

Company Presentation

# QRISP

## die Schnittstelle der nächsten Generation zur Umsetzung von Quantenalgorithmen

Dr. Ulrich Meier


Business Development Manager

September 2025



A top-down view of four people (three men and one woman) sitting around a dark table. They are all looking down at a tablet or document on the table. The woman is on the left, wearing a floral patterned top. The man next to her is wearing a dark shirt. The man on the right is wearing a blue hoodie. The woman at the bottom is wearing a dark top. The background is dark and out of focus.

## Our Mission

 We build world leading quantum computers for the well-being of humankind, now and for the future

# IQM in numbers

IQM × **qrisp**

**330+**

Employees

**135+**

PhDs

**200M€  
+275M€**

investments

**30+**

full systems  
built

**15+**

full systems  
sold

**2018**

Founded

**50+**

Nationalities

**150**

qubit chip in  
development

**500+**

Cloud users

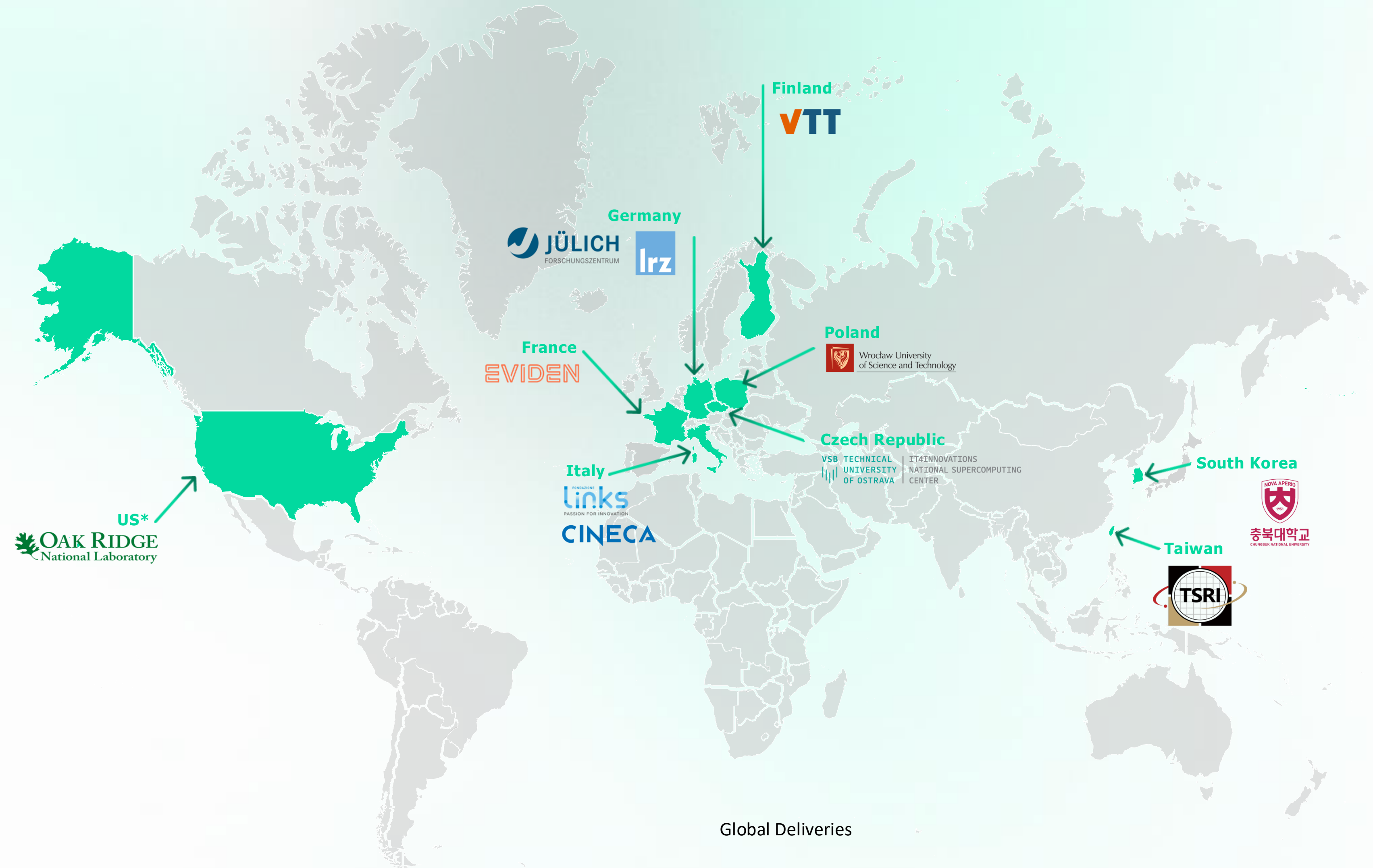
**On-prem  
& cloud**

IQM



# IQM Customer Deliveries

IQM × **qrisp**



**10+ full systems  
delivered and installed  
15+ systems sold**

\*US Oak Ridge is Cloud delivery

Global Deliveries

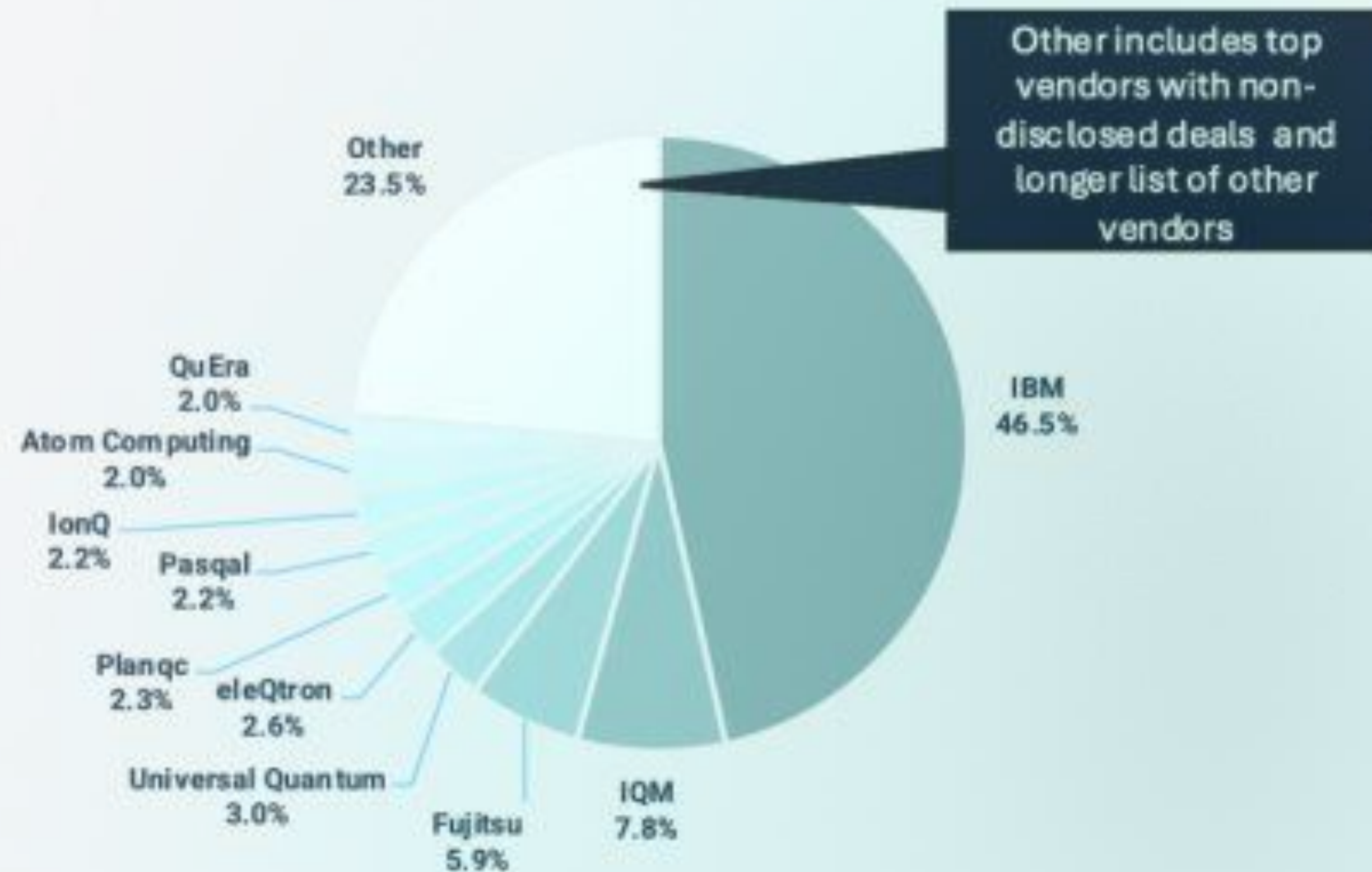
IQM

# IQM leads in number of systems installed

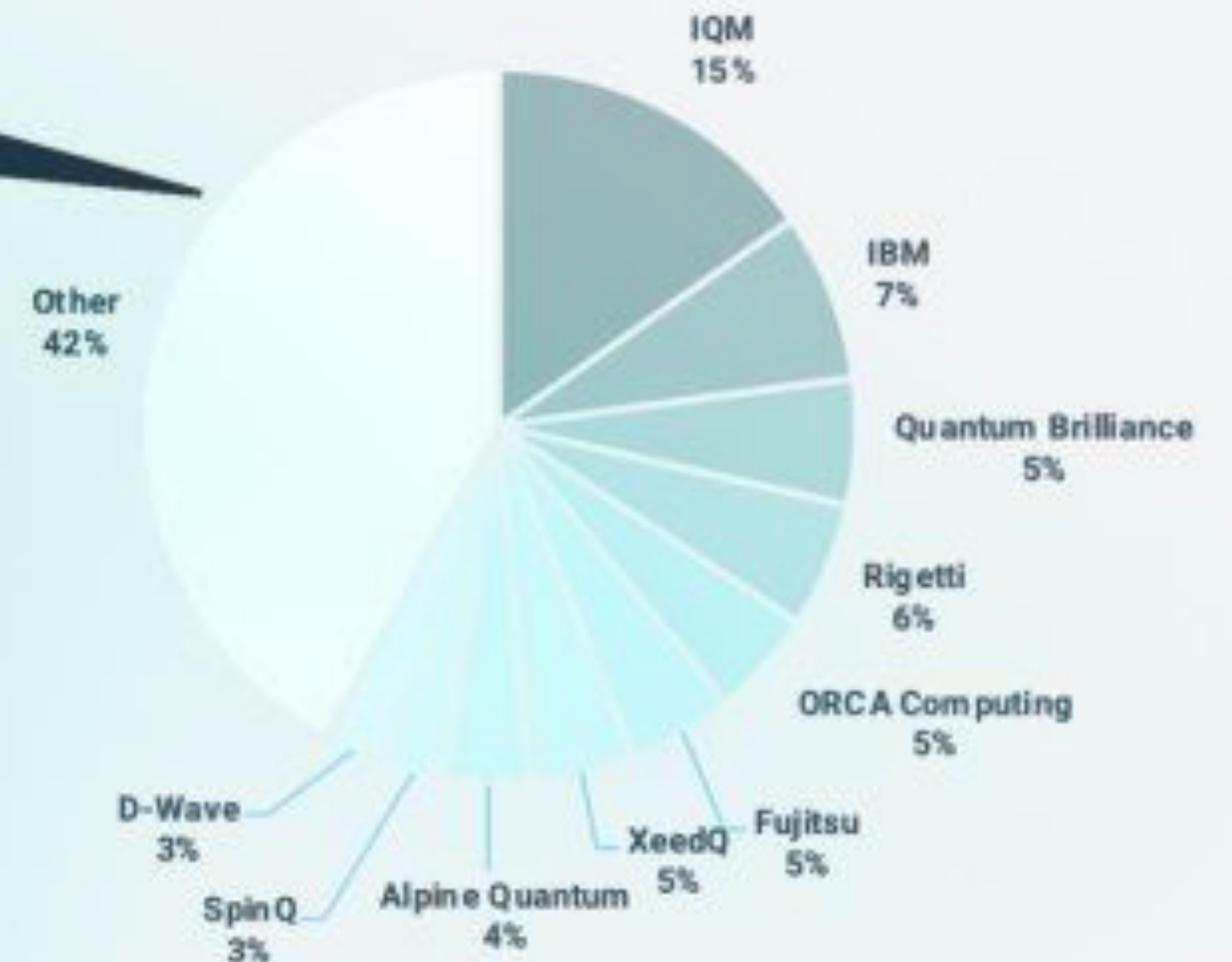
Data published by "The Quantum Insider"

IQM × **QRISP**

Top-10 vendors by share of total deals amount, 2020 to June 2025




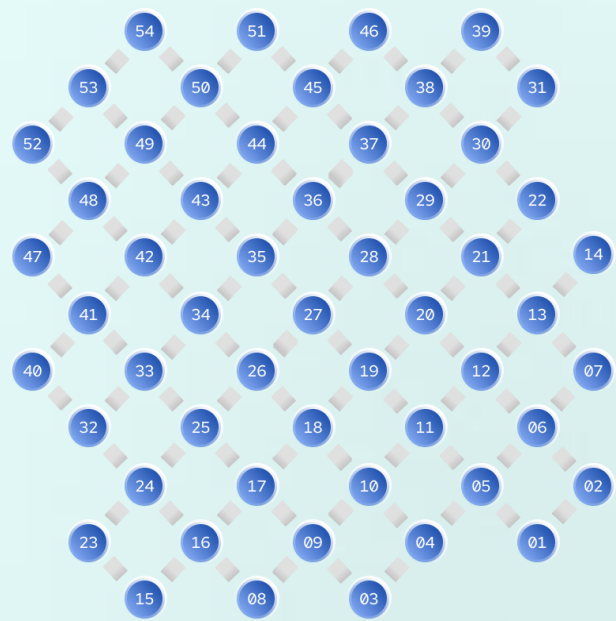
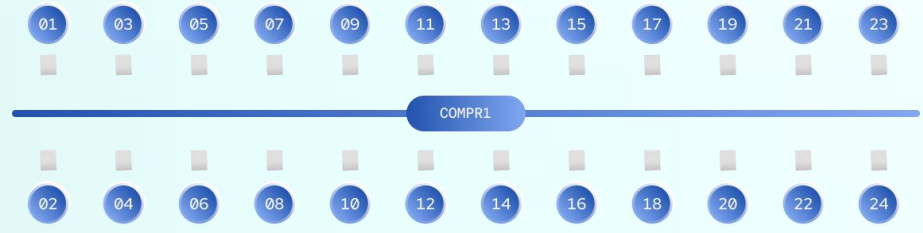
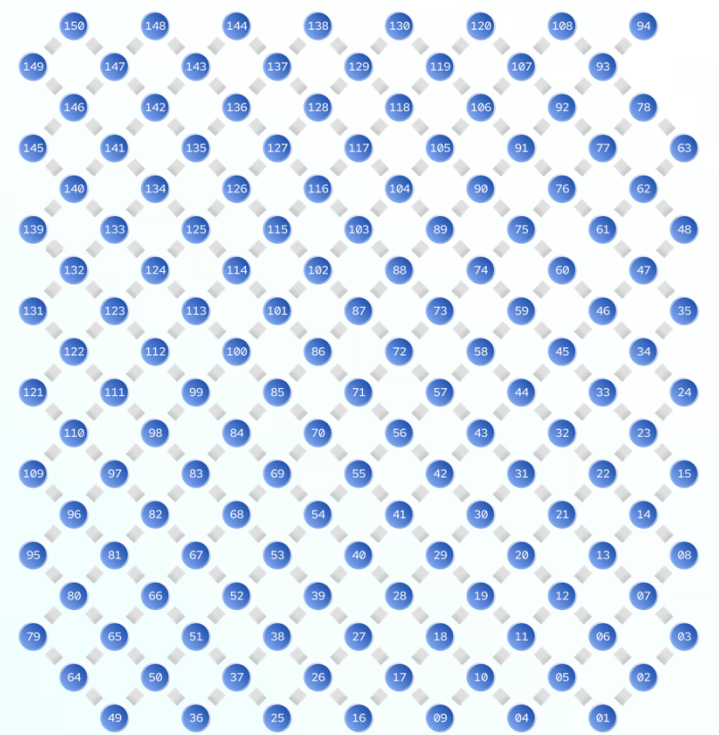
Top-10 vendors by share of total systems sold, 2020 to June 2025



# Our road to a superior quantum hardware platform



## IQM Hardware Roadmap

	<b>IQM Garnet</b> (Crystal 20)	<b>IQM Crystal 54</b>	<b>IQM Star 24</b>	<b>IQM Crystal 150</b>
<b>Qubit Count</b>	<b>20</b> computational qubits <b>30</b> coupler qubits	<b>54</b> computational qubits <b>90</b> coupler qubits	<b>24</b> qubits <b>1</b> computational resonator	<b>150</b> computational qubits <b>266</b> coupler qubits
				
<b>Availability</b>	Available ✓	Available ✓	Available ✓	H1 2026 ✓

# Quantum Algorithm Development

- **Algorithm development via circuit construction is slow, unstructured and can not be modularized!**
- More **systematic** quantum software development will be an important part in achieving Quantum Advantage for many fields of applications.

- D-Wave – Ocean
  - NVIDIA – CuDA-Q
  - Rigetti – Forest
  - IBM – Qiskit
  - Google – Cirq
  - Microsoft – Quantum Development Kit (QDK)
  - Zapata – Orquestra
  - 1QBit – 1QBit SDK
  - Amazon – Braket SDK
  - ETH Zurich – ProjectQ
  - Xanadu – PennyLane
  - Riverlane – Anian
- and more

## Abstraction Level

- Logical Gates
- Qubits

## Physical Level for Superconducting Qubits

- Microwave Pulses
- Accessible on IQM through Pulla

# Single Qubit Gates

What can we do with a single classical bit?

Nothing    Identity = 1    set to bit = 1  
 Bit Flip    = NOT    set to bit = 0

That's it

Qubits have more to offer

Identity :  $1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$   $|0\rangle \rightarrow |0\rangle$   
 $|1\rangle \rightarrow |1\rangle$

Bit-flip NOT :  $X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$   $|0\rangle \rightarrow |1\rangle$   
 $|1\rangle \rightarrow |0\rangle$

Bit-phase-flip :  $Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$   $|0\rangle \rightarrow i|1\rangle$   
 $|1\rangle \rightarrow -i|0\rangle$

Phase-flip:  $Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$   $|0\rangle \rightarrow |0\rangle$   
 $|1\rangle \rightarrow -|1\rangle$

Pauli  
Operators

Hadamard :  $H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$   $|0\rangle \rightarrow \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle)$   
 $|1\rangle \rightarrow \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)$   $H = \frac{1}{\sqrt{2}} (X + Z)$

Hadamard creates an equal superposition

Phase-shift:  $P_\phi = \begin{bmatrix} 1 & 0 \\ 0 & e^{-i\phi} \end{bmatrix}$   $|0\rangle \rightarrow |0\rangle$   
 $|1\rangle \rightarrow e^{-i\phi} |1\rangle$

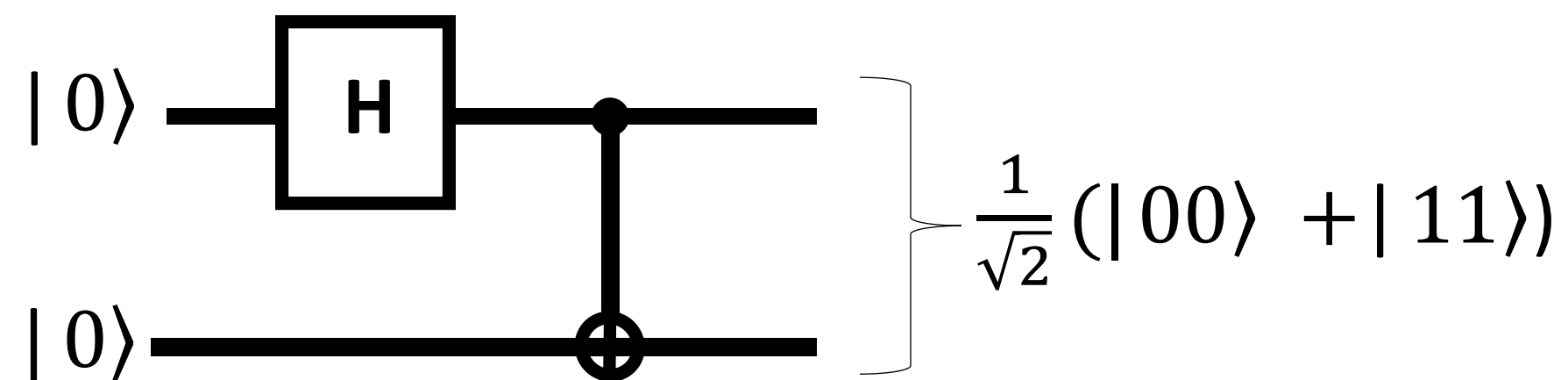
IQM × **QRISP**

# How to become Entangled?

We need the Controlled NOT action on 2 Qubits

$$\text{C-NOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

If flips the 2<sup>nd</sup> Qubit (**Target**) if the 1<sup>st</sup> Qubit (**Control**) is  $|1\rangle$   
 If does nothing if the 1<sup>st</sup> Qubit (**Control**) is  $|0\rangle$



In the 1930s when scientists, including Albert Einstein and Erwin Schrödinger, first discovered the phenomenon of entanglement, they were perplexed. Entanglement, disturbingly, required two separated particles to remain connected without being in direct contact.

Einstein famously called entanglement "spooky action at a distance (spukhafte Fernwirkung)" since the particles seemed to be communicating faster than the speed of light.

$$|0\rangle |0\rangle \xrightarrow{H} \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) |0\rangle \xrightarrow{\text{C-NOT}} \frac{1}{\sqrt{2}} (|0\rangle |0\rangle + |1\rangle |1\rangle)$$

**No Hidden Variables**  
**Shared Randomness**

# Introducing Qrisp

IQM × **qrisp**



- Qrisp is a fully **compilable, high-level** programming framework.
- Central building block is the **QuantumVariable**.
- **Significantly enhances development aspects** like prototyping, code size, maintainability, bug-fixing/testing, modularity, readability, refactoring etc.

# What is Qrisp ?

- **Structured Programming for Quantum Algorithms.**
  - Simplifies development & maintenance, easy scalable
- **Hardware Agnostic.**
  - Qrisp is directly compatible with devices manufactured by IQM, IBM, eleQtron and AQT.
- **Integrated Quantum State Simulator**
  - Powerful tool for testing and debugging
- **Developed by Fraunhofer FOKUS**
- **Open Source**
  - Democratizing access to quantum computing.

[www.qrisp.eu](http://www.qrisp.eu), Setup: `pip install qrisp`

# Qrisp core elements

- **QuantumVariable**

- Can be of Quantum Types: QuantumFloat, QuantumBool, QuantumModulus, QuantumChar
- Users can create custom quantum types by inheriting from QuantumVariable specifying which values should be represented and how they are encoded. i.e QuantumColor

- **QuantumSession**

- Each QuantumVariable is registered in exactly one QuantumSession. The QuantumSession manages the lifetime cycle of the QuantumVariables

- **QuantumArray**

- Multiple QuantumVariables can be handled within a QuantumArray

- **QuantumDictionary**

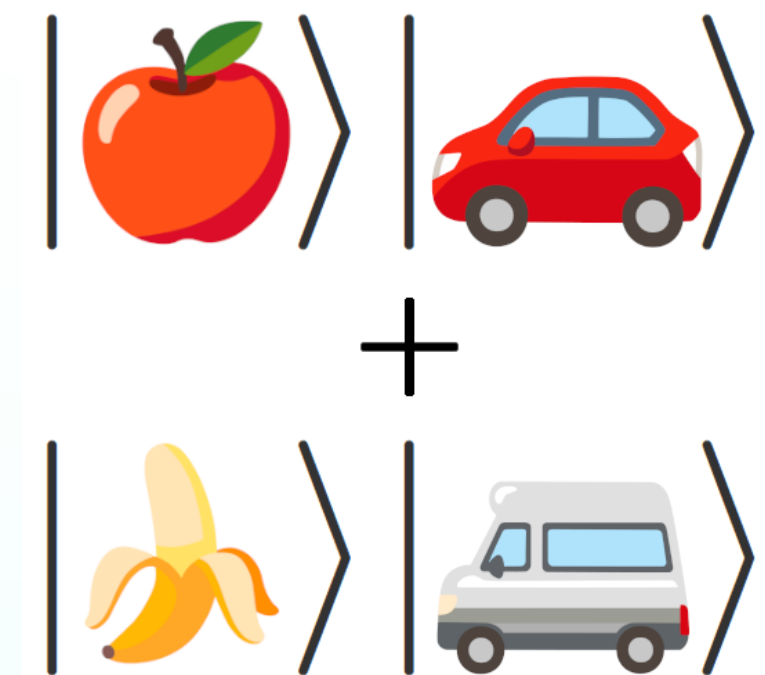
- extends a classical python dictionary, by allowing QuantumVariables as keys

- **Uncomputation**

- auto\_uncompute or manual uncompute to free resources

- **Session Merging**

- Automatically by entangling operation, manually with merge function



# QuantumFloat

- **class QuantumFloat(msize, exponent=0, qs=None, name=None, signed=False)**

```
>>>> from qrisp import QuantumFloat
```

```
>>>> a = QuantumFloat(3, -1, signed = False)
```

3 indicates the amount of mantissa qubits and the -1 indicates the exponent

$$f_k(i) = i2^k$$

```
>>>> for i in range(2**a.size): print(a.decoder(i))
```

0.0

0.5

1.0

1.5

2.0

2.5

3.0

3.5

# QuantumFloat signed

- **class QuantumFloat(msize, exponent=0, qs=None, name=None, signed=True)**

```
>>>> from qrisp import QuantumFloat
>>>> b= QuantumFloat(2, -1, signed = True)
```

2 indicates the amount of mantissa qubits and the -2 indicates the exponent.  
Here we have  $2^2=4$  values and their signed equivalents. The precision is  $0.25 = 2^{-2}$

$$f_k^n(i) = \begin{cases} i2^k & \text{if } i < 2^n \\ (i - 2^{n+1})2^k & \text{else} \end{cases}$$

```
>>>> for i in range(2**b.size): print(b.decoder(i))
```

0.0  
0.25  
0.5  
0.75  
-1.0  
-0.75  
-0.5  
-0.25

```
>>> from qrisp import q_div,
>>> qf_inversion
>>> a = QuantumFloat(3)
>>> a[:] = 1
>>> b = QuantumFloat(3)
>>> b[:] = 7
>>> c = q_div(a, b, prec = 6)
>>> print(c)
{0.140625: 1.0}
```

```
>>> 1/7 - 0.140625 = 0.002232...
Result is less than  $2^{-6}$ 
```

# Code Example

- Simple addition

```
%%capture
!pip install qrisp[iqm]
```

```
from qrisp import *
```

```
a = QuantumFloat(4)
a[:] = 2
b = a+a
b.get_measurement()
```

OUT: {4: 1.0}

From the simulator with 100%

- On a real machine

```
%%capture
!pip install qrisp[iqm]
```

```
from qrisp.interface import IQMBackend
token = "YOUR_TOKEN"
iqm_emerald = IQMBackend(api_token = token,
                        device_instance = "emerald:timeslot")
```

```
a = QuantumFloat(4)
a[:] = 2
b = a + a
b.get_measurement(backend = iqm_emerald)
```

OUT: {4: 0.554, 3: 0.101, 0: 0.073, 1: 0.07, 6: 0.055,  
10: 0.045, 4: 0.027, 7: 0.015, 5: 0.012, 11: 0.011,  
12: 0.009, 14: 0.009, 15: 0.007, 8: 0.006, 9: 0.003, 13: 0.003}

More shots: b.get\_measurement(backend = iqm\_emerald, shots=2000)

OUT: {4: 0.6365, 0: 0.066, 3: 0.0595, 10: 0.047, 1: 0.035, 6: 0.033,  
11: 0.023, 2: 0.018, 12: 0.0165, 5: 0.013, 7: 0.0125, 8: 0.012,  
14: 0.0095, 9: 0.0075, 13: 0.0055, 15: 0.0055}

# Code Example Quantum Loops

```
from qrisp import QuantumFloat, qRange, h
# Create some QuantumFloats
n = QuantumFloat(3)
qf = QuantumFloat(5)
# Initialize the value 4
n[:] = 4
# Apply H-gate to 0-th qubit
h(n[0])
print(n) # Yields {4.0: 0.5, 5.0: 0.5}
# Perform successive addition of increasing numbers
for i in qRange(n):
    qf += i
print(qf) # Yields {10.0: 0.5, 15.0: 0.5}
# Agrees with the expectation of  $qf = n*(n+1)/2$ 
```

# Code Comparison



```
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]
```

# Next-gen SDK Qrisp as default



Potential  
Developer Market  
Share Migration  
**>70%**

Based on early usage metrics  
and developer feedback

## Easy to learn and super quick to develop

50x more compact and 10x better performance outcomes, even as systems grow

## Powerful features for the future

HPC Integration, Pulse level access, Hybrid runtime, error mitigation integration and designed for error-correction era

## Open & Integrated

Cloud access, open source, real-world ready

**10 times fewer** entangling Gates and **1.000.000.000.000 faster resource** estimation for simulation of Beryllium Hydride compared to Qiskit.

Exponentially **faster** arithmetic through carry-lookahead adders

**3 times faster** quantum circuit simulation than Qiskit to facilitate development workflow and testing.

**10 times faster** quantum compilation for amplitude amplification problems compared to PennyLane or Qiskit

World's most efficient quantum implementation available with ~2% of code base vs Qiskit



11

50x Productivity Gain

500

Lines of code required for the implementation of Shor's algorithm

Source: Public Information, GitHub

# Curios? Try it out for free!

- **30 free credits per month** to use on as many jobs as you need
- Access to IQM's **unique** quantum computer **topologies** and learning resources
- Limited-time access to our **flagship Crystal 54** device for all Starter users

→ [Sign up now](#)

→ <https://resonance.meetiqm.com/sign-up>

- **IQM Academy**  
Your one-stop shop for online quantum education  
Including Qrisp

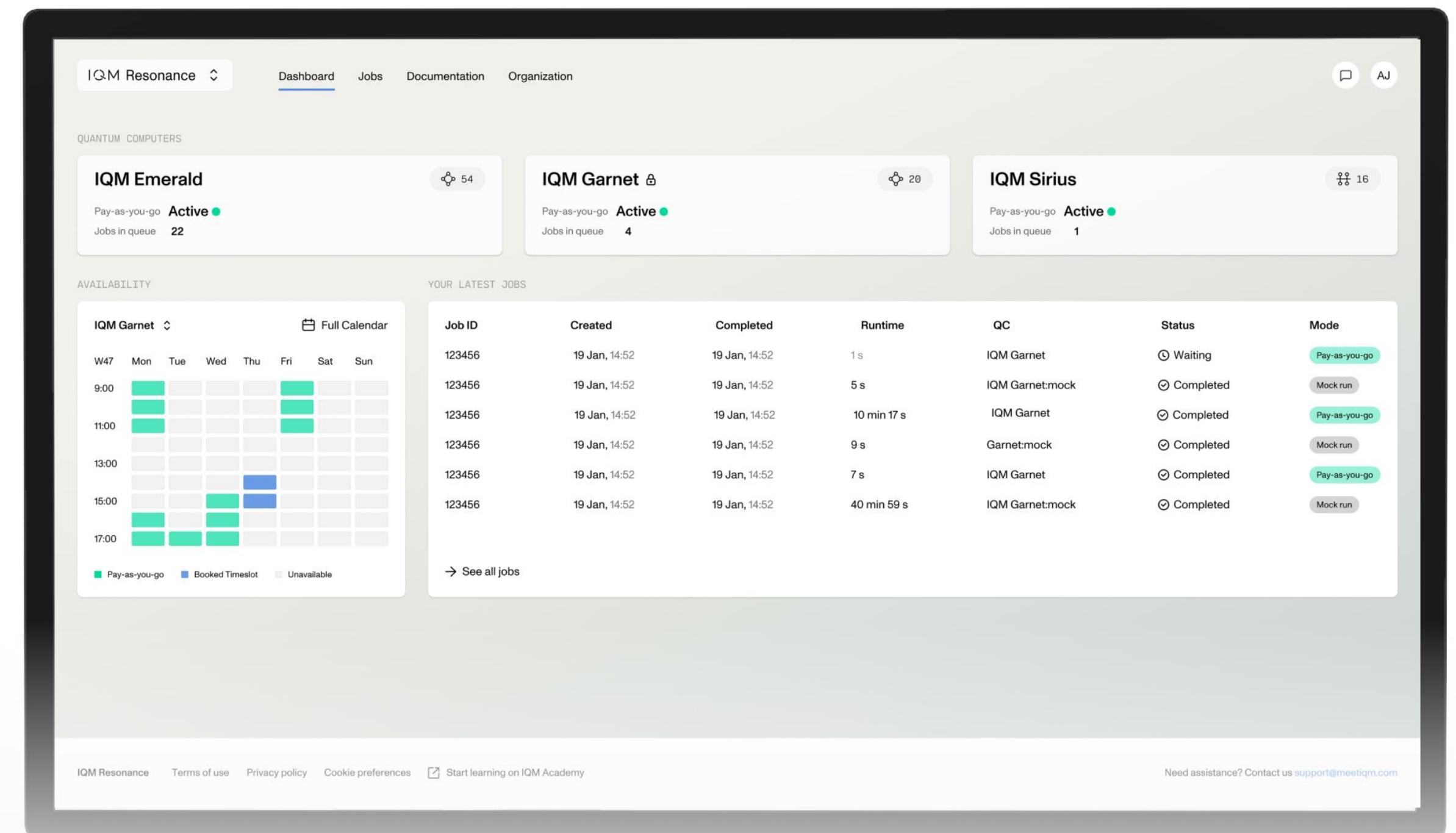
→ <https://academy.meetiqm.com/>

## A major upgrade to IQM Resonance

Access to our 54-qubit Crystal system, pulse-level access, advanced error handling, upgrades to IQM Academy, a new QAOA library, and more.

Free sign up →

Experience Resonance →



# Let's start your quantum journey, together

---

**Ulrich Meier**

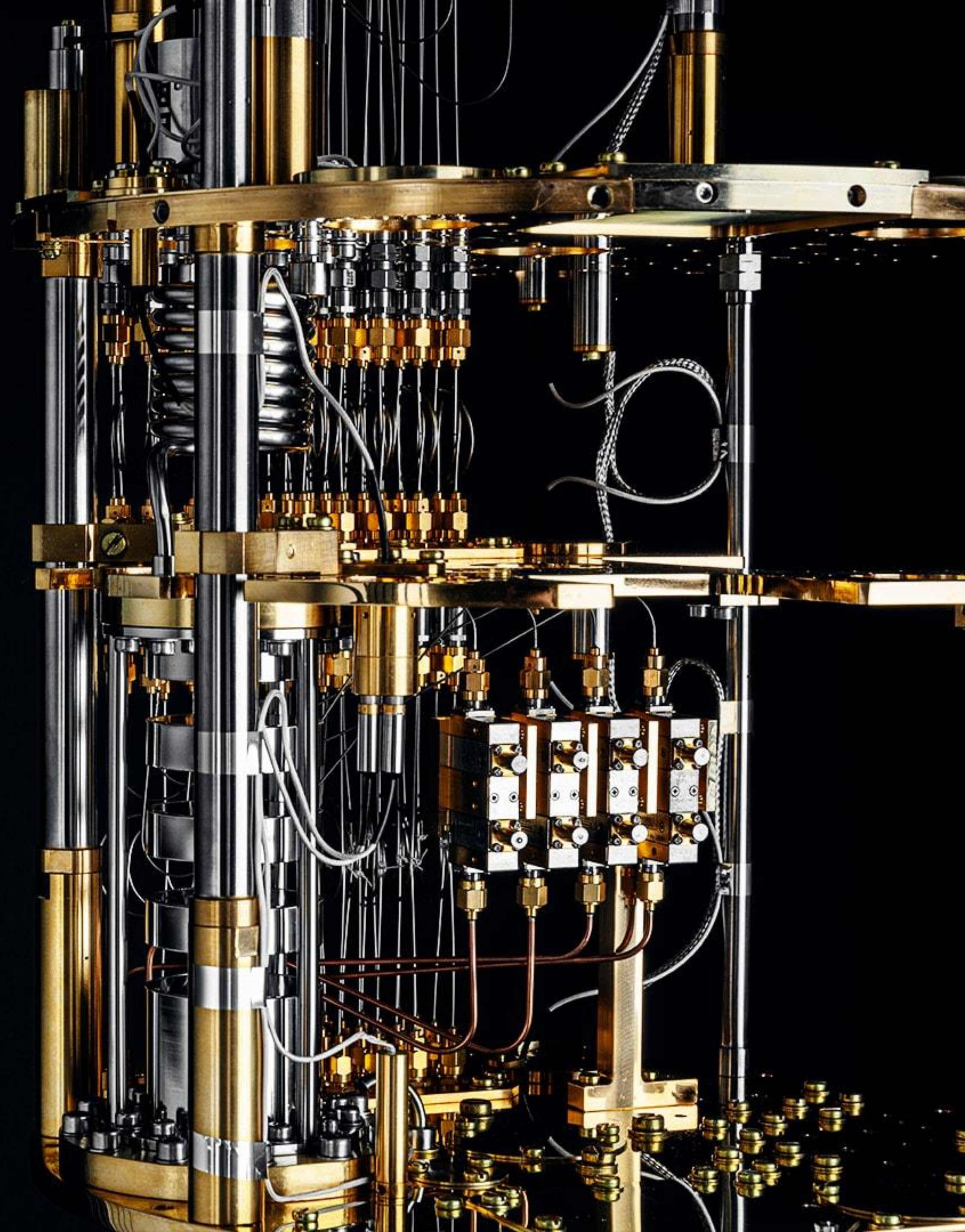
+49 160 98191171

[ulrich.meier@meetiqm.com](mailto:ulrich.meier@meetiqm.com)

**Got questions? Just contact us!**

Follow us on [LinkedIn](#) to stay informed with our latest news.

[www.meetiqm.com](http://www.meetiqm.com)



# We have a unique combination of critical infrastructure



## Chip fabrication facility

**200 mm fabrication facility  
Optimized for QPU piloting and  
production**

- Location: Espoo, Finland
- Area: 560 m<sup>2</sup> - +200 m<sup>2</sup> expansion underway
- Next generation chip factory for error-correction roadmap planned



## Assembly line

**Full stack model benefitting from  
holistic view and control over the  
build and processes**

- Lower production costs due to value chain control
- Customers benefit from better system performance and system stability
- Capacity for up to 20 systems per year



## Quantum Data Center

**Safe and secure  
environment for hosting  
quantum cloud**

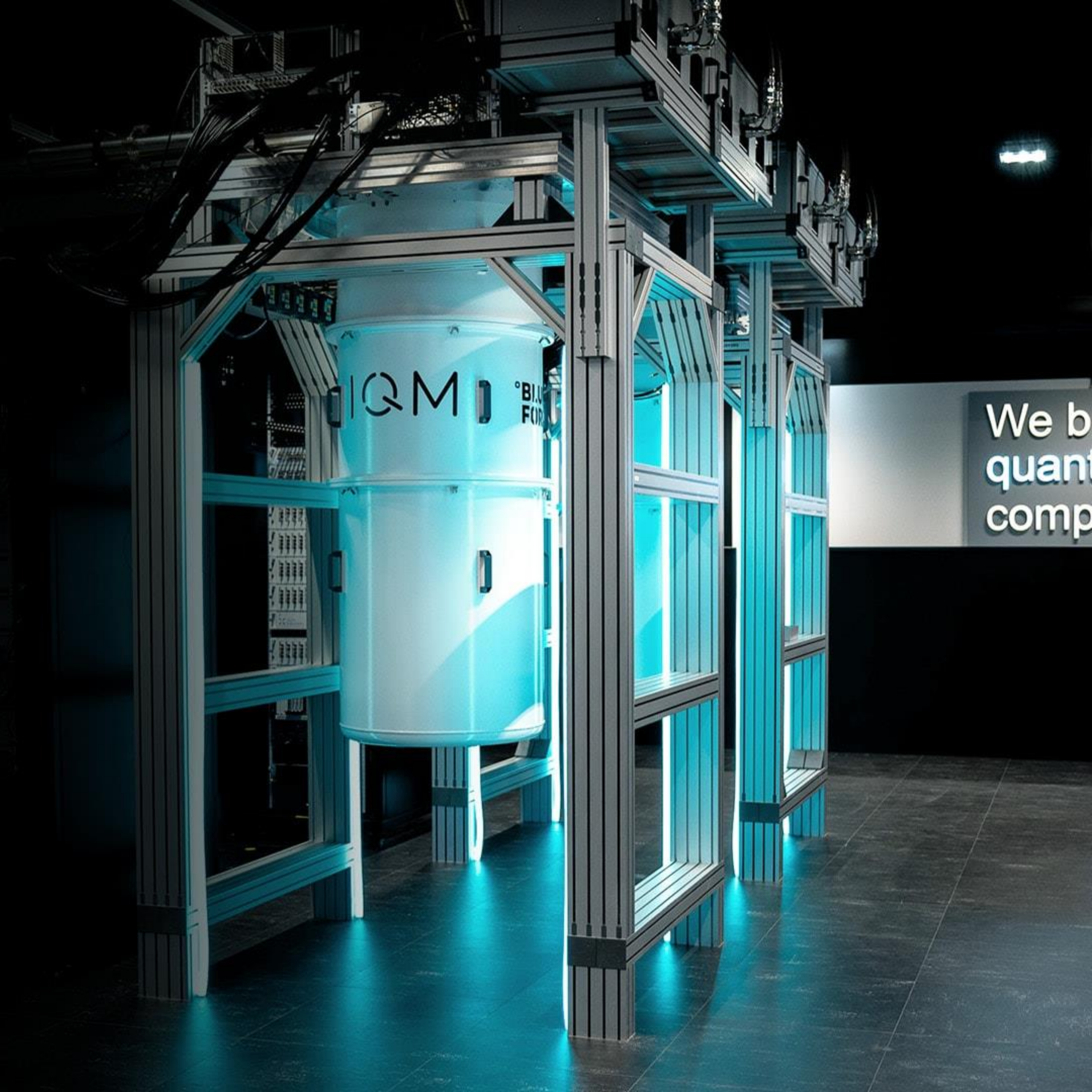
- Location: Munich, Germany
- Hosting IQM's quantum cloud - Resonance™
- Further investment plans to expand capabilities

# State-of-the-art operations site for IQM Quantum Cloud

Full and secure operations sites in  
Finland and Germany

Full-service support by IQM

IQM



# — The world's largest assembly line, designed to deliver on-premises quantum computers

Capacity to build 20 full-stack quantum computers per year

