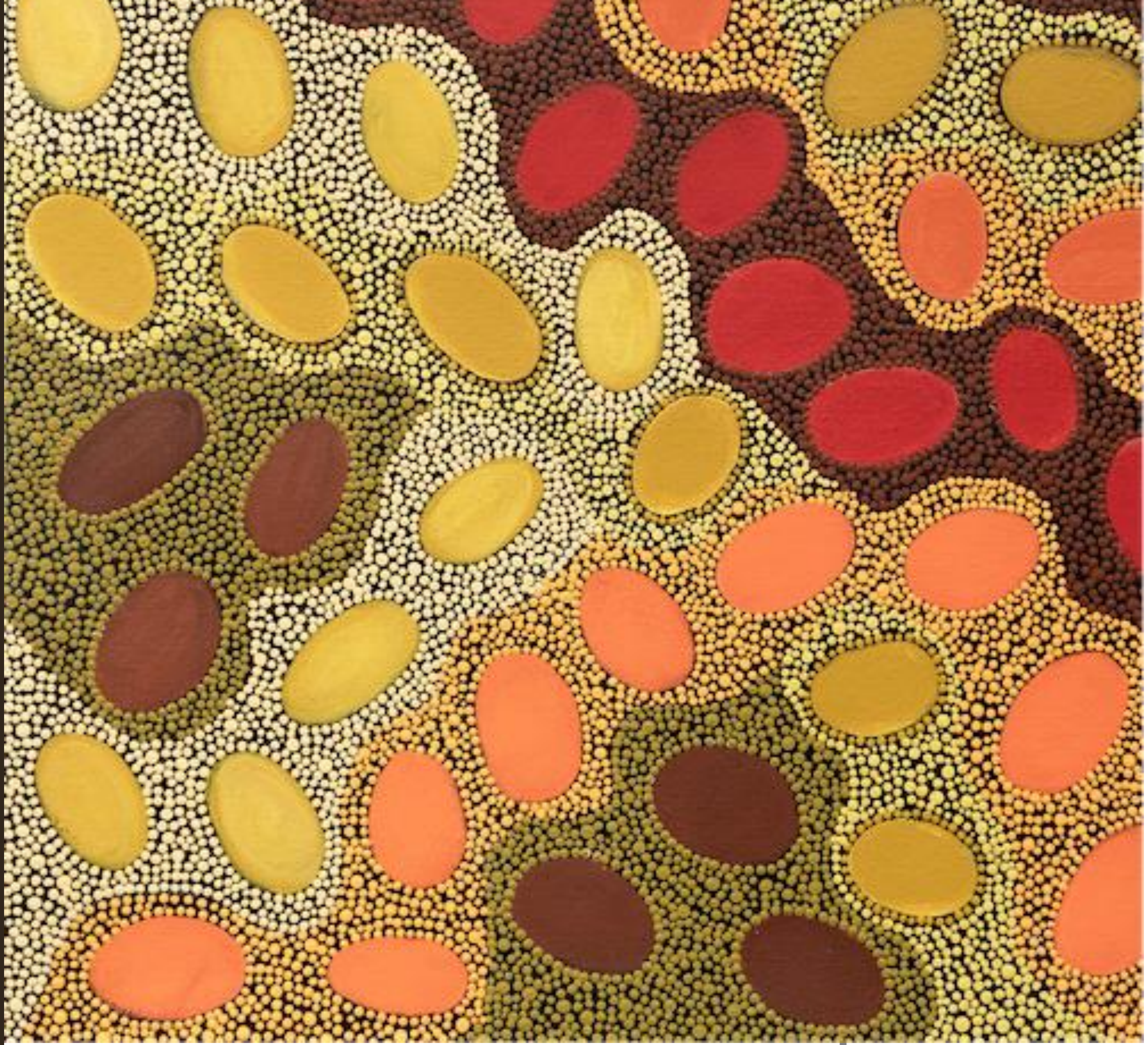
Introduction to Quantum Computing

Dr. Edric Matwiejew

Pawsey



Meteorites by Wajarri Yamatji artist Margaret Whitehurst

Acknowledgment of Country

Ngaala Kaaditji Noongar moort keyen kaadak nidja boodja

We acknowledge the traditional owners of land on which Pawsey is located, the Noongar Whadjuk People– their ancestors and elders, past, and present – as the original custodians of this land.

We pay our respects to the traditional owners of the land on which we meet today.

Outline

- Introduction to the Pawsey Quantum Supercomputing Innovation Hub
- Quantum Computing:
  - Bits vs qubits, superposition, entanglement
  - Quantum circuits and quantum gates.
  - Quantum computing landscape: NISQ devices, analog vs digital
- Quantum Algorithms:
  - Quantum parallelism
  - Adiabatic quantum computing
  - Hamiltonian Simulation
  - Variational quantum circuits (VQCs)
- Hands-On:
  - Programming quantum circuits with Qiskit
  - Solving the max-cut problem with QAOA

Quantum Computing at Pawsey

Pawsey Supercomputing Research Centre

- National High Performance Computing Centre headquartered in Perth, Western Australia on Whadjuk Noongar country
- Launched as Pawsey in 2014, UJV foundations back to 2000
- 50+ Staff employed via CSIRO, national science agency
- AU$70m capital refresh by Australian Government
- Houses the fastest Supercomputing in Australia
- **AU$5m NCRIS investment in developing a Quantum Supercomputing Innovations Hub, part of the National Quantum Strategy**

Pawsey strives to be at the cutting-edge of technology. Our domain is scientific computing: accelerating discovery through world-class supercomputing infrastructure and expertise.

The Pawsey Supercomputing Research Centre is an unincorporated joint venture between

Core Members

CSIRO
Curtin University
Murdoch University
The University of Western Australia

Founding Associate Member

ECU Edith Cowan University

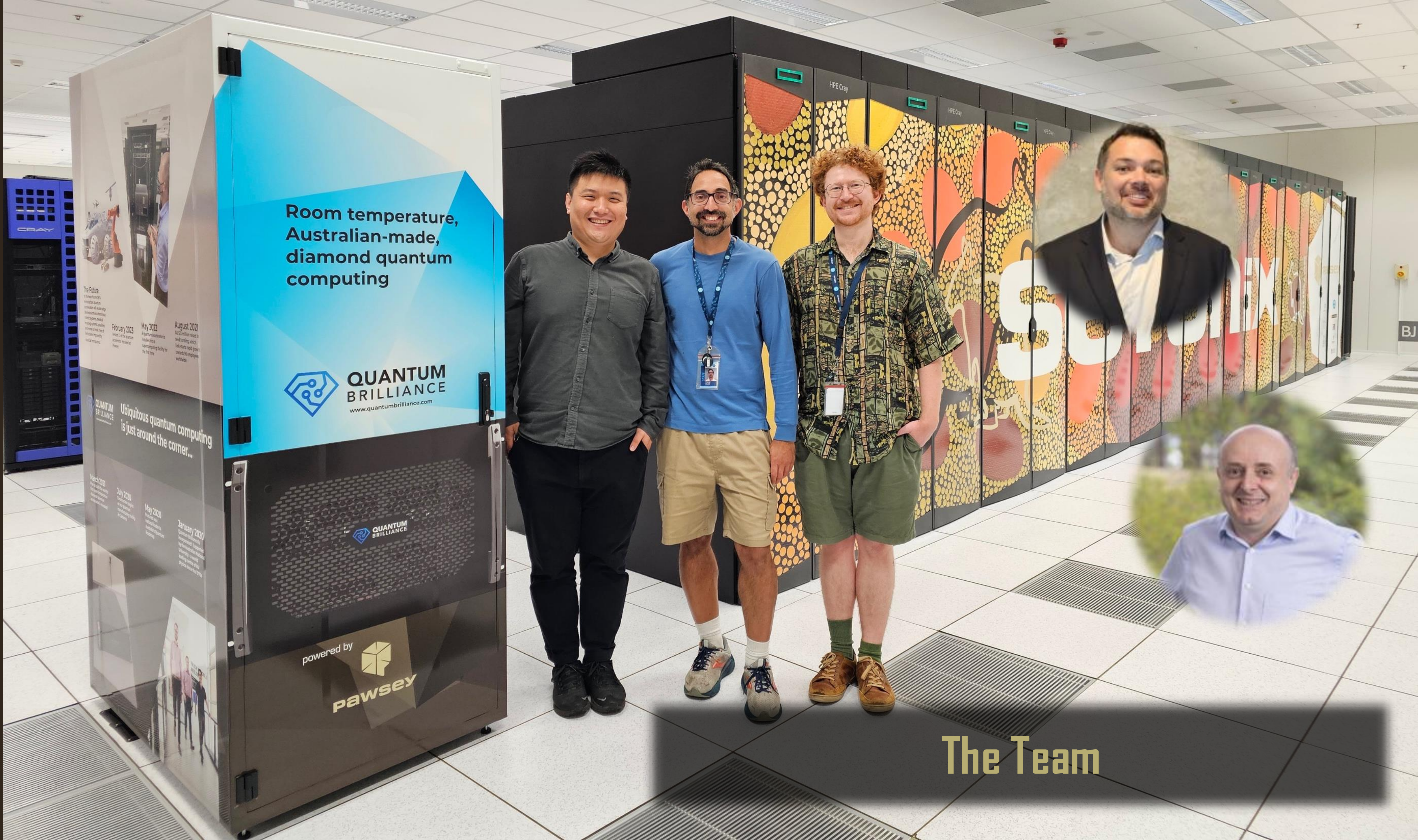
and proudly funded by

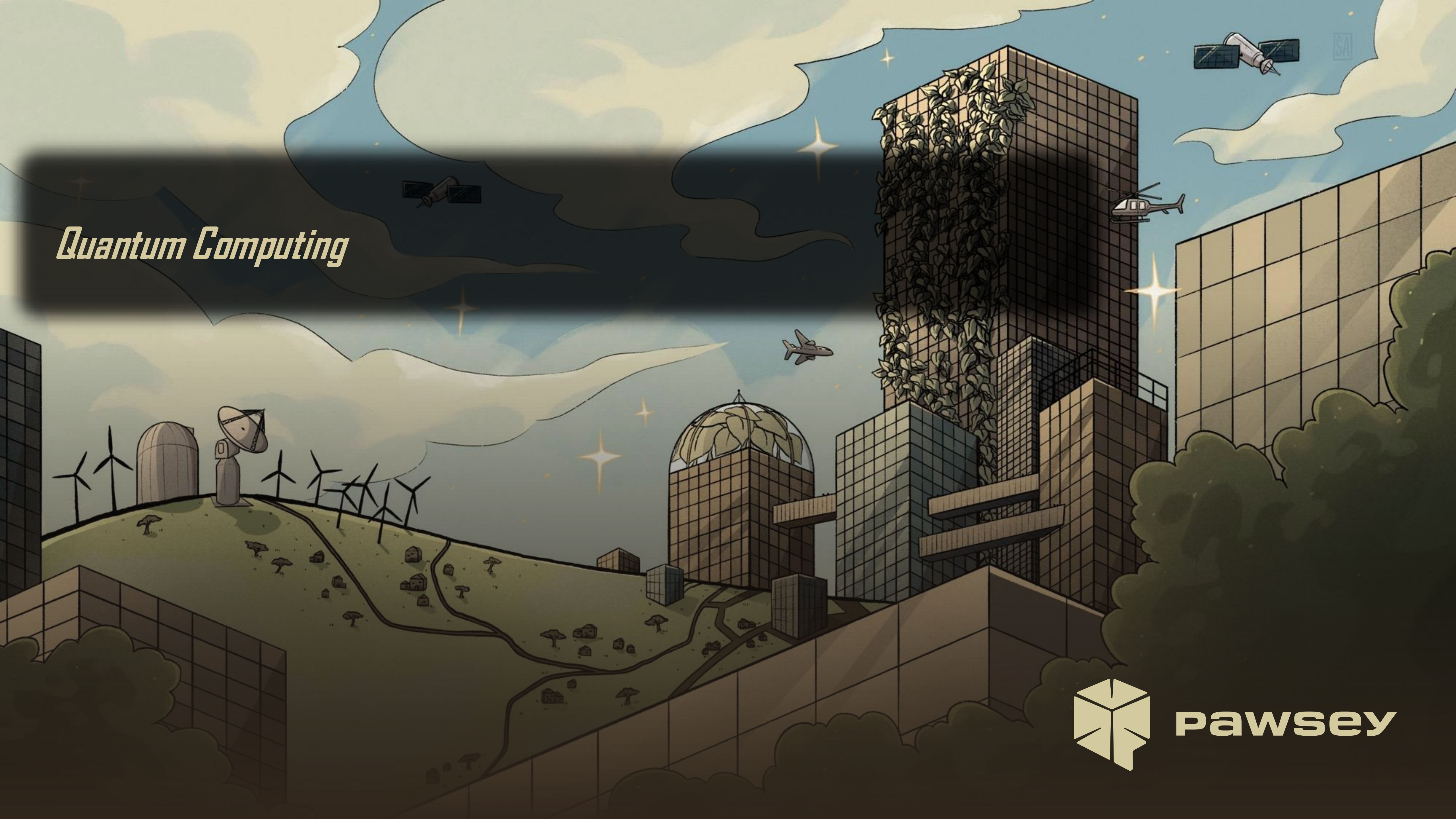
NCRIS National Research Infrastructure for Australia An Australian Government Initiative
Government of Western Australia

Quantum SUPERcomputing Innovation Hub

- Pawsey is on the cutting edge of computing technologies
- The Hub, supported by NCRIS funding, a collaborative platform uniting quantum & classical computing to drive scientific breakthroughs.
- Setonix-Q merit-allocation schemes provides access to quantum computing hardware and QC-dedicated classical compute.

The Team


Room temperature, Australian-made, diamond quantum computing
QUANTUM BRILLIANCE
www.quantumbrilliance.com
Ubiquitous quantum computing is just around the corner...
QUANTUM BRILLIANCE
powered by
Pawsey
HPE Cray
CRAY



Quantum Computing

Quantum Computing

- Computation on information encoded in quantum states – typically **qubits**.
- Leverage quantum phenomena (superposition, interference and entanglement) to solve tough problems faster than classical computers.
- Quantum hardware and algorithms differ fundamentally from classical approaches.
- Potential for major speedups—though not for every problem.
- The challenge of realising practical quantum computing spans physics, mathematics, engineering, and computer science.
- No widely adopted high-level languages - programming with quantum “assembly” or “machine-level” instructions (e.g. laser pulse shaping)
- *Requires rethinking how problems are formulated and tackled, in a rapidly evolving field.*

The Computational Basis

- A quantum computer uses the quantised states of a quantum object.
- Atomic energy levels, photon polarisation, molecular spins, or any other physical system that can be precisely controlled.
- In principle this quantum object can have any number of basis states.
- Most commonly the fundamental unit of information is a “qubit” (two basis states, often referred to as the **computational basis**):

$$\left\{ |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right\}$$

The Computational Basis

Quantum basis states combined according to the **tensor product** ⊗:

$$|0\rangle \otimes |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} (1) \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ (0) \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = |00\rangle$$

$$|0\rangle \otimes |1\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} (1) \times \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ (0) \times \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = |01\rangle$$

The Computational Basis

- An n-qubit **computational basis state** is often represented using a single base-10 integer.

$n = 2$:

$$
\begin{aligned}
|0\rangle \otimes |0\rangle = |00\rangle = |0\rangle \\
|0\rangle \otimes |1\rangle = |01\rangle = |1\rangle \\
|1\rangle \otimes |0\rangle = |10\rangle = |2\rangle \\
|1\rangle \otimes |1\rangle = |11\rangle = |3\rangle
\end{aligned}
$$

The Computational Basis

- An n-qubit **computational basis state** is often represented using a single base-10 integer.

$n = 3$:

$$
\begin{aligned}
|0\rangle \otimes |0\rangle \otimes |0\rangle &= |000\rangle = |0\rangle \\
|0\rangle \otimes |0\rangle \otimes |1\rangle &= |001\rangle = |1\rangle \\
|0\rangle \otimes |1\rangle \otimes |0\rangle &= |010\rangle = |2\rangle \\
|0\rangle \otimes |1\rangle \otimes |1\rangle &= |011\rangle = |3\rangle \\
|1\rangle \otimes |0\rangle \otimes |0\rangle &= |100\rangle = |4\rangle \\
|1\rangle \otimes |0\rangle \otimes |1\rangle &= |101\rangle = |5\rangle \\
|1\rangle \otimes |1\rangle \otimes |0\rangle &= |110\rangle = |6\rangle \\
|1\rangle \otimes |1\rangle \otimes |1\rangle &= |111\rangle = |7\rangle
\end{aligned}
$$

The Computational Basis

- Each additional qubit **doubles** the number of possible states.
- For a 2 qubit system $|00\rangle, |10\rangle, |01\rangle, |11\rangle$
- Adding one qubit doubles the number of possible states.
- For a 3 qubit system: $|000\rangle, |100\rangle, |010\rangle, |001\rangle, |101\rangle, |011\rangle, |110\rangle, |111\rangle$
- Consequently, representing a system with $\boldsymbol{n}$ **qubits requires on the order of** $\mathbf{2^n}$ **classical bits**.

There's an ancient story on the origins of chess that shows the power of doubling. Start with 1 grain of rice, then doubled it each time on the next square. What do you need at the end?

The last square needs 18,446,744,073,709,551,615 grains, weighing the same as 2000 billion people!

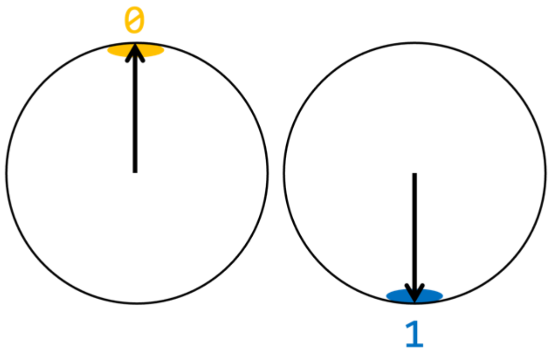
State-Vectors: Superposition

- The state of a quantum computer is given by its state-vector $|\psi\rangle$.
- $|\psi\rangle$ can be expressed as a **superposition** (linear-combination), of computational basis states.
- A system of qubits is finite-dimensional so we can always represent $|\psi\rangle$ as a **column vector**.
- The possible states for a **single** qubit:

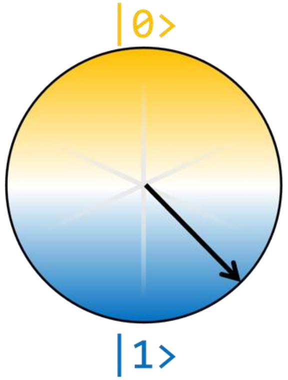
$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

- Coefficients α and β are complex numbers with a **magnitude** and **phase**.
- A qubit can store information in the magnitude and phase of its coefficients – **more information than a classical bit.**

Classical Bit



Qubit



State-Vectors: Superposition

- Putting together two qubits:

$$|\psi\rangle = (\alpha_0|0\rangle + \beta_0|1\rangle\,) \otimes (\alpha_1|0\rangle + \beta_1|1\rangle\,)$$

$$= \alpha_0\alpha_1|00\rangle + \alpha_0\beta_1|01\rangle + \beta_0\alpha_1|10\rangle + \beta_0\beta_1|11\rangle$$

$$= \alpha_0\alpha_1\begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \alpha_0\beta_1\begin{bmatrix}0\\1\\0\\0\end{bmatrix} + \beta_0\alpha_1\begin{bmatrix}0\\0\\1\\0\end{bmatrix} + \beta_0\beta_1\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$$

Superposition

State-Vectors: Interference

- Differences in the complex **phase** of α_i and β_i cause **constructive or destructive interference,** altering the magnitude of basis state in the superposition.

$$|\psi\rangle = (\alpha_0|0\rangle + \beta_0|1\rangle) \otimes (\alpha_1|0\rangle + \beta_1|1\rangle)$$

$$= \alpha_0\alpha_1|00\rangle + \alpha_0\beta_1|01\rangle + \beta_0\alpha_1|10\rangle + \beta_0\beta_1|11\rangle$$

$$= \alpha_0\alpha_1\begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \alpha_0\beta_1\begin{bmatrix}0\\1\\0\\0\end{bmatrix} + \beta_0\alpha_1\begin{bmatrix}0\\0\\1\\0\end{bmatrix} + \beta_0\beta_1\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$$

Interference

State-Vectors: Measurement

- Quantum computing relies on quantum superposition—a single quantum object can occupy multiple states simultaneously.
- This phenomenon has been extensively experimentally confirmed.
- Equally well-established is the phenomenon of **collapse.**

When we **measure** the state of a quantum computer, we obtain **one** bit string that corresponds to just **one** of an exponential number of possible basis states.

State-Vectors: Measurement

- But which state does it collapse to?
- The coefficients of $|\psi\rangle$ are **probability amplitudes** for the corresponding basis states.
- To compute the probability of measuring a particular state $|\boldsymbol{k}\rangle$, we inner-product $|\psi\rangle$ with its corresponding **column vector** $\langle\boldsymbol{k}|$, compute the complex modulus and square the result:

$$\mathrm{Prob}(|01\rangle) = |\langle 01|\psi\rangle|^2 = \left|\begin{bmatrix}0 & 1 & 0 & 0\end{bmatrix}.\left(\alpha_0\alpha_1\begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \alpha_0\beta_1\begin{bmatrix}0\\1\\0\\0\end{bmatrix} + \beta_0\alpha_1\begin{bmatrix}0\\0\\1\\0\end{bmatrix} + \beta_0\beta_1\begin{bmatrix}0\\0\\0\\1\end{bmatrix}\right)\right|^2$$

$$= |\alpha_0\beta_1|^2$$

- **Note 1:** As the state-vector fully-describes a quantum system:

$$|\alpha_0\alpha_1|^2 + |\alpha_0\beta_1|^2 + |\beta_0\alpha_1|^2 + |\beta_0\beta_1|^2 = 1$$

State-Vectors: Measurement

- But which state does it collapse to?
- The coefficients of $|\psi\rangle$ are **probability amplitudes** for the corresponding basis states.
- To compute the probability of measuring a particular state $|\boldsymbol{k}\rangle$, we inner-product $|\psi\rangle$ with its corresponding **column vector** $\langle\boldsymbol{k}|$, compute the complex modulus and square the result:

$$\text{Prob}(|01\rangle) = |\langle 01|\psi\rangle|^2 = \left|\begin{bmatrix}0 & 1 & 0 & 0\end{bmatrix}.\left(\alpha_0\alpha_1\begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \alpha_0\beta_1\begin{bmatrix}0\\1\\0\\0\end{bmatrix} + \beta_0\alpha_1\begin{bmatrix}0\\0\\1\\0\end{bmatrix} + \beta_0\beta_1\begin{bmatrix}0\\0\\0\\1\end{bmatrix}\right)\right|^2$$

$$= |\alpha_0\beta_1|^2$$

- **Note 2:** Computational basis states are **orthonormal:**

$$\langle k|k'\rangle = \begin{cases} 1, & \text{if } k = k' \\ 0, & \text{otherwise} \end{cases}$$

State-Vectors: Entanglement

- Superposition interference and collapse can together give rise to **entanglement**.
- In an entangled state, a measurement outcome on one qubit determines the possible outcomes of another.
- Famously referred to as “spooky action at a distance” by Einstein – it is a uniquely quantum phenomena.

A two-qubit “Bell state”:

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

- Computationally, entanglement enables “controlled” operations between superposed basis states. Analogous to classical “if then” statements.

Unitaries and Quantum Circuits

- $|\psi\rangle$ is manipulated by **unitary operators U**
- U transforms $|\psi\rangle$ while ensuring that its measurement probabilities still sum to one.
- A unitary operator for a system with n qubits can be represented as a $2^n \times 2^n$ matrix.
- By definition, the inverse of any U is its conjugate transpose (or “adjoint”).
- So, we can always retrieve the initial state vector:

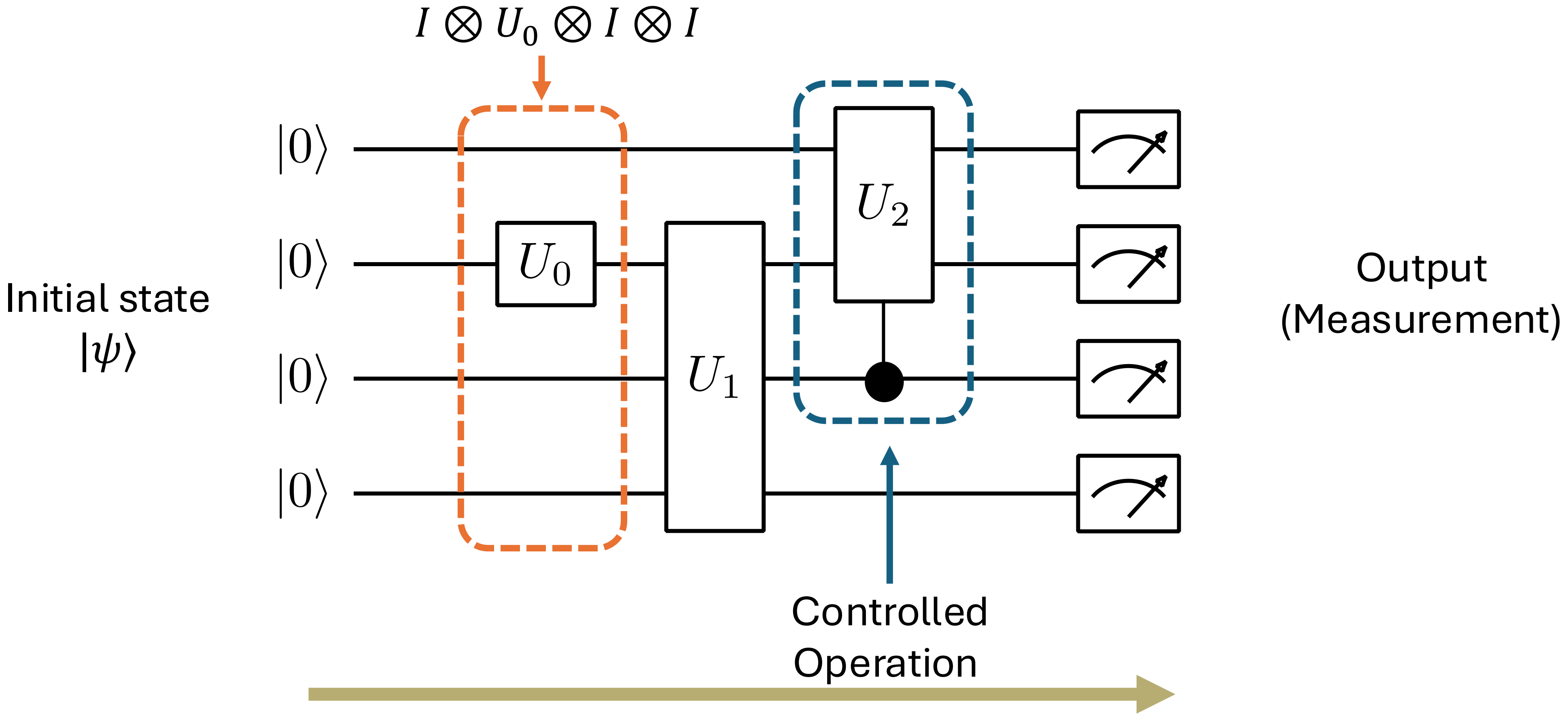
$$|\psi'\rangle = U|\psi\rangle$$

$$|\psi\rangle = U^{-1}|\psi'\rangle$$

- Quantum computing is a type of **reversible** computation.

Quantum Circuits

- Quantum circuits are a diagrammatic representation of a quantum program.
- They show how unitaries are applied to “registers” of qubits and where measurement occurs.

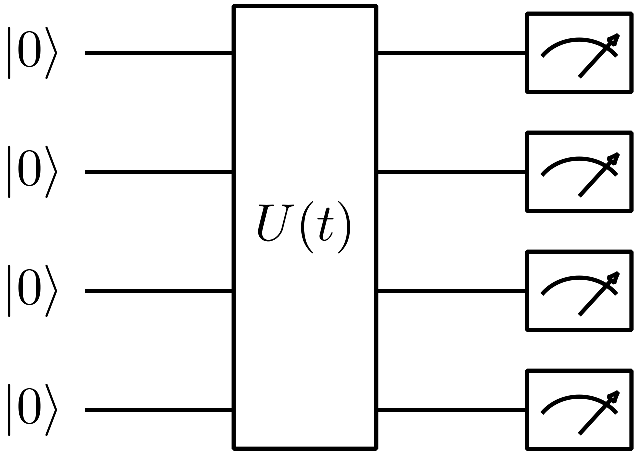


Quantum Computing Models: Analogue

- Continuously manipulates quantum states through engineered interactions between particles (e.g. atoms or ions). Viewed as applying a time-dependent unitary to prepare a target state:

$$U(t)|\psi\rangle = |\psi'\rangle$$

- Relies on the physical characteristics of the system and is less flexible for general-purpose computation.
- Applications include optimisation, graph-based problems and reservoir computing.

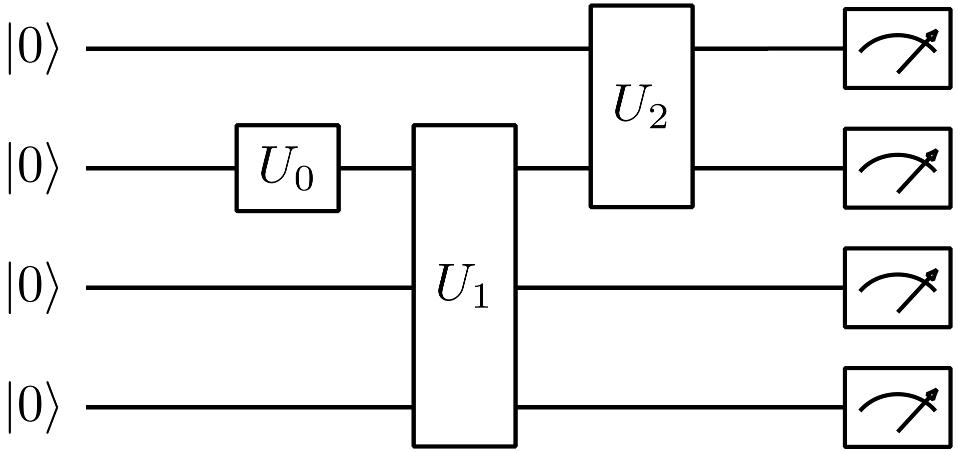


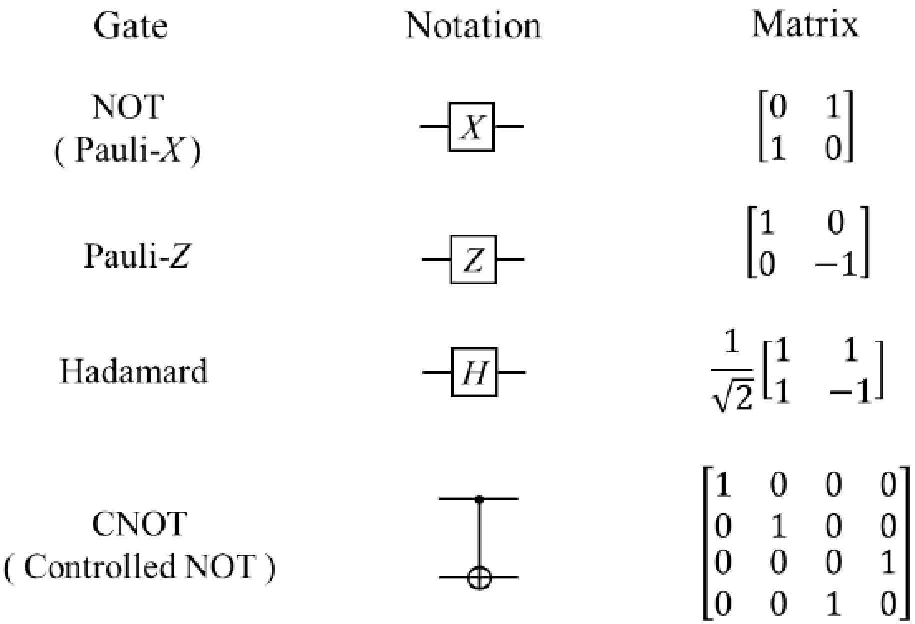
Quantum Computing Models: Digital

- Uses discrete **quantum gates** - analogous to classical logic gates. Viewed a sequence of unitaries, called quantum “gates”, that act on one or more qubits:

$$U_2 U_1 U_0 |\psi\rangle = |\psi'\rangle$$

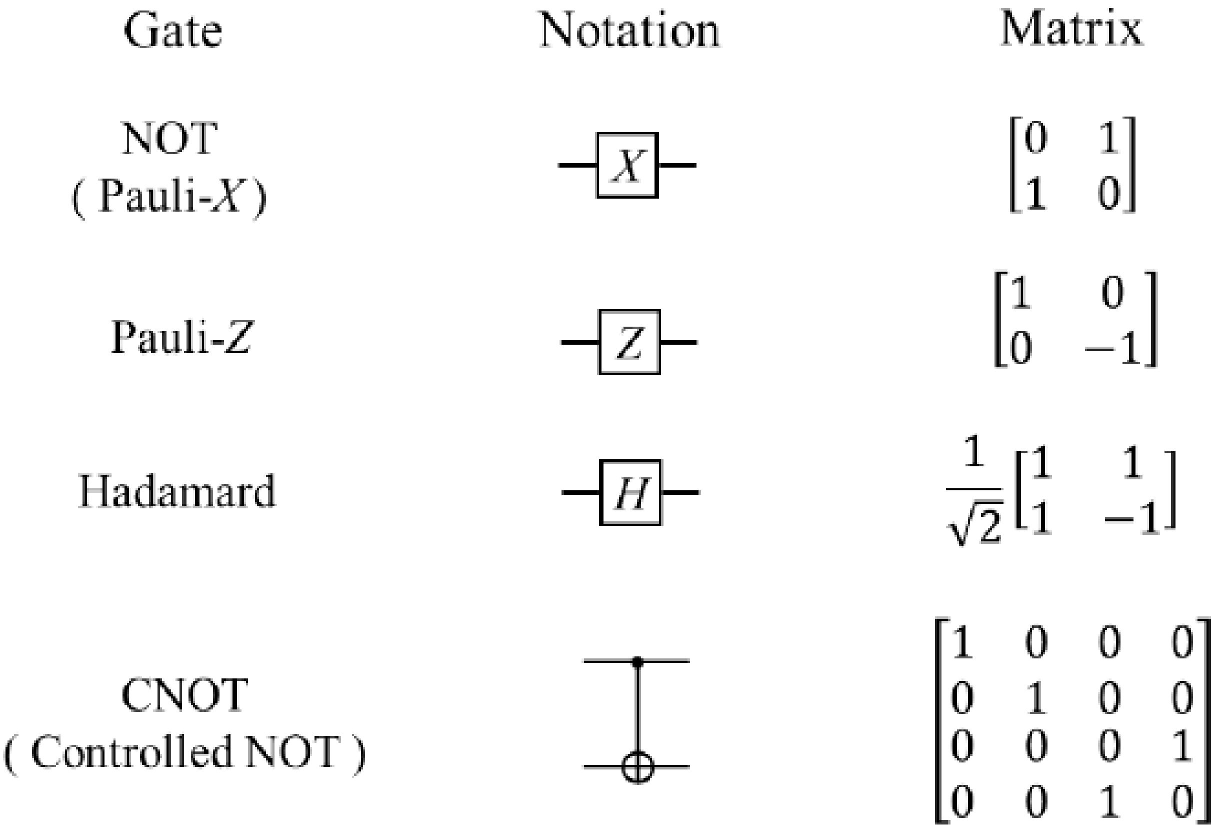
- Digital systems are more versatile and are the foundation for universal quantum computers.
- More dependent on error mitigation or correction.





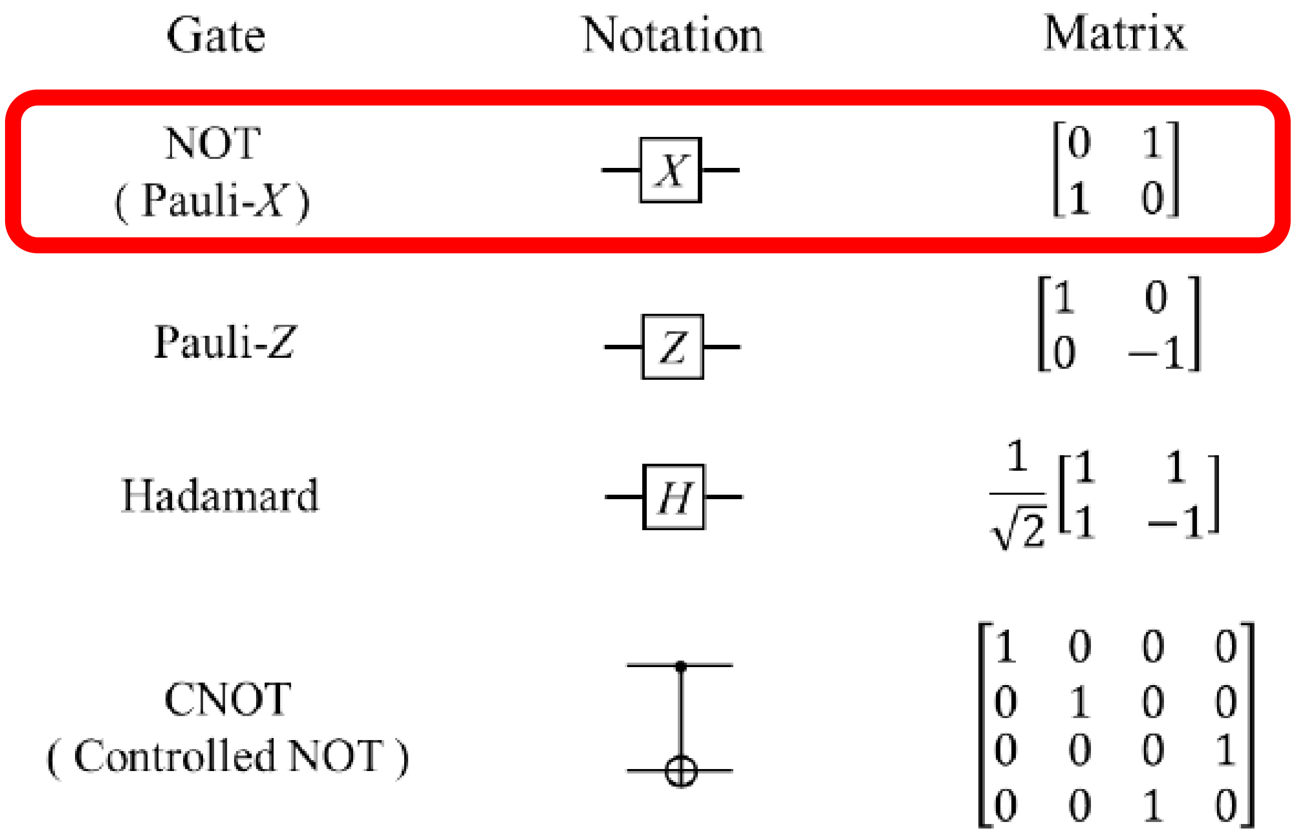
Gate	Notation	Matrix
NOT (Pauli-X)	X	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Z	Z	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard	H	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
CNOT (Controlled NOT)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

Quantum Computing Models: Digital

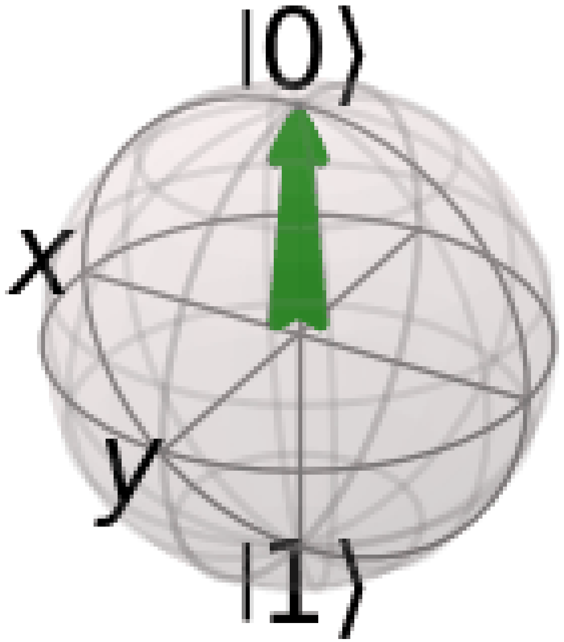


Gate	Notation	Matrix
NOT (Pauli-X)	$\boxed{X}$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Z	$\boxed{Z}$	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard	$\boxed{H}$	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
CNOT (Controlled NOT)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

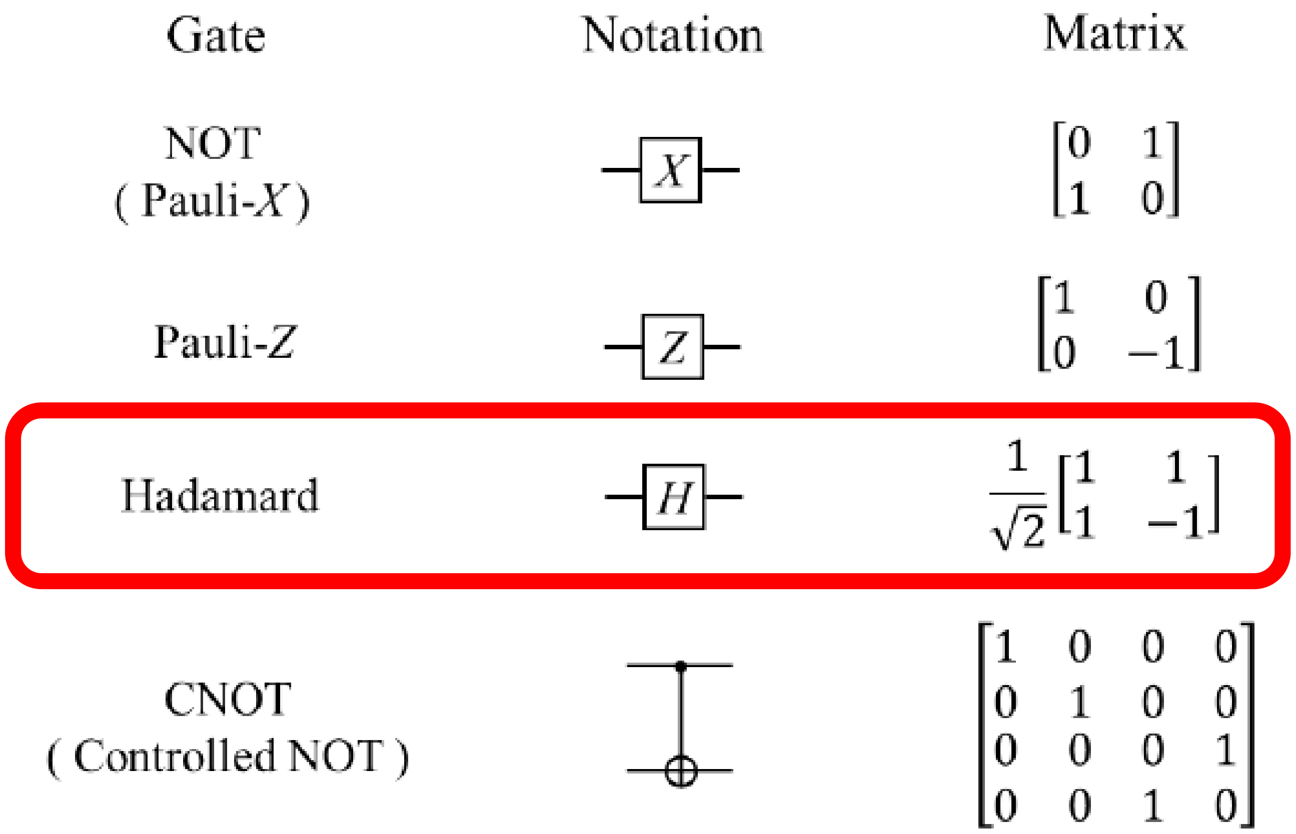
Quantum Computing Models: Digital



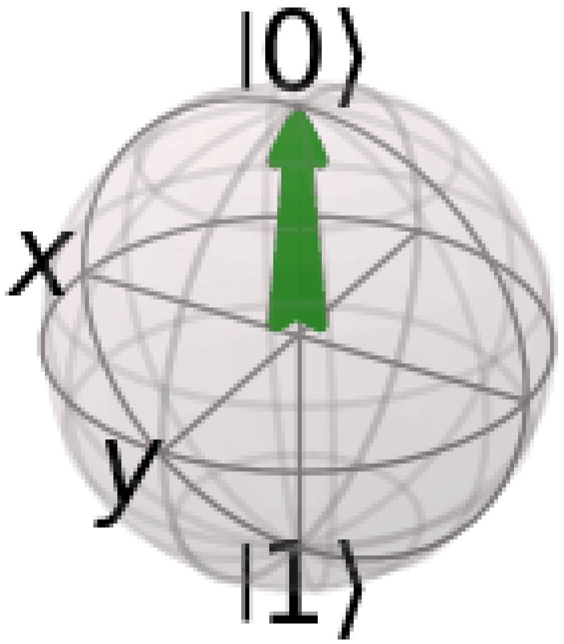
Gate	Notation	Matrix
NOT (Pauli-X)	$-\boxed{X}-$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Z	$-\boxed{Z}-$	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard	$-\boxed{H}-$	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
CNOT (Controlled NOT)	$\begin{array}{c} -\bullet- \\ \vert \\ -\oplus- \end{array}$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$



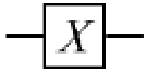
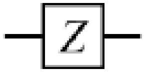
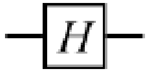
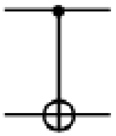
Quantum Computing Models: Digital



Gate	Notation	Matrix
NOT (Pauli-X)	$\boxed{X}$	$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$
Pauli-Z	$\boxed{Z}$	$\begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}$
Hadamard	$\boxed{H}$	$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}$
CNOT (Controlled NOT)		$\begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 0 & 1\\0 & 0 & 1 & 0\end{bmatrix}$



Quantum Computing Models: Digital

Gate	Notation	Matrix
NOT (Pauli-X)		$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Z		$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard		$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
CNOT (Controlled NOT)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

$$\mathrm{CNOT}\,|00\rangle = |00\rangle$$

$$\mathrm{CNOT}\,|01\rangle = |01\rangle$$

$$\mathrm{CNOT}\,|10\rangle = |11\rangle$$

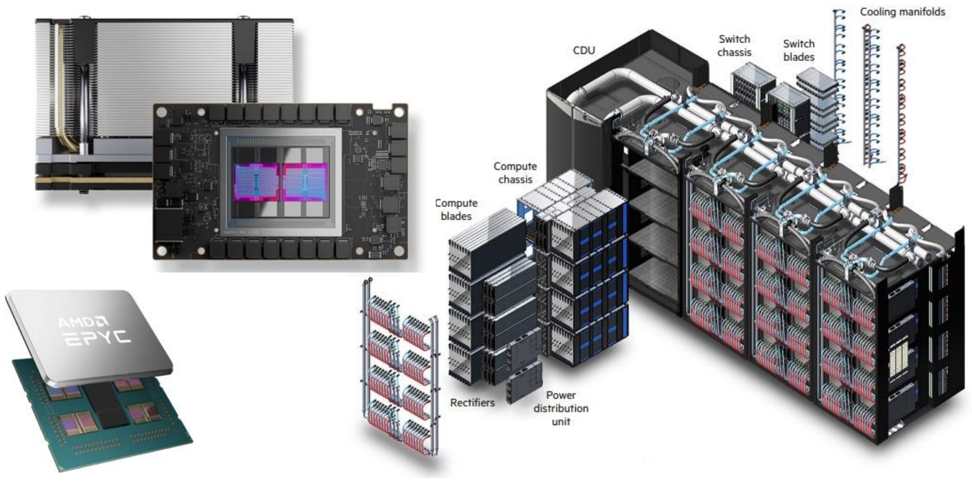
$$\mathrm{CNOT}\,|\underset{\uparrow\atop\textbf{Control}}{1}\underset{\uparrow\atop\textbf{Target}}{1}\rangle = |10\rangle$$

Classical computers

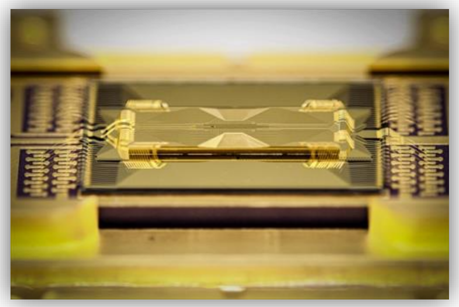
VS

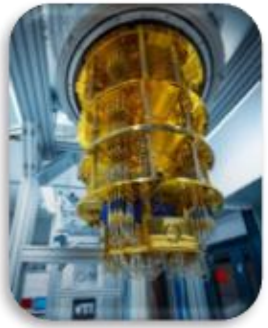
Quantum computers

- Stores bits to represent data, applies bit manipulation to represent operations
- Really fast (GHz)
- Good at arithmetic, 2 + 2 = 4 (almost) always
- Production scale, with systems relying on silicon transistors.
- Some variety in hardware but all pretty similar.



- Operates on “qubits”
- Not good at arithmetic BUT (much) better at other problems.
- We are in Noisy Intermediate Scale Quantum (NISQ) Era.
  - Current technologies are all noisy (90-99% fidelity on single gate operations).
  - Low qubit counts (100)
  - Quite slow (100 circuits per second)
- Lots of variety in technologies, all single experiments





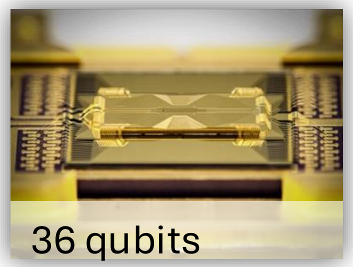
Quantum Computing Hardware

Analogue

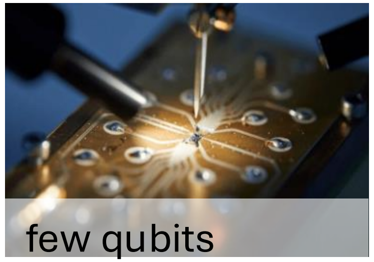
- Superconducting Flux Qubits: D-Wave Systems
  - Needs Helium 3 cooling
- Neutral Atoms: QuEra Computing, Pasqal, Infleqtion
  - Needs vacuum chamber, lasers


20-50 qubits

130-150 qubits


36 qubits

256 qubits,
analog computer


few qubits

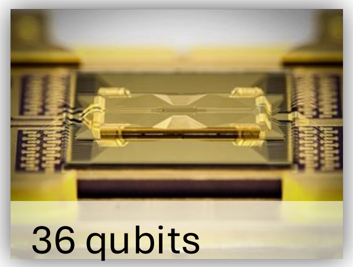
Quantum Computing Hardware

Digital

- Superconducting Qubits: IQM, IBM, Google Quantum AI, Rigetti Computing
  - Needs Helium 3 cooling
- Trapped Ions: IonQ, Quantinuum
  - Needs vacuum chamber, magnetic traps, lasers
- Neutral Atoms: QuEra Computing, Pasqal, Infleqtion
  - Needs vacuum chamber, magnetic traps, lasers
- Silicon Spin Qubits: Silicon Quantum Computing (SQC), Diraq
  - Needs Helium 3 cooling

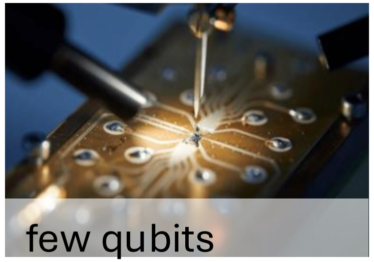
20-50 qubits

130-150 qubits



36 qubits

256 qubits,
analog computer



few qubits

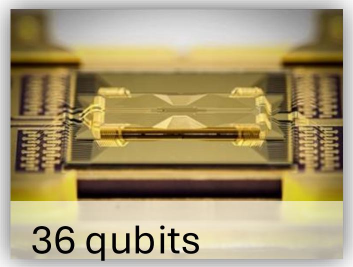
Quantum Computing Hardware

Digital (cont.)

- Diamond-Based Qubits: Quantum Brilliance
  - Currently limited scaling
- Photonic Qubits: ORCA, Xanadu
- Silicon Photonic Qubits: PsiQuantum
  - Needs Helium 3 cooling

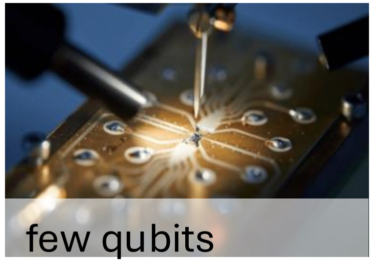
20-50 qubits

130-150 qubits

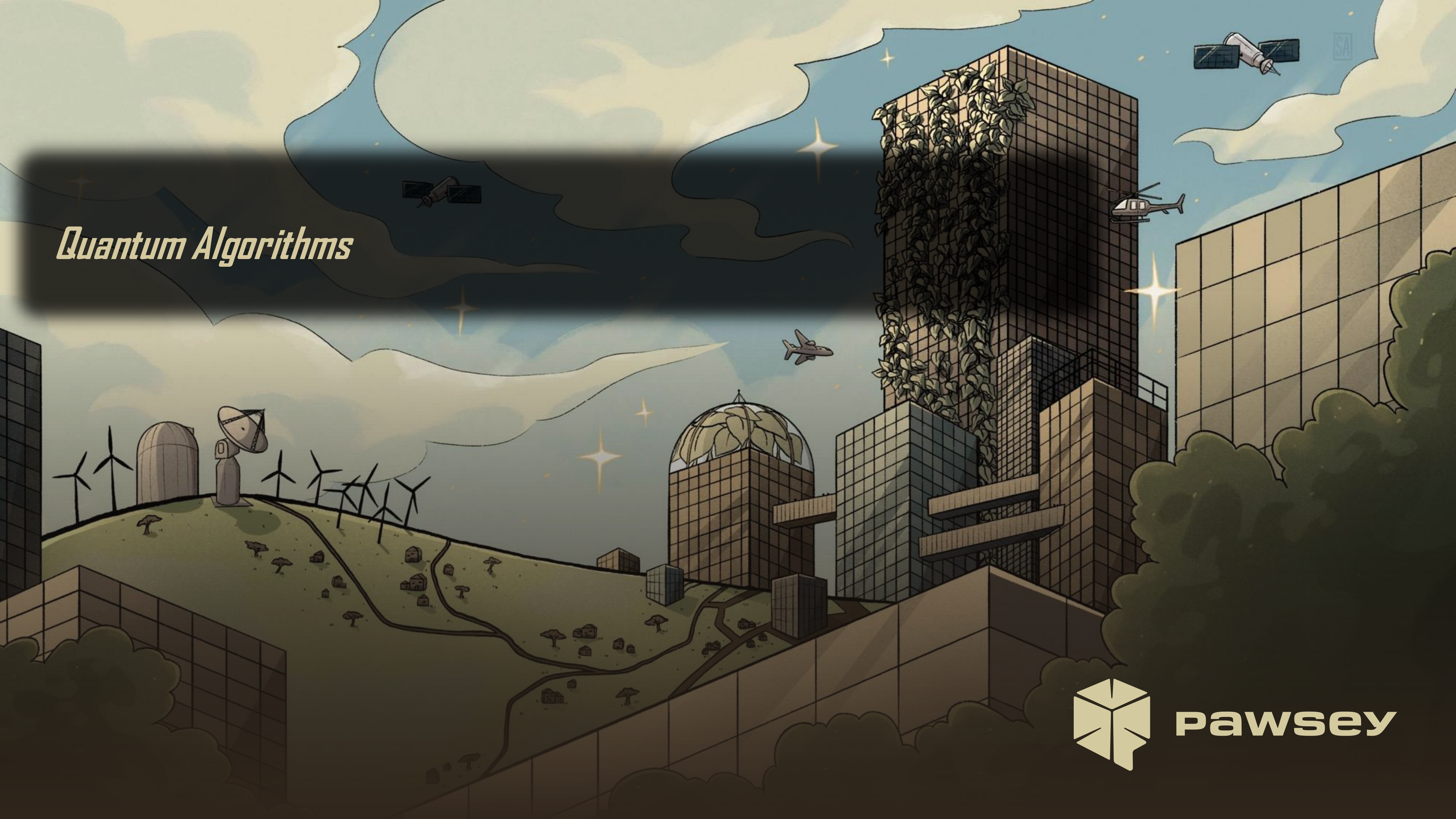


36 qubits

256 qubits,
analog computer



few qubits



Quantum Algorithms

Pawsey

Quantum Parallelism

- Consider a bijective function on binary strings of length n,

$$f:\{0,1\}^n \to \{0,1\}^n.$$

- The domain of f is **exponentially large**, with $N = 2^n$ elements.
- We can model this function on a quantum computer by a unitary U that acts on computational basis states $|x\rangle$ as,

$$U|x\rangle = |f(x)\rangle.$$

- Let $|s\rangle$ be a **uniform superposition** over all n-qubit $|x\rangle$,

$$|s\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle\,.$$

- We can evaluate f on its entire domain in one step,

$$U|s\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|f(x)\rangle\,.$$

Quantum Parallelism

- **Quantum parallelism** through quantum superposition underpins the potential for quantum algorithms to provide a speedup over classical methods.

Quantum Parallelism

- **Quantum parallelism** through quantum superposition underpins the potential for quantum algorithms to provide a speedup over classical methods.

Beware: Quantum parallelism alone does not provide an algorithmic advantage.

We also require:

- Implementation of U using a non-exponential number of qubits and gates.
- Manipulation of $U|x\rangle$ so useful information can be retrieved with a small number of measurements.

Adiabatic Quantum Computing

Adiabatic Quantum Computing (AQC) is an analogue method commonly used for combinatorial optimisation (e.g. logistics problems or Ising models).

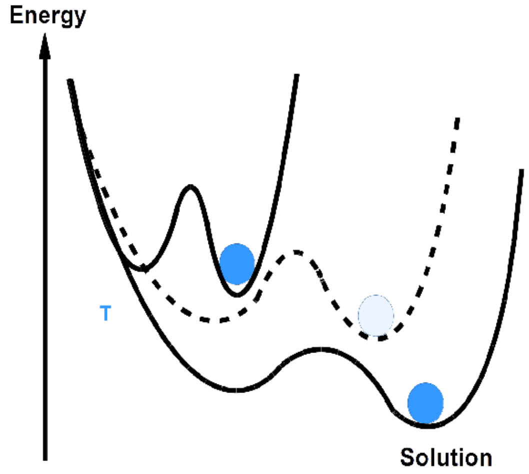
The Method:

1. Define an initial Hamiltonian H_0 that is easy to prepare on a given device
2. Define a final Hamiltonian H_1 that encodes the objective function $f(x)$, with optimal solutions corresponding to low-energy/ground states:

$$H_1 = \sum_{x=0}^{N-1} f(x)|x\rangle\langle x|$$

3. Start in H_0

(cont.)



Adiabatic evolution

A **Hamiltonian** is an operator that describes the energy of a quantum system and determines how it evolves over time.

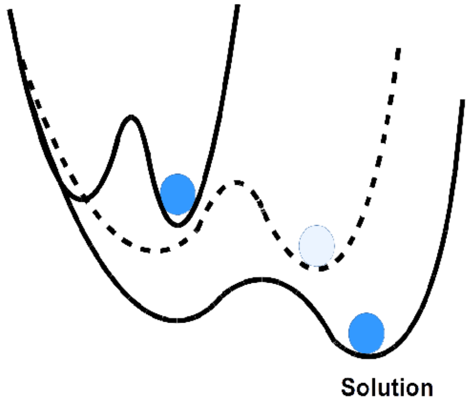
Adiabatic Quantum Computing

(cont.)

3. Slowly evolve to H_1 via

$$H(t) = (1 - s(t))H_0 + s(t)H_1, \quad s(t): 0 \rightarrow 1$$

4. If evolution is sufficiently slow, the system remains in the ground state and measurement reveals the solution.



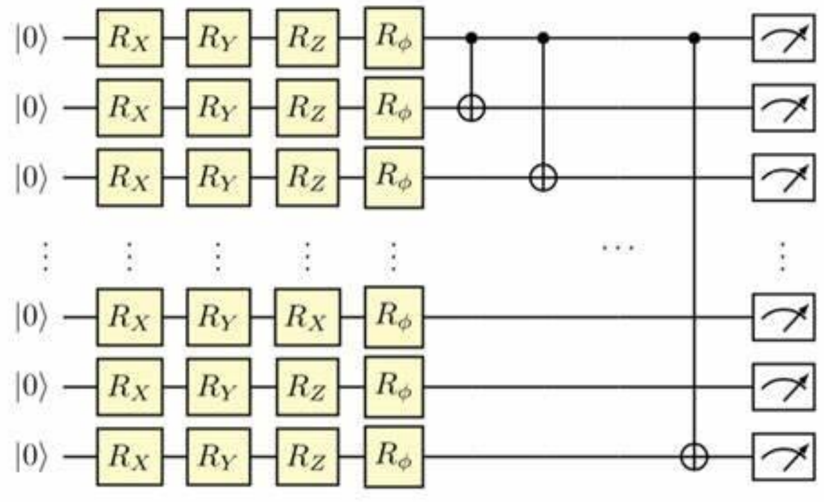
Adiabatic evolution

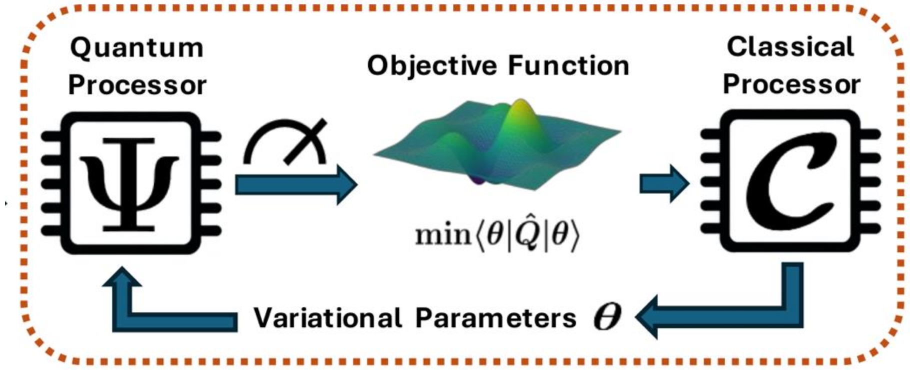
- Quantum Annealers (e.g., D-Wave) are specialised devices for AQC.
- Challenges associated with scaling of the annealing time with problem size.

A **Hamiltonian** is an operator that describes the energy of a quantum system and determines how it evolves over time.

Variational Quantum Circuits

- Quantum circuit with adjustable parameters (e.g. angles in rotation gates).
- Built up in layers, the depth of the circuit is adjusted to match problem complexity and hardware constraints.
- Measurement and feedback loop:
  1. Quantum circuit prepares a state
  2. Measure to compute an objective function
  3. A *classical optimiser* updates parameters
  4. Repeat until convergence.
- Algorithms for combinatorial optimisation, eigenvalue solving and machine learning fall within this model.
- These are a form of hybrid quantum-classical computation.

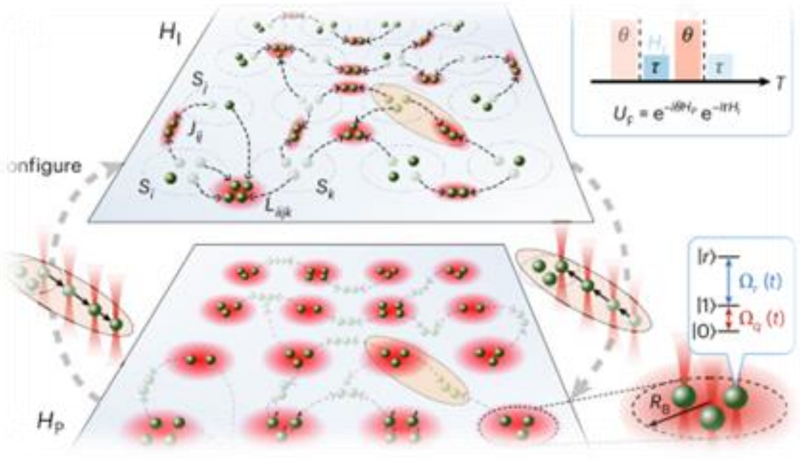




Hybrid Loop

Hamiltonian Simulation

- Compute $|\psi(t)\rangle = \exp(-itH)|\psi(0)\rangle$
- Applications quantum chemistry, condensed matter physics, and materials science—some of the most anticipated real-world applications.
- Takes advantage of memory-efficient representation of the Hamiltonian using qubits.
- Can be performed natively on analogue quantum computers by tuning the hardware's Hamiltonian to match or approximate a Hamiltonian of interest.
- Can be performed on digital quantum computers – to varying degrees of accuracy and efficiency depending on the chosen method and properties of H (as with classical computers).



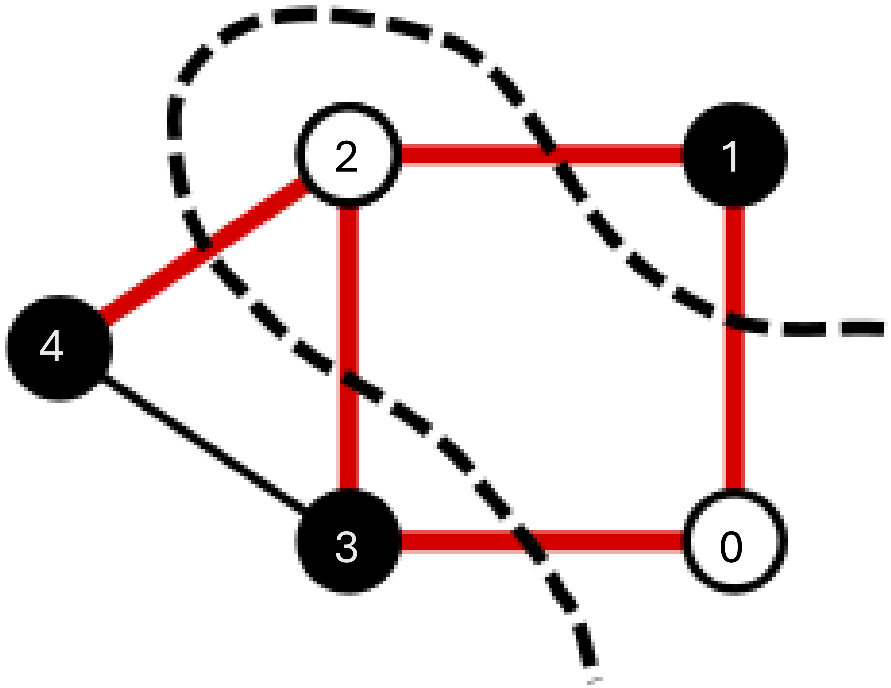
The Max-Cut Problem

Problem Statement:

- For a given graph, find an assignment of vertices to two set such that the number of adjacent vertices in different sets is maximised.

Why do we care?

- A simple to state problem that is computationally hard (NP-hard).
- Other difficult optimisation problems can be framed as a max-cut problem: electrical circuit layouts, spin models, portfolio optimisation…
- Straight-forward to problem express on a quantum computer.



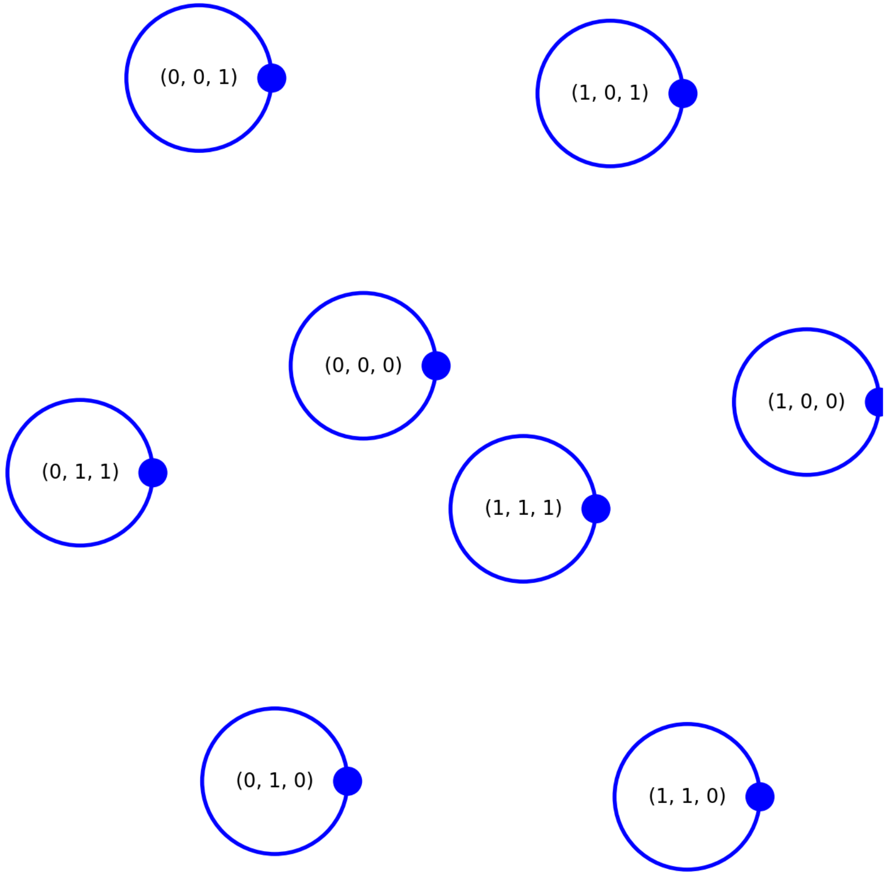
Solution: $\boldsymbol{z} = 10110$

Cut-value: $C(\boldsymbol{z}) = 5$

Encoding: $\boldsymbol{z} \rightarrow |\boldsymbol{z}\rangle = |10110\rangle$

The Quantum Approximate Optimisation Algorithm

- The Quantum Approximate Optimisation Algorithm (QAOA) aims to find optimal or near-optimal solutions to combinatorial optimisation problems. We apply it here to the MaxCut problem.
- Possible are mapped to the basis states of qubits. Each basis state is a possible cut.
- A variational quantum circuit "**ansatz**" prepares a state where the probability of measuring large-cut solutions are amplified.



The Quantum Approximate Optimisation Algorithm

- The quantum circuit of the QAOA has three main components.

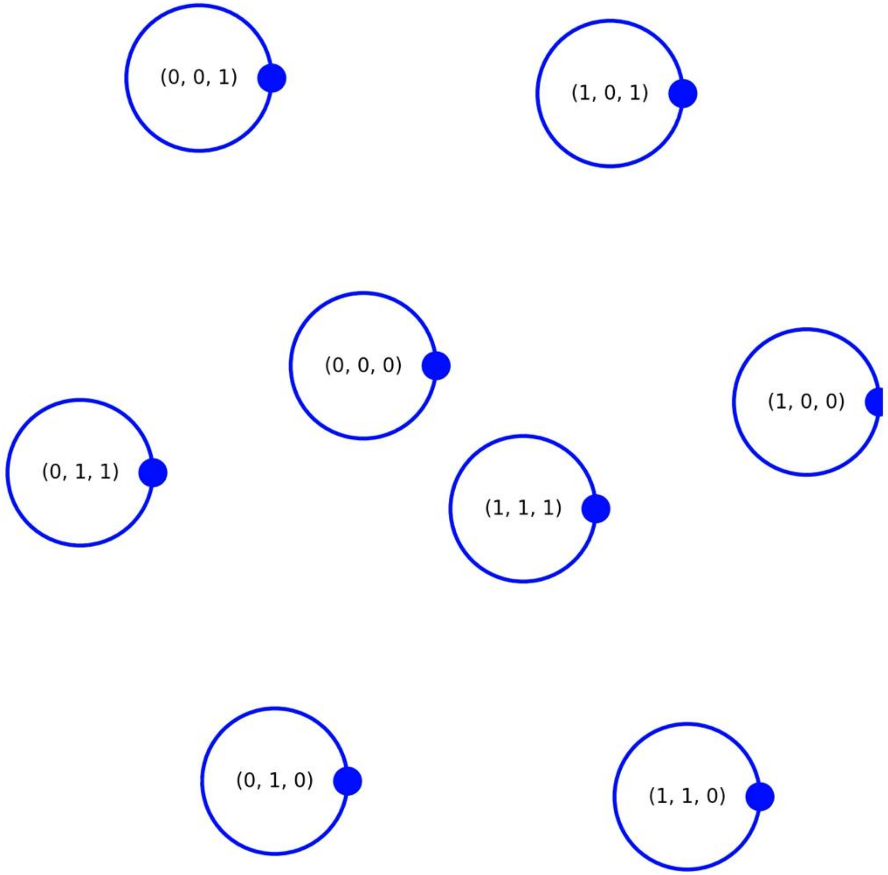
- **First:** preparation of a uniform superposition over all possible solutions/basis states.

- **Second:** a phase-shift unitary rotations the phase of each state proportionally to its assigned cost.

- For the max-cut problem the rotation angle is

$$\gamma \, C(\boldsymbol{z})$$

- γ is a variational parameter and $C(\boldsymbol{z})$is the value of the cut for solution $\boldsymbol{z}$.



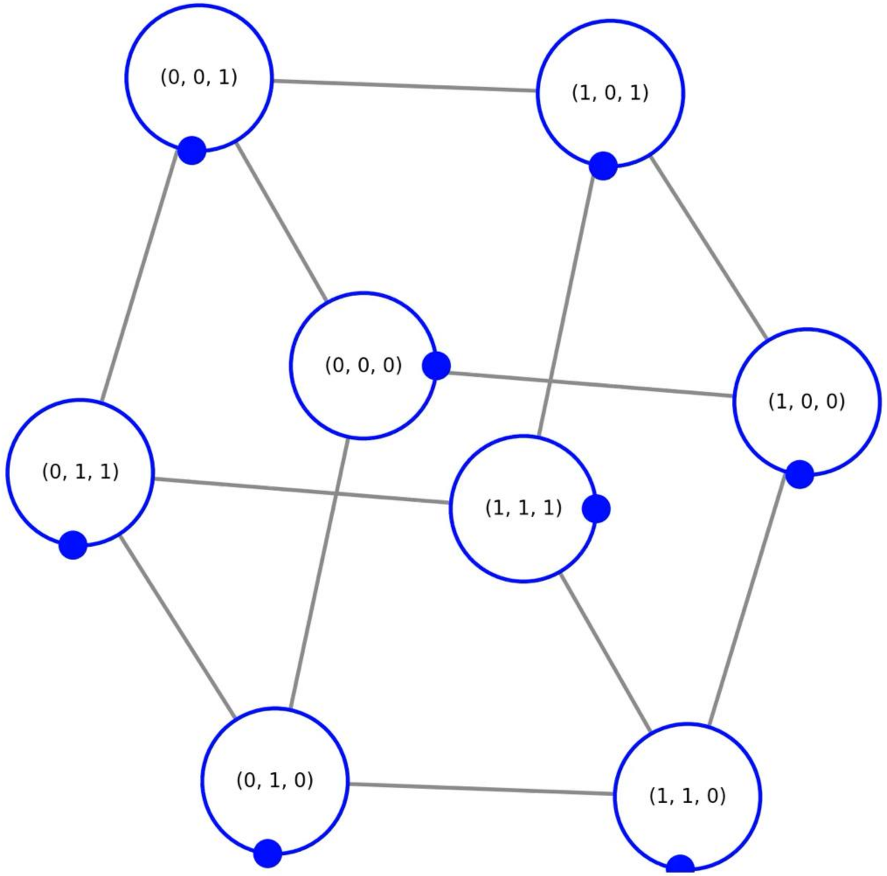
The Quantum Approximate Optimisation Algorithm

- **Third:** a mixing-unitary drives the transfer of amplitude between basis states.

- This is typically achieved by applying a Pauli-X rotation to each qubit for angle β, where β is another variational parameter.

- During this step, phase differences due to the phase-encoded cut-values cause destructive and constructive interference.

- The phase-shift and mixing unitary are applied for p iterations. As p increases, so does the degree of possible interference.



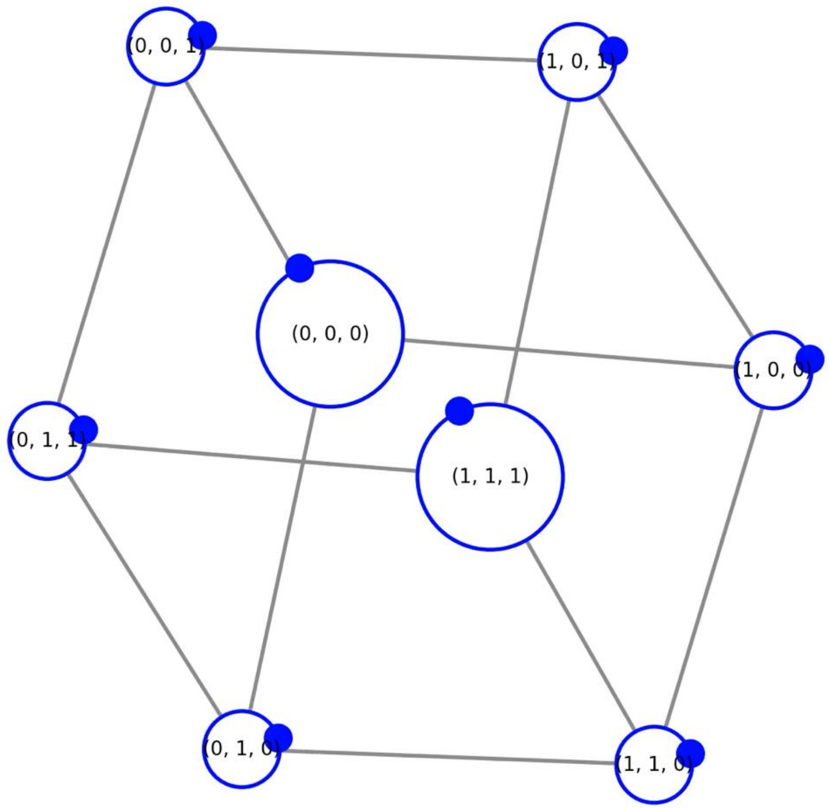
The Quantum Approximate Optimisation Algorithm

- The full set of variational parameters is:

$$\gamma = (\gamma_1, \dots \gamma_p)$$
$$\beta = (\beta_1, \dots \beta_p)$$

- These are tuned by a classical optimiser whose objective function is the average measured (expectation value) of the cut-values.

- As the objective function increases, so does the probability of measuring an optimal, or near-optimal, solution.



Conclusion

- Quantum computing is progressing rapidly, but is still not production-ready.
- Current efforts focus on near-term systems: how to construct and use them effectively.
- Computing is a broad, collaborative field—quantum computing is no different, drawing on physics, math, engineering, and beyond.
- You can get involved: domain-specific applications (beyond physics) are driving interest, while tools from theoretical physics are increasingly being repurposed for new algorithmic insights.

Workshop Activity: Hands-On Quantum Computing

https://github.com/PawseySC/PHYS4004_quantum_computing