



浙江大學  
ZHEJIANG UNIVERSITY

# HPCA 2025 Tutorial

## Topic 1. Introduction of Janus Quantum Cloud Platform and Installing JanusQ



Speaker: Liqiang Lu

College of Computer Science and Technology  
Zhejiang University (ZJU)

[https://janusq.github.io/HPCA\\_2025\\_Tutorial/](https://janusq.github.io/HPCA_2025_Tutorial/)

# Presenter

---



**Tianyao Chu** is a first-year PhD student at the College of Computer Science, Zhejiang University. He is interested in distributed quantum computing and quantum network system. He is currently working on designing a quantum network on chip based on enhanced QST.

Tianyao Chu

[tianyao\\_chu@zju.edu.cn](mailto:tianyao_chu@zju.edu.cn)

# Presenter

---



**Kaiwen Zhou** is a first-year PhD student at the College of Computer Science, Zhejiang University. He is interested in quantum computer architecture and high-performance computing. He is currently working on designing a QLDPC decoding accelerator.

Kaiwen Zhou

[@zju.edu.cn](mailto:@zju.edu.cn)

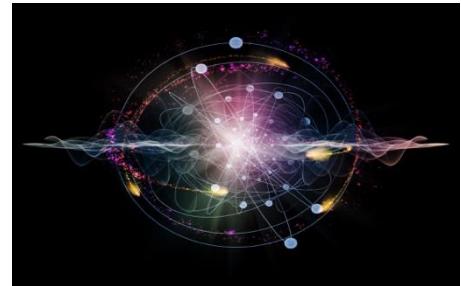


浙江大學  
ZHEJIANG UNIVERSITY

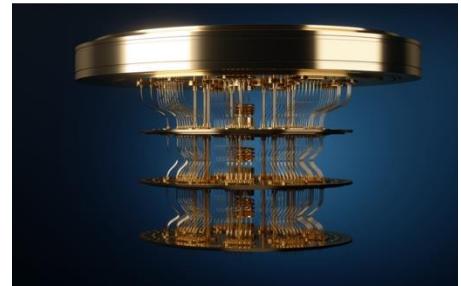
# Background Knowledge



Development of  
Classical Computing



Motivation of  
Quantum Computing



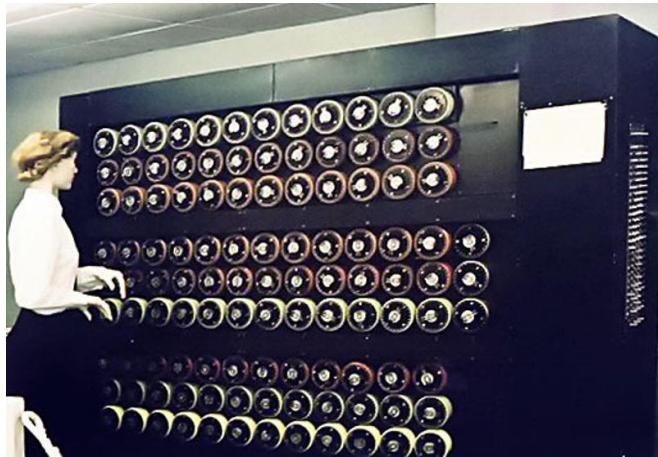
Development of  
Quantum Computing

# Development Of Classical Computing

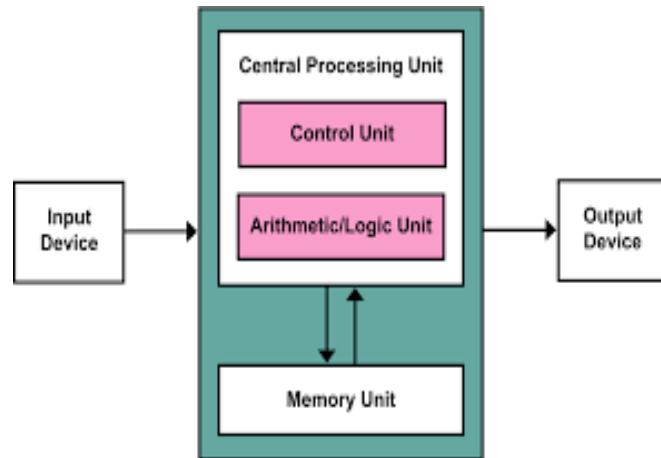


浙江大學  
ZHEJIANG UNIVERSITY

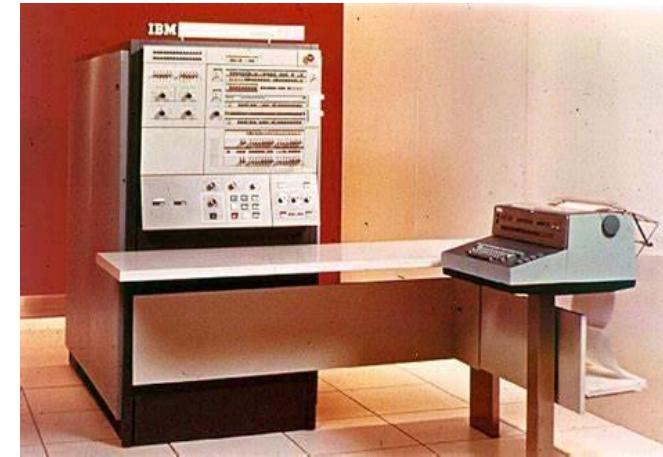
Domain-specific calculator (1939)



Von Neumann architecture (1947)



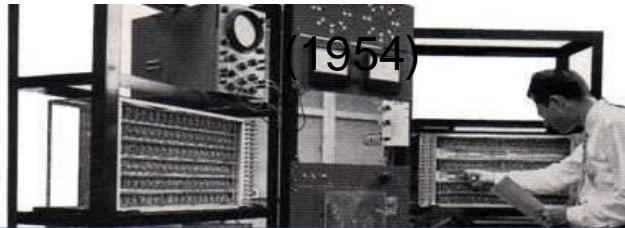
IBM360 Integrated Circuit (1964)



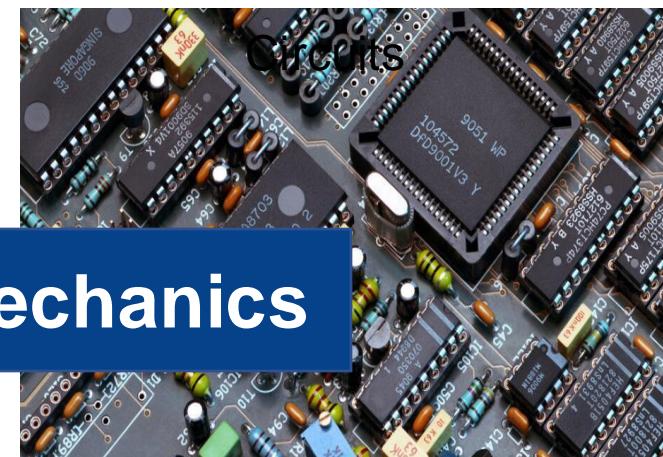
Vacuum Tube Computer ENIAC (1942)



Transistor Computer TRADIC



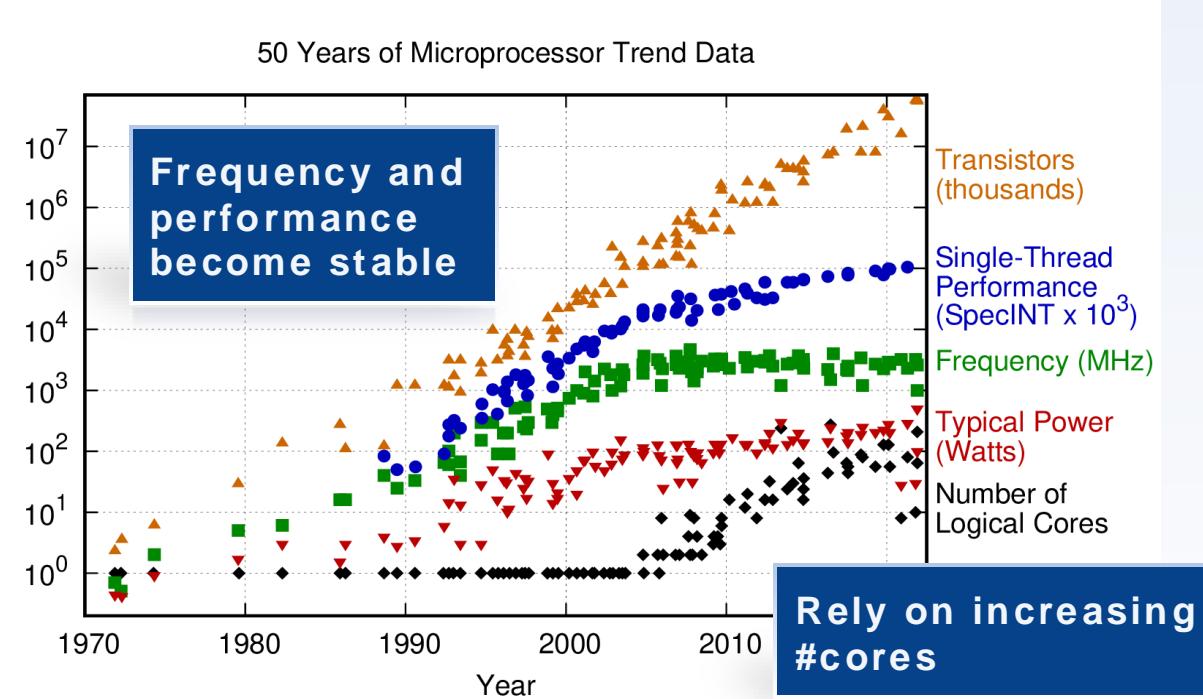
Large Scale Integrated Circuits



Macroscopic Effects of Quantum Mechanics

# Motivation Of Quantum Computing

## Computation barrier

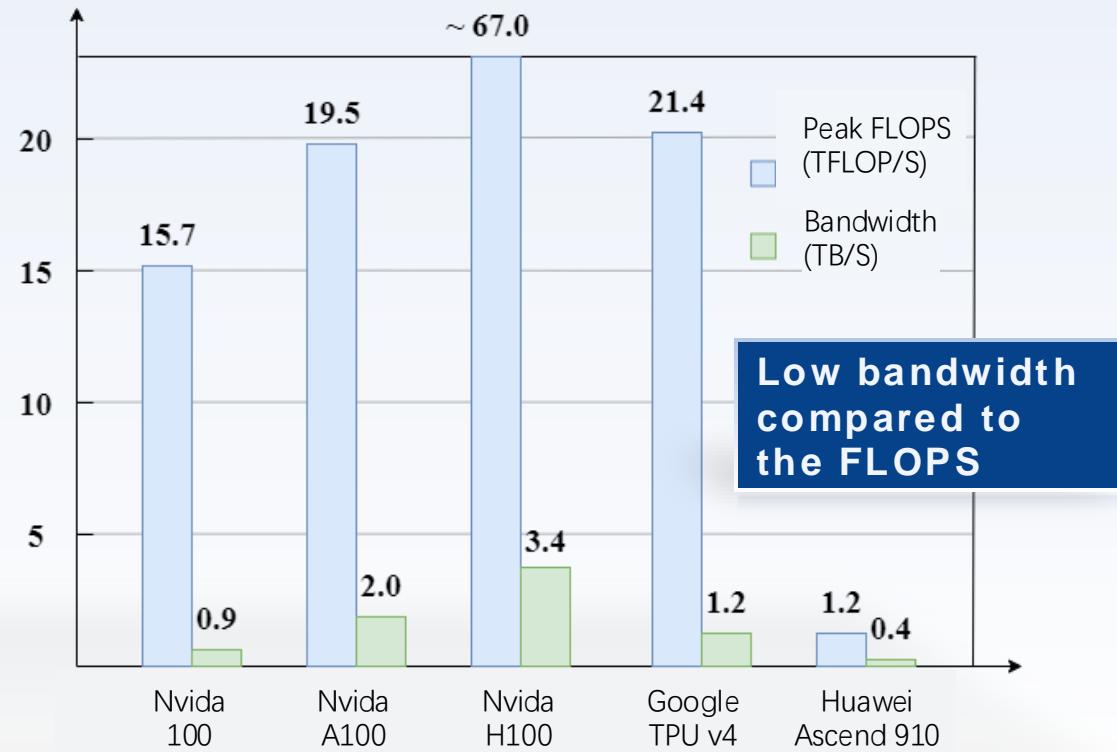


Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonne, O. Shacham, K. Olukolu, L. Hammond, and C. Balen. New plot and data collected for 2010-2021 by K. Rupp.

- The research costs and cycles of advanced chip processes are continuously increasing, Moore's Law is approaching obsolescence.
- The computing systems cannot rely solely on the development of traditional single chips. Instead, it requires new chip design methods and computing principles.

# Motivation Of Quantum Computing

## Storage barrier



- Computation power and bandwidth is not matched.
- Latency of memory access is high, limiting CPU performance.
- Quantum computing is in memory.
- Non-von Neumann architectures.

## Power wall

Thermal design power exponentially increases.



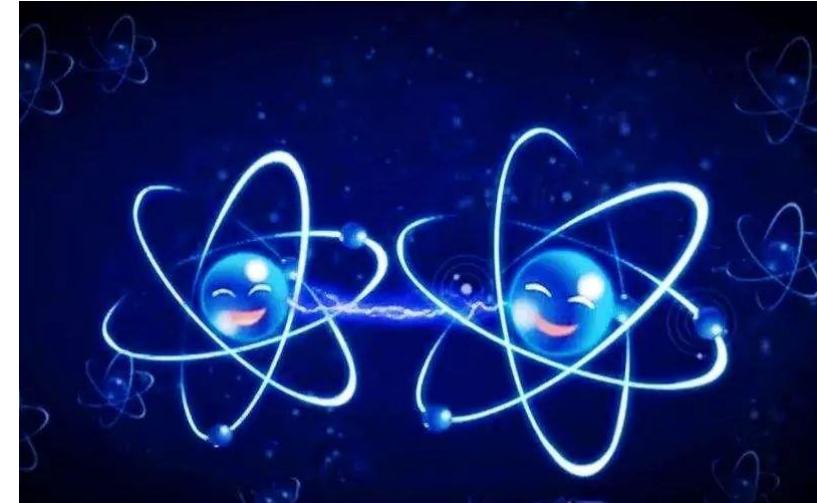
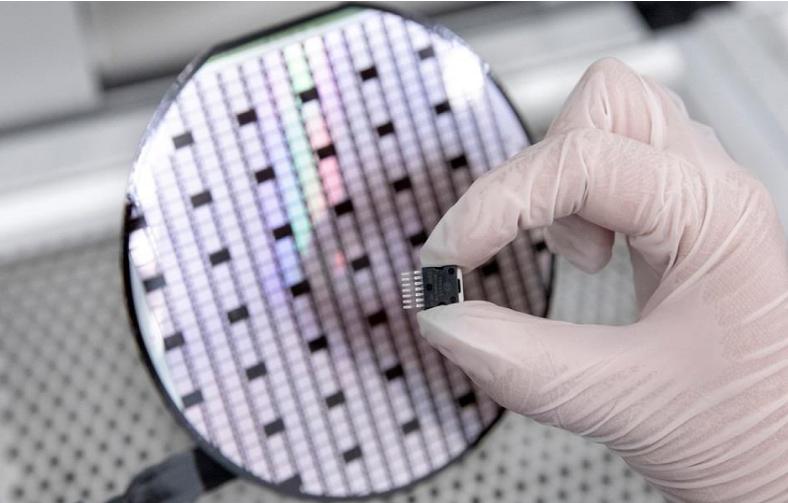
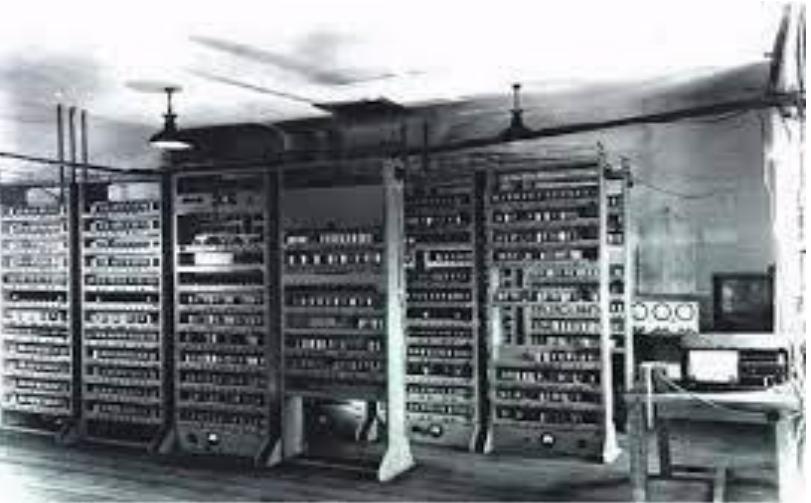
From <https://www.blueocean-china.net/faq3/241.html>

- The power consumption **increases exponentially with computing power.**
- Energy consumption **restricts computing power.**
- Systematical resource scheduling at the architectural level.

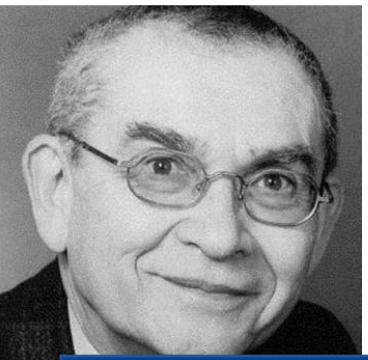
# Proposal of Quantum Computing



浙江大學  
ZHEJIANG UNIVERSITY



- Classical physics is no longer able to fully describe the underlying physical mechanisms at its core.



**Yuri Manin**

- Algebraic Geometry
- Discrete Geometry (Diophantine Geometry)
- Manin (1980) and Feynman (1982) proposed the first quantum computation algorithm

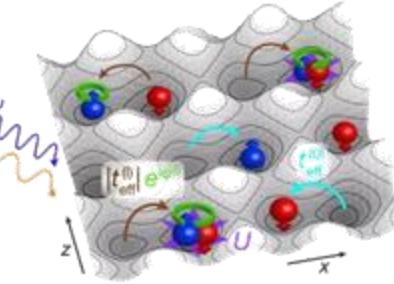


**Richard Feynman**

- Feynman Path Integral, Feynman Diagram, Feynman Parton Model
- Quantum Electrodynamics, Nobel Prize in Physics

generalize to other domains such as the database search and cryptography

# Basic Concepts of Quantum Computing



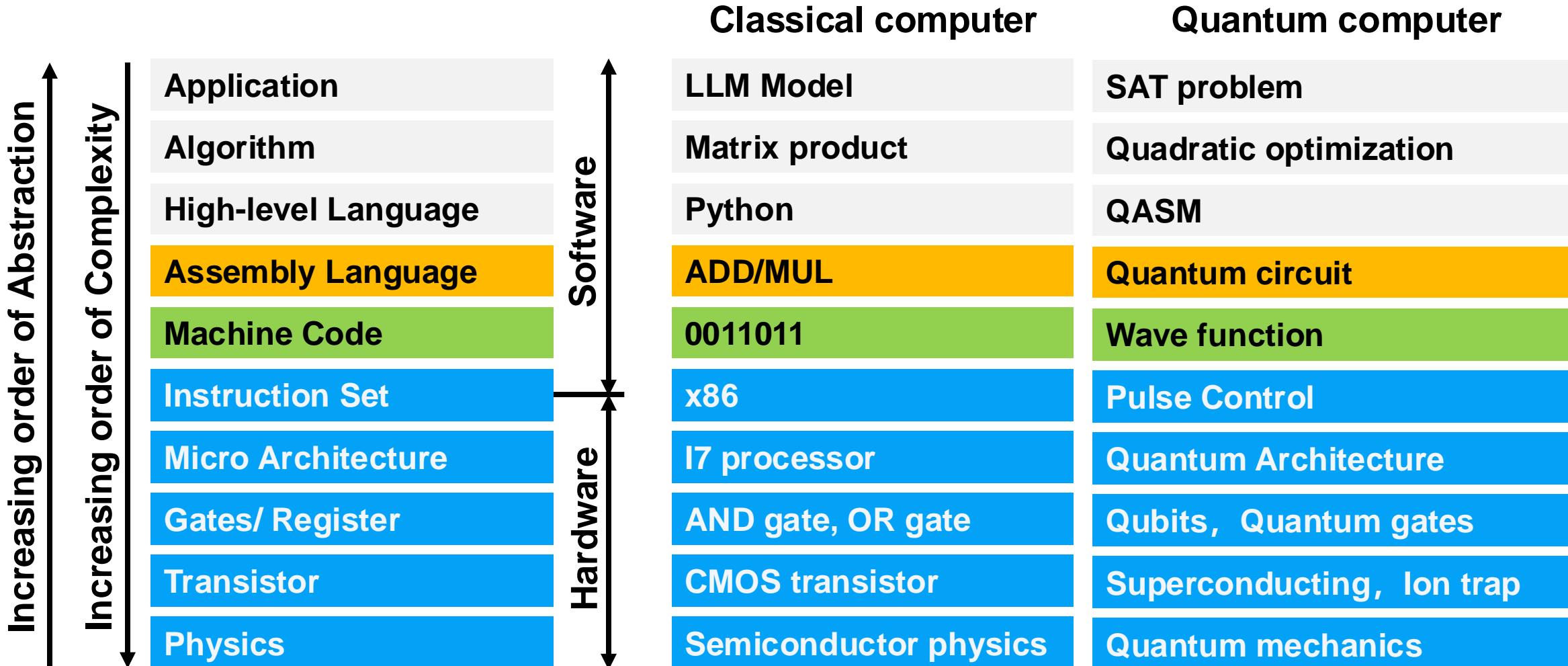
Physical  
Simulation



Factor  
Application

		Computing theory	Encoding	Physical implementation
Physical Simulation	Classical program	Classical Turing machine	Program based on binary encoding	CMOS (0/1 )
	Quantum program	Quantum Turing machine	Circuit-based quantum program	Superconducting, photon (superposition state)
		<p>lead to high computational advantages via parallelism.</p>	<p>suffer from complexity due to quantum collapse and entanglement</p>	<p>has low energy consumption due to reversibility and superconductivity.</p>

# Quantum Computing Architecture

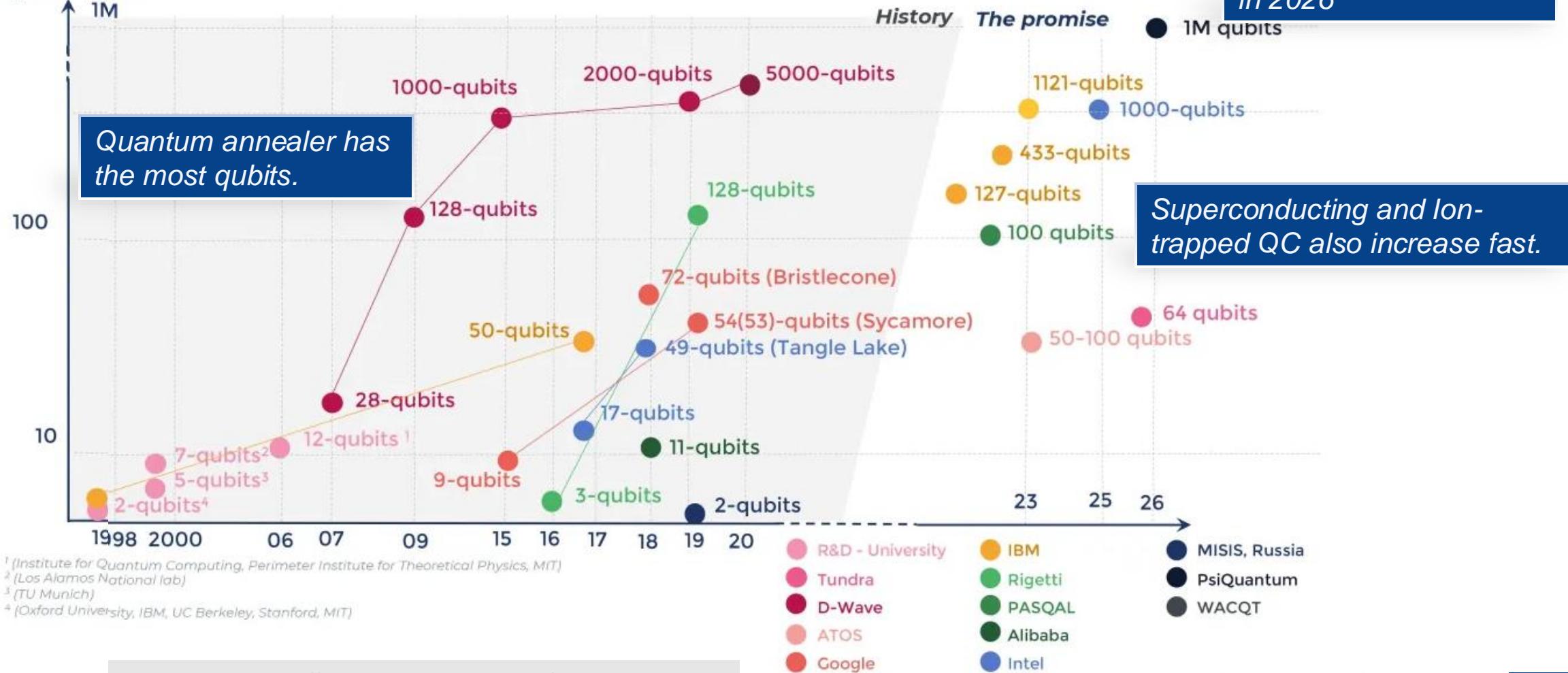


From  
<https://www.secplcity.org/2018/09/19/understanding-the-layers-of-a-computer-system/>

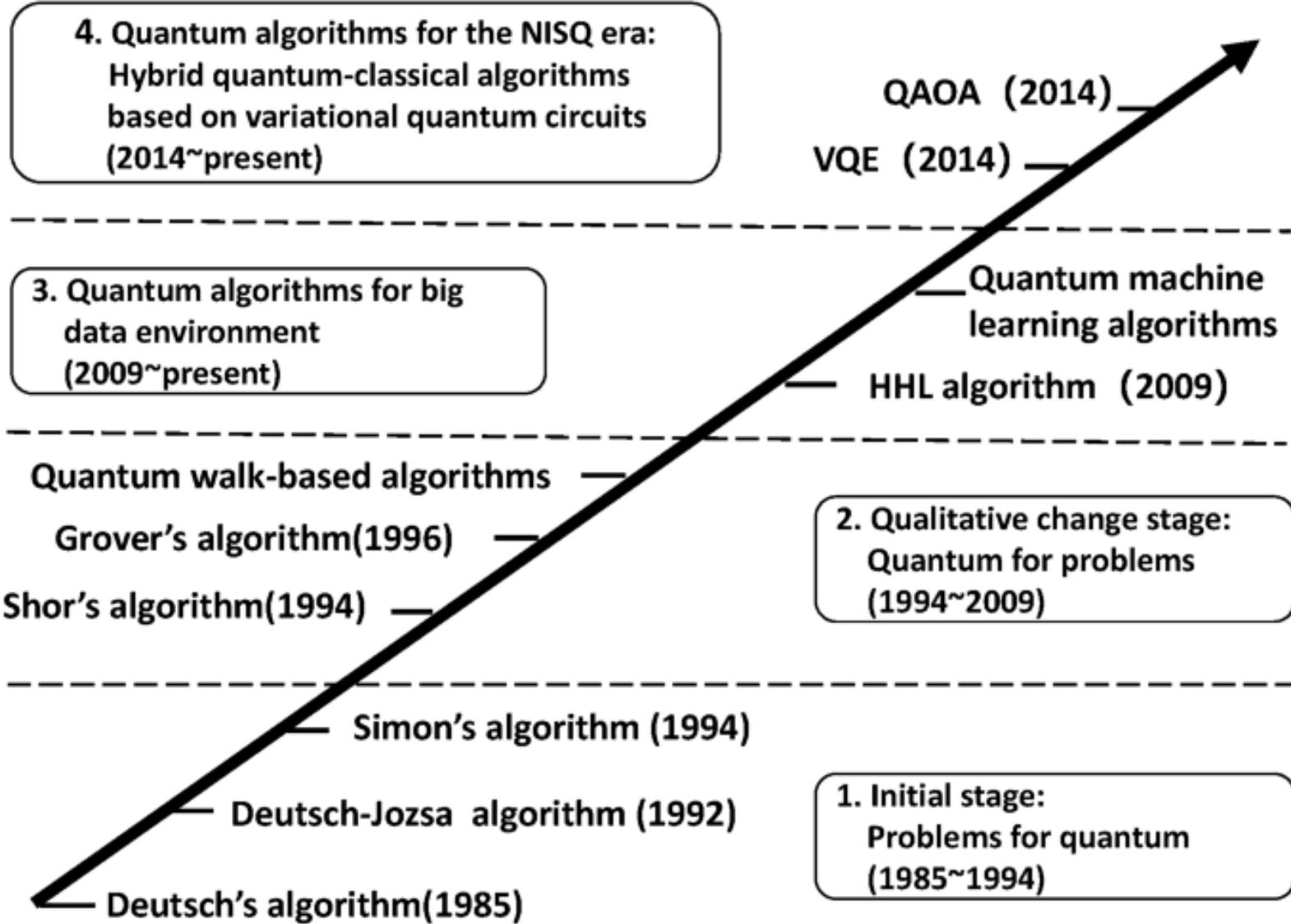
# Development of Quantum Computer



Graph below shows physical qubit roadmap (Note: for a quantum computer, 50 logical qubits minimum are required → it means 50 000 physical qubits)  
Physical qubits



# Development of Quantum Algorithm

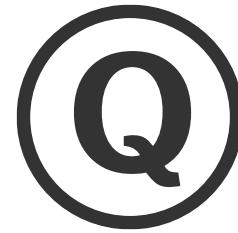


Zhang, S., Li, L. A brief introduction to quantum algorithms. *CCF Trans. HPC* 4, 2022



浙江大學  
ZHEJIANG UNIVERSITY

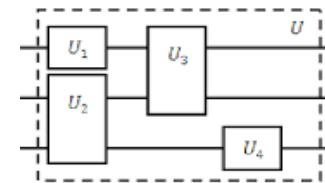
# Mathematical Model of Quantum Computing



Qubits



Quantum Evolution



Quantum Circuit

# Quantum Bit (Qubit)



## Single qubit

A **qubit** has two bases  $|0\rangle$  and  $|1\rangle$ . The information stored in its **superposition state** is represented as a **2-dimension state vector**  $|\varphi\rangle$ .

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

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

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

$$|\varphi\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

$$\text{subject to } |\alpha|^2 + |\beta|^2 = 1$$

# Quantum Bit (Qubit)



## Single qubit

A **qubit** has two bases  $|0\rangle$  and  $|1\rangle$ . The information stored in its **superposition state** is represented as a **2-dimension state vector**  $|\varphi\rangle$ .

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

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

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

$$|\varphi\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

$$\text{subject to } |\alpha|^2 + |\beta|^2 = 1$$

## Multiple qubits

**$N$  qubits** has  $2^N$  bases  $|00\cdots 0\rangle$ ,  $|00\cdots 1\rangle\cdots$ ,  $|11\cdots 1\rangle$ . The information stored in their **superposition state** is represented as a  **$2^N$ -dimension state vector**.

$$|00\cdots 0\rangle = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad |00\cdots 1\rangle = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \quad \cdots \quad |11\cdots 1\rangle = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

$$|\varphi\rangle = \alpha_0|00\cdots 0\rangle + \alpha_1|00\cdots 1\rangle + \cdots + \alpha_{2^N}|11\cdots 1\rangle$$

$$|\varphi\rangle = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{2^N} \end{bmatrix}$$

$$\text{subject to } |\alpha_0|^2 + |\alpha_1|^2 + \cdots + |\alpha_{2^N}|^2 = 1$$

## Unitary matrix

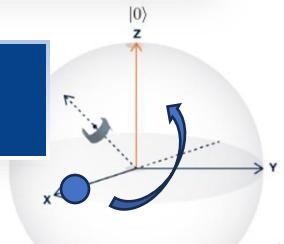
A quantum evolution caused by **quantum gates** is represented as a **unitary matrix (unitary)**, which is a square matrix whose conjugate transpose is its inverse.

$$UU^\dagger = I$$

The evolution of qubit state  $|\varphi\rangle$  is represented as:

Bloch sphere

X+Z axes

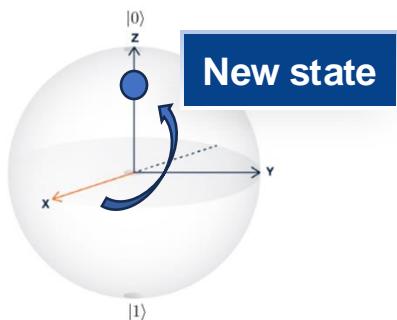


State

$$|\varphi'\rangle = U|\varphi\rangle$$

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$\xrightarrow{\pi \text{ rotation around X+Z axes}}$



New state

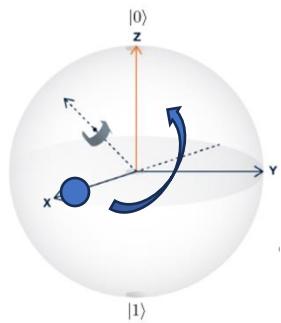
## Unitary matrix

A quantum evolution caused by **quantum gates** is represented as a **unitary matrix (unitary)**, which is a square matrix whose conjugate transpose is its inverse.

$$UU^\dagger = I$$

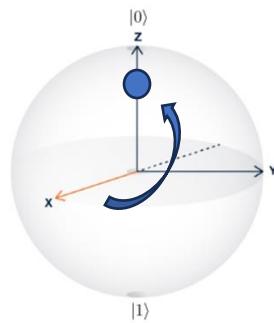
The evolution of qubit state  $|\varphi\rangle$  is represented as:

$$|\varphi'\rangle = U|\varphi\rangle$$



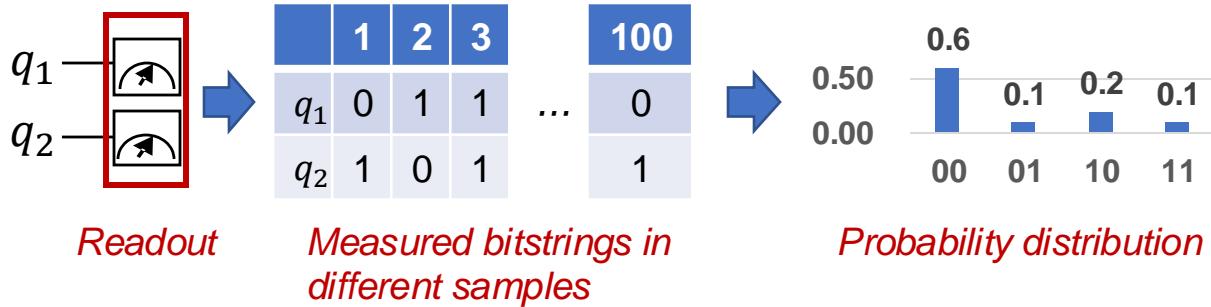
$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$\pi$  rotation around  
X+Z axis:  
Exchanges X and  
Z



## Quantum readout

A sampling of a quantum state is a bitstring. Multiple sampling of this state composes a probability distribution of measuring different bitstrings.



# Quantum Circuit



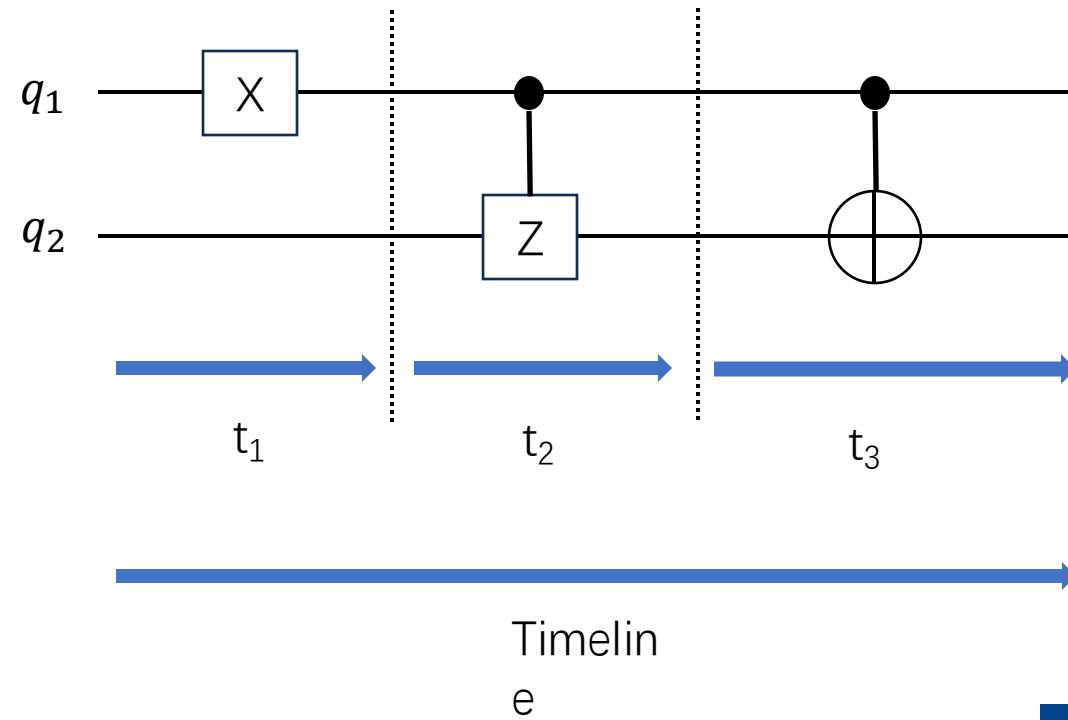
## Quantum gates

In the quantum circuit model of computation, a **quantum gate** is a **basic quantum circuit operating** on a small number of qubits.

Operator	Gate(s)	Matrix
Pauli-X (X)		$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Y (Y)		$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
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)		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$
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)		$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

## Qubit timeline

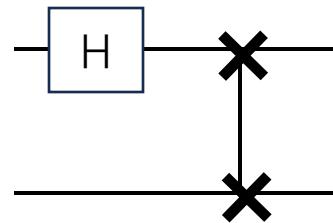
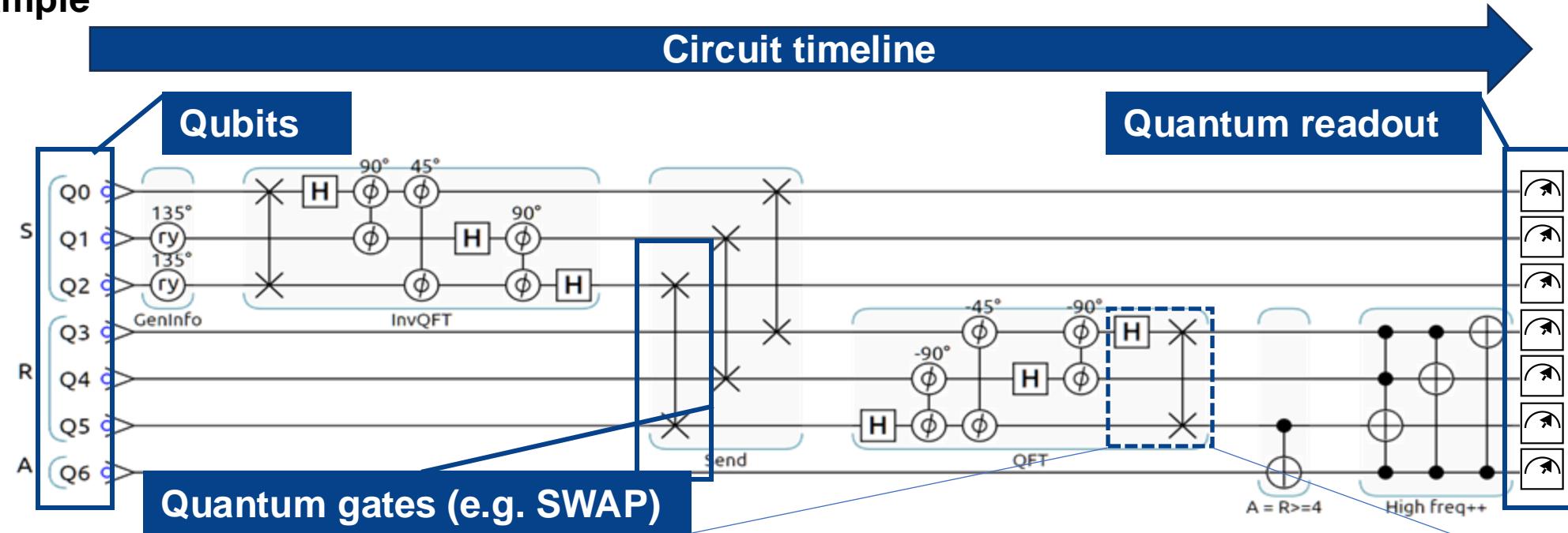
Each line in a quantum circuit represents a **qubit**. The quantum gate in a line is applied on the same qubits **from left to right in time direction**.



# Quantum Circuit



For example



=

$$SWAP \cdot (H \otimes I) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \end{bmatrix}$$

# Implementation of Quantum Circuit

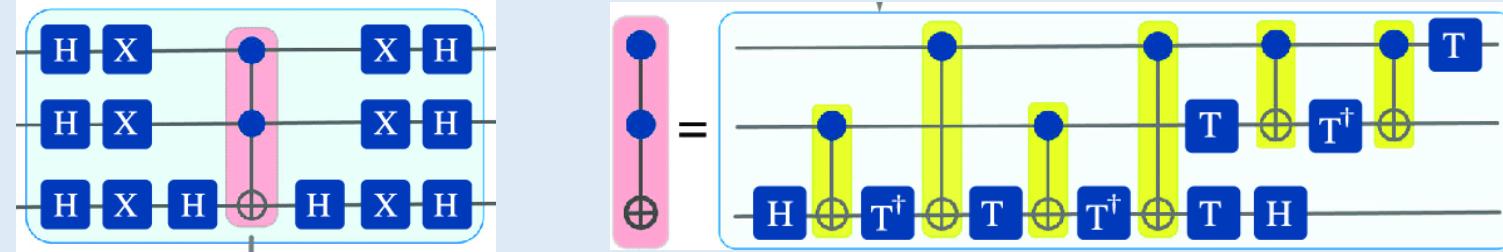


## On superconducting quantum computer

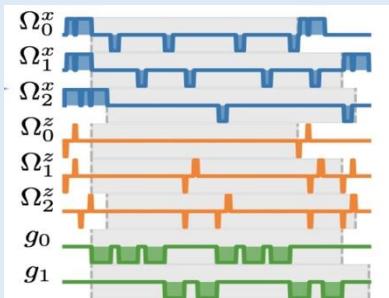
### Step 1. Circuit statement

```
OPENQASM 2.0;  
qreg qubits[3];  
H qubits[1];H qubits[2];  
H qubits[3];  
X qubits[1];X qubits[2];  
X qubits[3];  
H qubits[3];  
Toffoli qubits[1], qubits[2], qubits[3];  
H qubits[3];  
X qubits[1];X qubits[2];  
X qubits[3];  
H qubits[1];H qubits[2];  
H qubits[3];
```

### Step 2. Circuit compilation



### Step 3. Circuit execution



Pulse generation

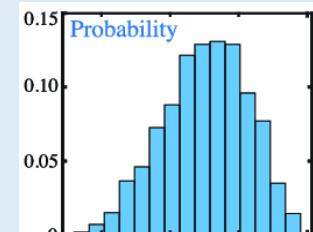


Quantum device

### Step 4. Result processing

Error emigration

Visualization



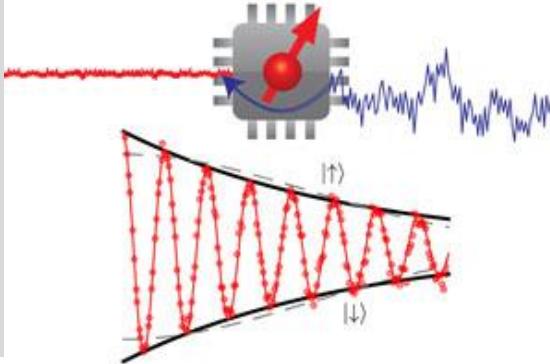
Probability distribution

# Challenge in Quantum Computing



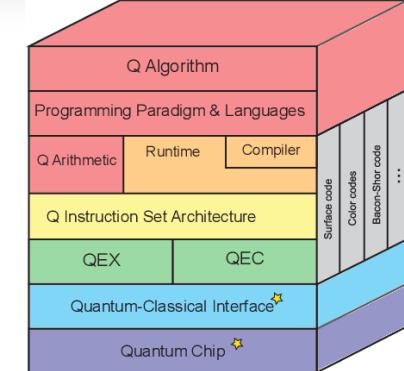
## Noise

- Coherence error
- Gate error
- State preparation error
- Readout error
- Crosstalk

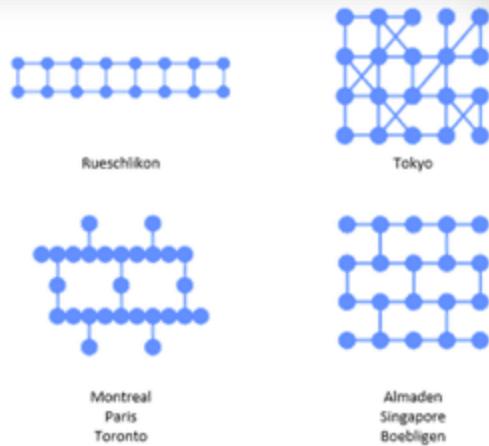


## Instructions

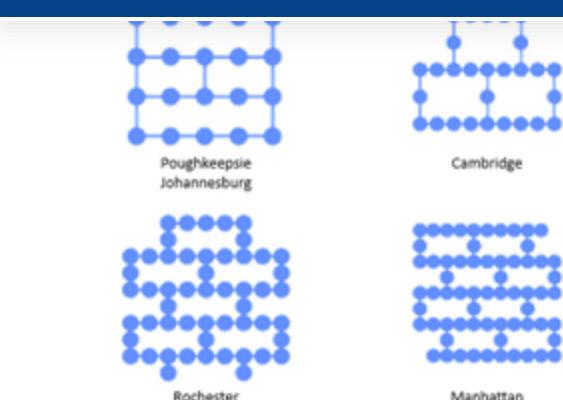
- Multi-level compilation
- Micro-architecture instructions
- Quantum-classical communication



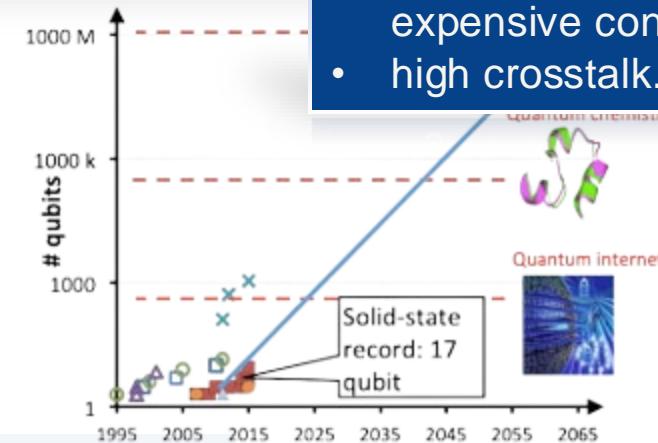
## Topology



## Locality in communication



## Scalability



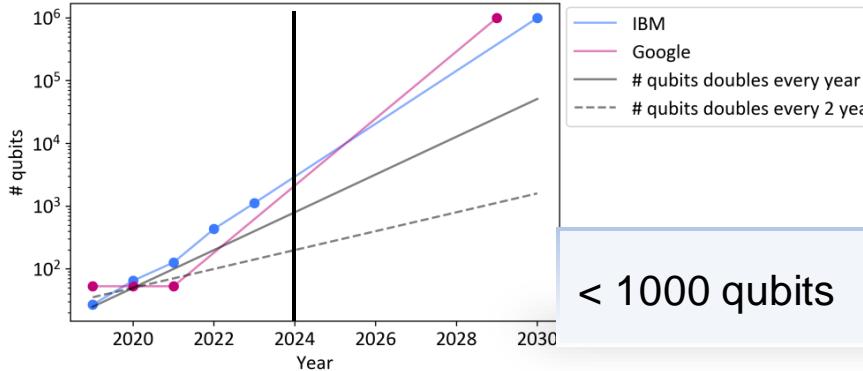
- poor fabrication techniques
- expensive control systems
- high crosstalk.

Constrained by physical implementation

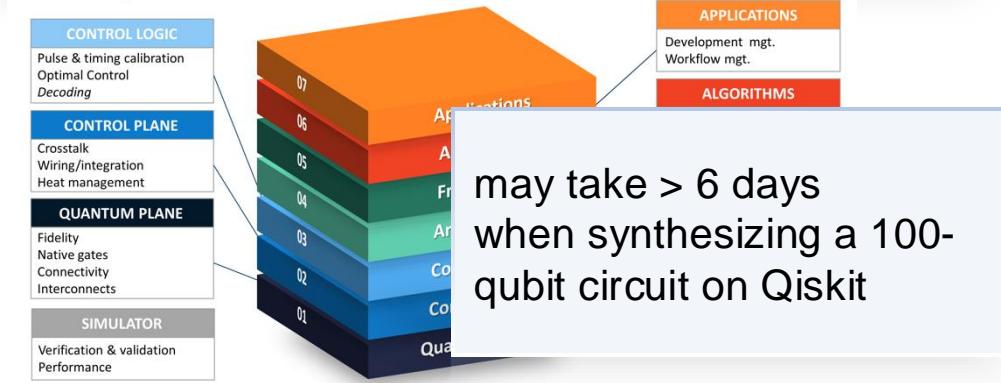
# Results of Challenges



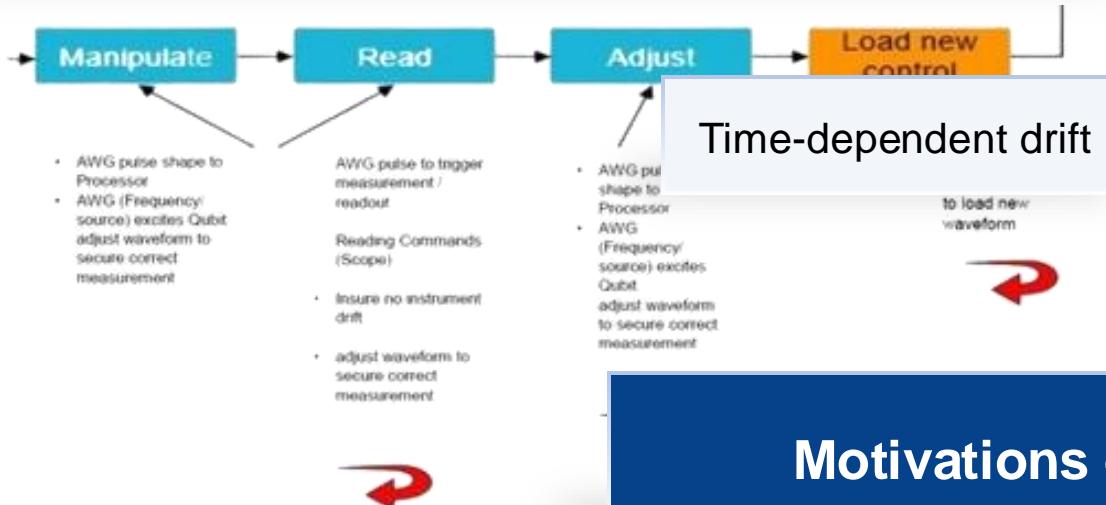
## Limited Hardware Resource



## Compilation Complexity



## Difficulty Of Hardware Calibration



## Rare quantum advantage

- Limited qubits resources
- High error rate
- Hard to ensure quantum advantage in real applications

Long calibration time (e.g., readout calibration)



## Motivations of JanusQ

# Janus (Taiyuan) Quantum Cloud Platform





**Tai Yuan** is a Chinese gold who is the wife of Pan Gu (the god that created the world). She was delivered of Yin and Yang, which is entangled like a quantum superposition state



**Janus** is a two-faced god of the ancient Romans. He is the god of beginning, transition, time, change, dualism, and end. He has two faces simultaneously.

# Development History of Janus Cloud



## Four updates since July 2023

Taiyuan Quantum  
Provide real-time quantum computing services  
Get Start

**July 2023**

1. Redesign the cloud interface.
2. Adding the support of English.

Document

**Quicode API**

**1. Circuit Commands**

Preparation work before performing quantum operations, applying for and initializing the quantum circuit.

`new(qubit_number, name)`

Apply for a variable for the quantum circuit, which will contain `qubit_number` quantum bits.

Parameters

`qubit_number` – the number of requested quantum bits.  
`name` – quantum circuit variable name.

Example

```
var a = new(3,'a')
```

Request a variable 'a' with 3 qubits.

`reset(qubit_number)`

Initialize `qubit_number` quantum bits to the |0⟩ state for the quantum circuit. This function should be called before adding a quantum gate.

**August 2023**

1. Update the document

Project List

serial number	project name	task number	creation time	operation
1	336	0	2023-11-09 16:11:46	<a href="#">edit item</a> <a href="#">view details</a> <a href="#">delete</a>
2	8899	0	2023-11-08 16:11:5	<a href="#">edit item</a> <a href="#">view details</a> <a href="#">delete</a>
3	test20231027	0	2023-10-27 17:10:8	<a href="#">edit item</a> <a href="#">view details</a> <a href="#">delete</a>
4	20230923	0	2023-09-23 20:09	<a href="#">edit item</a> <a href="#">view details</a> <a href="#">delete</a>
5	0923-1	0	2023-09-23 20:09	<a href="#">edit item</a> <a href="#">view details</a> <a href="#">delete</a>
6	23	0	2023-09-23 20:09	<a href="#">edit item</a> <a href="#">view details</a> <a href="#">delete</a>
7	通信	0	2023-09-16 09:09	<a href="#">edit item</a> <a href="#">view details</a> <a href="#">delete</a>

**September 2023**

1. Added multi-level project management.

Janus 2.0: A Software Framework for Analyzing Optimizing and Implementing Quantum Circuit

**Janus-SAT**: Solves Boolean satisfiability problems, including hybrid quantum-classical solvers and portfolio solvers for various applications like portfolio optimization, cryptography, and combinatorial analysis.

**Janus-Q Cloud**: Manages quantum cloud resources, including a software stack with a RESTful API, cloud middleware (using AWS Lambda, QM Manager, Lambda Scheduler), and hardware interfaces (QPU, Hardware Controller).

**Janus-CT**: Compiles quantum circuits, featuring compilation passes (Quantum Decomposition, Type check, Constant folding, Optimizations, Error reduction, and Accuracy) and a compiler back-end (Quantum Compiler).

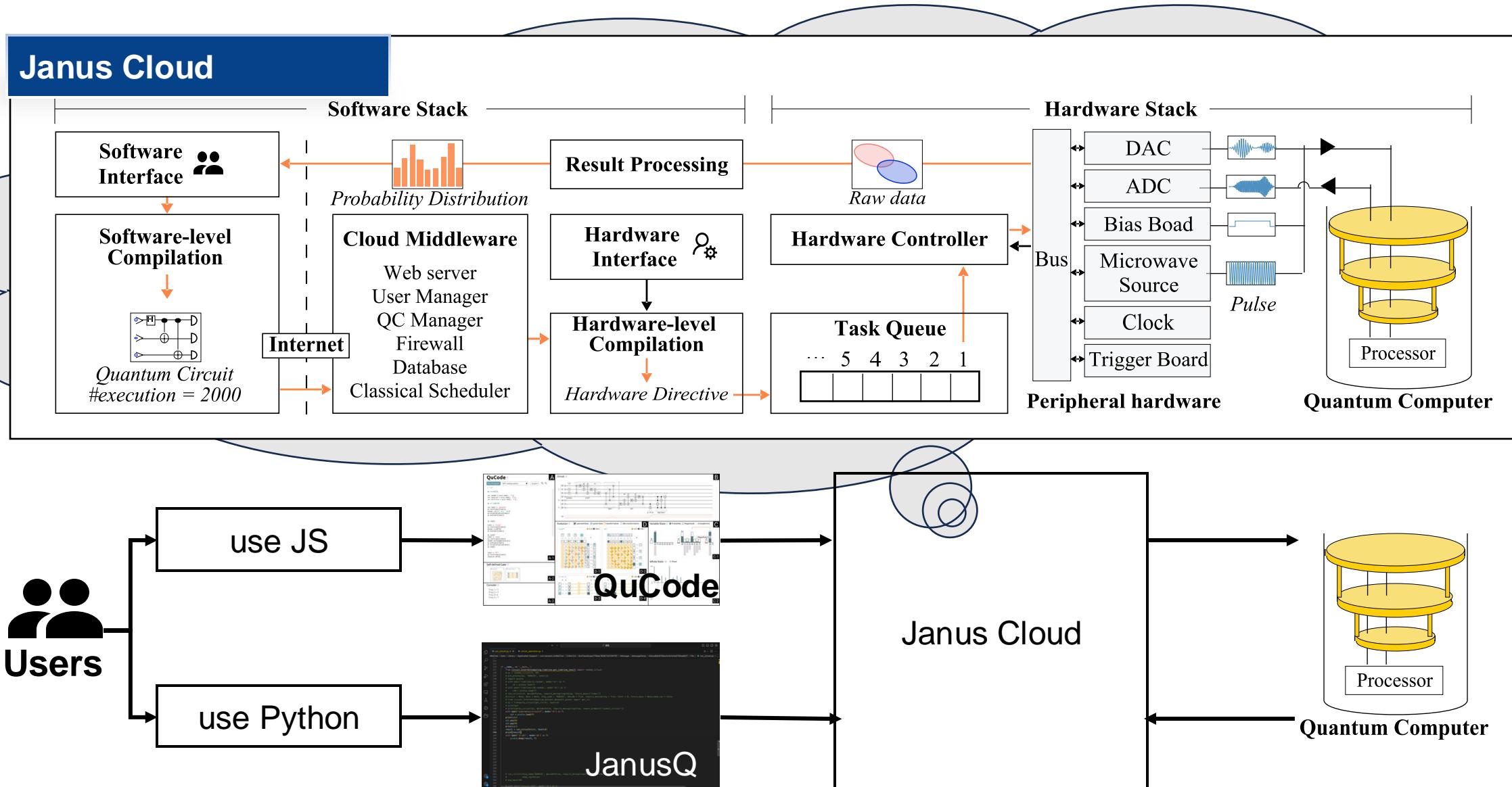
**Janus-PulseLab**: Manages quantum pulses, including a pulse library, pulse manager, and pulse generator.

**Abstract**: The design of quantum computing exhibits a high potential to outperform classical computing in solving complex problems, e.g., combinatorial optimization, and network analysis. However, achieving entanglement breakup on the real-world quantum device is difficult, involves a high degree of noise, high compilation overhead, and high cost of managing the quantum device. In this tutorial, we present Janus 2.0, an open-source framework for analyzing and optimizing quantum circuit. This tutorial begins with introduction of JanusQ cloud platform, which is equipped with several superconducting quantum chips, developed by Zhejiang University (Science 2019). Then, we present the tutorial of Janus 2.0 toolkit, from application-level to hardware-level. First, we introduce Janus application-software codesign technique for accelerating the solving of Boolean satisfiability (SAT) problems (HPCA 2023). We will provide code and demo to demonstrate the speedup of Janus-SAT. Second, we introduce Janus-CT, a unified compilation framework that integrates the upstream vectorization model and downstream models (MICRO 2023). In this tutorial, we provide the code and

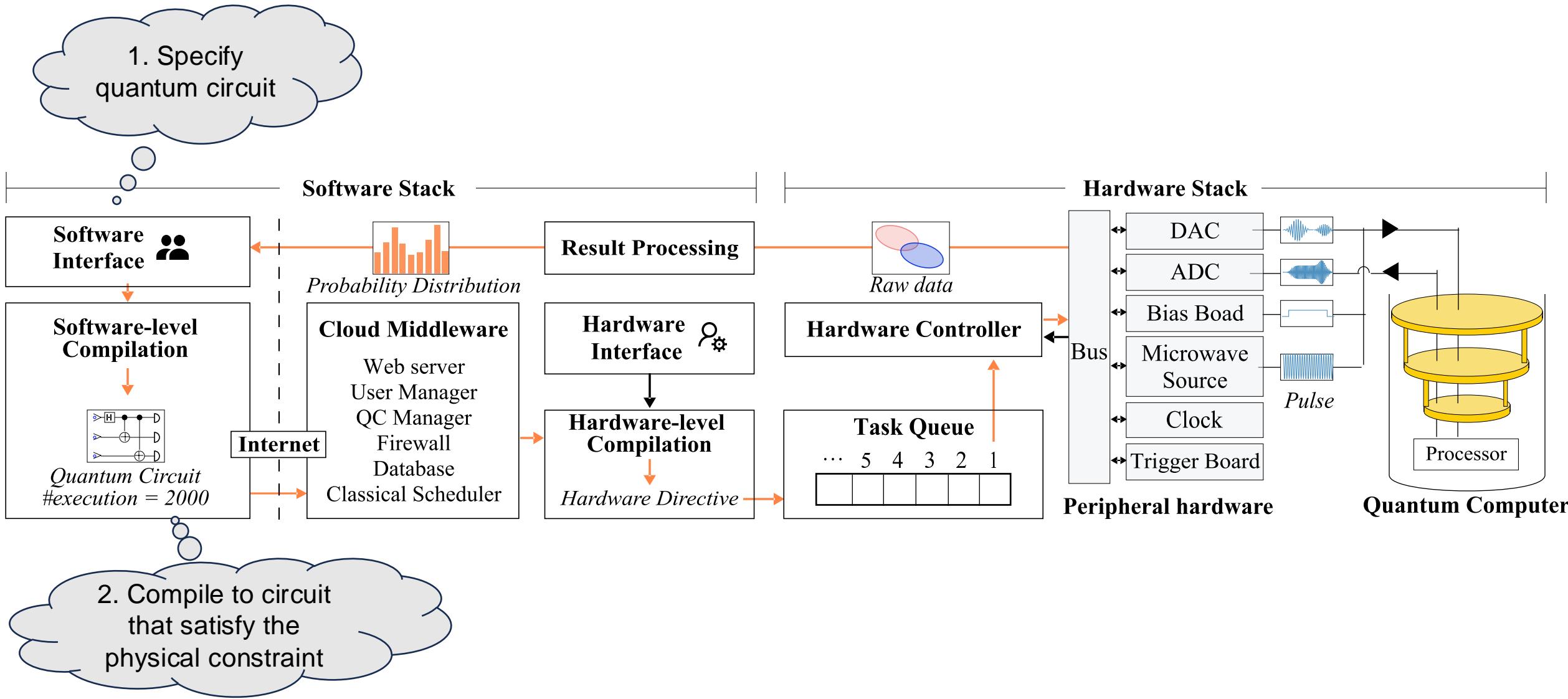
**October 2023**

1. Enable the online Janus-CT and Janus-SAT

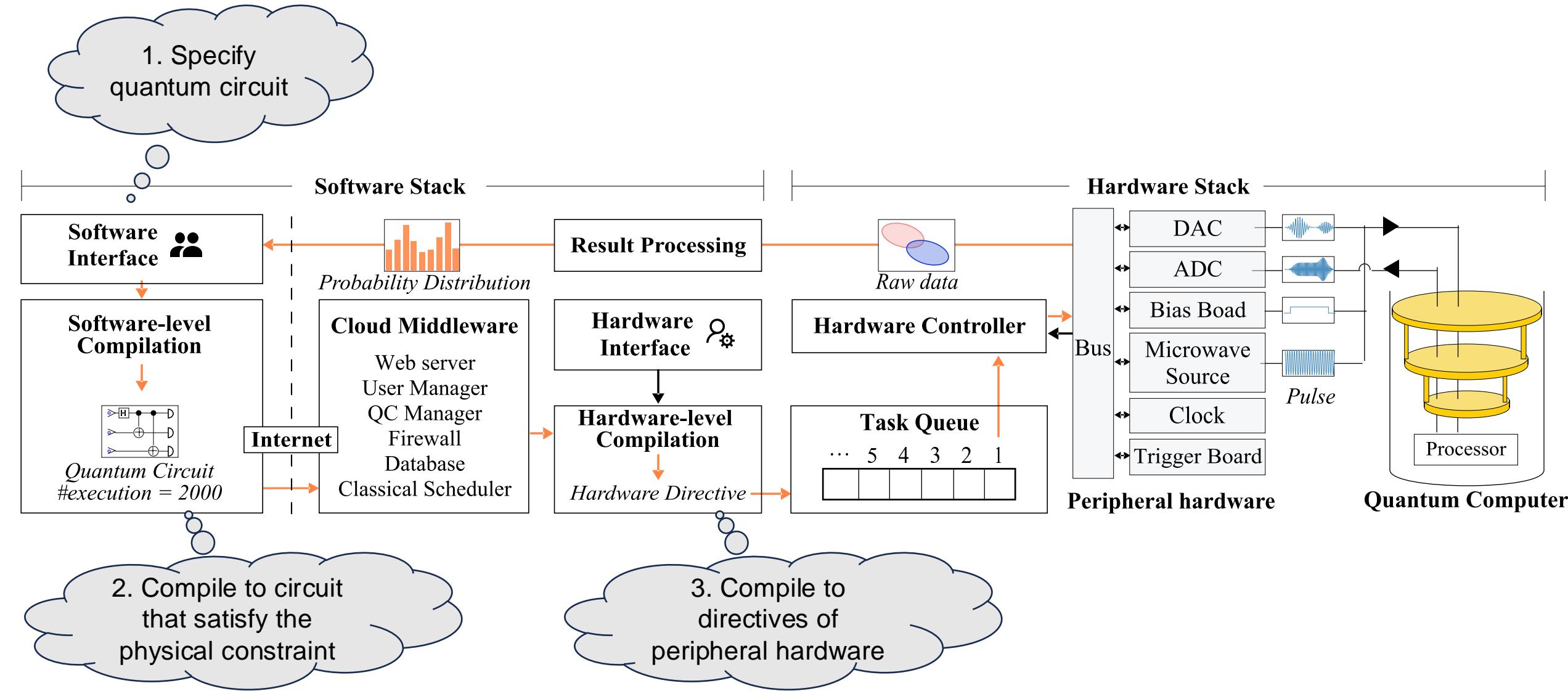
# What we can do on Janus Platform



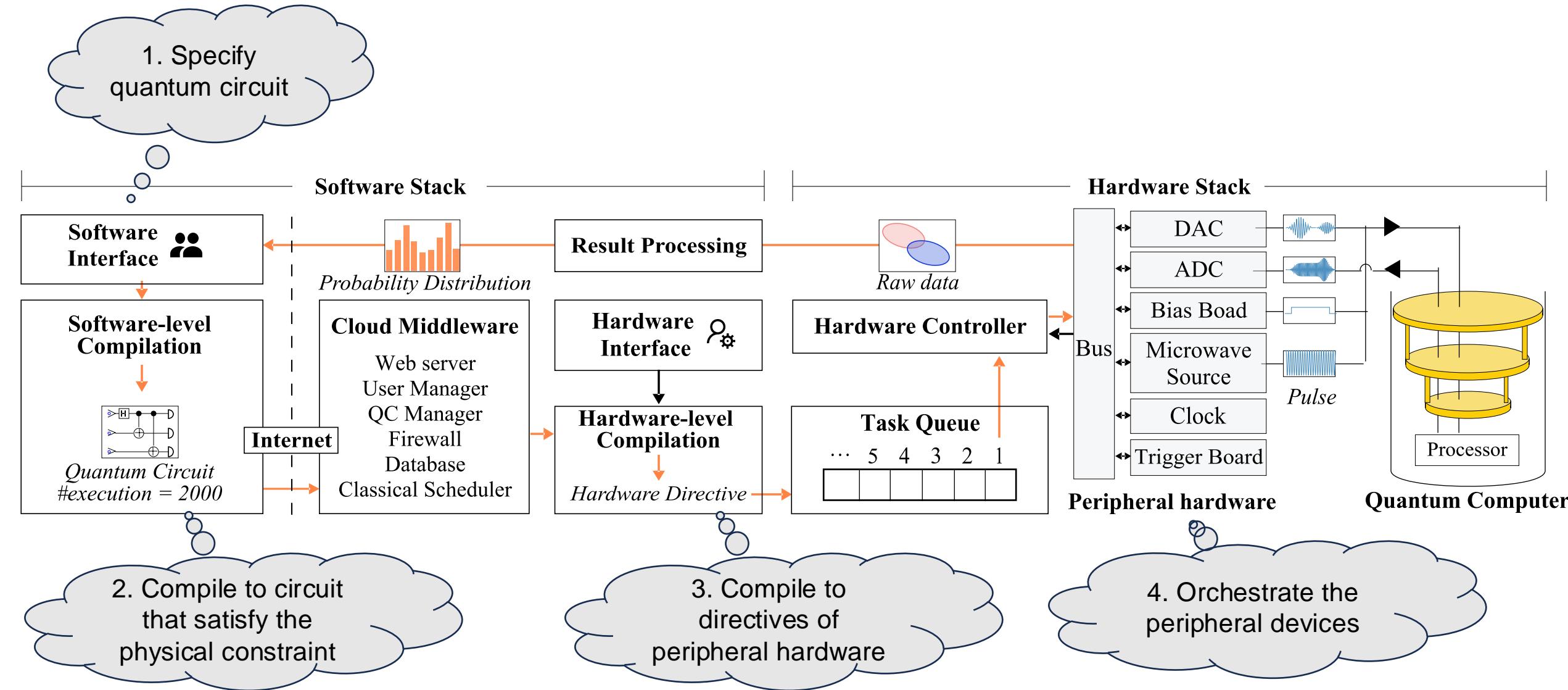
# What we can do on Janus Platform



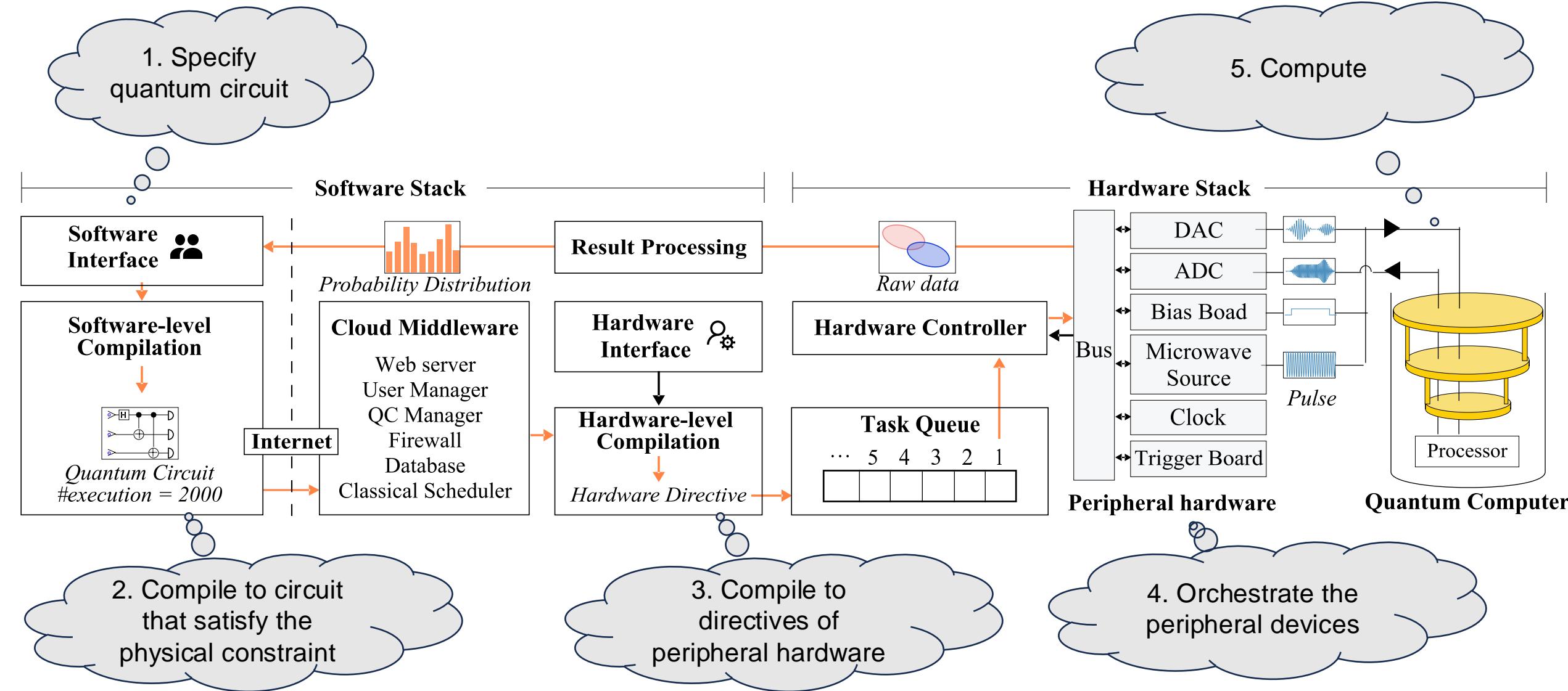
# What we can do on Janus Platform



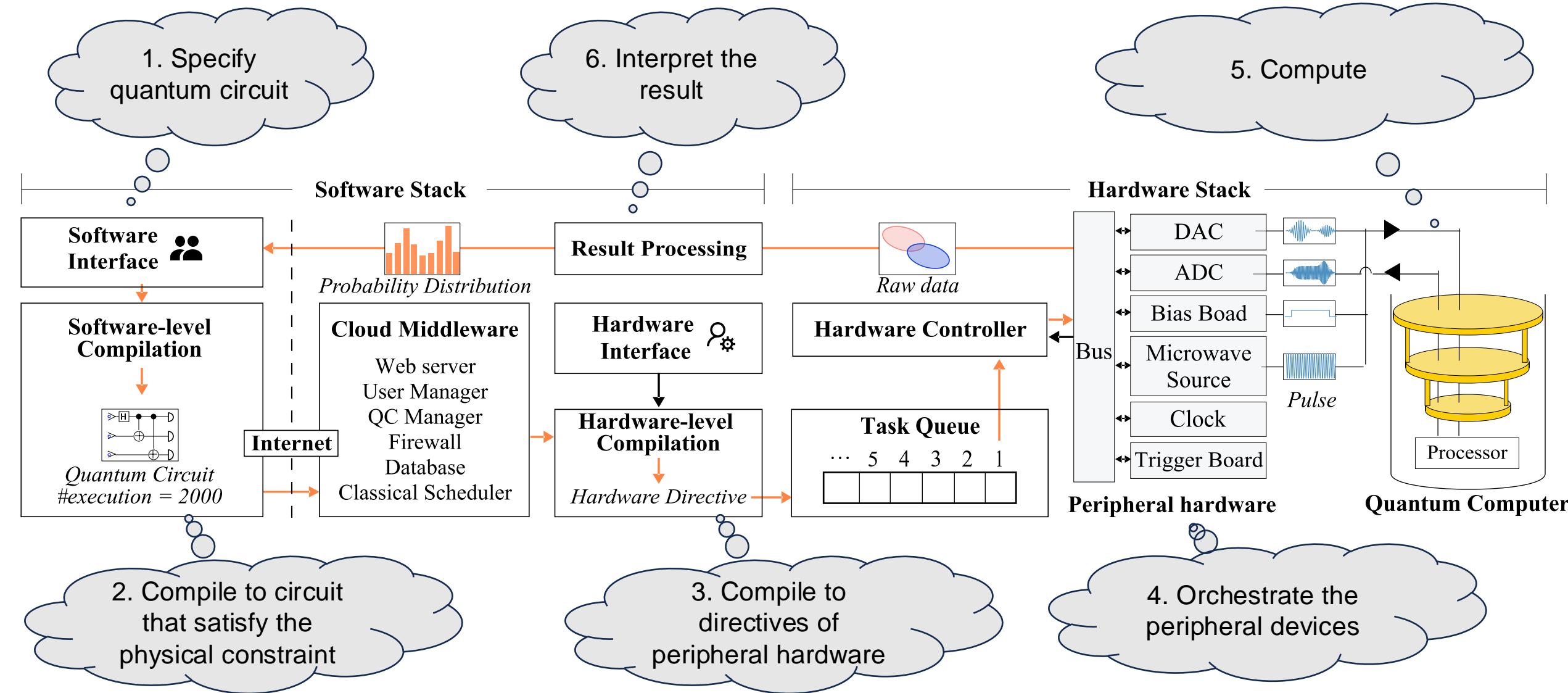
# What we can do on Janus Platform



# What we can do on Janus Platform



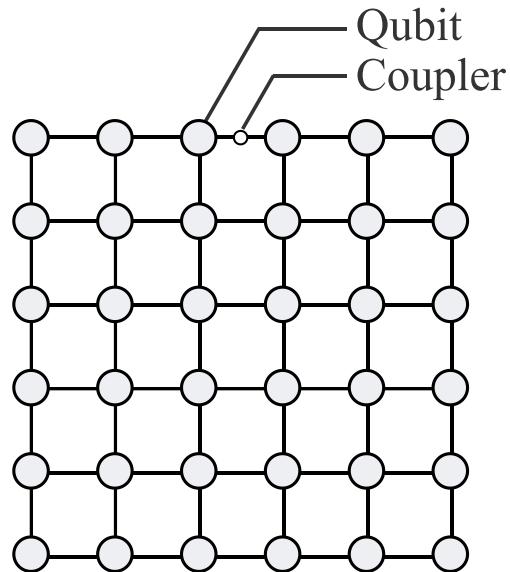
# What we can do on Janus Platform



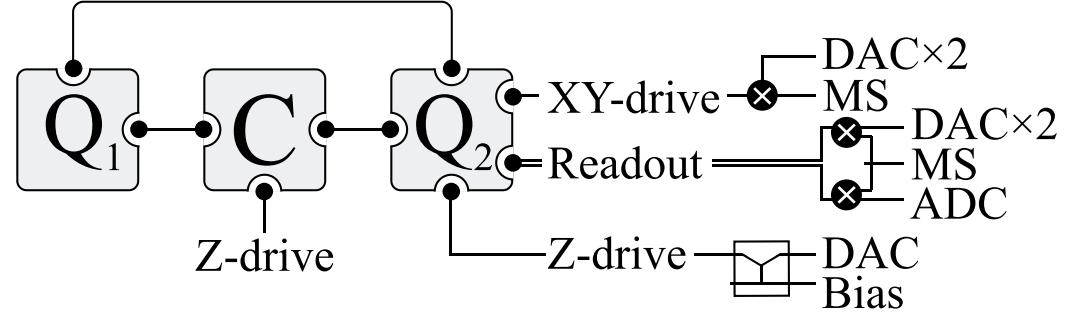
# Different compared to other quantum cloud platforms



Custom hardware & more flexibility to improve the fidelity



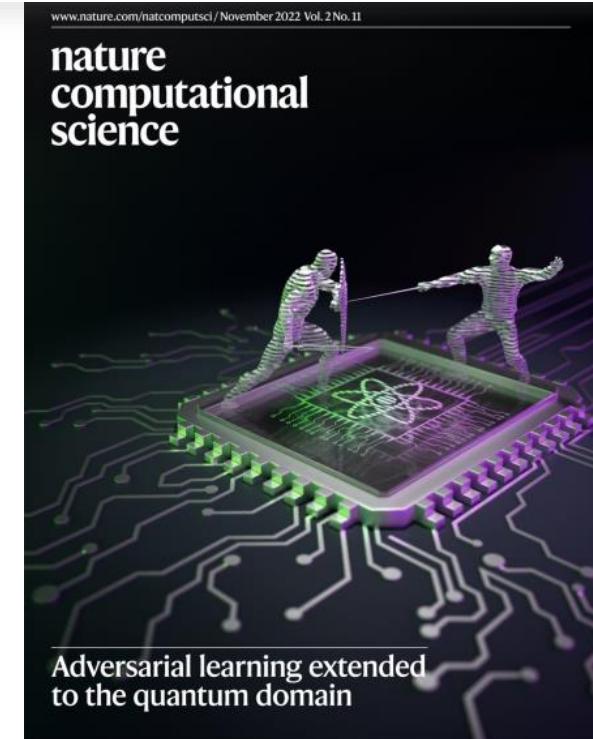
(a) Processor topology.



(b) Sketch of two qubits and a coupler.

***Customize topology, processor topology, qubit frequency for academic researches.***

Achieving 99% accuracy in the medical image classification.



Cover of Volume 2 Issue 11, November 2022, Nature computational science

# Creating an Account



浙江大學  
ZHEJIANG UNIVERSITY

Go to Janus Cloud

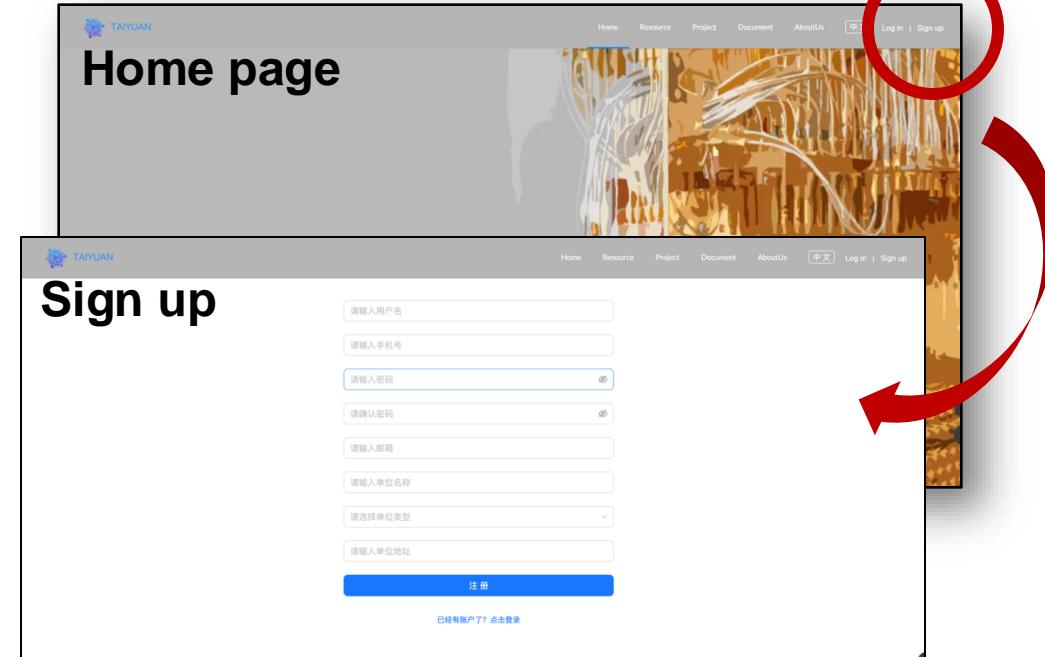
Open <http://janusq.zju.edu.cn/home>



QR code

Sign up

Click sign up button



- After signing up, you will get an API key for task submission.
- **The quantum computer can be accessed on the website during the tutorial !!**

# Running Program On the Website



## Submitting on website

Open <http://janusq.zju.edu.cn/home>

TAIYUAN

### Home page

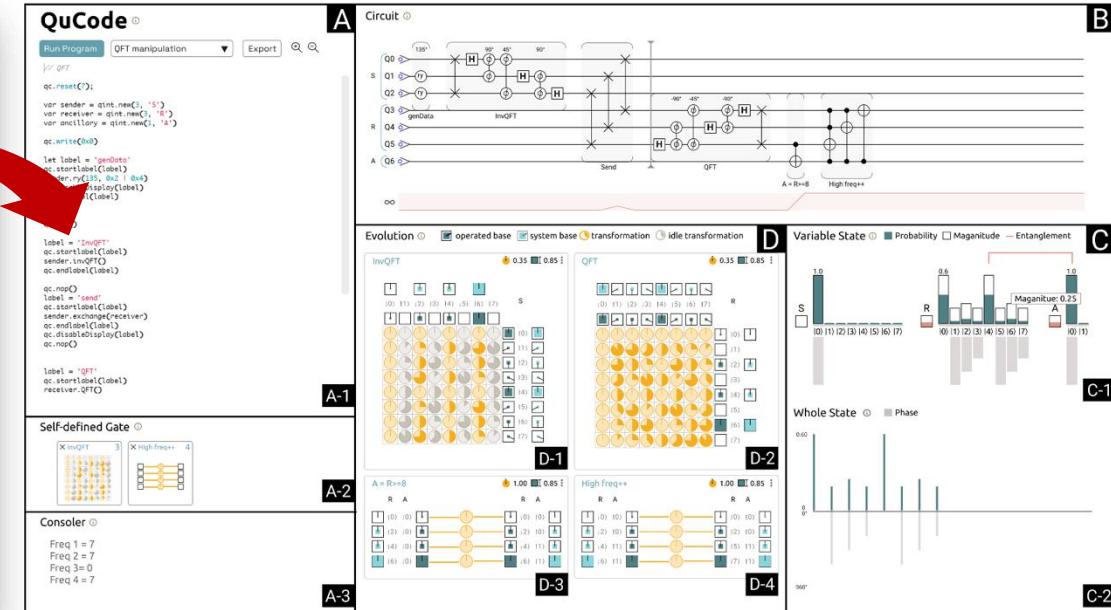
Taiyuan Quantum

Provide real-time quantum computing service

Get Start

Click

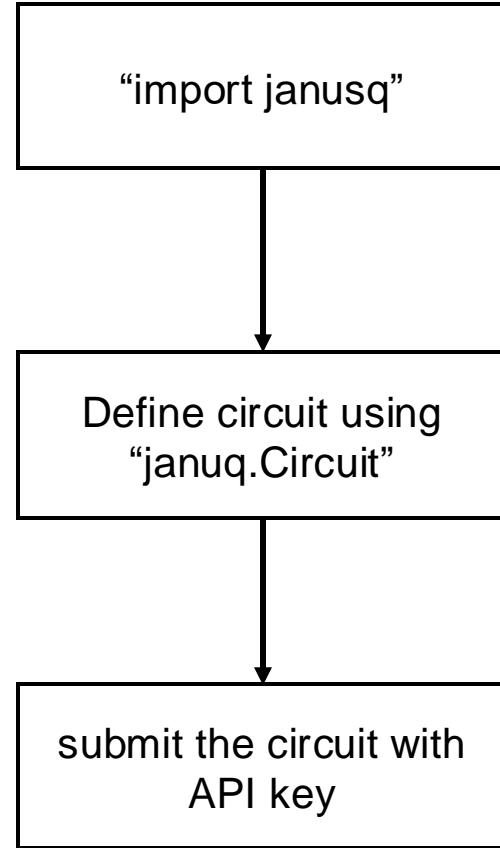
## Programming on the Code editor



- The quantum computer can be accessed on the website during the tutorial !!

# Running Program By Python API

## Submitting by Python



Import package {

```
import matplotlib.pyplot as plt
from janusq.cloud_interface import submit, get_result
from janusq.data_objects.circuit import Circuit
```

Define a circuit {

```
qc = Circuit([], n_qubits = 4)
qc.h(0, 0)
qc.cx(0, 1, 1)
qc.cx(1, 2, 2)
qc.cx(2, 3, 3)
print(qc)
```

Submit circuit {

```
result = submit(circuit=qc, label= 'GHZ', shots= 3000,
run_type='simulator', API_TOKEN="")
```

Get result {

```
result = get_result(result['data']['result_id'],
run_type='simulator', result_format='probs')
print(result)
```

# QuCode: Online Quantum Program Editor



浙江大學  
ZHEJIANG UNIVERSITY

**QuCode**

Run Program QFT manipulation Export

V V QFT

**Program editor**

```
var sender = qint.new(3, 'S')
var receiver = qint.new(3, 'R')
var ancillary = qint.new(1, 'A')

qc.write(0x0)

let label = 'genData'
qc.startlabel(label)
sender.ry(135, 0x2 | 0x4)
qc.disableDisplay(label)
qc.endlabel(label)

qc.nop()

label = 'InvQFT'
qc.startlabel(label)
sender.invQFT()
qc.endlabel(label)

qc.nop()
label = 'send'
qc.startlabel(label)
sender.exchange(receiver)
qc.endlabel(label)
qc.disableDisplay(label)
qc.nop()

label = 'QFT'
qc.startlabel(label)
receiver.QFT()
```

**Circuit reuse**

**Self-defined Gate**

**Console**

```
Freq 1 = 7
Freq 2 = 7
Freq 3 = 0
Freq 4 = 7
```

**A**
**B**

**Circuit**

**Quantum circuit view**

**D**
**C**

**D-1**
**D-2**
**D-3**
**D-4**
**C-1**
**C-2**

**Evolution view**

InvQFT      QFT

**Variable State**

S      R      A

**A = R>=8**
**High freq++**

**D-1**
**D-2**
**D-3**
**D-4**

**Whole State**
**State view**

**C-1**
**C-2**

Whole State

State view

HPCA 2025

37

# QuCode: Online Quantum Program Editor



浙江大學  
ZHEJIANG UNIVERSITY

Program editor

A

**Circuit**

QC code:

```

qc.reset(7);

var sender = qint.new(3, 'S')
var receiver = qint.new(3, 'R')
var ancillary = qint.new(1, 'A')

qc.write(0x0)

let label = 'genData'
qc.startlabel(label)
sender.ry(135, 0x2 | 0x4)
qc.disableDisplay(label)
qc.endlabel(label)

qc.nop()

label = 'InvQFT'
qc.startlabel(label)
sender.invQFT()
qc.endlabel(label)

qc.nop()
label = 'send'
qc.startlabel(label)
sender.exchange(receiver)
qc.endlabel(label)
qc.disableDisplay(label)
qc.nop()

label = 'QFT'
qc.startlabel(label)
receiver.QFT()
    
```

**Evolution**

- operated base
- system base
- transformation
- idle transformation

**Variable State**

- Probability
- Magnitude
- Entanglement

**D**

**C**

**C-1**

**C-2**

**D**

**D-1**

**D-2**

**D-3**

**D-4**

**D-5**

**D-6**

**D-7**

**D-8**

**D-9**

**D-10**

**D-11**

**D-12**

**D-13**

**D-14**

**D-15**

**D-16**

**D-17**

**D-18**

**D-19**

**D-20**

**D-21**

**D-22**

**D-23**

**D-24**

**D-25**

**D-26**

**D-27**

**D-28**

**D-29**

**D-30**

**D-31**

**D-32**

**D-33**

**D-34**

**D-35**

**D-36**

**D-37**

**D-38**

**D-39**

**D-40**

**D-41**

**D-42**

**D-43**

**D-44**

**D-45**

**D-46**

**D-47**

**D-48**

**D-49**

**D-50**

**D-51**

**D-52**

**D-53**

**D-54**

**D-55**

**D-56**

**D-57**

**D-58**

**D-59**

**D-60**

**D-61**

**D-62**

**D-63**

**D-64**

**D-65**

**D-66**

**D-67**

**D-68**

**D-69**

**D-70**

**D-71**

**D-72**

**D-73**

**D-74**

**D-75**

**D-76**

**D-77**

**D-78**

**D-79**

**D-80**

**D-81**

**D-82**

**D-83**

**D-84**

**D-85**

**D-86**

**D-87**

**D-88**

**D-89**

**D-90**

**D-91**

**D-92**

**D-93**

**D-94**

**D-95**

**D-96**

**D-97**

**D-98**

**D-99**

**D-100**

**D-101**

**D-102**

**D-103**

**D-104**

**D-105**

**D-106**

**D-107**

**D-108**

**D-109**

**D-110**

**D-111**

**D-112**

**D-113**

**D-114**

**D-115**

**D-116**

**D-117**

**D-118**

**D-119**

**D-120**

**D-121**

**D-122**

**D-123**

**D-124**

**D-125**

**D-126**

**D-127**

**D-128**

**D-129**

**D-130**

**D-131**

**D-132**

**D-133**

**D-134**

**D-135**

**D-136**

**D-137**

**D-138**

**D-139**

**D-140**

**D-141**

**D-142**

**D-143**

**D-144**

**D-145**

**D-146**

**D-147**

**D-148**

**D-149**

**D-150**

**D-151**

**D-152**

**D-153**

**D-154**

**D-155**

**D-156**

**D-157**

**D-158**

**D-159**

**D-160**

**D-161**

**D-162**

**D-163**

**D-164**

**D-165**

**D-166**

**D-167**

**D-168**

**D-169**

**D-170**

**D-171**

**D-172**

**D-173**

**D-174**

**D-175**

**D-176**

**D-177**

**D-178**

**D-179**

**D-180**

**D-181**

**D-182**

<div data-bbox="310 6100 850 6130" style="border: 1px solid black; padding:

# QuCode: Online Quantum Program Editor



浙江大学  
ZHEJIANG UNIVERSITY

**Program editor**

**A**

Run Program QFT manipulation Export 🔍 🔍

```
✓ QFT
qc.reset();
var sender = qint.new(3, 'S')
var receiver = qint.new(3, 'R')
var ancillary = qint.new(1, 'A')

qc.write(0x0)
let label = 'genData'
qc.startLabel(label)
sender.ry(135, 0x2 | 0x4)
qc.disableDisplay(label)
qc.endLabel(label)

qc.nop()

label = 'InvQFT'
qc.startLabel(label)
sender.invQFT()
qc.endLabel(label)

qc.nop()
label = 'send'
qc.startLabel(label)
sender.exchange(receiver)
qc.endLabel(label)
qc.disableDisplay(label)
qc.nop()

label = 'QFT'
qc.startLabel(label)
receiver.QFT()
```

Provide 14 examples:

1. QFT
2. teleportation
3. Grover
- ...

**B**

Circuit

**C**

Variable State

**D**

QFT

**D-1**

**D-2**

**D-3**

**D-4**

**A-1**

**A-2**

**A-3**

**A-4**

**A-5**

**A-6**

**A-7**

**A-8**

**A-9**

**A-10**

**A-11**

**A-12**

**A-13**

**A-14**

**A-15**

**A-16**

**A-17**

**A-18**

**A-19**

**A-20**

**A-21**

**A-22**

**A-23**

**A-24**

**A-25**

**A-26**

**A-27**

**A-28**

**A-29**

**A-30**

**A-31**

**A-32**

**A-33**

**A-34**

**A-35**

**A-36**

**A-37**

**A-38**

**A-39**

**A-40**

**A-41**

**A-42**

**A-43**

**A-44**

**A-45**

**A-46**

**A-47**

**A-48**

**A-49**

**A-50**

**A-51**

**A-52**

**A-53**

**A-54**

**A-55**

**A-56**

**A-57**

**A-58**

**A-59**

**A-60**

**A-61**

**A-62**

**A-63**

**A-64**

**A-65**

**A-66**

**A-67**

**A-68**

**A-69**

**A-70**

**A-71**

**A-72**

**A-73**

**A-74**

**A-75**

**A-76**

**A-77**

**A-78**

**A-79**

**A-80**

**A-81**

**A-82**

**A-83**

**A-84**

**A-85**

**A-86**

**A-87**

**A-88**

**A-89**

**A-90**

**A-91**

**A-92**

**A-93**

**A-94**

**A-95**

**A-96**

**A-97**

**A-98**

**A-99**

**A-100**

**A-101**

**A-102**

**A-103**

**A-104**

**A-105**

**A-106**

**A-107**

**A-108**

**A-109**

**A-110**

**A-111**

**A-112**

**A-113**

**A-114**

**A-115**

**A-116**

**A-117**

**A-118**

**A-119**

**A-120**

**A-121**

**A-122**

**A-123**

**A-124**

**A-125**

**A-126**

**A-127**

**A-128**

**A-129**

**A-130**

**A-131**

**A-132**

**A-133**

**A-134**

**A-135**

**A-136**

**A-137**

**A-138**

**A-139**

**A-140**

**A-141**

**A-142**

**A-143**

**A-144**

**A-145**

**A-146**

**A-147**

**A-148**

**A-149**

**A-150**

**A-151**

**A-152**

**A-153**

**A-154**

**A-155**

**A-156**

**A-157**

**A-158**

**A-159**

**A-160**

**A-161**

**A-162**

**A-163**

**A-164**

**A-165**

**A-166**

**A-167**

**A-168**

**A-169**

**A-170**

**A-171**

**A-172**

**A-173**

**A-174**

**A-175**

**A-176**

**A-177**

**A-178**

**A-179**

**A-180**

**A-181**

**A-182**

**A-183**

**A-184**

**A-185**

**A-186**

**A-187**

**A-188**

**A-189**

**A-190**

**A-191**

**A-192**

**A-193**

**A-194**

**A-195**

**A-196**

**A-197**

**A-198**

**A-199**

**A-200**

**A-201**

**A-202**

**A-203**

**A-204**

**A-205**

**A-206**

**A-207**

**A-208**

**A-209**

**A-210**

**A-211**

**A-212**

**A-213**

**A-214**

**A-215**

**A-216**

**A-217**

**A-218**

**A-219**

**A-220**

**A-221**

**A-222**

**A-223**

**A-224**

**A-225**

**A-226**

**A-227**

**A-228**

**A-229**

**A-230**

**A-231**

**A-232**

**A-233**

**A-234**

**A-235**

**A-236**

**A-237**

**A-238**

**A-239**

**A-240**

**A-241**

**A-242**

**A-243**

**A-244**

**A-245**

**A-246**

**A-247**

**A-248**

**A-249**

**A-250**

**A-251**

**A-252**

**A-253**

**A-254**

**A-255**

**A-256**

**A-257**

**A-258**

**A-259**

**A-260**

**A-261**

**A-262**

**A-263**

**A-264**

**A-265**

**A-266**

**A-267**

**A-268**

**A-269**

**A-270**

**A-271**

**A-272**

**A-273**

**A-274**

**A-275**

**A-276**

**A-277**

**A-278**

**A-279**

**A-280**

**A-281**

**A-282**

**A-283**

**A-284**

**A-285**

**A-286**

**A-287**

**A-288**

**A-289**

**A-290**

**A-291**

**A-292**

**A-293**

**A-294**

**A-295**

**A-296**

**A-297**

**A-298**

**A-299**

**A-300**

**A-301**

**A-302**

**A-303**

**A-304**

**A-305**

**A-306**

**A-307**

**A-308**

**A-309**

**A-310**

**A-311**

**A-312**

**A-313**

**A-314**

**A-315**

**A-316**

**A-317**

**A-318**

**A-319**

**A-320**

**A-321**

**A-322**

**A-323**

**A-324**

**A-325**

**A-326**

**A-327**

**A-328**

**A-329**

**A-330**

**A-331**

**A-332**

**A-333**

**A-334**

**A-335**

**A-336**

**A-337**

**A-338**

**A-339**

**A-340**

**A-341**

**A-342**

**A-343**

**A-344**

**A-345**

**A-346**

**A-347**

**A-348**

**A-349**

**A-350**

**A-351**

**A-352**

**A-353**

**A-354**

**A-355**

**A-356**

**A-357**

**A-358**

**A-359**

**A-360**

**A-361**

**A-362**

**A-363**

**A-364**

**A-365**

**A-366**

**A-367**

**A-368**

**A-369**

**A-370**

**A-371**

**A-372**

**A-373**

**A-374**

**A-375**

**A-376**

**A-377**

**A-378**

**A-379**

**A-380**

**A-381**

**A-382**

**A-383**

**A-384**

**A-385**

**A-386**

**A-387**

**A-388**

**A-389**

**A-390**

**A-391**

**A-392**

**A-393**

**A-394**

**A-395**

**A-396**

**A-397**

**A-398**

**A-399**

**A-400**

**A-401**

**A-402**

**A-403**

**A-404**

**A-405**

**A-406**

**A-407**

**A-408**

**A-409**

**A-410**

**A-411**

**A-412**

**A-413**

**A-414**

**A-415**

**A-416**

**A-417**

**A-418**

**A-419**

**A-420**

**A-421**

**A-422**

**A-423**

**A-424**

**A-425**

**A-426**

**A-427**

**A-428**

**A-429**

**A-430**

**A-431**

**A-432**

**A-433**

**A-434**

**A-435**

**A-436**

**A-437**

**A-438**

**A-439**

**A-440**

**A-441**

**A-442**

**A-443**

**A-444**

**A-445**

**A-446**

**A-447**

**A-448**

**A-449**

**A-450**

**A-451**

**A-452**

**A-453**

**A-454**

**A-455**

**A-456**

**A-457**

**A-458**

**A-459**

**A-460**

**A-461**

**A-462**

**A-463**

**A-464**

**A-465**

**A-466**

**A-467**

**A-468**

**A-469**

**A-470**

**A-471**

**A-472**

**A-473**

**A-474**

**A-475**

**A-476**

**A-477**

**A-478**

**A-479**

**A-480**

**A-481**

**A-482**

**A-483**

**A-484**

**A-485**

**A-486**

**A-487**

**A-488**

**A-489**

**A-490**

**A-491**

**A-492**

**A-493**

**A-494**

**A-495**

**A-496**

**A-497**

**A-498**

**A-499**

**A-500**

**A-501**

**A-502**

**A-503**

**A-504**

**A-505**

**A-506**

**A-507**

**A-508**

**A-509**

**A-510**

**A-511**

**A-512**

**A-513**

**A-514**

**A-515**

**A-516**

**A-517**

**A-518**

**A-519**

**A-520**

**A-521**

**A-522**

**A-523**

**A-524**

**A-525**

**A-526**

**A-527**

**A-528**

**A-529**

**A-530**

**A-531**

**A-532**

**A-533**

**A-534**

**A-535**

**A-536**

**A-537**

**A-538**

**A-539**

**A-540**

**A-541**

**A-542**

**A-543**

**A-544**

**A-545**

**A-546**

**A-547**

**A-548**

**A-549**

**A-550**

**A-551**

**A-552**

**A-553**

**A-554**

**A-555**

**A-556**

**A-557**

**A-558**

**A-559**

**A-560**

**A-561**

**A-562**

**A-563**

**A-564**

**A-565**

**A-566**

**A-567**

**A-568**

**A-569**

**A-570**

**A-571**

**A-572**

**A-573**

**A-574**

**A-575**

**A-576**

**A-577**

**A-578**

**A-579**

**A-580**

**A-581**

**A-582**

**A-583**

**A-584**

**A-585**

**A-586**

**A-587**

**A-588**

**A-589**

**A-590**

**A-591**

**A-592**

**A-593**

**A-594**

**A-595**

**A-596**

**A-597**

**A-598**

**A-599**

**A-600**

**A-601**

**A-602**

**A-603**

**A-604**

**A-605**

**A-606**

**A-607**

**A-608**

**A-609**

**A-610**

**A-611**

**A-612**

**A-613**

**A-614**

**A-615**

**A-616**

**A-617**

**A-618**

**A-619**

**A-620**

**A-621**

**A-622**

**A-623**

**A-624**

**A-625**

**A-626**

**A-627**

**A-628**

**A-629**

**A-630**

**A-631**

**A-632**

**A-633**

**A-634**

**A-635**

**A-636**

**A-637**

**A-638**

**A-639**

**A-640**

**A-641**

**A-642**

**A-643**

**A-644**

**A-645**

**A-646**

**A-647**

**A-648**

**A-649**

**A-650**

**A-651**

**A-652**

**A-653**

**A-654**

**A-655**

**A-656**

**A-657**

**A-658**

**A-659**

**A-660**

**A-661**

**A-662**

**A-663**

**A-664**

**A-665**

**A-666**

**A-667**

**A-668**

**A-669**

**A-670**

**A-671**

**A-672**

**A-673**

**A-674**

**A-675**

**A-676**

**A-677**

**A-678**

**A-679**

**A-680**

**A-681**

**A-682**

**A-683**

**A-684**

**A-685**

**A-686**

**A-687**

**A-688**

**A-689**

**A-690**

**A-691**

**A-692**

**A-693**

**A-694**

**A-695**

**A-696**

**A-697**

**A-698**

**A-699**

**A-700**

**A-701**

**A-702**

**A-703**

**A-704**

**A-705**

**A-706**

**A-707**

**A-708**

**A-709**

**A-710**

**A-711**

**A-712**

**A-713**

**A-714**

**A-715**

**A-716**

**A-717**

**A-718**

**A-719**

**A-720**

**A-721**

**A-722**

**A-723**

**A-724**

**A-725**

**A-726**

**A-727**

**A-728**

**A-729**

**A-730**

**A-731**

**A-732**

**A-733**

**A-734**

**A-735**

**A-736**

**A-737**

**A-738**

**A-739**

**A-740**

**A-741**

**A-742**

**A-743**

**A-744**

**A-745**

**A-746**

**A-747**

**A-748**

**A-749**

**A-750**

**A-751**

**A-752**

**A-753**

**A-754**

**A-755**

**A-756**

**A-757**

**A-758**

**A-759**

**A-760**

**A-761**

**A-762**

**A-763**

**A-764**

**A-765**

**A-766**

**A-767**

**A-768**

**A-769**

**A-770**

**A-771**

**A-772**

**A-773**

**A-774**

**A-775**

**A-776**

**A-777**

**A-778**

**A-779**

**A-780**

**A-781**

**A-782**

**A-783**

**A-784**

**A-785**

**A-786**

**A-787**

**A-788**

**A-789**

**A-790**

**A-791**

**A-792**

**A-793**

**A-794**

**A-795**

**A-796**

**A-797**

**A-798**

**A-799**

**A-800**

**A-801**

**A-802**

**A-803**

**A-804**

**A-805**

**A-806**

**A-807**

**A-808**

**A-809**

**A-810**

**A-811**

**A-812**

**A-813**

**A-814**

**A-815**

**A-816**

**A-817**

**A-818**

**A-819**

**A-820**

**A-821**

**A-822**

**A-823**

**A-824**

**A-825**

**A-826**

**A-827**

**A-828**

**A-829**

**A-830**

**A-831**

**A-832**

**A-833**

**A-834**

**A-835**

**A-836**

**A-837**

**A-838**

**A-839**

**A-840**

**A-841**

**A-842**

**A-843**

**A-844**

**A-845**

**A-846**

**A-847**

**A-848**

**A-849**

**A-850**

**A-851**

**A-852**

**A-853**

**A-854**

**A-855**

**A-856**

**A-857**

**A-858**

**A-859**

**A-860**

**A-861**

**A-862**

**A-863**

**A-864**

**A-865**

**A-866**

**A-867**

**A-868**

**A-869**

**A-870**

**A-871**

**A-872**

**A-873**

**A-874**

**A-875**

**A-876**

**A-877**

**A-878**

**A-879**

**A-880**

**A-881**

**A-882**

**A-883**

**A-884**

**A-885**

**A-886**

**A-887**

**A-888**

**A-889**

**A-890**

**A-891**

**A-892**

**A-893**

**A-894**

**A-895**

**A-896**

**A-897**

**A-898**

**A-899**

**A-900**

**A-901**

**A-902**

**A-903**

**A-904**

**A-905**

**A-906**

**A-907**

**A-908**

**A-909**

**A-910**

**A-911**

**A-912**

**A-913**

**A-914**

**A-915**

**A-916**

**A-917**

**A-918**

**A-919**

**A-920**

**A-921**

**A-922**

**A-923**

**A-924**

<

# QuCode: Online Quantum Program Editor



浙江大學  
ZHEJIANG UNIVERSITY

**QuCode** ⓘ

**Circuit View**

```
// QFT
qc.reset(7);

var sender = qint.new(3, 'S')
var receiver = qint.new(3, 'R')
var ancillary = qint.new(1, 'A')

qc.write(0x0)

let label = 'genData'
qc.startlabel(label)
sender.ry(135, 0x2 | 0x4)
qc.disableDisplay(label)
qc.endlabel(label)

qc.nop()

label = 'InvQFT'
qc.startlabel(label)
sender.invQFT()
qc.endlabel(label)

qc.nop()
label = 'send'
qc.startlabel(label)
sender.exchange(receiver)
qc.endlabel(label)
qc.disableDisplay(label)
qc.nop()

label = 'QFT'
qc.startlabel(label)
receiver.QFT()
```

**Grouping**

**Purity**

**Evolution** ⓘ

**Variable State** ⓘ

**D**

**C**

**A-1**

**A-2**

**A-3**

**D-1**

**D-2**

**D-3**

**D-4**

**C-1**

**C-2**

# QuCode: Online Quantum Program Editor



浙江大學  
ZHEJIANG UNIVERSITY

**QuCode**

Run Program   QFT manipulation   Export   🔍   🔍

```

// QFT
qc.reset(7);

var sender = qint.new(3, 'S')
var receiver = qint.new(3, 'R')
var ancillary = qint.new(1, 'A')

qc.write(0x0)

let label = 'genData'
qc.startlabel(label)
sender.ry(135, 0x2 | 0x4)
qc.disableDisplay(label)
qc.endlabel(label)

qc.nop()

label = 'InvQFT'
qc.startlabel(label)
sender.invQFT()
qc.endlabel(label)

qc.nop()
label = 'send'
qc.startlabel(label)
sender.exchange(receiver)
qc.endlabel(label)
qc.disableDisplay(label)
qc.nop()

label = 'QFT'
qc.startlabel(label)
receiver.QFT()
      
```

**A**

**Circuit**

**Evolution**

InvQFT

0.35 0.85

QFT

0.35 0.85

A = R>=8

1.00 0.85

High freq++

1.00 0.85

**B**

**A-1**

**D-1**

**A-2**

**D-2**

**A-3**

Console

```

Freq 1 = 7
Freq 2 = 7
Freq 3 = 0
Freq 4 = 7
      
```

**D-3**

**D-4**

**State View**

partial trace

The whole state

R

Magnitude   Entanglement

Whole State

Magnitude   Phase

# QuCode: Online Quantum Program Editor



浙江大學  
ZHEJIANG UNIVERSITY

**QuCode**

Run Program   QFT manipulation   Export   🔍   🔍

```

V// QFT

qc.reset(7);

var sender = qint.new(3, 'S')
var receiver = qint.new(3, 'R')
var ancillary = qint.new(1, 'A')

qc.write(0x0)

let label = 'genData'
qc.startlabel(label)
sender.ry(135, 0x2 | 0x4)
qc.disableDisplay(label)
qc.endlabel(label)

qc.nop()

label = 'InvQFT'
qc.startlabel(label)
sender.invQFT()
qc.endlabel(label)

qc.nop()
label = 'send'
qc.startlabel(label)
sender.exchange(receiver)
qc.endlabel(label)
qc.disableDisplay(label)
qc.nop()

label = 'QFT'
qc.startlabel(label)
receiver.QFT()
      
```

A
B

**Circuit**

**Evolution View**

D
C

**INVQFT**

**QFT**

D-1
D-2
D-3
D-4

C-1
C-2

1. Visualize the unitary

2. Visualize the change of qubit states

A-1
A-2
A-3

D
C

A-1
A-2
A-3

D-1
D-2
D-3
D-4

C-1
C-2

D-1
D-2
D-3
D-4

A-1
A-2
A-3

D-1
D-2
D-3
D-4

C-1
C-2

# Simulation of Time crystal



## Mathematical formulation

$$H(t) = \begin{cases} H_1, & \text{for } 0 \leq t < T_1 \\ H_2, & \text{for } T_1 \leq t < T \end{cases}$$

$$H_1 \equiv \left(\frac{\pi}{2}\right) \sum_k \sigma_k^x$$

$$U_1(t) = e^{-iH_1 t}$$

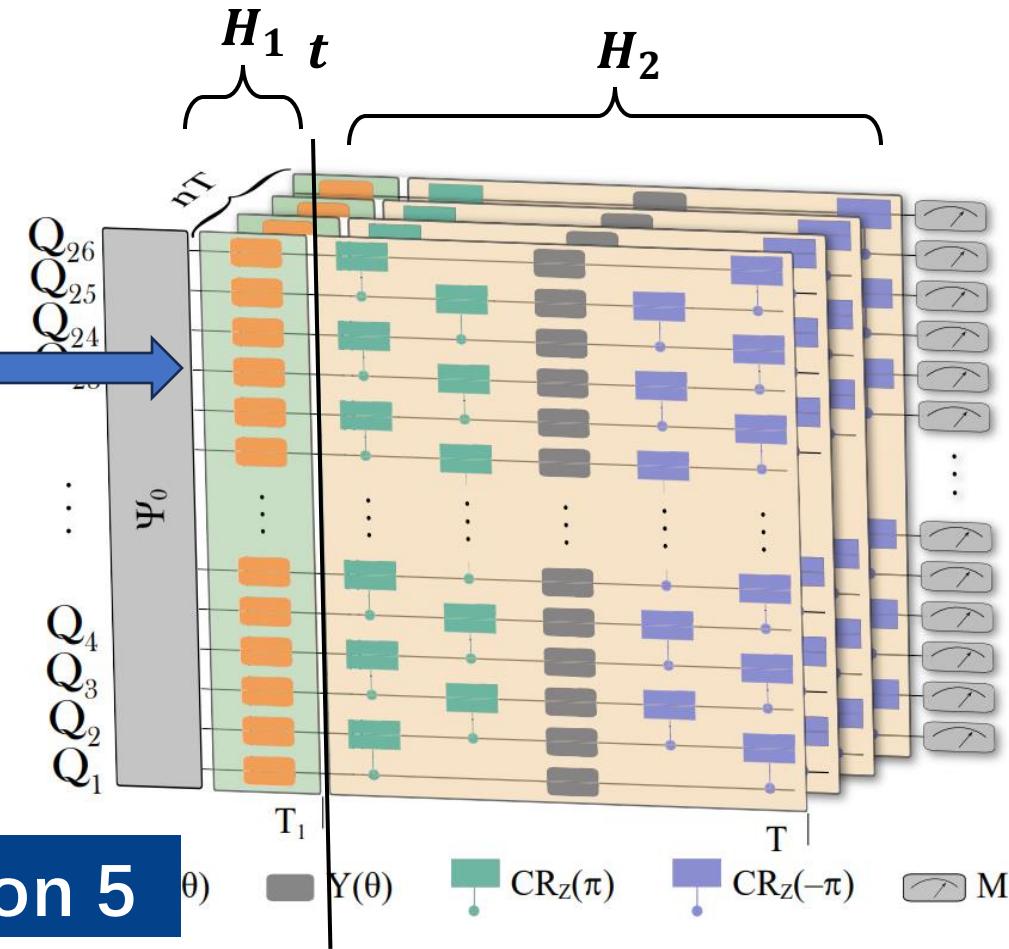
$$H_2 \equiv - \sum_k J_k \sigma_{k-1}^z \sigma_k^x \sigma_{k+1}^z$$

$$J_k \in [0, 2]$$

20 random disorder instances ( $J_k$ )

Introduced In Session 5

## Circuit Implementation





浙江大學  
ZHEJIANG UNIVERSITY

# Installing JanusQ



<https://github.com/JanusQ/JanusQ>



**Language:** Python 3, C++

**Available platforms:** Linux, Mac, Windows

**Hardware requirements:**

- Classical computer:  $\geq 16$  GB Memory, Intel i5 10 Gen

**Hardware used in the experiment**

- A server with two AMD EPYC 2.25GHz 64-core CPUs and 1.6TB DDR 5 memory is preferable.
- Superconducting, ion-trapped quantum computer and D-Wave quantum annealer.

**Installation:**

**From Docker (recommend)**

*Data collected from the quantum hardware are put into the framework.*

**From Source code**

*Wheel is provided in <https://github.com/JanusQ/JanusQ/tree/main/dist>. But Janus-SAT is disabled.*

# Installation From Docker (Preferred)



## Step 1. download docker-desktop

<https://www.docker.com/products/docker-desktop/>

*Windows Subsystem for Linux (WSL) are required for windows.*

## Step 2. pull JanusQ image

docker pull janusq/janusq



## Step 3. start image

docker run -itd -p 8888:22 -p 9999:23 --name tutorial  
janusq/janusq

## Step 4. Use SSH/VSCode/PyCharm to connect the docker

ssh root@localhost -p 8888

Docker page of JanusQ:

<https://hub.docker.com/r/janusq/janusq>

## Step 5. Or use Jupyter Lab to connect the docker

<http://localhost:9999/lab>

*The password of the docker is "" (empty).*

# Installation From Source Code



## Step 1. clone the source code

```
git clone git@github.com:JanusQ/JanusQ.git
```

## Step 2. install requirements (virtual environment is preferred)

```
cd ./JanusQ  
pip install -r requirements.txt
```

## Step 3. compile Janus-SAT

```
cd ./JanusQ/janusq/application/hyqsat/solver  
cmake .  
make install
```



Page of github:  
<https://github.com/JanusQ/JanusQ>

implemented based on C++.

# Testing JanusQ



On “./JanusQ/examples/ipynb/1\_1\_install\_janusq.ipynb”

## Test Janus-CT

```
from janusq.analysis.vectorization import *
vec_model = RandomwalkModel()
vec_model.train()
```

## Test Janus-SAT

```
from janusq.application.hyqsat import readCNF
readCNF("./data/cnf_examples/test/uf100-01.cnf")
```

## Test Janus-FEM

```
from janusq.calibration.readout_mitigation.fem
import EnumeratedProtocol
protocol = EnumeratedProtocol(4)
```

## Test Janus-Cloud

```
from janusq.objects.circuit import Circuit
from janusq.cloud_interface import submit, get_result
qc = Circuit([], n_qubits = 3)
qc.h(0, 0)
result = submit(circuit=qc, label= 'GHZ', shots= 3000,
run_type='simulator', API_TOKEN="")
result = get_result(result['data']['result_id'],
run_type='simulator', result_format='probs')
print(result)
```

# Document and Example



On: [JanusQ/examples](#) or <https://janusq.github.io/tutorials/Demonstrations>

Janus Quantum

In [1]:

```
%matplotlib inline

import os
os.chdir('..')

from analysis.vectorization import RandomwalkModel, extract_device

from data_objects.random_circuit import random_circuits, random_circuit
from data_objects.backend import GridBackend

import random
import numpy as np
```

Vectorization Model Of Janus-CT

Author: Siwei Tan

Date: 7/4/2024

Based on "[QuCT: A Framework for Analyzing Quantum Circuits](#)"

In the current Noisy Intermediate-Scale Quantum (NISQ) era, analyzing quantum programs is challenging due to the tradeoff between fidelity and efficiency. To address this, we propose Janus-CT, a unified framework that can vectorize each gate with each element, quantitatively analyze the circuit, and develop multiple downstream models. By combining these capabilities, we can leverage the strengths of different models to achieve better results than individual ones. In this demonstration, we will introduce the vectorization model of Janus-CT.

def solve\_by\_minisat(cnf\_file, save=False, result\_dir=".", verb=1, cpu\_lim=0, mem\_lim=0, strictp=False):
 ...
 description: using minisat method to solve sat domain problem.
 param {str} cnf\_file: input a cnf file, which needs to be solve.
 param {bool} save: weather save result in result dir.
 param {str} result\_dir: save result in result dir.
 param {bool} verb: weather print log.
 param {int} cpu\_lim: cpu limit(core).
 param {int} mem\_lim: memory limit(MB).
 param {bool} strictp: weather strict.
 ...
 if verb:
 verb = 1

JanusQ Demo Resources Github Team

URL

Document of functions



浙江大學  
ZHEJIANG UNIVERSITY

# Thanks for listening!