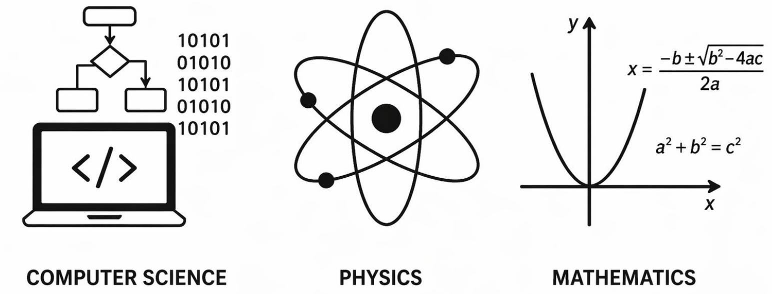
Hes·so

Introduction to quantum computing from a software engineering perspective

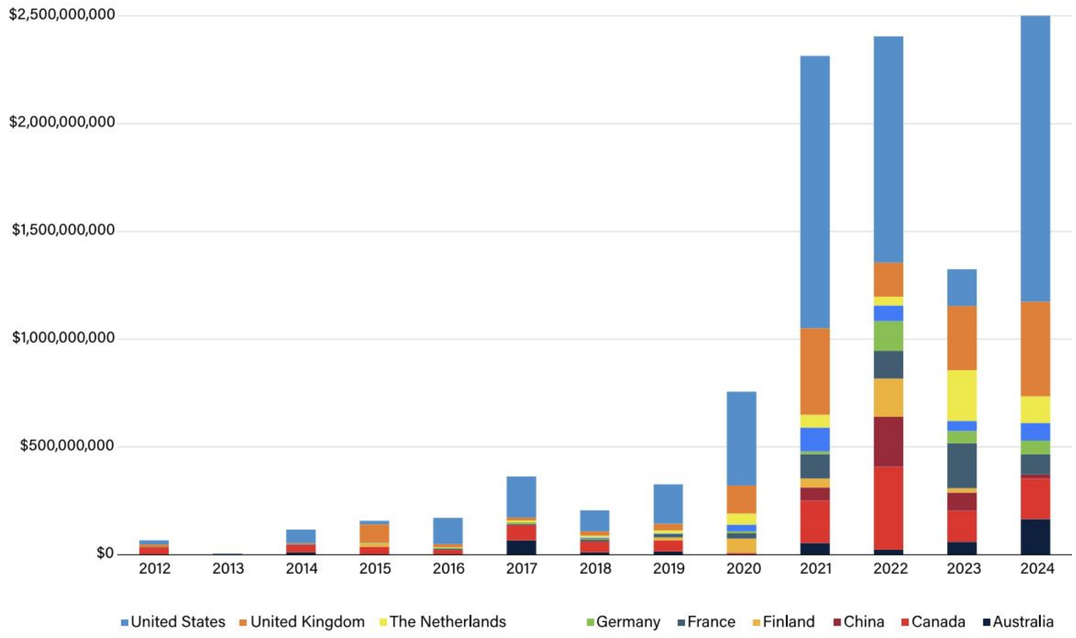
Nabil Abdennadher
University of Applied Sciences, Western Switzerland
(HES-SO, HEPIA)

What is quantum computing ?

- Using principles of quantum mechanics to:
  - solve **some complex** problems faster than on classical computers.
  - encode and process information
- Unlike classical computers, which use binary bits (0, 1), quantum computers uses qubits which are governed by laws of quantum mechanics.
- A multidisciplinary field covering computer sciences, physics, mathmatics
- Includes hardware research and application development



Why now ?



QIR, Quantum Idex report MIT 2025 p.48

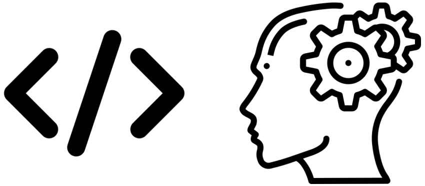
In Switzerland …

- Swiss Quantum initiative
- Open Quantum institute

Why now ?

1. Quantum computing is moving beyond its theoretical niche and becoming a practical reality.
2. The first quantum computers are already on the market, and in the cloud

A new paradigm for programming and software engineering



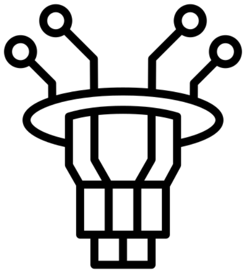
Quantum-based technologies

1. **Ultra-sensitive sensors**, for cutting-edge sensing
2. **Quantum communication**, for secure information exchange
3. **Quantum computers**, to solve our most complex calculations

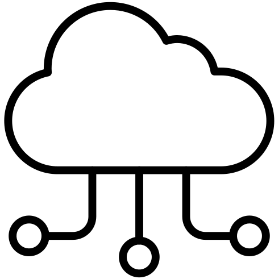
We are interested in quantum computers

Hes·so

Quantum computers/programming: Why now ?



First generation of Quantum Hardware



Software Development Kit & Cloud access

IQM

D-Wave

XANADU

PENNYLANE

aws

Microsoft

Google

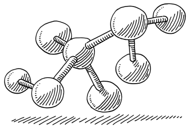
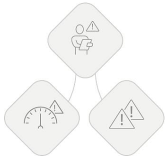
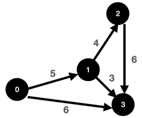
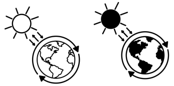
IBM

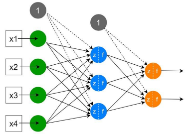
IONQ

Hes·so

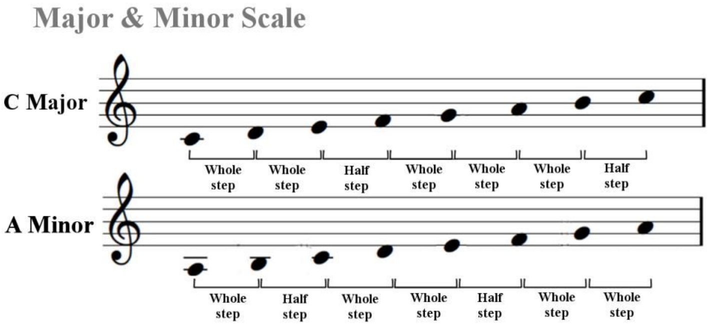
Most promising domains of application

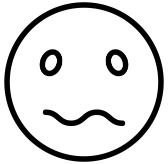
1. Simulation of **chemical processes** to develop new medicines or create new materials,
2. Simulation of scenarios involving multiple variables to enable **risk calculations** in the banking and insurance sectors.
3. In logistics, quantum computers are able to solve **traffic or supply chain optimisation problems**.
4. **Climate modelling**
5. Production of more **environmentally friendly agricultural fertilisers**.
6. Training **artificial intelligence models**.



From theory to practice





Hes·so

My promises …

- We don't deal with quantum physics
- We rely on mathmatic theory to represent and model quantum physics

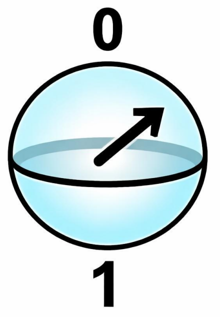
- No maths (theory) ... unless it's necessary (unavoidable)
- To make good progress in this tutorial, We need your trust.

The promise you made, Cock Robin, 1985

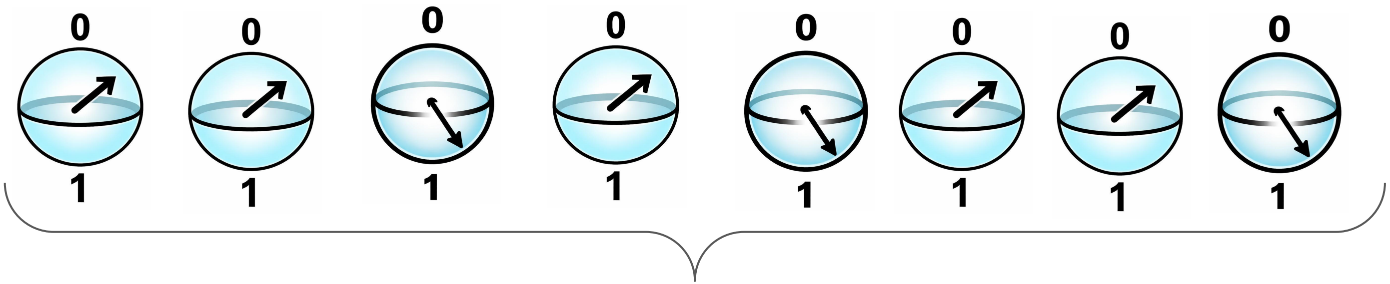
Plan

- 14h00 - 15h00: Fondamentals of quantum computing
- 15h00 - 16h00 : Keynote …
- 16h00 - 17h00 : Deutsch algorithm + Grover Algorithm

Qubit



- A quantum particle that has two distinct states at the same time: 0 **and** 1



These 8 qubits contains, **at the same time**, all values ranging from 00000000 to 11111111

- When we measure a Qubit, we can only obtain one state (0 **or** 1)
- Once a Qubit is measured, it **collapses** into one of the two states: 0 **or** 1

Superposition

$$|\psi> = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = a_0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + a_1 \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

a_0 and a_1 are complex numbers

$$|a_0|^2 + |a_1|^2 = 1$$

State “0” → $|0\rangle$

State “1” → $|1\rangle$

Ket notation

$$|\psi> = a_0 |0> + a_1 |1>$$

Dirac notation

Superposition

$$|\psi> = a_0 |0> + a_1 |1>$$

$$|a_0|^2 + |a_1|^2 = 1$$

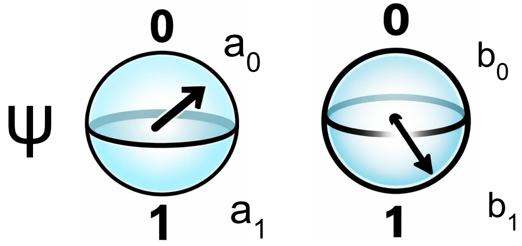
When measuring |ψ⟩, we obtain:

- The “0” (|0⟩) state with a probability of $|a_0|^2$ and |ψ⟩ collapses to |0⟩ state
- The “1” (|1⟩) state with a probability of $|a_1|^2$ and |ψ⟩ collapses to |1⟩ state

Superposition

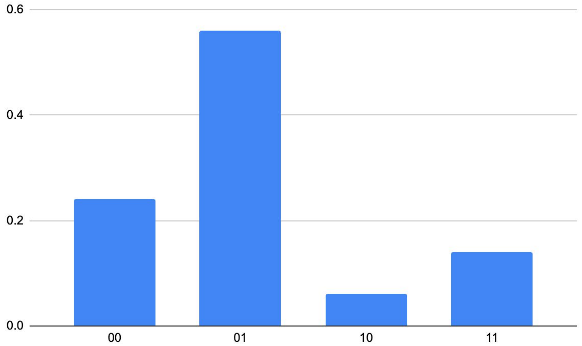
With two qubits (*a* and *b*), we have now 4 states:

- "00" state with a probability of $P_{00} = (a_0 b_0)^2$
- "01" state with a probability of $P_{01} = (a_0 b_1)^2$
- "10" state with a probability of $P_{10} = (a_1 b_0)^2$
- "11" state with a probability of $P_{11} = (a_1 b_1)^2$



$$\Psi = P_{00}|00\rangle + P_{01}|01\rangle + P_{10}|10\rangle + P_{11}|11\rangle$$

$$|P_{00}| + |P_{01}| + |P_{10}| + |P_{11}| = 1$$



Why absolute values ?

With *n* qubits, we have 2^n possible states

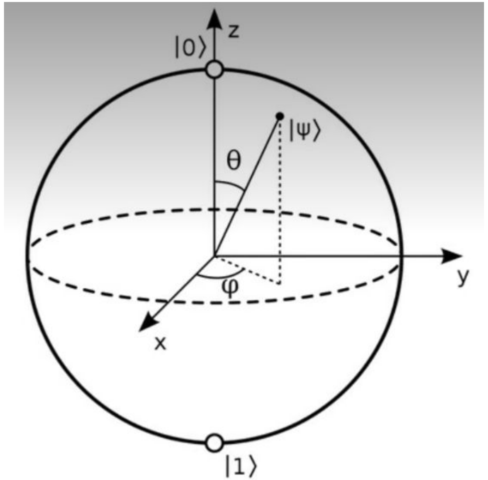
Bloch sphere

$$|\psi> = a_0 |0> + a_1 |1>$$

Since a_0 and a_1 are complex, we have 4 Degrees of Freedom (DoF)

$$|\psi> = \cos\left(\frac{\theta}{2}\right)|0> + e^{i\varphi}\sin\left(\frac{\theta}{2}\right)|1>$$

… And here we have 2 DoF: θ and φ



For further details, scan me

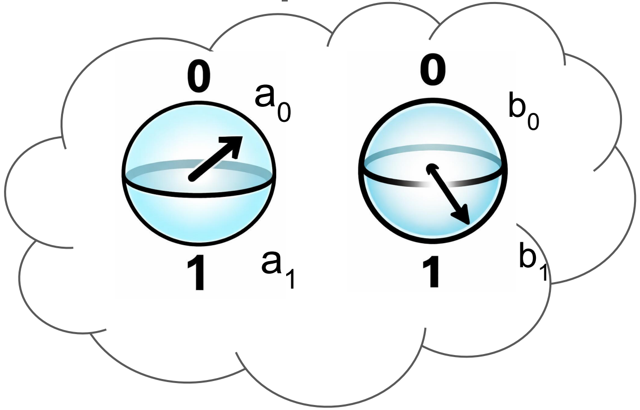
We go from 4 DoF to 2 DoF thanks to two constraints:

- Only relative phase matters
- Normalization condition

Entanglement

- In classical computers, the bits are independent from each others
- In quantum computers, Qubits can be entangled:
  - They become part of one large quantum state together
  - If we update a_0, a_1, b_0 or b_1, we update the probability distribution of the whole system
- Entanglement is:
  - Distance-independent
  - Sensitive to noise
- The real enemy of entanglement is the environmental interaction

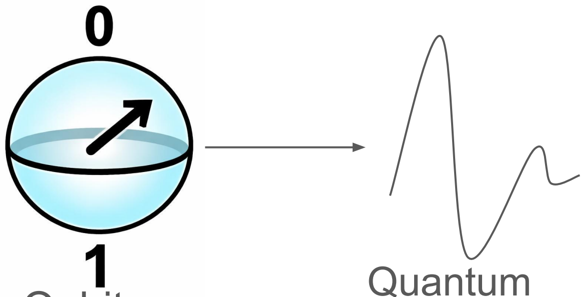
$\Psi = |a\ b\rangle$



https://www.youtube.com/watch?v=-UlxHPIEVqA&t=221s

For further details, scan me (first 7 minutes)

Interference

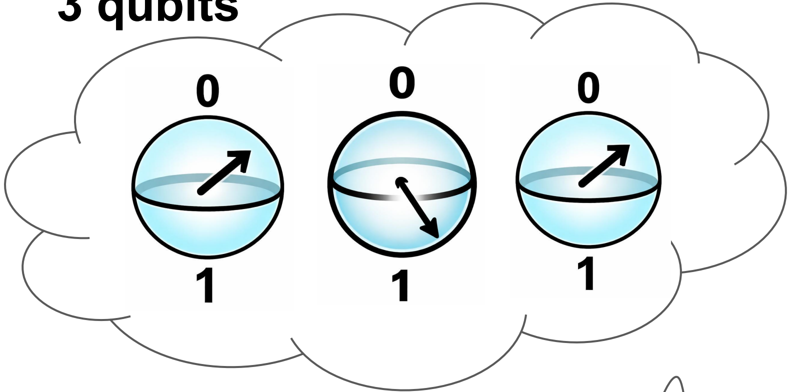


Qubit

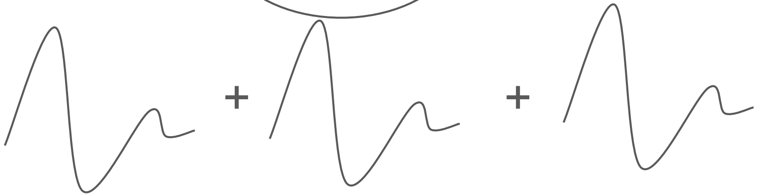
Quantum wavefunction

Entangled Quantum system: 3 qubits

Qubits



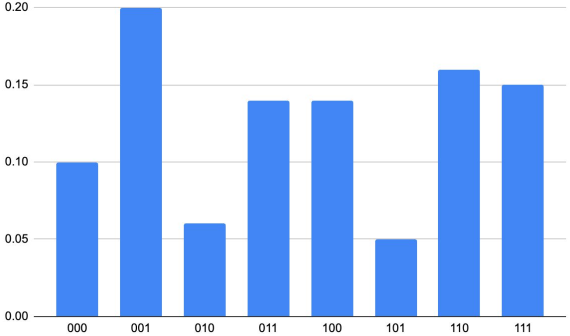
Quantum wavefunction



Interference

Quantum aggregation wavefunction

Probability distribution

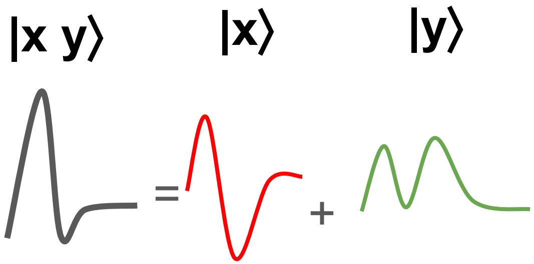


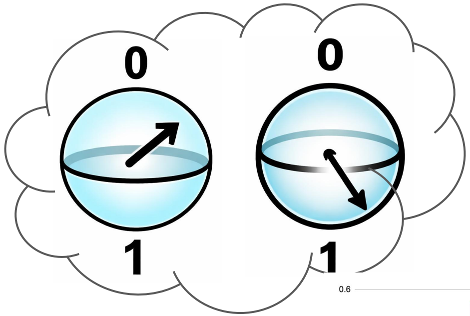
https://www.youtube.com/watch?v=-UIxHPIEVqA&t=221s

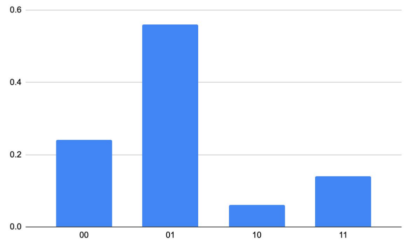
For further details, scan me (first 7 minutes)

Interference

Quantum system: 2 qubits: |x y⟩

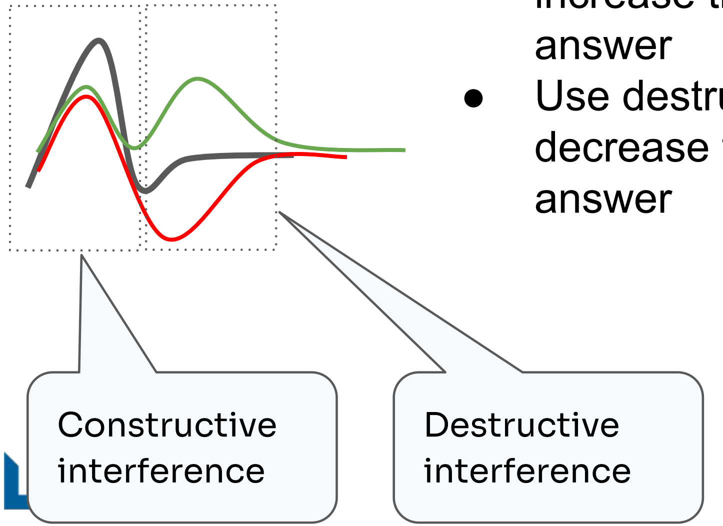






- Use constructive interference to increase the probability of correct answer
- Use destructive interference to decrease the probability of incorrect answer

Probability distribution

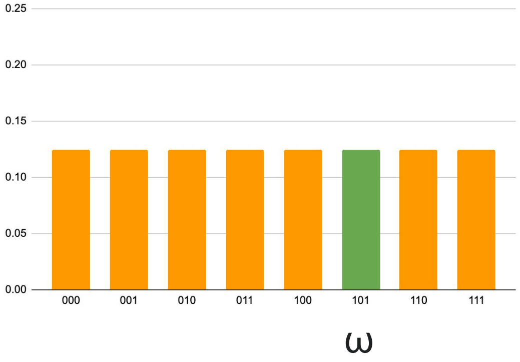


https://www.youtube.com/watch?v=-UIxHPIEVqA&t=221s

For further details, scan me (first 7 minutes)

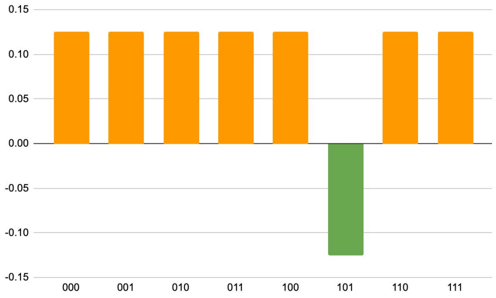
Example: Interference in Grover algorithm

000	0.125
001	0.125
010	0.125
011	0.125
100	0.125
101	0.125
110	0.125
111	0.125



Marking the solution

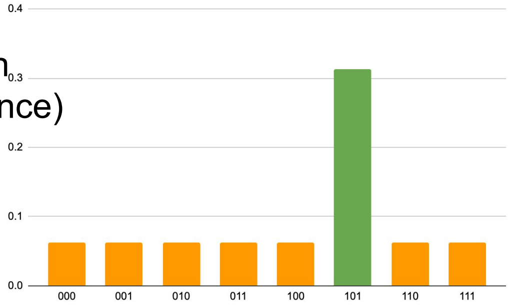
000	0.125
001	0.125
010	0.125
011	0.125
100	0.125
101	-0.125
110	0.125
111	0.125



$a_i \leftarrow 2*\text{average} - a_i$

Amplifying the solution
(constructive interference)

000	0.0625
001	0.0625
010	0.0625
011	0.0625
100	0.0625
101	0.3125
110	0.0625
111	0.0625



Tensor product

What does $|a\ b\rangle$ mean ?

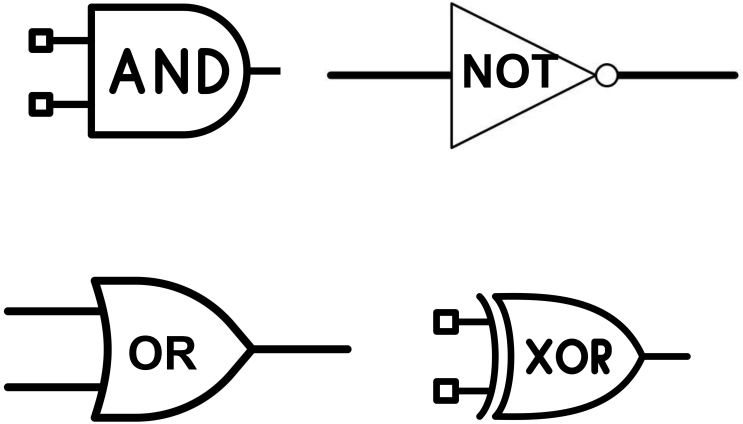
$|a\rangle = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix}$ $|b\rangle = \begin{pmatrix} b_0 \\ b_1 \end{pmatrix}$

$|a\ b\rangle = |a\rangle \otimes |b\rangle = \begin{pmatrix} a_0 b_0 \\ a_0 b_1 \\ a_1 b_0 \\ a_1 b_1 \end{pmatrix}$

Tensor product notation

Do not confuse with $\oplus$ (XOR)

Classical vs. Quantum gates



Operator	Gate(s)		Matrix
Pauli-X (X)	X	$\oplus$	$\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$
Pauli-Y (Y)	Y		$\begin{bmatrix}0 & -i\\i & 0\end{bmatrix}$
Pauli-Z (Z)	Z		$\begin{bmatrix}1 & 0\\0 & -1\end{bmatrix}$
Hadamard (H)	H		$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}$
Phase (S, P)	S		$\begin{bmatrix}1 & 0\\0 & i\end{bmatrix}$
$\pi/8$ (T)	T		$\begin{bmatrix}1 & 0\\0 & e^{i\pi/4}\end{bmatrix}$
Controlled Not (CNOT, CX)			$\begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 0 & 1\\0 & 0 & 1 & 0\end{bmatrix}$
Controlled Z (CZ)	Z		$\begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\0 & 0 & 1 & 0\\0 & 0 & 0 & -1\end{bmatrix}$
SWAP			$\begin{bmatrix}1 & 0 & 0 & 0\\0 & 0 & 1 & 0\\0 & 1 & 0 & 0\\0 & 0 & 0 & 1\end{bmatrix}$
Toffoli (CCNOT, CCX, TOFF)			$\begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\0 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\\0 & 0 & 0 & 0 & 0 & 0 & 1 & 0\end{bmatrix}$

Quantum gates: reversability, norm-preservation

Each quantum gate is represented by a matrix A which has two characteristics:

- Norm-preservation

$$|a> = \sum_i a_i \cdot |i> \ and \ |b> = A\ |a> \ with \ |b> = \sum_i b_i \cdot |i>$$

$$with\ |i> = |0000\ldots00> \ to \ |1111\ldots11>$$

$$\sum_i a_i = \sum_i b_i = 1$$

- Reversability: ∃ A’, such as A * A’ = I

Quantum gates: reversability, norm-preservation

A unitary matrix **U** guarantees these two characteristics:

- Norm-preservation
- Reversability: ∃ U', such as U * U' = I

$$U^{\dagger} = U^{-1} \Rightarrow U^{\dagger} * U = I$$

$$U^{\dagger} = (U^{*})^{T}$$

- U^{*} is the matrix conjugate of U: the matrix obtained by taking the complex conjugate of each element of U
  - $a + i * b \rightarrow a - i * b$
- U^{T} is the transpose of U: swap rows and columns

Single Qubit gates

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$X\,(\,|\Psi>) = X\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \beta \\ \alpha \end{pmatrix}$$

$$X|0> \; = |1>$$

$$X|1> \; = |0>$$

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$Y\,(\,|\Psi>) = Y\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = i\begin{pmatrix} -\beta \\ \alpha \end{pmatrix}$$

$$Y|0> \; = i\;|1>$$

$$Y|1> \; = -\,i\;|0>$$

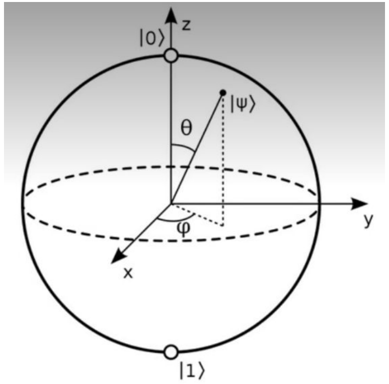
$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$Z\,(\,|\Psi>) = Z\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \alpha \\ -\beta \end{pmatrix}$$

$$Z|0> \; = |0>$$

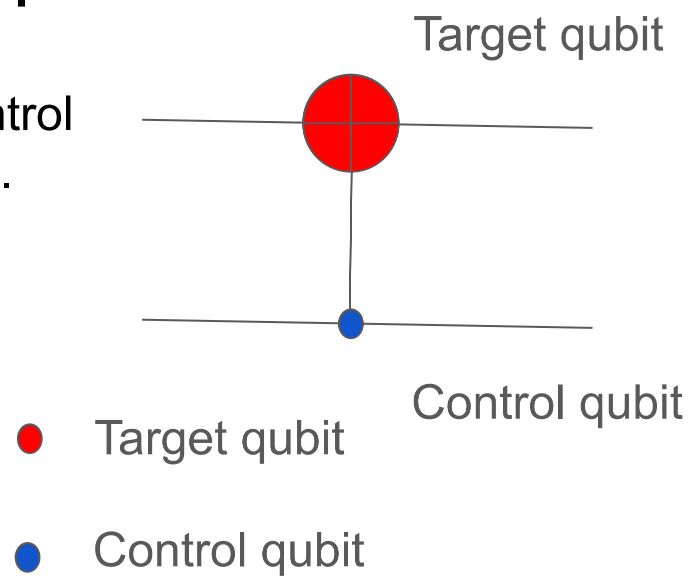
$$Z|1> \; = -\,|1>$$

X, Y and Z rotate the qubit π radians around the specified axis



Multi-Qubit gates: CNOT

CNOT applies X gate on the target qubit if the control qubit is a 1, does nothing if the control qubit is a 0.



Example:

$$CNOT\left(\frac{1}{\sqrt{2}}\,|001> + \frac{1}{\sqrt{2}}|010>\right) =$$

$$\frac{1}{\sqrt{2}}\,(\,CNOT\,|001> + CNOT\,|010>\,) =$$

$$\frac{1}{\sqrt{2}}\,(\,|011> + |010>\,)$$

$$\textbf{CNOT}\,|a, b\rangle = |a, a \oplus b\rangle$$

Control qubit

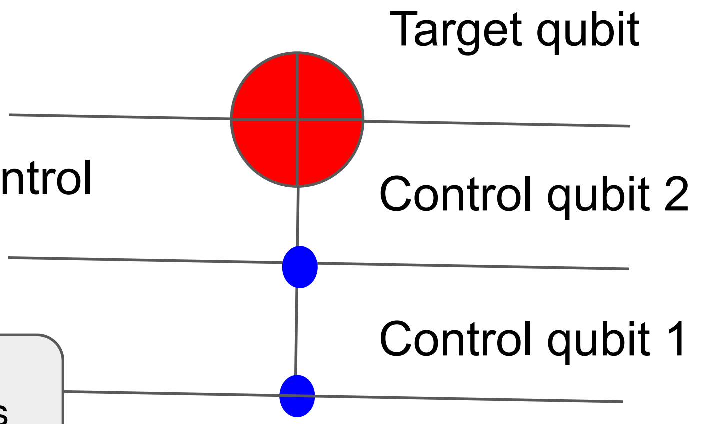
Target qubit

XOR

Multi-Qubit gates: Toffolli

Same as CNOT, but has two control qubits.

In this example, the two first qubits are the control qubits. The 3rd qubit is the target qubit.



$$|\Psi> = \frac{1}{\sqrt{3}}|000> + \frac{\sqrt{2}}{\sqrt{3}}|110>$$

The 3rd qubit shifts from 0 to 1

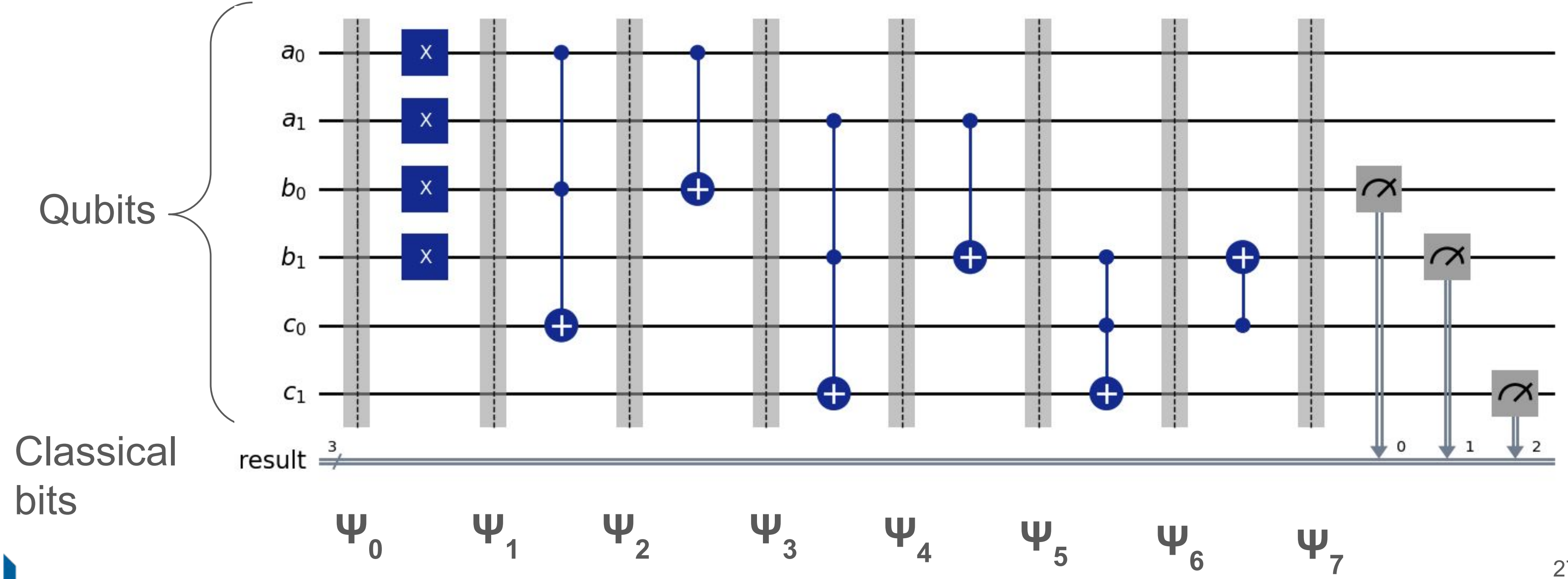
$$TOFFOLI\ (|\psi \ldots \frac{1}{\sqrt{3}}|000> + \frac{\sqrt{2}}{\sqrt{3}}|111>$$

The target qubit does not shift from 0 to 1, because the two control qubits are not 1

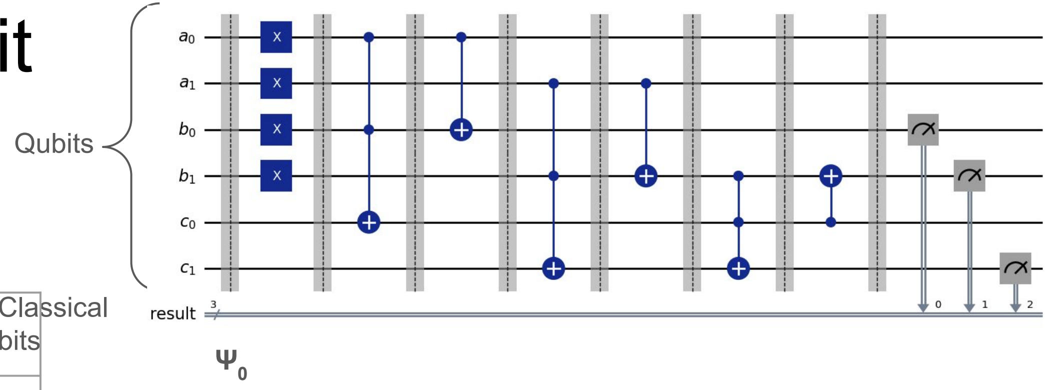
Toffoli |a, b, c⟩= |a, b, c⊕(a AND b)⟩

Quantum Circuit

- Quantum circuits are for quantum computing what algorithms are for classical computing
- The horizontal is the order in which we apply the gates (from left to right)

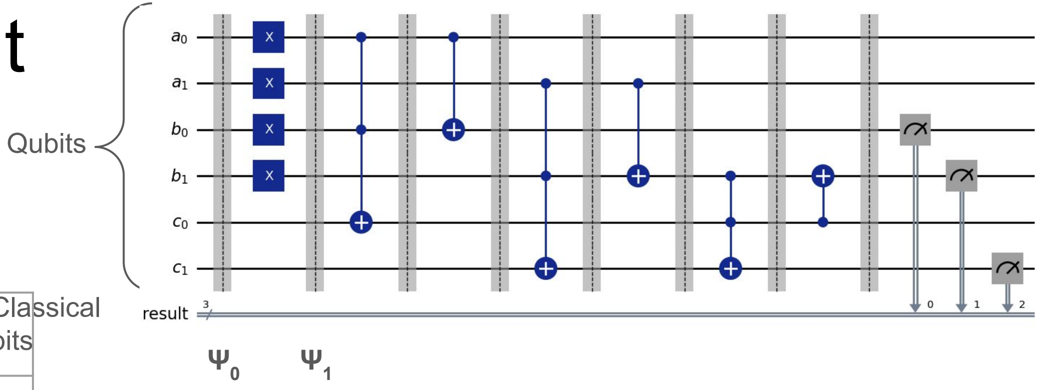


Quantum Circuit



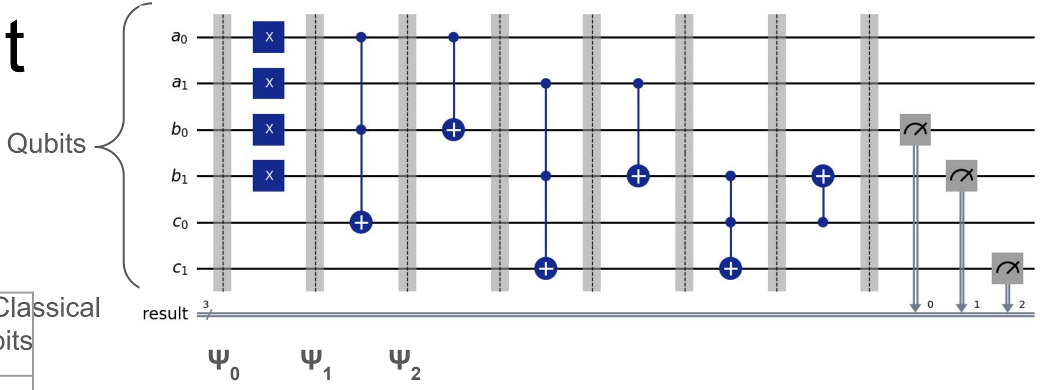
Ψ_i	a_0	a_1	b_0	b_1	c_0	c_1
Ψ_0	0	0	0	0	0	0

Quantum Circuit



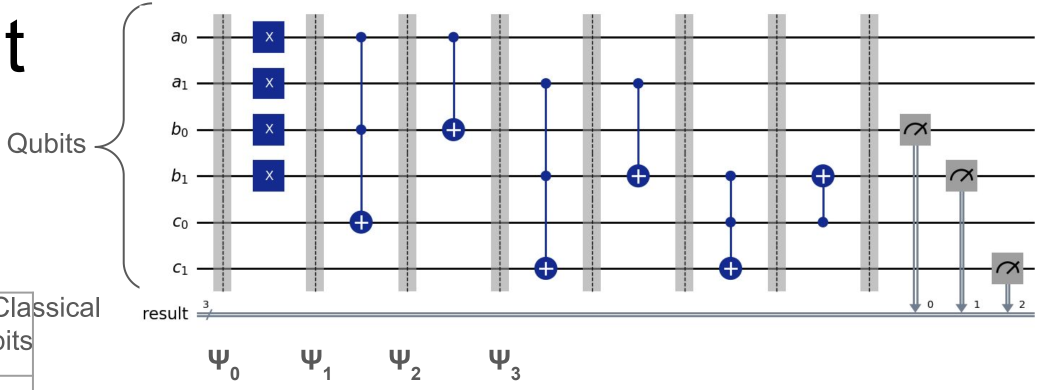
ψ_i	a_0	a_1	b_0	b_1	c_0	c_1
ψ_0	0	0	0	0	0	0
ψ_1	1	1	1	1	0	0

Quantum Circuit



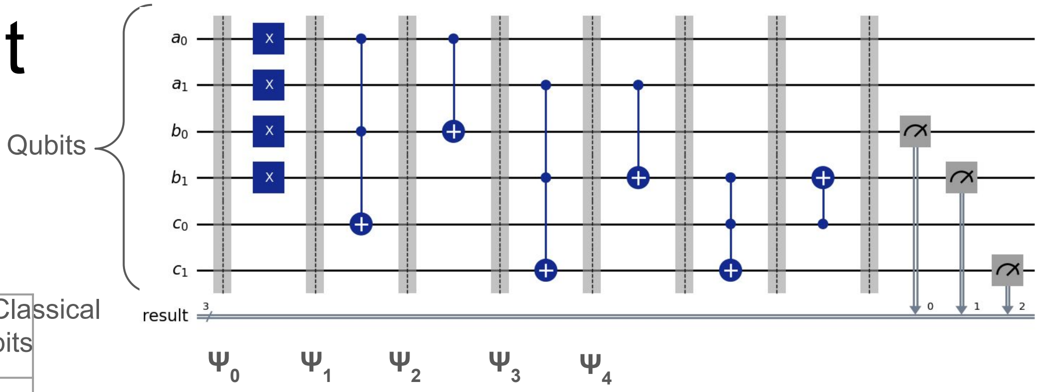
ψ_i	a_0	a_1	b_0	b_1	c_0	c_1
ψ_0	0	0	0	0	0	0
ψ_1	1	1	1	1	0	0
ψ_2	1	1	1	1	1	0

Quantum Circuit



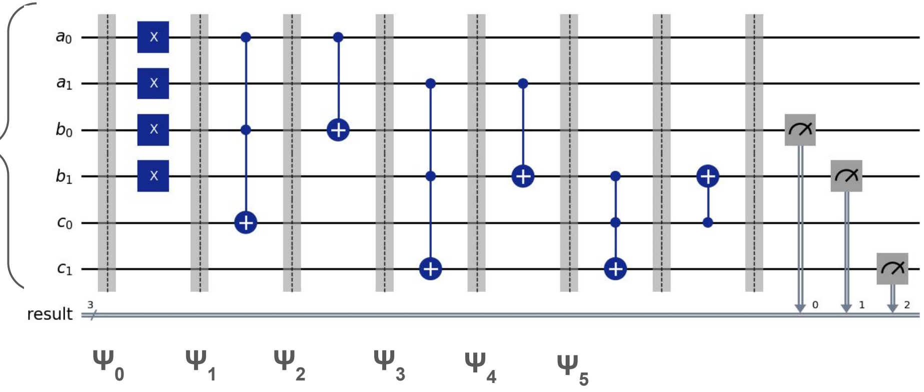
ψ_i	a_0	a_1	b_0	b_1	c_0	c_1
ψ_0	0	0	0	0	0	0
ψ_1	1	1	1	1	0	0
ψ_2	1	1	1	1	1	0
ψ_3	1	1	0	1	1	0

Quantum Circuit



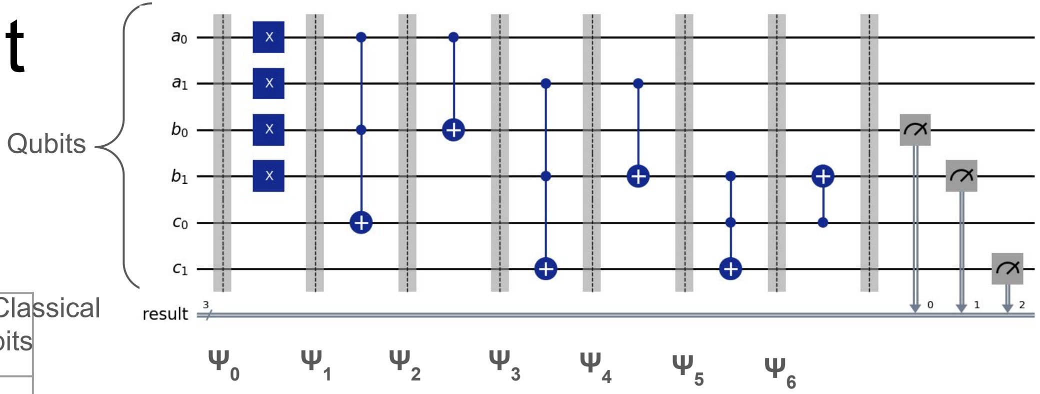
ψ_i	a_0	a_1	b_0	b_1	c_0	c_1
ψ_0	0	0	0	0	0	0
ψ_1	1	1	1	1	0	0
ψ_2	1	1	1	1	1	0
ψ_3	1	1	0	1	1	0
ψ_4	1	1	0	1	1	1

Quantum Circuit



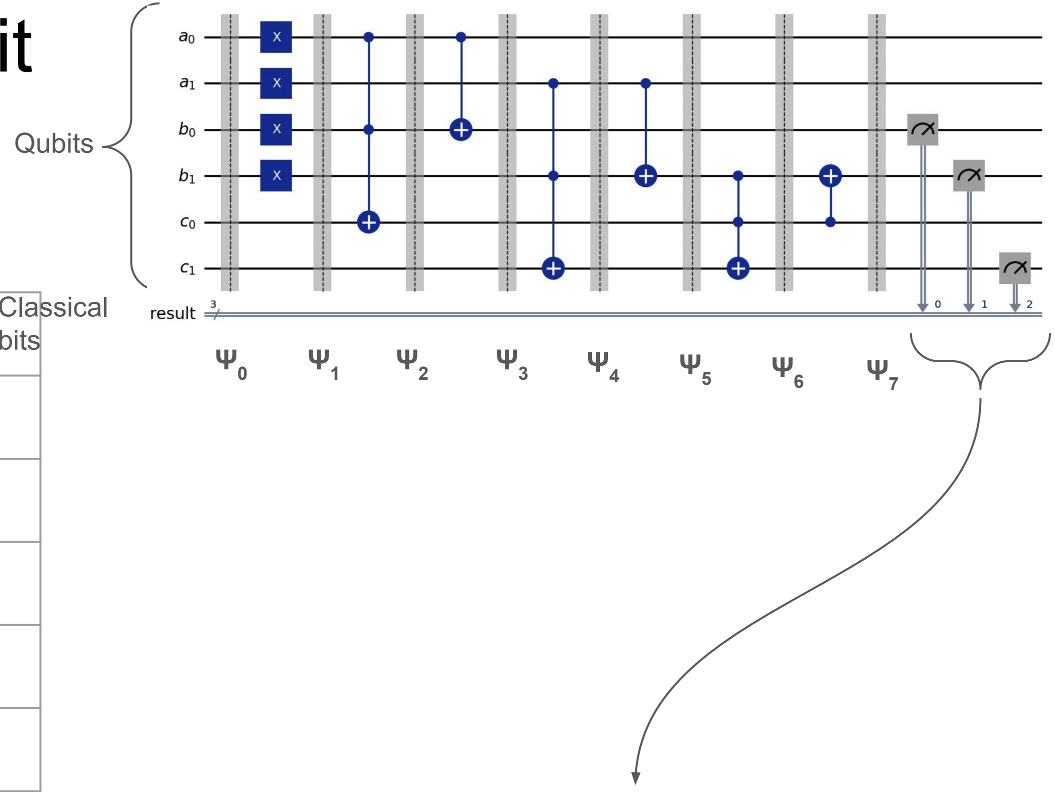
ψ_i	a_0	a_1	b_0	b_1	c_0	c_1
ψ_0	0	0	0	0	0	0
ψ_1	1	1	1	1	0	0
ψ_2	1	1	1	1	1	0
ψ_3	1	1	0	1	1	0
ψ_4	1	1	0	1	1	1
ψ_5	1	1	0	0	1	1

Quantum Circuit



Ψ_i	a_0	a_1	b_0	b_1	c_0	c_1
Ψ_0	0	0	0	0	0	0
Ψ_1	1	1	1	1	0	0
Ψ_2	1	1	1	1	1	0
Ψ_3	1	1	0	1	1	0
Ψ_4	1	1	0	1	1	1
Ψ_5	1	1	0	0	1	1
Ψ_6	1	1	0	0	1	1

Quantum Circuit



ψ_i	a_0	a_1	b_0	b_1	c_0	c_1
ψ_0	0	0	0	0	0	0
ψ_1	1	1	1	1	0	0
ψ_2	1	1	1	1	1	0
ψ_3	1	1	0	1	1	0
ψ_4	1	1	0	1	1	1
ψ_5	1	1	0	0	1	1
ψ_6	1	1	0	0	1	1
ψ_7	1	1	0	1	1	1

result[0] ← $|b_0>$
result[1] ← $|b_1>$
result[2] ← $|c_1>$

Let’s execute this quantum circuit …

- **Qiskit** : open-source, Python-based, software stack for quantum computing, originally developed by IBM Research and first released in 2017
- A Hello World program available on moodle
- We will be using a simulator

Let's execute this quantum circuit …

```
#create a quantum simulator that behaves like a quantum computer
(on your laptop / Colab)
Simulator= AerSimulator()
#Converts your circuit into a form the simulator understands
compile=transpile(qc,Simulator)
#run(...): Executes the circuit
#shots=20: Runs the circuit 20 times
#result(): Gets the execution result
#get_counts(): Returns how many times each result appeared
result= Simulator.run(compile, shots=20).result().get_counts()
```

qc : the quantum circuit to execute

Let’s execute this quantum circuit …

```
qc.measure(b[0], res[0])
qc.measure(b[1], res[1])
qc.measure(c[1], res[2])
```

To summarize

- Three concepts: Superposition, Entanglement, interference
- Tensor product: A mathematical operation that combines qubits into a single, larger multi-qubits.
- Basic gates : mono-qubits and multi-qubits
- Quantum circuits : quantum algorithms

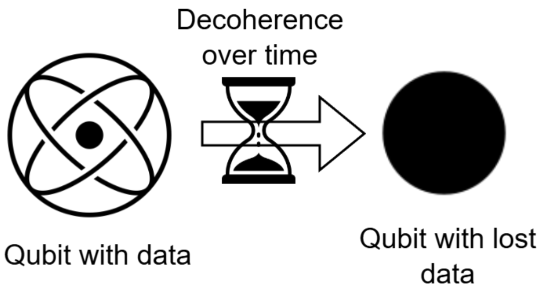
To summarize: Classical vs Quantum Computing ?

Classical	**Quantum**
Bit : 0 or 1	Qubit : 0,1 or Both (Superposition)
Bits are independent	Qubits can be correlated (Entanglement)
Logical Gate (XOR, AND, …)	Quantum Logical Gate (C-NOT, Toffoli, …)
Massive computation : implementation through parallelism	Massive computation : Superposition
Program, Function	Circuit, Quantum Algorithm
Result : Deterministic	Result : Probabilistic

Research topics

Current Problems and Researches

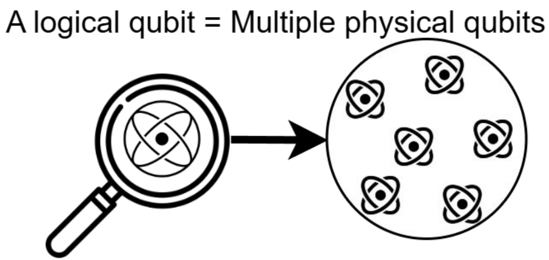
1. Decoherence



Quantum states are unstable
(Collapse after few ms)

Nielsen & Chuang
[20]

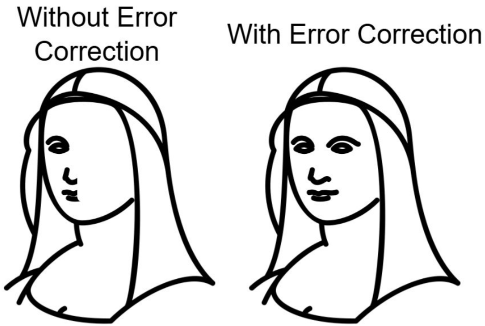
2. Limited Q resources



Lack of Logical Qubits

Preskill [51]

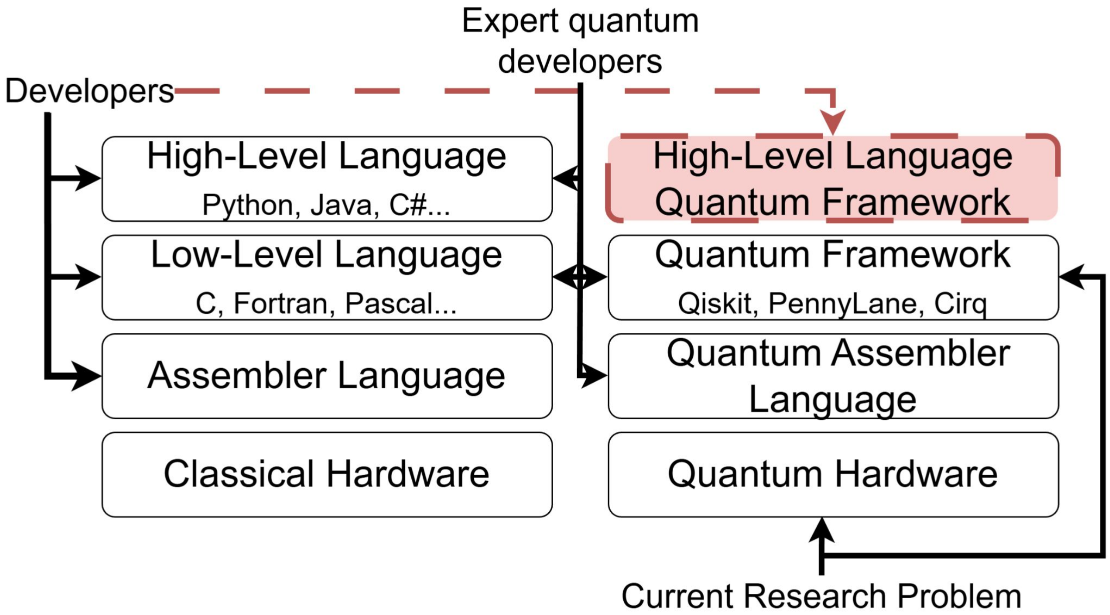
3. Error Correction



Noise from operations
(Increase logical qubit Reliability)

Preskill [51], Cerezo et al. [21]

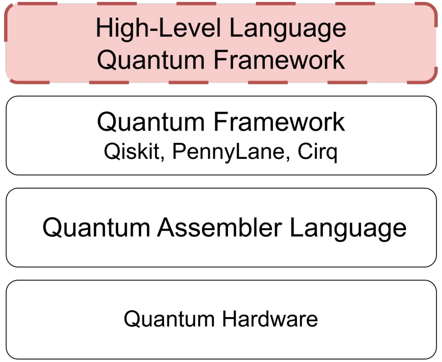
What is missing from a software engineering perspective …



Following the classical IT evolution, Quantum IT is currently missing its top layer
The one that would give every developer access to quantum computing

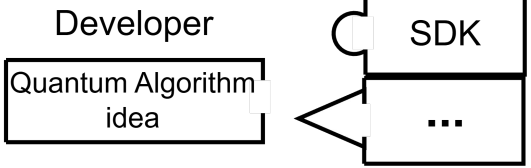
What is missing from a software engineering perspective …

Lack of High-Level Abstractions



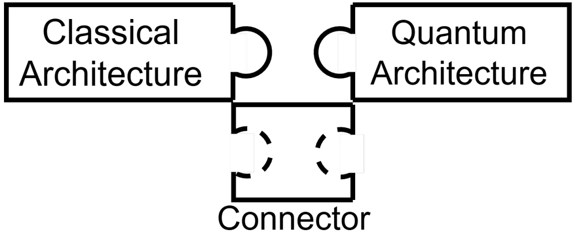
Lot of knowledge required [24,36]

Fragmentation of the Quantum Software Ecosystem



Poor portability among Vendor [24,27]

Managing Hybrid Workflows



Classical and Quantum can not be joint easily [18,21]

References

- [11] IBM, IBM - Qiskit. [Online]. Available: https://www.ibm.com/quantum/qiskit.
- [14] Xanadu, PennyLane. [Online]. Available: https://pennylane.ai/.
- [18] V. Stirbu and T. Mikkonen, "Software Architecture Challenges in Integrating Hybrid Classical-Quantum Systems," in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, 2023, pp. 203–204.
- [20] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*, Cambridge University Press, 2010.
- [21] M. Cerezo et al., "Variational Quantum Algorithms," *Nat Rev Phys*, vol. 3, no. 9, pp. 625–644, 2021.
- [24] M. A. Serrano, J. A. Cruz-Lemus, R. Perez-Castillo, and M. Piattini, "Quantum Software Components and Platforms: Overview and Quality Assessment," *ACM Comput. Surv.*, vol. 55, no. 8, 164:1–164:31, 2022.
- [27] R. LaRose, "Overview and comparison of gate-level quantum software platforms," *Quantum*, vol. 3, p. 130, 2019.
- [30] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, 2009, pp. 169–178.

References

- [31] A. Broadbent, J. Fitzsimons, and E. Kashefi, "Universal blind quantum computation," in *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, 2009, pp. 517–526.
- [36] J. M. Murillo et al., "Quantum Software Engineering: Roadmap and Challenges Ahead," *ACM Trans. Softw. Eng. Methodol.*, vol. 34, no. 5, pp. 1–48, 2025.
- [38] Qiskit Community, *Qiskit: An open-source framework for quantum computing*, 2017.
- [39] V. Bergholm et al., *PennyLane: Automatic differentiation of hybrid quantum-classical computations*, 2022.
- [41] R. Wille et al., "The MQT handbook: A summary of design automation tools and software for quantum computing," in *IEEE International Conference on Quantum Software (QSW)*, 2024.
- [51] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, 2018.

Useful links

- Swiss Quantum initiative
- Open Quantum institute
- Tutorial for beginners