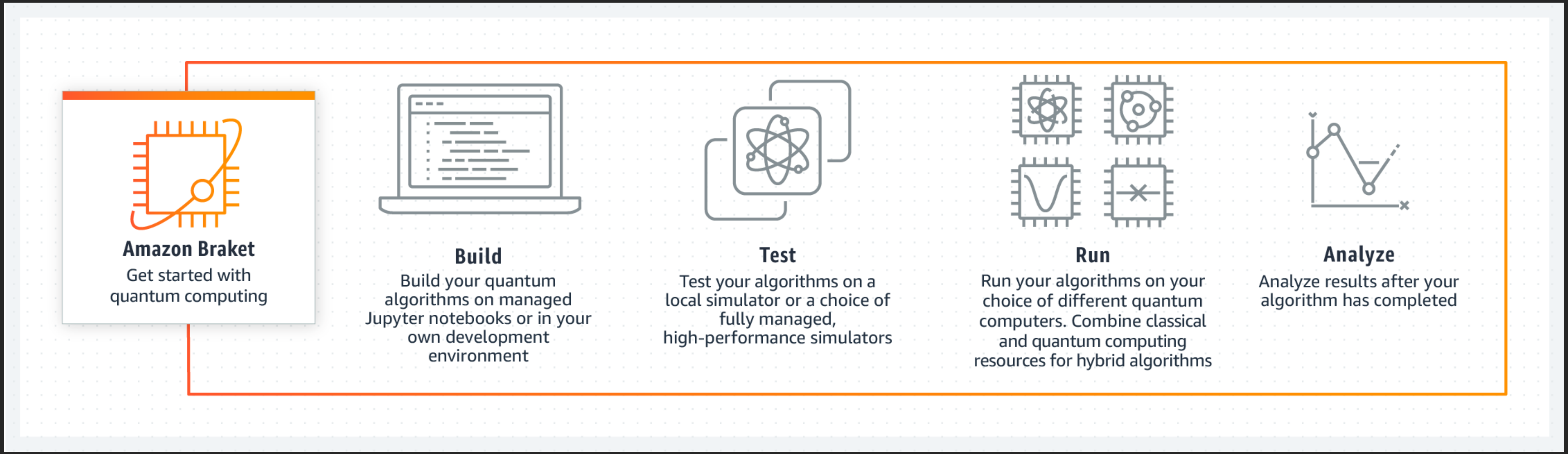
Credit: [IBM - Qiskit]

IBM Quantum



Credit: [IBM – Qiskit documentation]

AWS Braket



Credit: [Amazon AWS - Braket]

AWS Braket

Amazon Braket Hardware Providers

Amazon Braket provides AWS customers access to multiple types of quantum computing technologies from quantum hardware providers, including gate-based quantum computers and quantum annealing systems. Learn more about these quantum hardware providers below.

D-Wave's technology uses quantum annealing to solve problems represented as mathematical functions (resembling a landscape of peaks and valleys). Their QPUs are built from a network of interconnected superconducting flux qubits. Each qubit is made from a tiny loop of metal interrupted by a Josephson Junction.

Learn more »

IonQ's trapped-ion approach to quantum computing starts with ionized ytterbium atoms. Two internal states of these identical atoms make up the qubits, the basic unit of quantum information. The execution of computational tasks is accomplished by programming the sequence of laser pulses used to implement each quantum gate operation.

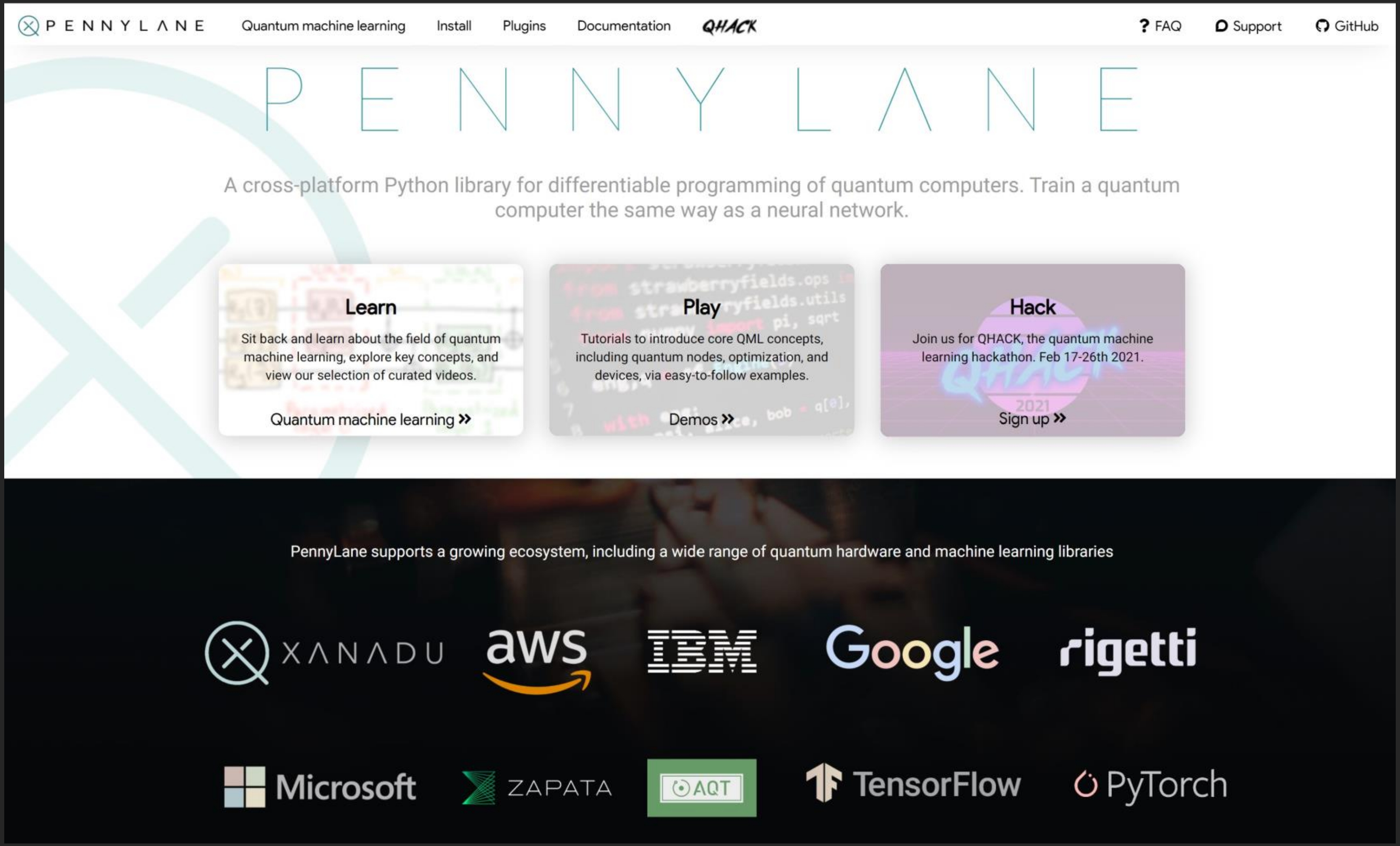
Learn more »

Rigetti quantum processors are universal, gate-based machines based on superconducting qubits. The Rigetti Aspen series of chips feature tileable lattices of alternating fixed-frequency and tunable superconducting qubits within a scalable architecture.

Learn more »

Credit: Amazon AWS – Braket

PennyLane



Credit: [Xanadu – PennyLane]

Google Cirq

Cirq

An open source framework for programming quantum computers

Cirq is a Python software library for writing, manipulating, and optimizing quantum circuits, and then running them on quantum computers and quantum simulators. Cirq provides useful abstractions for dealing with today's noisy intermediate-scale quantum computers, where details of the hardware are vital to achieving state-of-the-art results.

Get started with Cirq

GitHub repository

```python
import cirq

# Pick a qubit.
qubit = cirq.GridQubit(0, 0)

# Create a circuit
circuit = cirq.Circuit(
    cirq.X(qubit)**0.5,  # Square root of NOT.
    cirq.measure(qubit, key='m')  # Measurement.
)
print("Circuit:")
print(circuit)

# Simulate the circuit several times.
simulator = cirq.Simulator()
result = simulator.run(circuit, repetitions=20)
print("Results:")
print(result)
```

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.

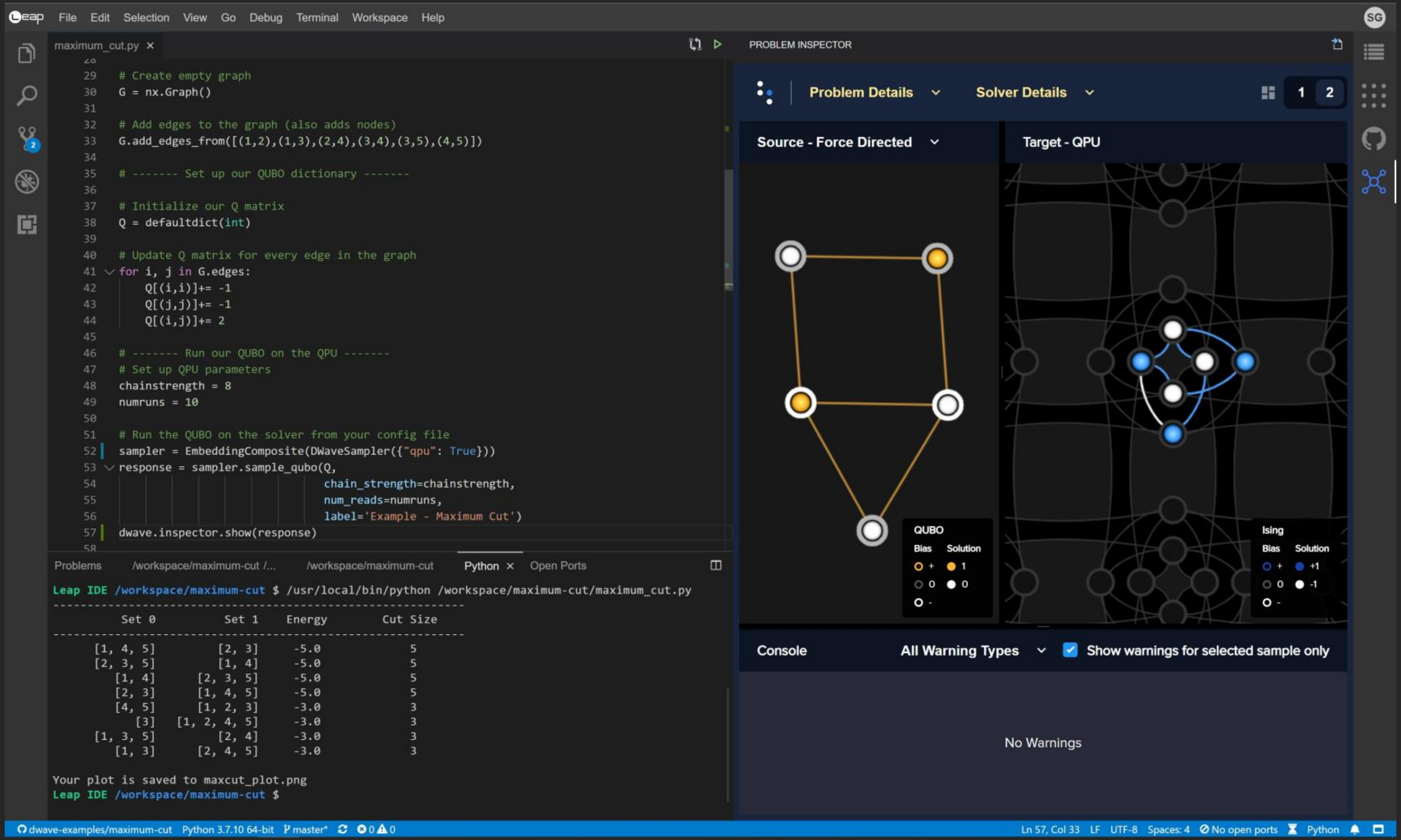
```
# A hybrid quantum-classical model.
model = tf.keras.Sequential([
    # Quantum circuit data comes in inside of tensors.
    tf.keras.Input(shape=(), dtype=tf.dtypes.string),

    # 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]