

Introduction to Quantum Computing

A lecture series by IQM and HS RM

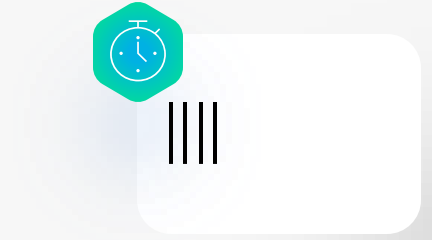
Authors: Stefan Seegerer, Mikio Nakahara, Nikolay Tcholchev

Last Updated 06/2025



**Why think about
different
computing
paradigms?**

Sorting a list of numbers



8

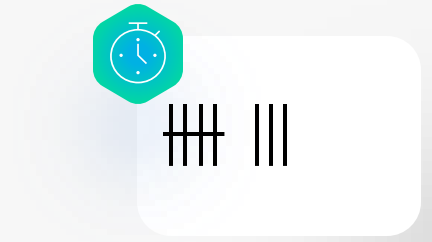
2

5

3



Sorting a list of numbers



2

8

5

3



Sorting a list of numbers



|||| |



2

3

5

8



Thinking about new
computing paradigms can
lead to more efficient
solutions



There are problems, a classical computer cannot (efficiently) solve

Problem with current computers:

- Intractable problems
- Fundamental limits
- Time & power limits

Current approach is not scalable!



“Latest findings suggest global computing is more likely responsible for between 2.1% and 3.9% of greenhouse gas emissions.”

Overview

- **Computational complexity** depends on the resource to implement the algorithm.
- **Physical systems** may be employed for information processing. They sometimes give different computational complexity than digital information processing.

<https://www.miraikan.jst.go.jp/exhibitions/future/internet/>

Photos courtesy of
Miraikan - The National
Museum of Emerging
Science and Innovation



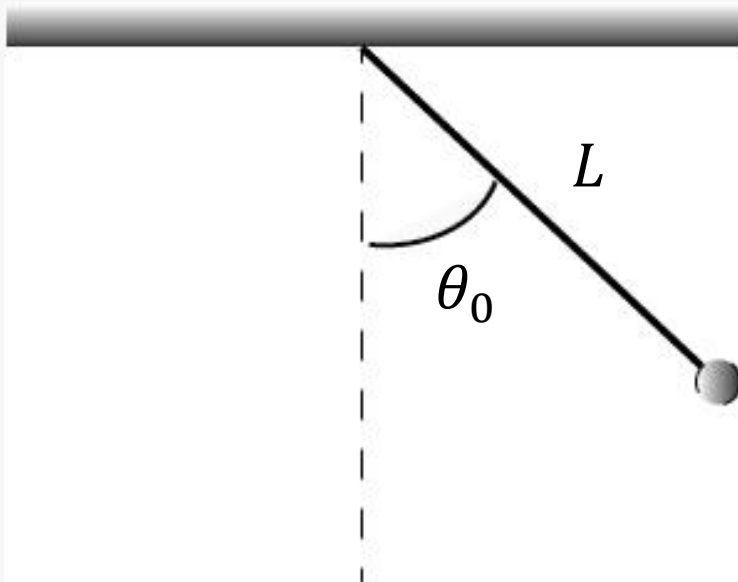
- **Quantum mechanics** describes **microscopic systems**, such as electrons and photons. A state is represented by a **complex vector** and an operation by **a matrix**.
- Quantum information processing and quantum computation employ **quantum systems to store and process information**, which may **reduce computational complexity** for **some** tasks.

Information is Physical. (R Landauer (IBM), 1991)

- **Physics laws** can be used for information processing/computation.
- Example: Evaluate the complete elliptic integral of the first kind

$$K(k) = \int_0^{\pi/2} \frac{1}{\sqrt{1 - k^2 \sin^2 \phi}} d\phi \quad . \quad K(k) = \frac{\pi}{2} \sum_{n=0}^{\infty} \left[\frac{(2n-1)!!}{(2n)!!} \right]^2 k^{2n} .$$

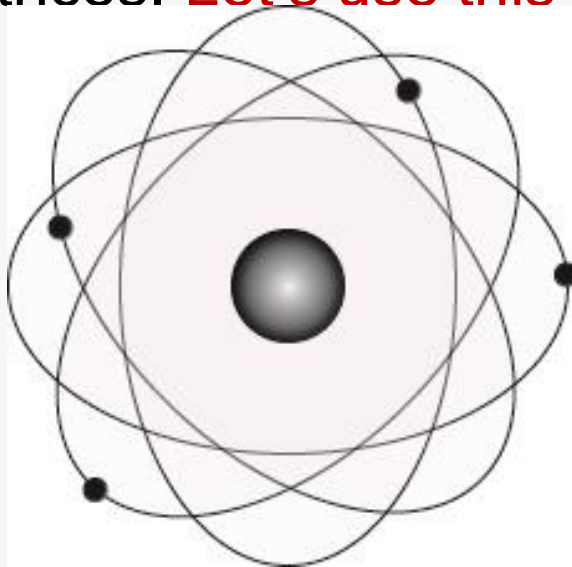
- Use **physics**. The period of a pendulum is $T = 4 \sqrt{\frac{L}{g}} K(k)$, $k = \sin \frac{\theta_0}{2}$.



- It may take a few minutes to set up a pendulum. It is much shorter than the time required to build a digital computer from scratch.
- A physical system may be used as a computational resource. Its **usage is limited** in general though.

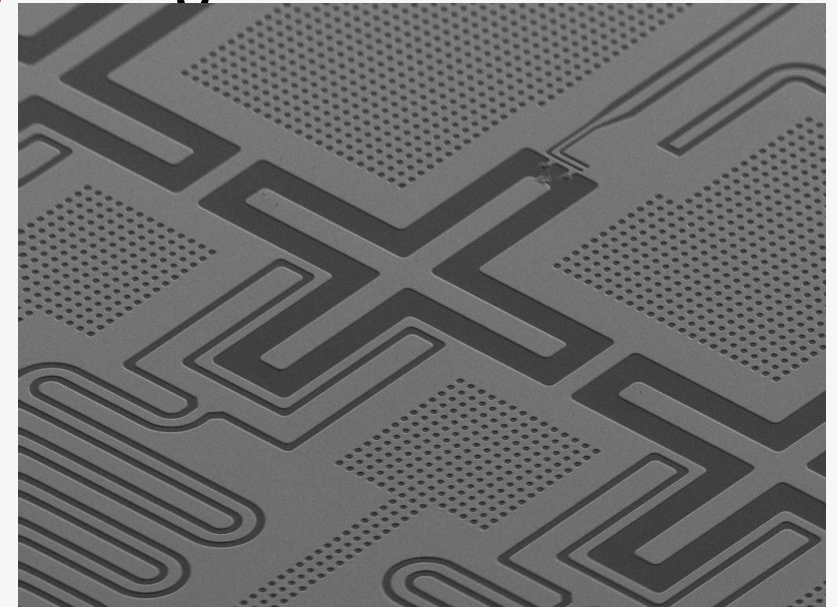
Complexity depends on the device for processing

- Complexity depends on **a system to implement an algorithm**.
- There are some algorithms whose complexity may be reduced if they are implemented **with complex vectors** and **matrices** acting on them.
- A **quantum physics**, the physics of the **microscopic world**, is described by complex vectors and matrices. **Let's use this system to execute such algorithms!**



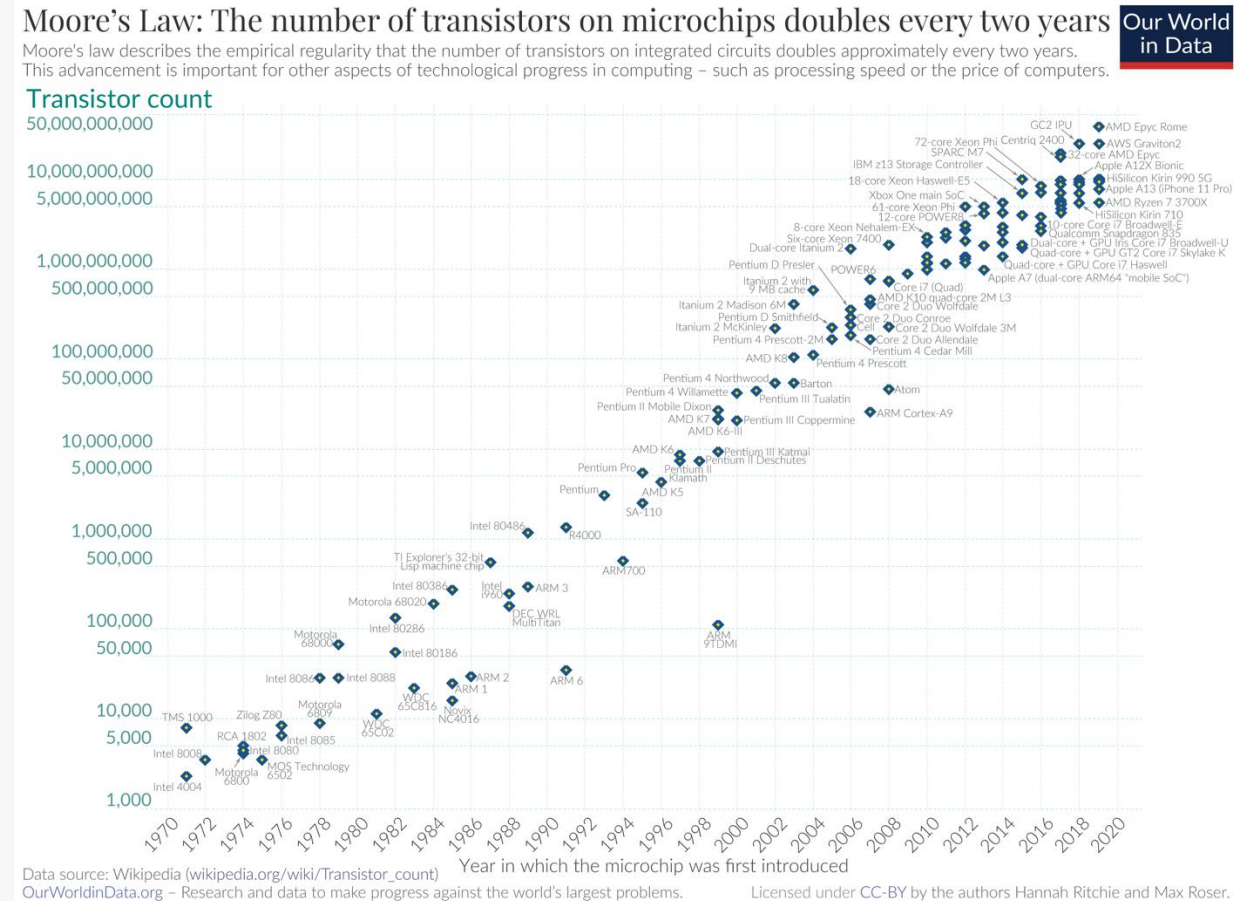
Quantum Information Processing

- **Semiconductor devices** also employ quantum physics to design but there are too many electrons to show **quantum properties**.
- We need a **single electron**, a **single nucleus** or a **single photon** to process information quantum mechanically. Preparation, control and measurement of such systems are **challenging** though.
- Thanks to the progress of nanotechnology, it is now possible to fabricate a nanoscopic system that follows the laws of quantum mechanics. **Superconducting qubits** (=quantum bit) employed in **IQM quantum computers** are an example of such devices.



Another Motivation for Quantum Computing

- **Moore's law:** The number of transistors that can be packed into a CPU is expected to **double every two years**.
- The size of one memory will become as small as an **atom**. Then quantum nature will manifest itself. Computation becomes probabilistic. **Good or bad?**
- Let's use quantum nature to **improve computational complexity**.



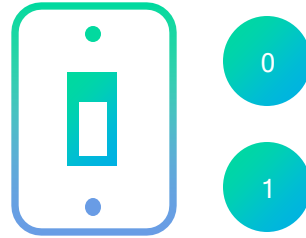
CC-BY 4.0 by Max Roser, Hannah Ritchie

Bits and Qubits

Classical Computer vs. Quantum Computer

Classical bit

2 distinct states

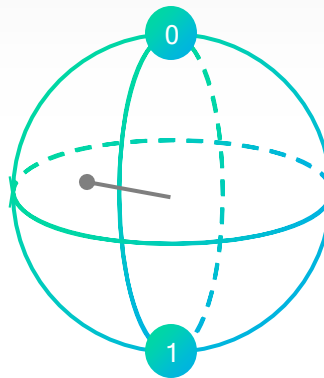


switch on/off

State can be
measured
repeatedly

Quantum bit

Superposition of
2 basis states



Point on the
surface of a
sphere

Measurement of
the state will
cause it to fall
back to 0 and 1,
no repeated
measurement

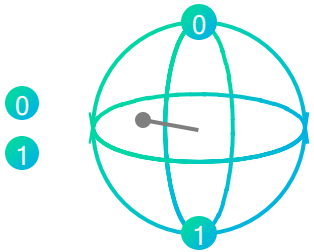
What is quantum computing?

Classical computer vs. Quantum computer

Quantum computers follow different rules than classical computers - those of quantum physics.

Bit vs. Qubit (quantum bit)

A bit can be either 0 or 1, while a qubit can be in a **superposition**: qubit can be both 0 and 1 at the same time. Measurement will yield either 0 or 1.



Two qubits can be **entangled**. Changing one directly impacts the other.



- 1 qubit can be in a superposition of 2 basis states
- A 1 bit state can be just one out them

Basis states:

- 0
- 1

1Qubit

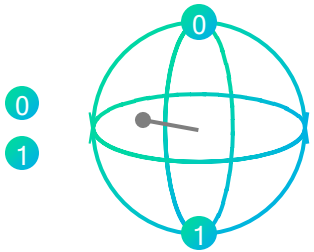
What is quantum computing?

Classical computer vs. Quantum computer

Quantum computers follow different rules than classical computers - those of quantum physics.

Bit vs. Qubit (quantum bit)

A bit can be either 0 or 1, while a qubit can be in a **superposition**: qubit can be both 0 and 1 at the same time. Measurement will yield either 0 or 1.



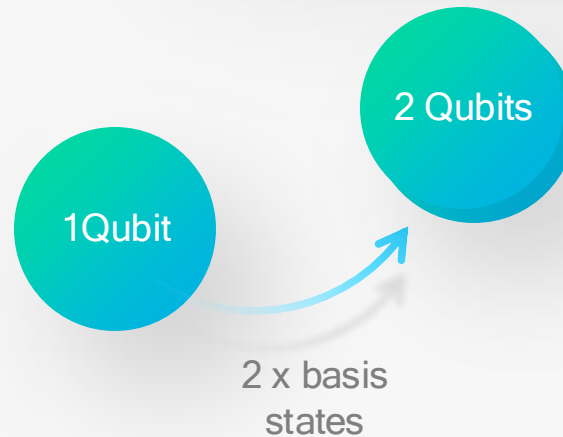
Two qubits can be **entangled**. Changing one directly impacts the other.



- 2 qubits can be in a superposition of all 4 basis states
- A 2-bit state can be just one out them

Basis states:

- 00
- 01
- 10
- 11



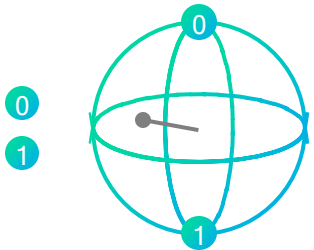
What is quantum computing?

Classical computer vs. Quantum computer

Quantum computers follow different rules than classical computers - those of quantum physics.

Bit vs. Qubit (quantum bit)

A bit can be either 0 or 1, while a qubit can be in a **superposition**: qubit can be both 0 and 1 at the same time. Measurement will yield either 0 or 1.



Two qubits can be **entangled**. Changing one directly impacts the other.



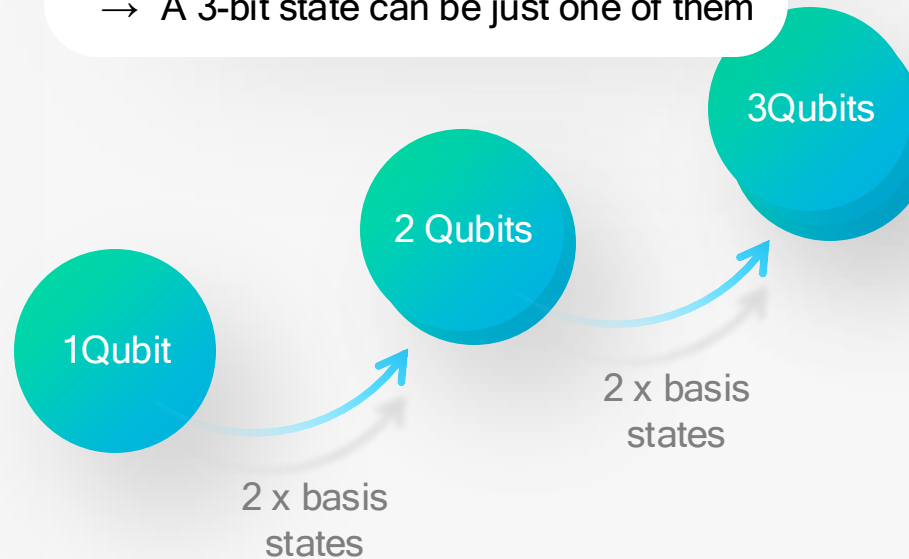
000	001	010	011
100	101	110	111

→ 3 qubits can be in a superposition of all $2^3 = 8$ basis states

→ A 3-bit state can be just one of them

Basis states:

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



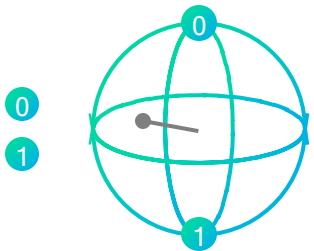
What is quantum computing?

Classical computer vs. Quantum computer

Quantum computers follow different rules than classical computers - those of quantum physics.

Bit vs. Qubit (quantum bit)

A bit can be either 0 or 1, while a qubit can be in a **superposition**: qubit can be both 0 and 1 at the same time. Measurement will yield either 0 or 1.



Two qubits can be **entangled**. Changing one directly impacts the other.



000	001	010	011
100	101	110	111

→ 3 qubits can be in a superposition of all $2^3 = 8$ basis states

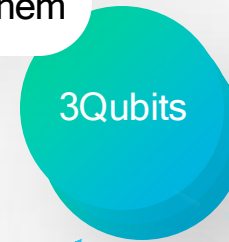
→ A 3-bit state can be just one of them



2 x basis states



2 x basis states



2^{50} states



Classical supercomputers hit a roadblock at 50 qubits (Technology Review)



51 Qubits



300 Qubits

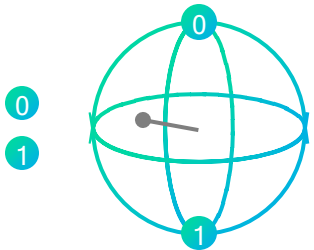
What is quantum computing?

Classical computer vs. Quantum computer

Quantum computers follow different rules than classical computers - those of quantum physics.

Bit vs. Qubit (quantum bit)

A bit can be either 0 or 1, while a qubit can be in a **superposition**: qubit can be both 0 and 1 at the same time. Measurement will yield either 0 or 1.



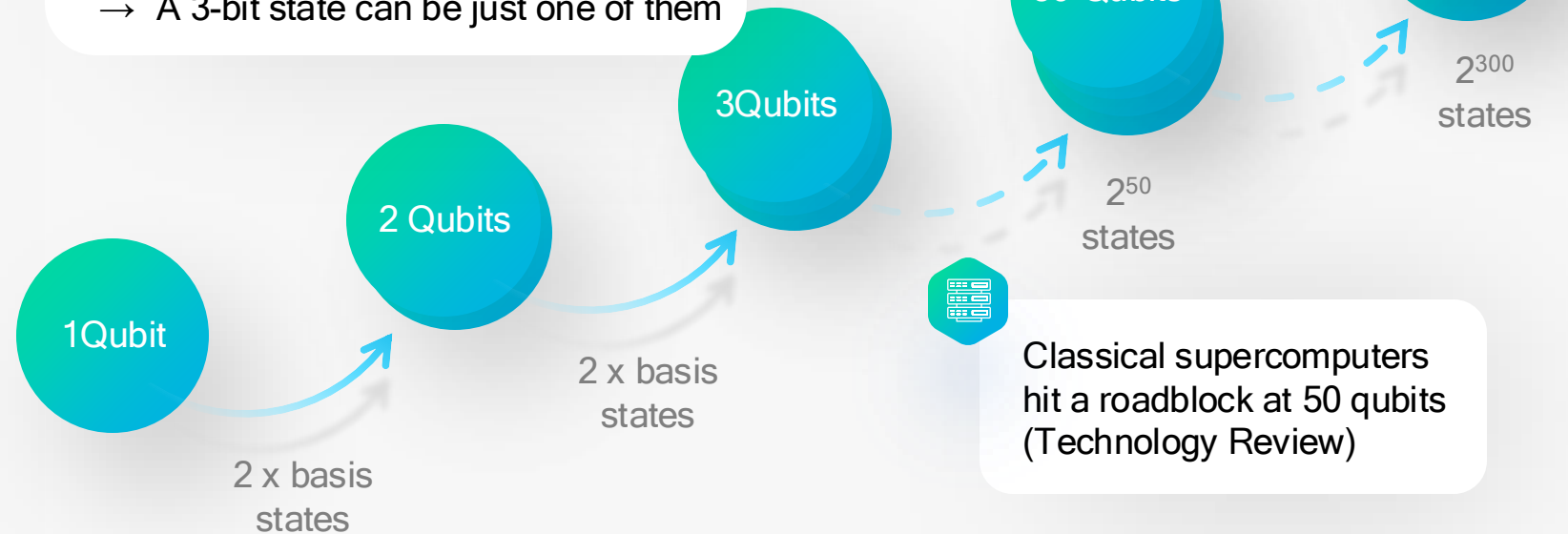
Two qubits can be **entangled**. Changing one directly impacts the other.



000	001	010	011
100	101	110	111

→ 3 qubits can be in a superposition of all $2^3 = 8$ basis states

→ A 3-bit state can be just one of them

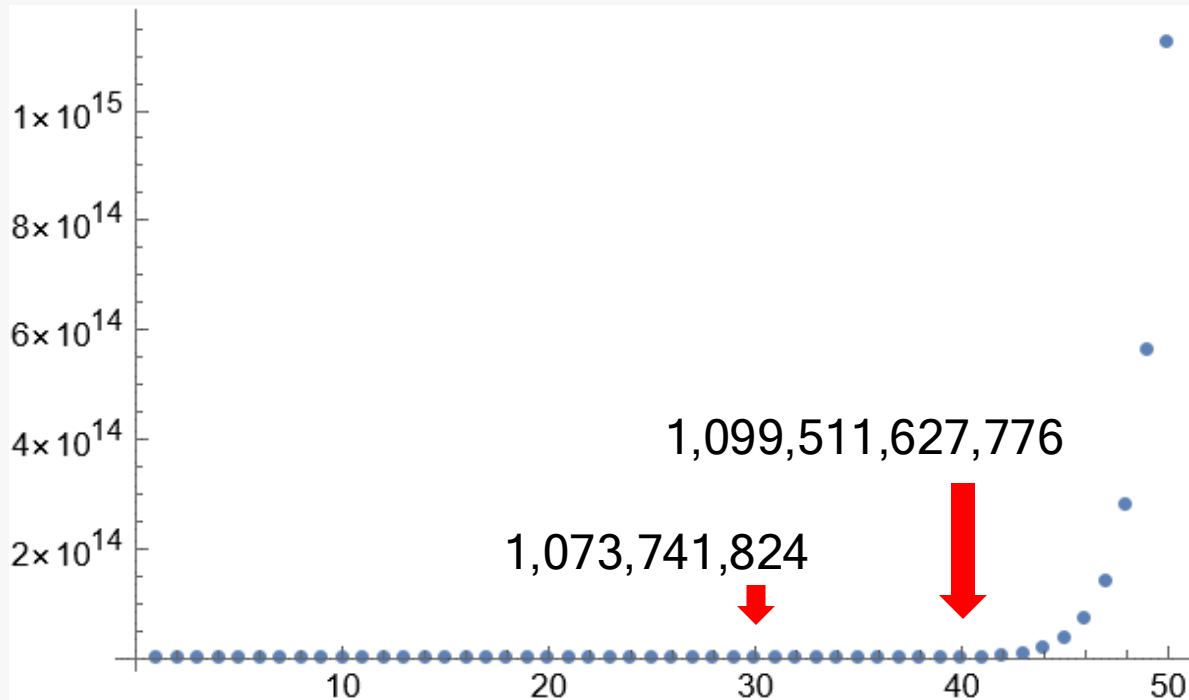


→ Exponential increase in computational capacity

→ Enables new algorithms and solutions to previously intractable problems

“Quantum Parallelism”

- Bit is replaced by **qubit** in quantum computing.
 n qubits represent up to 2^n different states
simultaneously.


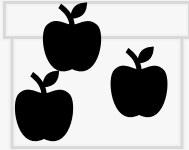

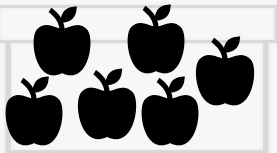






n	2^n
1	2
10	1,024
20	1,048,576
30	1,073,741,824
40	1,099,511,627,776
50	1,125,899,906,842,624 $\sim 10^{15}$
60	$\sim 10^{18}$
70	$\sim 10^{21}$
80	$\sim 10^{24}$
90	$\sim 10^{27}$
100	$\sim 10^{30}$
1,000	$\sim 10^{301}$

Eddington number $\sim 10^{80}$. The number of atoms in the visible universe.


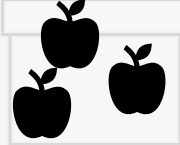

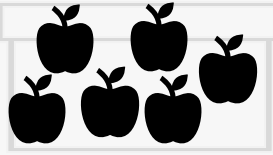




Computational Complexity

A suitable measure for complexity

	1 Person	3 Person
Sliced apple (space)		
Apple compote (space)		
Sliced apple (time)	 1 min	 3 min
Apple compote (time)	 2 min	 2 min

- In general, the effort required in terms of memory or time can be determined
- Time is often the limiting factor

A suitable measure for complexity

	1 Person	3 Person
Sliced apple (space)		
Apple compote (space)		
Sliced apple (time) $T(n) = 3n \in O(n)$	 1 min	 3 min
Apple compote (time) $T(n) = 1 \in O(1)$	 2 min	 2 min

IQM

- In general, the effort required in terms of memory or time can be determined
- Time is often the limiting factor
- The complexity is specified depending on the input size, but independently of the specific hardware
 - $T(n) = (n^2)! \cdot 4^{n^2}$
- The O notation $O(\dots)$ is widely used
 - $T(n) \in O((n^2)! \cdot 4^{n^2})$
- In most cases, it is not the exact effort that is relevant, but the order of magnitude
 - $O(5n + 1) = O(n)$
 - $O(2n^2 + n) = O(n^2)$

— Complexity of algorithms

Notation	Type	Example
$O(1)$	constant	Add an item to the end of a list
$O(n)$	Linear	Finding an item in an unsorted list
$O(n^c)$	Polynomial	Bubble Sort
$O(c^n)$	Exponential	Traveling Salesman Problem, using dynamic programming
$O(n!)$	Factorial	Traveling Salesman Problem, using brute force

Complexity of algorithms

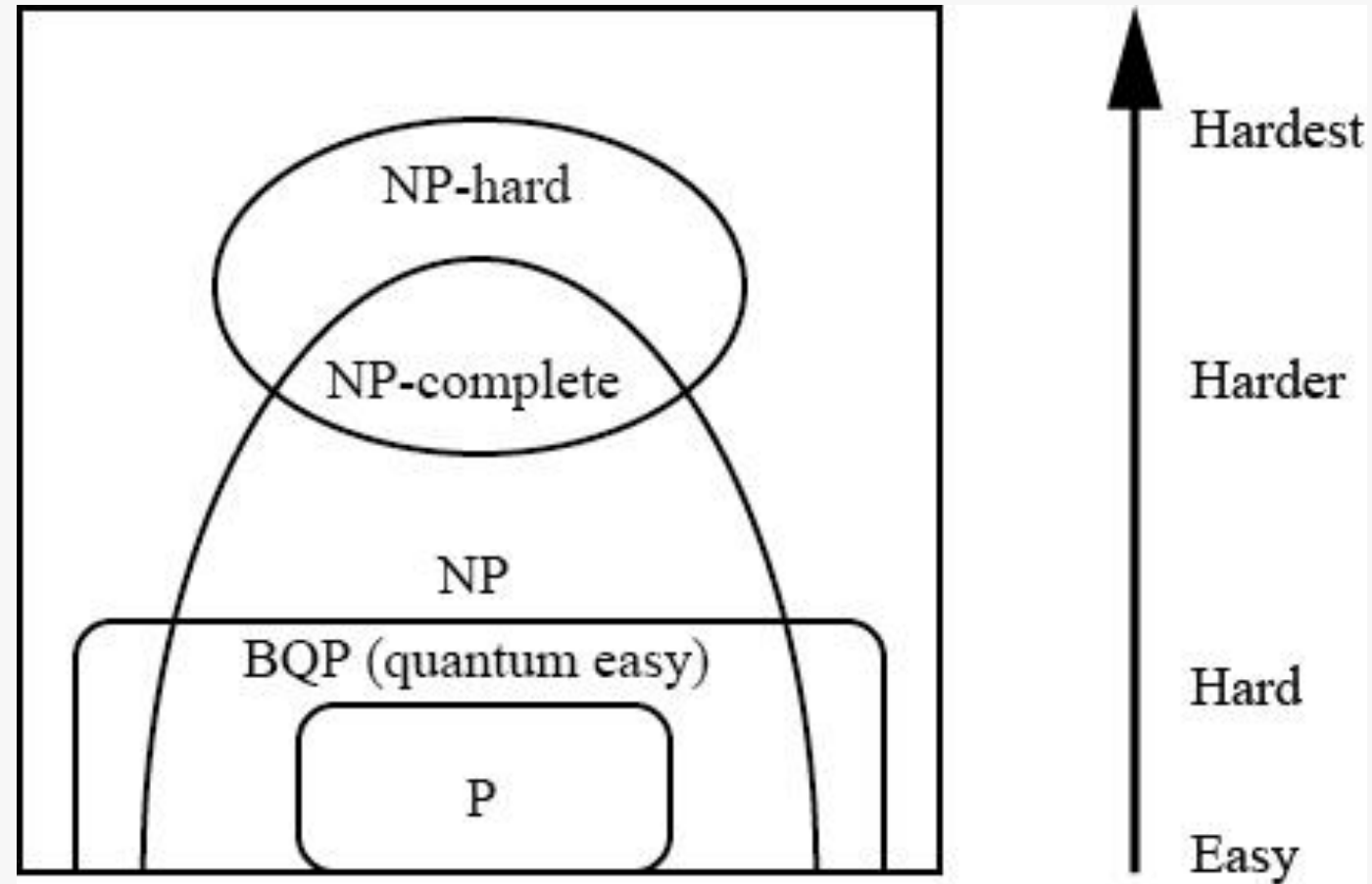
	n	n^2	2^n	$n!$
$n=10$	< 1s	< 1s	< 1s	4s
$n=30$	< 1s	< 1s	18 m	10^{25} y
$n=50$	< 1s	< 1s	36 y	Very long
$n=100$	< 1s	< 1s	10^{17} y	Very long
$n=1000$	< 1s	1s	Very long	Very long
$n=10000$	< 1s	2 m	Very long	Very long
$n=100000$	< 1s	3 h	Very long	Very long
$n=1000000$	1s	12 d	Very long	Very long
0				

Assuming one Million Instructions per second (MIPS)

The power of one [Motorola 68000](#) (current processors have 100 000 MIPS)

Computational Complexity

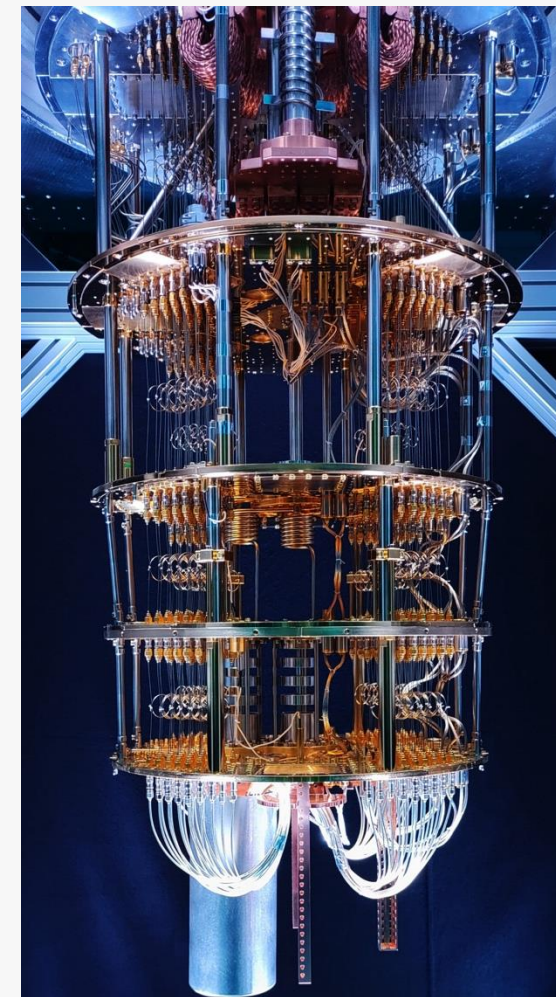
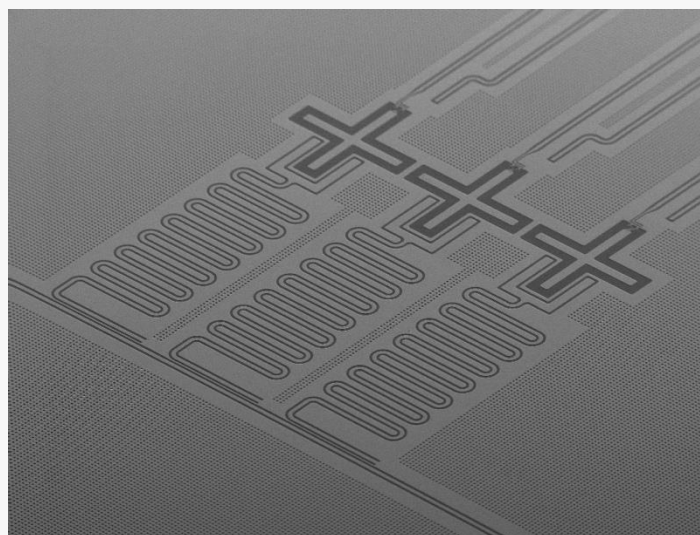
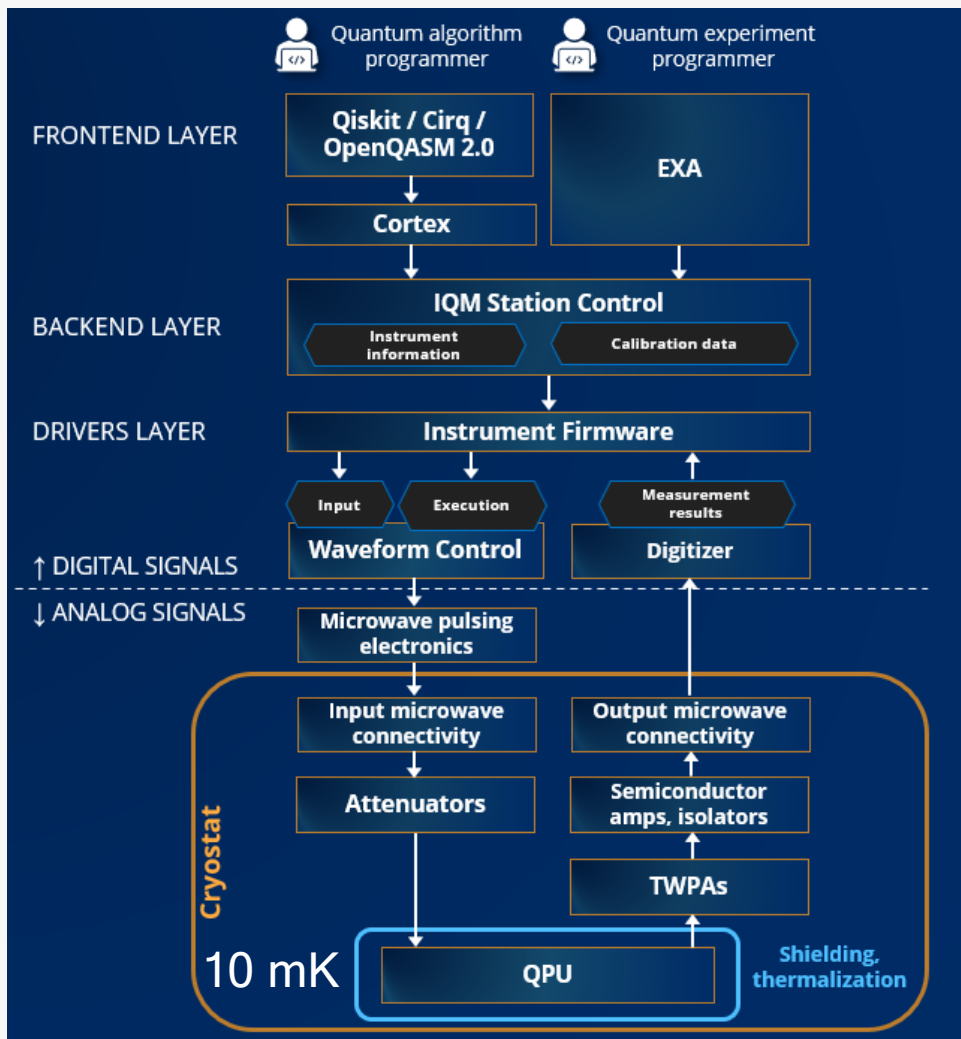
- **P**: Solved in polynomial time.
- **NP**: Verified in polynomial time.
- **NP-complete**: Most difficult in NP.
- **NP-hard**: At least as difficult as **NP-complete**. (Find the best solution among the NP-complete solutions, for example.)
- **BQP** (Bounded error, Quantum, Polynomial time): Solvable in polynomial time by a quantum computer.



— **Caution: Not All Algorithms Can Be Improved**

- A physical system cannot be an **all-purpose** computer (pendulum/spaghetti).
- A quantum computer makes use of **wave nature** of quantum states. There are **interference, superposition and entanglement** associated with this. They do not exist in a **classical** (i.e., digital) computer.
- Typical quantum algorithm; **Grover's database search, Shor's factorization**. They require a full-fledged quantum computer with built-in **quantum error correction (QECC)**.
- Currently available quantum computers, including the IQM quantum computer, are called the **NISQ (Noisy Intermediate-Scale Quantum)** computer. <1000 QB, no QECC. They are often used for the **quantum-classical hybrid computation** such as the **Variational Quantum Eigensolver (VQE)**.

Brief Summary of IQM Quantum Computer



Summary

Summary

- **Physical system** may be employed for information processing and computation.
- **Quantum computing** employs the law of **quantum physics** to store, process and transfer information.
- Quantum physics describes a **microscopic system** such as an electron, an atom and a photon. These systems are **hard** to control and measure.
- Thanks to the progress of nanotechnology, we can fabricate **nanoscale devices** that follow the law of quantum mechanics.
- **Superconducting qubit** in IQM quantum computer is an example of such systems.
- Currently available quantum computer is called **NISQ** (**Noisy Intermediate-Scale Quantum**) computer. No QECC (**N**). <1000 qubits (**IS**).
- We will see how quantum information processing is different from and superior to classical one in the following lectures.

Practical notes

Outline of the course

Lecture	Title	Presentation	Exercises
1	Introduction to Quantum Computing	PPTX PDF	LaTeX PDF
2	Quantum States and Quantum Operations	PPTX PDF	LaTeX PDF
3	Introduction to Quantum Algorithms	PPTX PDF	LaTeX PDF
4	Transpiling, compiling and optimizing quantum circuits	PPTX PDF	LaTeX PDF
5	Quantum Algorithms: Grover's search	PPTX PDF	LaTeX PDF
6	Variational quantum algorithms	PPTX PDF	LaTeX PDF
7	Error Reduction Strategies	PPTX PDF	LaTeX PDF
8	Benchmarking quantum computers	PPTX PDF	LaTeX PDF
9	Quantum Algorithms: Shor's Algorithm	PPTX PDF	LaTeX PDF
10	Data Encoding and Quantum Machine Learning	PPTX PDF	LaTeX PDF
11	Quantum Hardware and Architectures	PPTX PDF	LaTeX PDF
12	Quantum Error Correction and the Future of Quantum Computing	PPTX PDF	LaTeX PDF

— This course ...

- ... is intended to be an introduction to the computing (not the physics aspects) of a quantum computer
- ... assumes you get actively engage with the lectures and the exercises
- ... uses qrisp as the SDK for developing quantum algorithms
- ... assumes access to quantum computers via IQM Resonance (freemium available) / IQM Server (on-prem users)
- ... is supported by the material and applets on IQM Academy

— qrisp



- qrisp is a high level programming framework for working with quantum computing programs
- It's pythonic
- It's open source

IQM



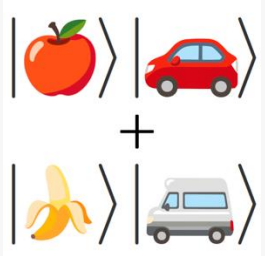
qrisp.eu



<https://github.com/eclipse-qrisp/Qrisp>

`pip install „qrisp[iqm]“`

Features of qrisp (you will learn about these throughout the course)



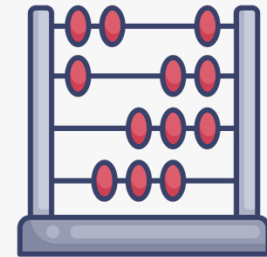
Typed quantum variables



automatic Uncomputation,
i.e. Garbage Collection



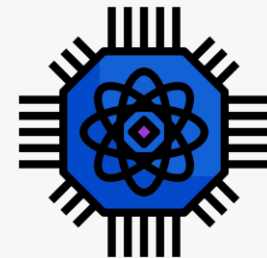
Modularity



Arithmetics



Compatibility and
interoperability



Works with IQM hardware

There are different quantum computing frameworks, but qrisp will enable you to learn what matters



```
from qrisp import QuantumFloat
n = 6
a = QuantumFloat(n)
b = QuantumFloat(n)
a[:] = 3
b[:] = 4
res = a*b
print(res)
#Yields: {12: 1.0}
```

```
from qiskit import (QuantumCircuit, QuantumRegister,
                    ClassicalRegister, Aer, execute)
from qiskit.circuit.library import RGQFTMultiplier
n = 6
a = QuantumRegister(n)
b = QuantumRegister(n)
res = QuantumRegister(2*n)
cl_res = ClassicalRegister(2*n)
qc = QuantumCircuit(a, b, res, cl_res)
for i in range(len(a)):
    if 3 & 1<<i: qc.x(a[i])
for i in range(len(b)):
    if 4 & 1<<i: qc.x(b[i])
qc.append(RGQFTMultiplier(n, 2*n),
          list(a) + list(b) + list(res))
qc.measure(res, cl_res)
backend = Aer.get_backend('qasm_simulator')
counts_dic = execute(qc, backend).result().get_counts()
print({int(k, 2) : v for k, v in counts_dic.items()})
#Yields: {12: 1024}
```

```
import pennylane as qml
import numpy as np
w_m = [0, 1, 2]
w_k = [3, 4, 5]
w_sol = [6, 7, 8, 9]
dev = qml.device("default.qubit", wires=w_m + w_k + w_sol, shots=1)
n_wires = len(dev.wires)
def add(k, wires):
    for j in range(len(wires)):
        qml.RZ(k * np.pi / (2**j), wires=wires[j])
def multiplication(w_m, w_k, w_sol):
    qml.QFT(wires=w_sol)
    for i in range(len(w_k)):
        for j in range(len(w_m)):
            coeff = 2 ** (len(w_m) + len(w_k) - i - j - 2)
            qml.ctrl(add, control=[w_k[i], w_m[j]])(coeff, w_sol)
    qml.adjoint(qml.QFT)(wires=w_sol)
@qml.qnode(dev)
def mul(m, k):
    qml.BasisEmbedding(m, wires=w_m)
    qml.BasisEmbedding(k, wires=w_k)
    multiplication(w_m, w_k, w_sol)
    return qml.sample(wires=w_sol)
print(f"The ket representation of 3*7 is {mul(3,4)}")
#Yields: The ket representation of 3*7 is [10101]
```

IQM Resonance / IQM Server

- A platform to run and organize your quantum circuits.
- Made for education and research.
- Support for multiple frameworks (qrisp, qiskit, Cirq and more)

