

Kawasaki Quantum Summer Camp

量子ゲート基礎 QiskitBlocks

沼田 祈史

Kifumi Numata
IBM Quantum



Day 1

13:00-13:30 Welcome

13:30-14:25 量子コンピューティング入門

14:25-15:35 Break

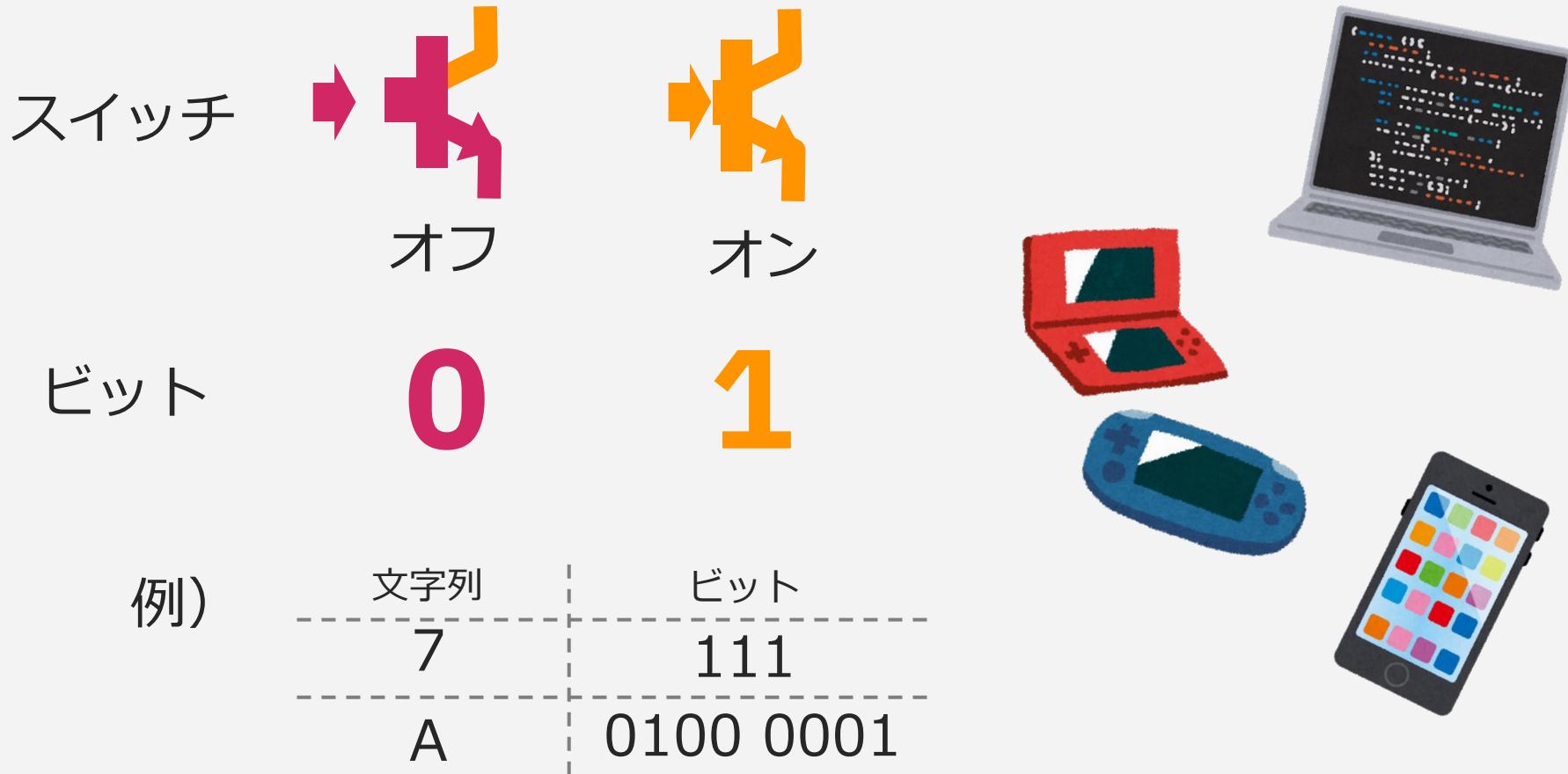
14:35-15:20 量子ゲーム - QiskitBlocks

15:20-16:10 量子ゲート - IBM Quantum Composer

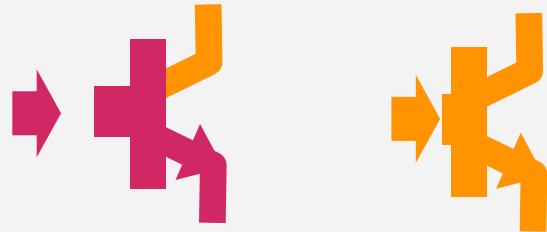
16:10-16:20 Break

16:20-16:50 量子テレポーテーション

コンピューターの中は、ビットで計算



いつも使っている コンピューターのビット



0 または 1

どちらか

いつも使っている
コンピューターのビット

0 または **1**

どちらか

量子コンピューターの
量子ビット

0 と **1**

両方

量子重ね合わせ

いつも使っている
コンピューターのビット

0 または **1**

どちらか

コイン

表

おもて



量子コンピューターの
量子ビット

0 と **1**

両方

コイン

裏

うら



いつも使っている
コンピューターのビット

0 または 1

どちらか

コイン

表

おもて



コイン

裏

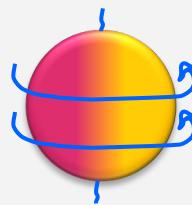
うら

量子コンピューターの
量子ビット

0 と 1

両方

くるくる回っているコイン（イメージ）



測定すると表か裏にバシッと決まる

いつも使っている
コンピューターのビット

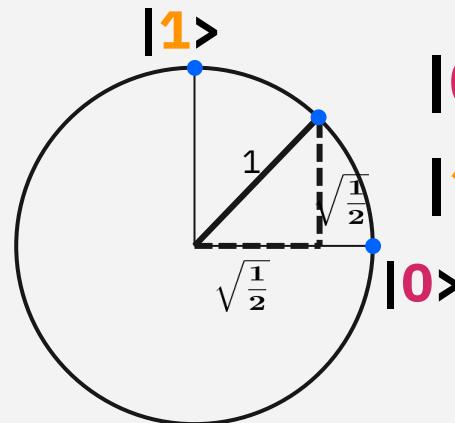
0 または 1

どちらか

量子コンピューターの
量子ビット

$\alpha \times |0\rangle + \beta \times |1\rangle$

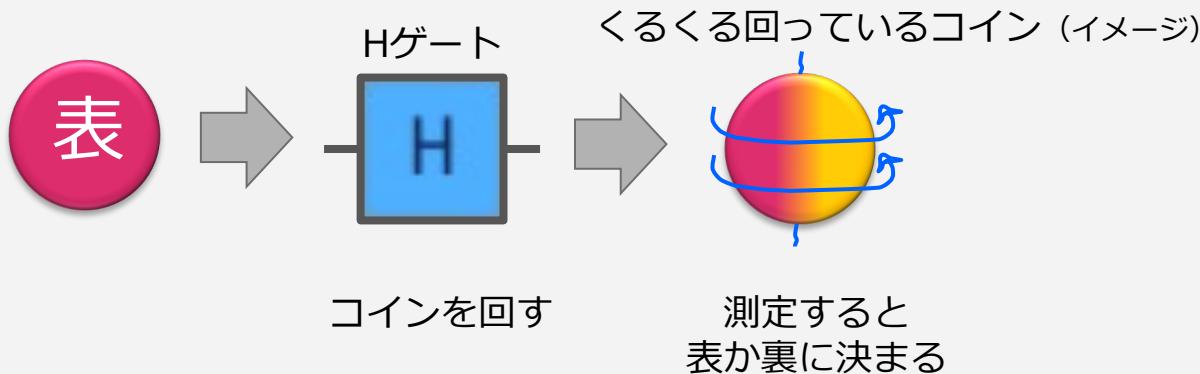
0 と 1 の「重ね合わせ」



$$|0\rangle = 1 \times |0\rangle + 0 \times |1\rangle$$

$$|1\rangle = 0 \times |0\rangle + 1 \times |1\rangle$$

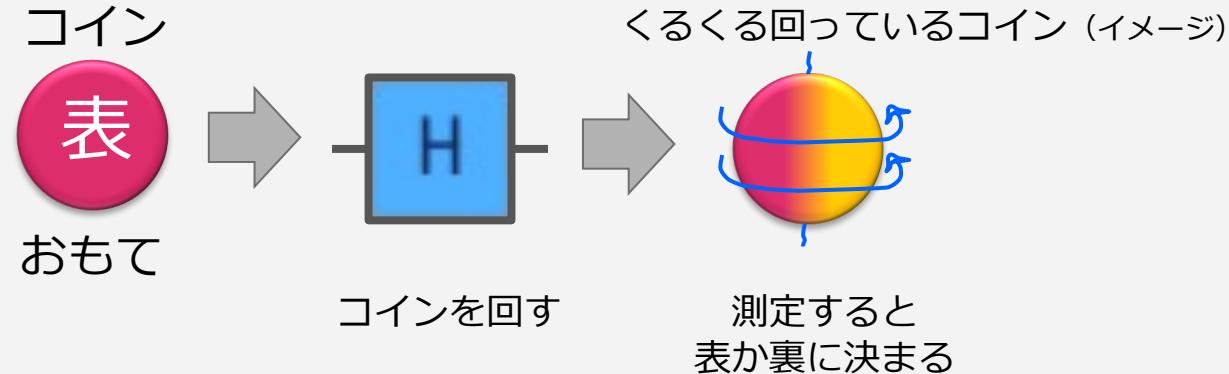
量子コンピューターの計算方法



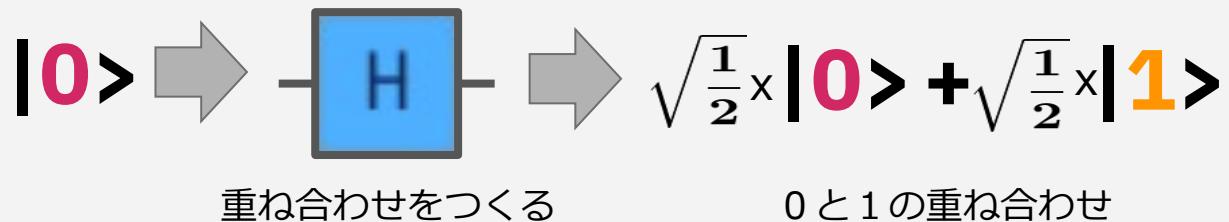
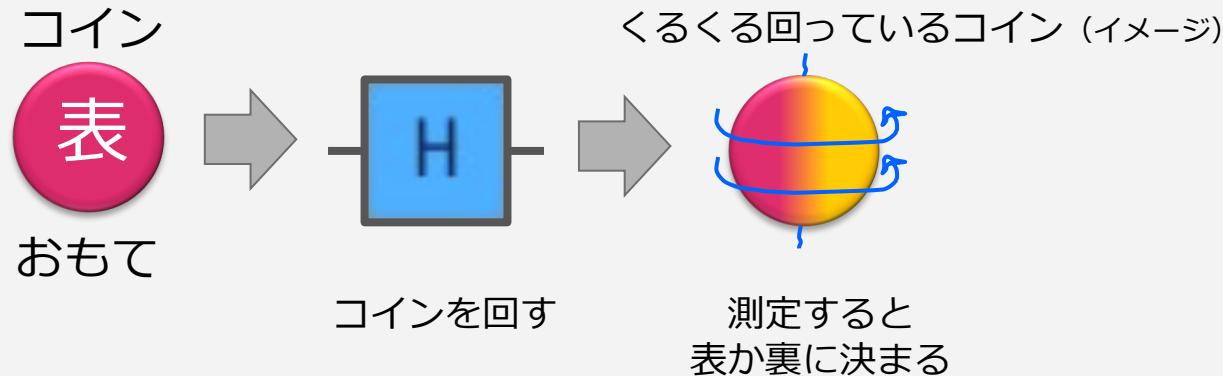
Xゲート



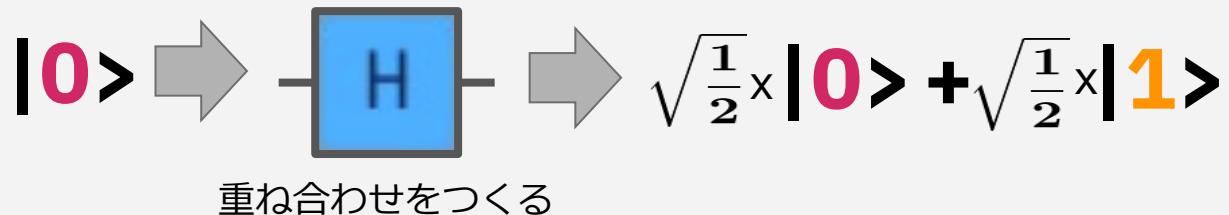
Hゲート



Hゲート



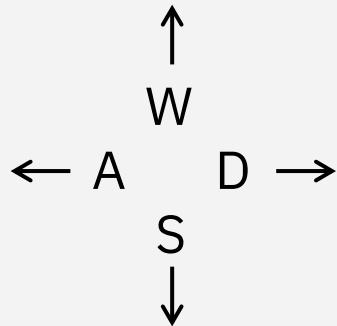
量子コンピューターの計算方法



量子ゲームQiskitBlocksで遊びます

<https://github.com/JavaFXpert/QiskitBlocks>
http://ibm.biz/qblocks_j19

ゲーム製作：
ゲームス・ウィーバーさん



Qiita ブログ コミュニティ Q キーワードを入力 ストッパー

15 LGTM 0 574 views @kifumi 2019年12月16日に更新

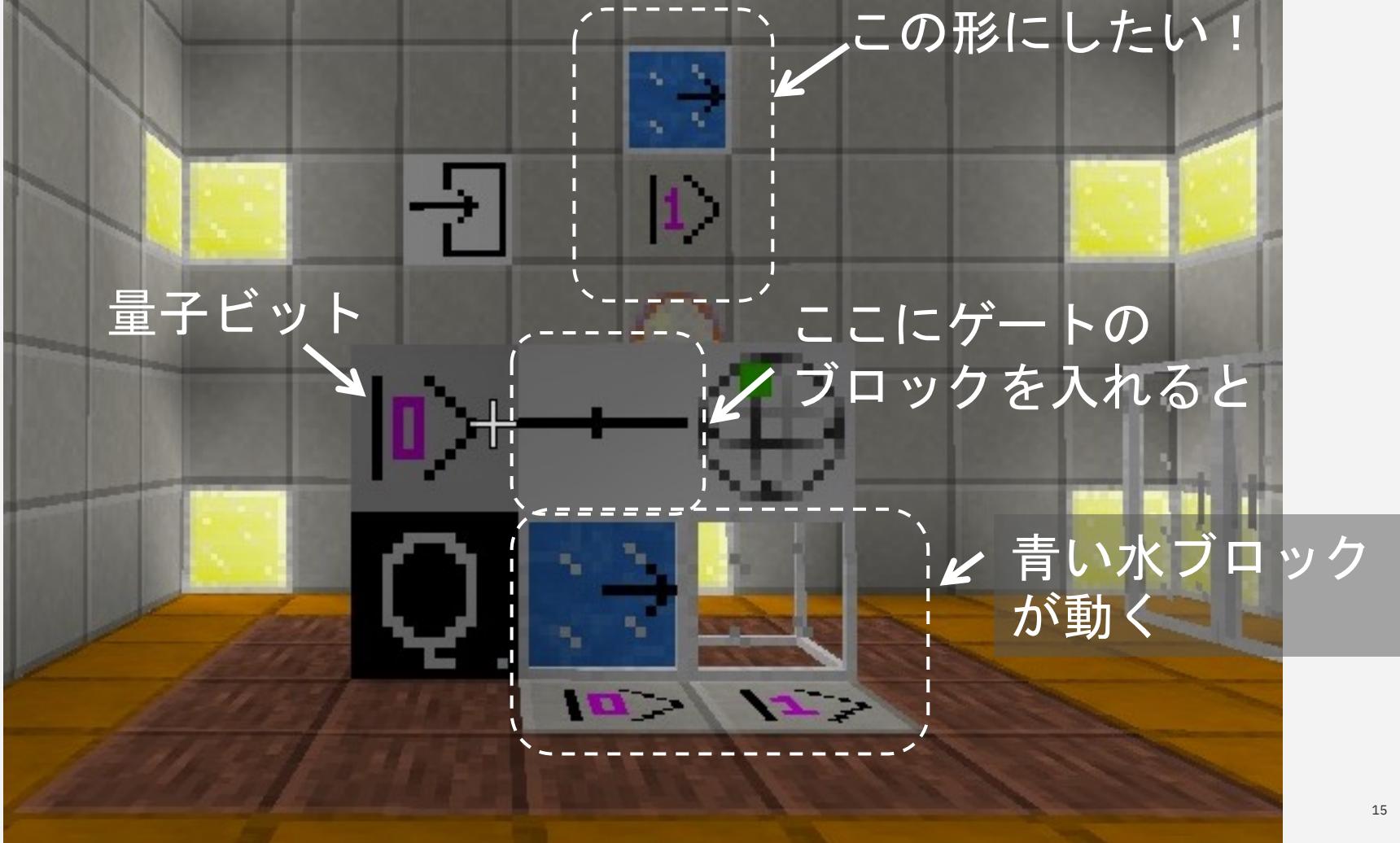
量子コンピューター Advent Calendar 2019 | 13日目

量子ゲーム QiskitBlocksの遊び方

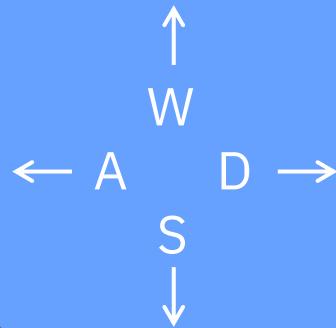
QISKIT IBMQ

量子コンピューターを学習するゲーム QiskitBlocks (<https://github.com/JavaFXpert/QiskitBlocks>) を紹介します。
ぱっと見、マイクラがないと使えないのでは？と思われる方もいるかもしれませんが、オープンソースの Minetest (<http://minetest.net>) を使っているので無料で遊べますので、ぜひ遊んでみてください！

A screenshot of the QiskitBlocks game, which is a mod for the game Minetest. It shows a colorful, blocky landscape with various structures made of blocks, including what look like quantum circuit diagrams and control panels. The interface includes a compass rose and some UI elements typical of a video game.



移動のキー



量子ビット

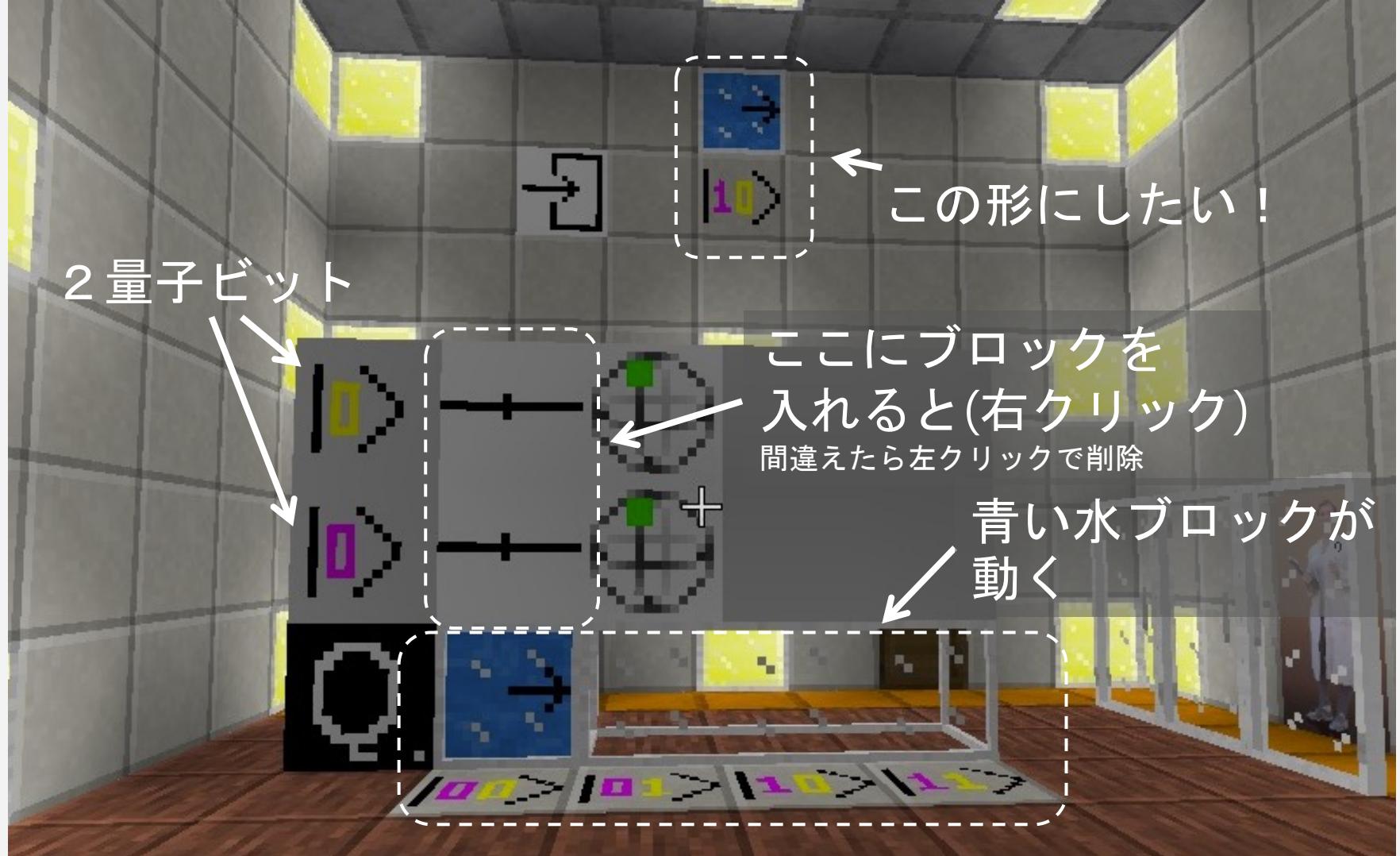
ブロックが入っている
チェスト(右クリックで開ける)

この形にしたい!

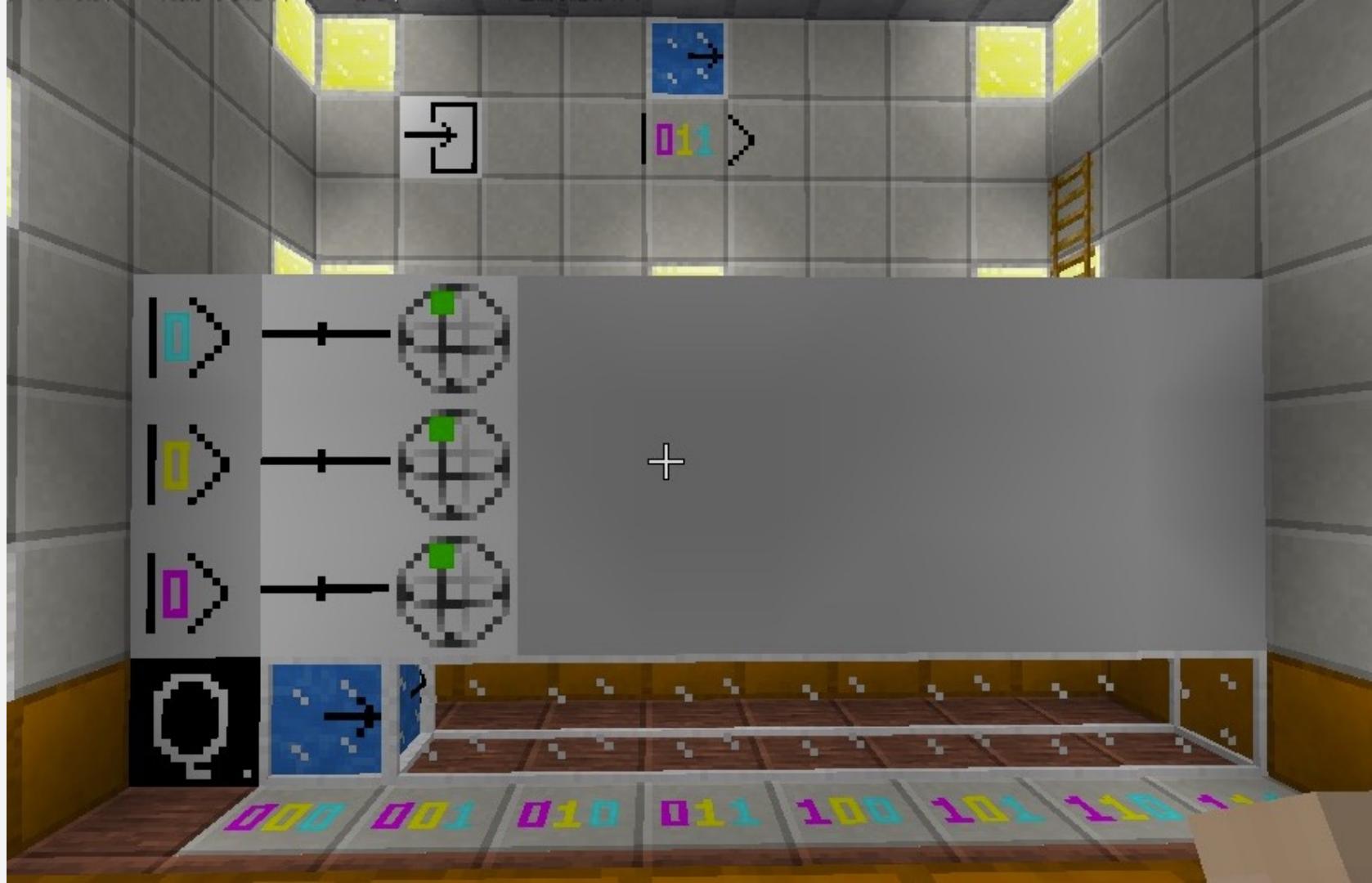
ここにゲート
ブロックを入れると
(右クリック)

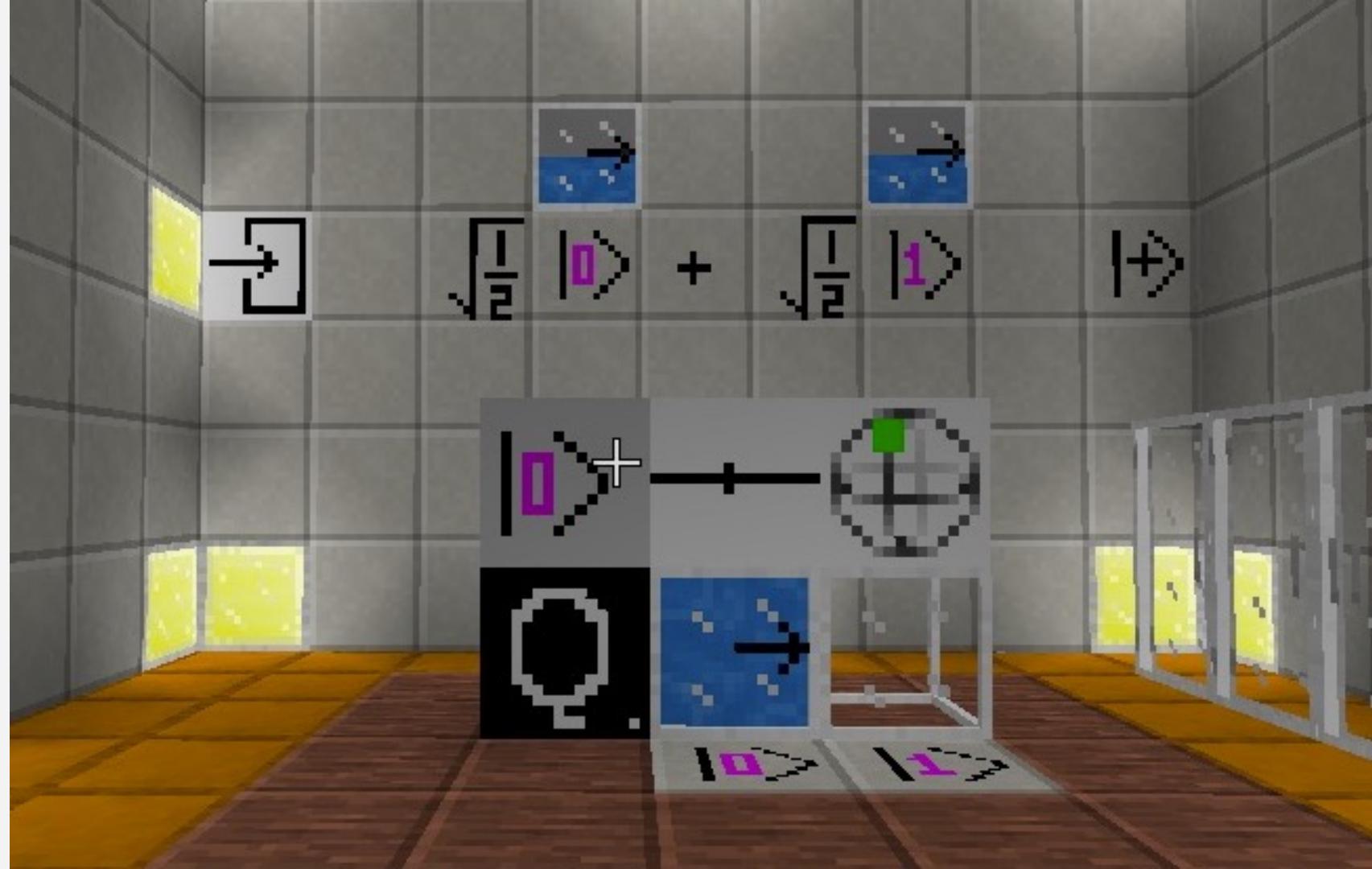
青い水ブロック
が動く

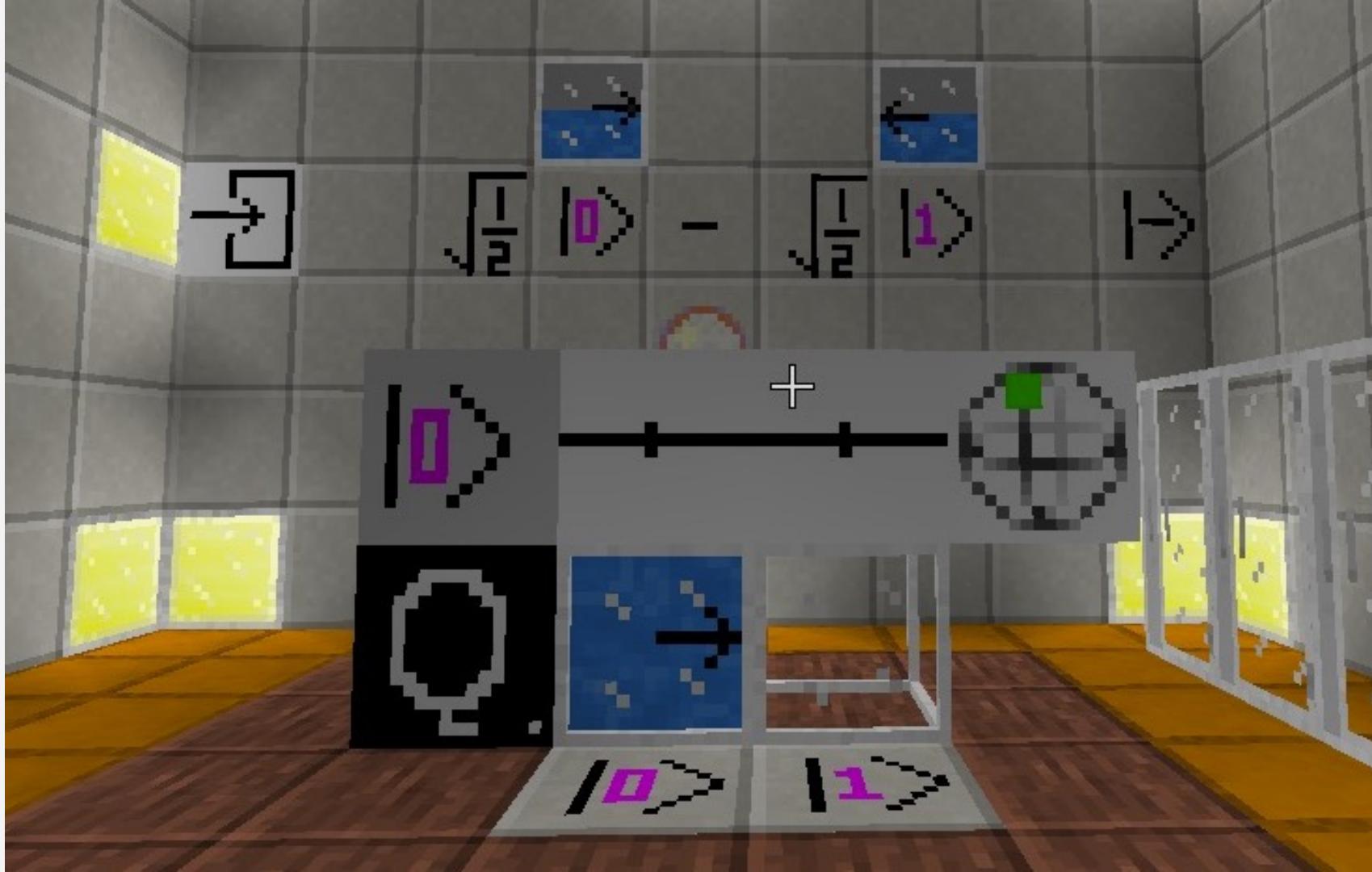




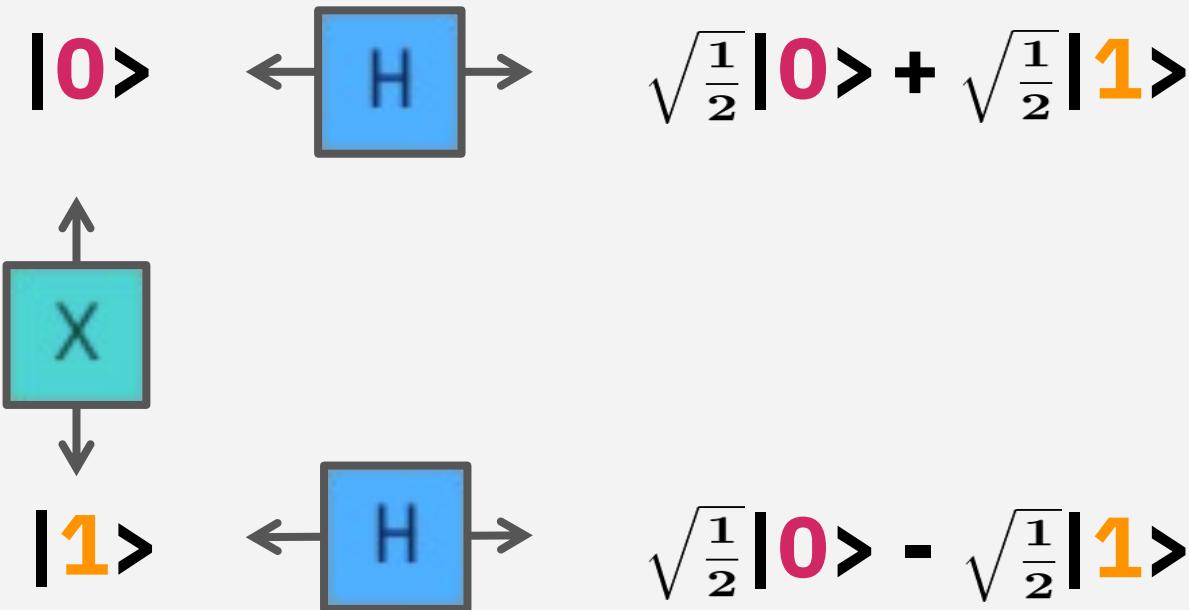
3



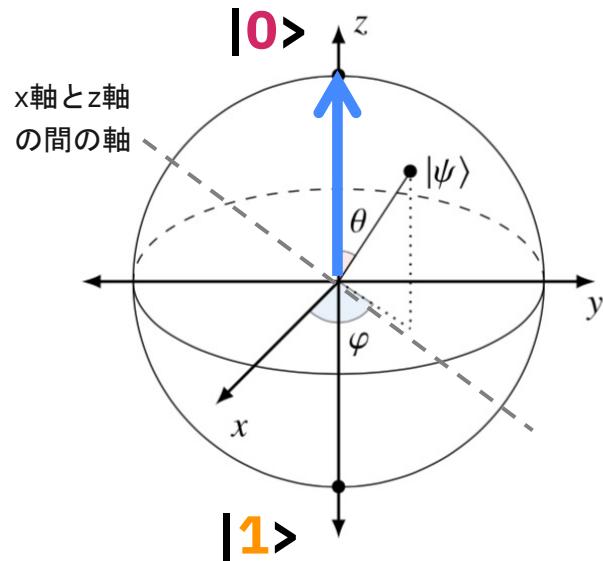




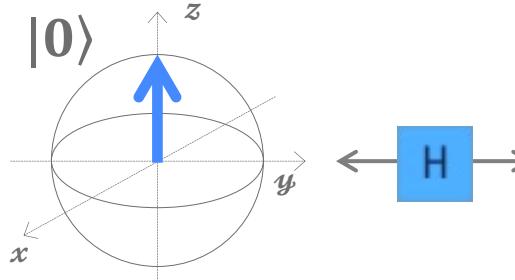
量子コンピューターの計算方法 まとめ



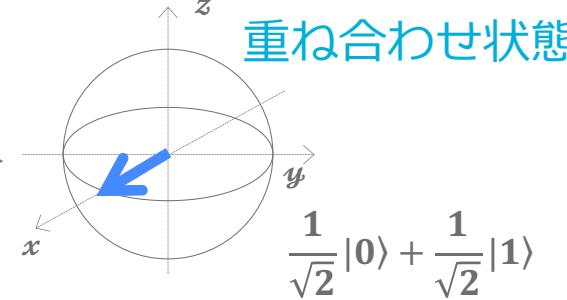
ブロツ木球



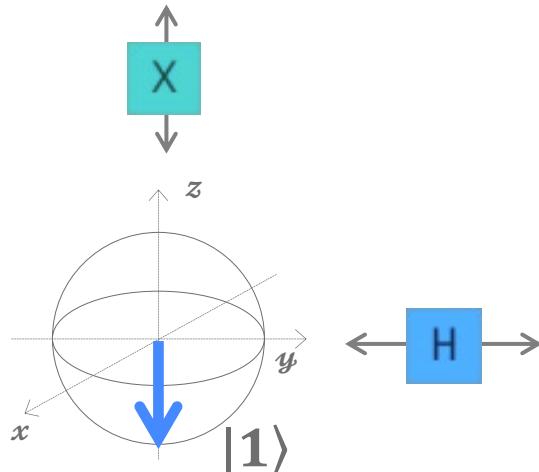
北極 : 0の確率が100%



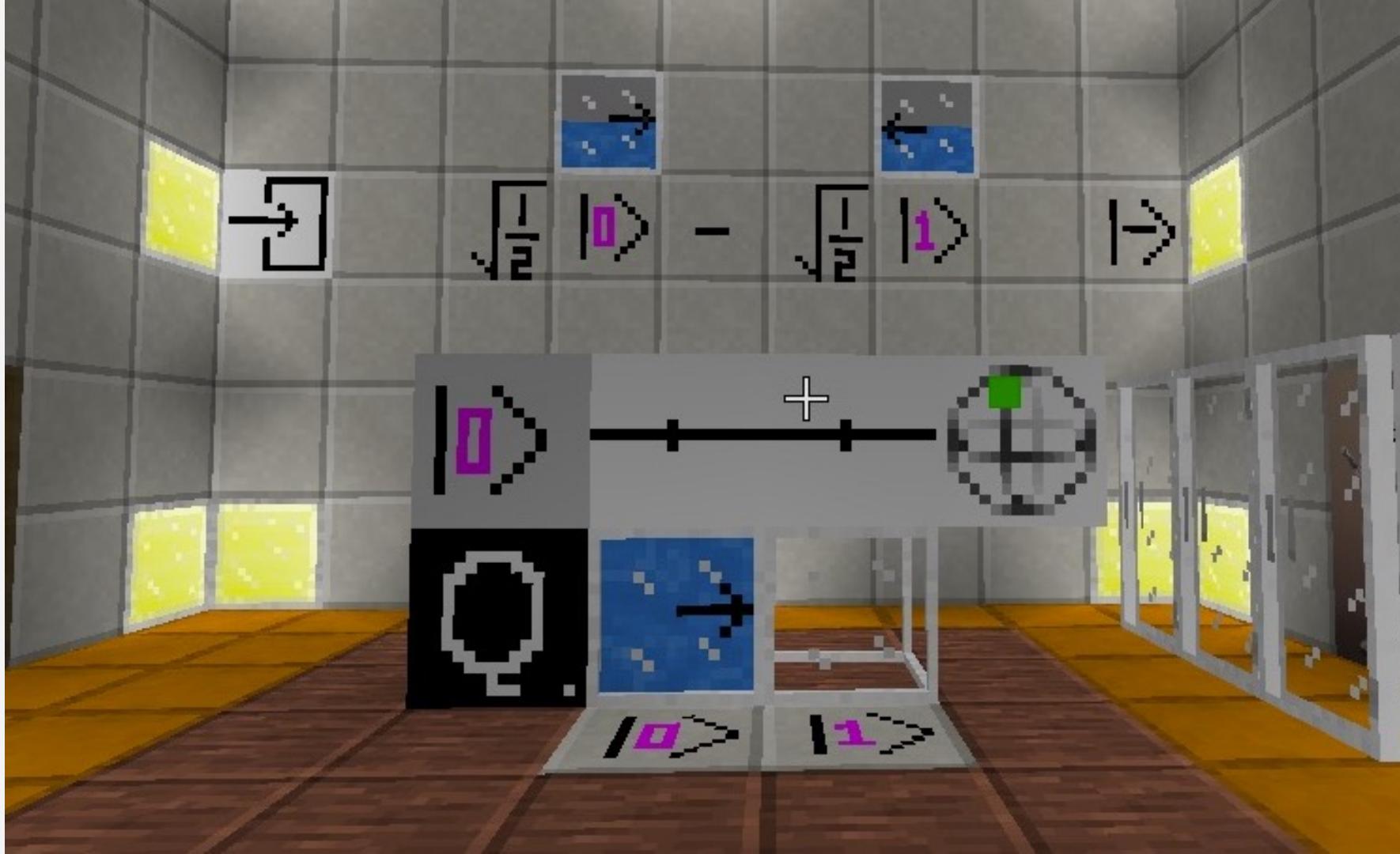
赤道 : 0と1が50%ずつの
重ね合わせ状態



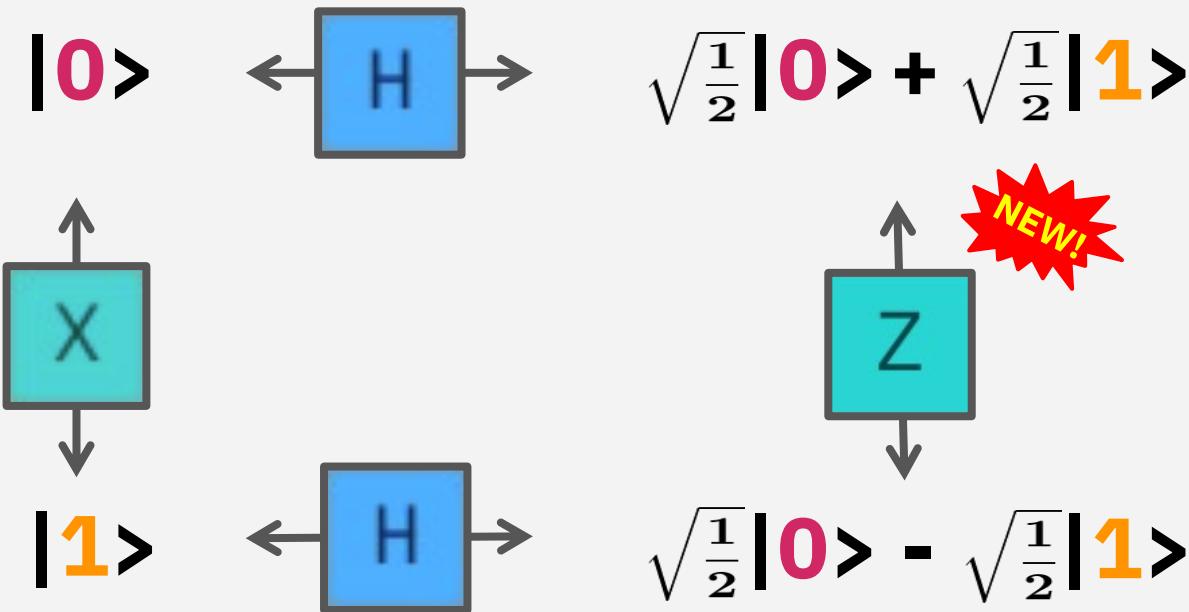
南極 : 1の確率が100%



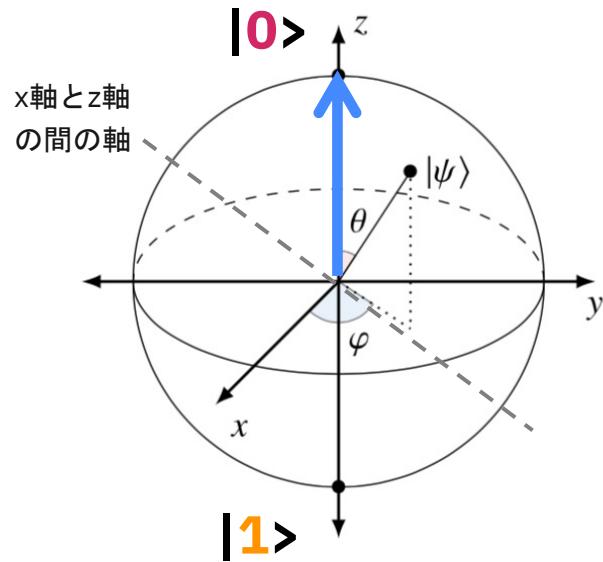
$$\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$



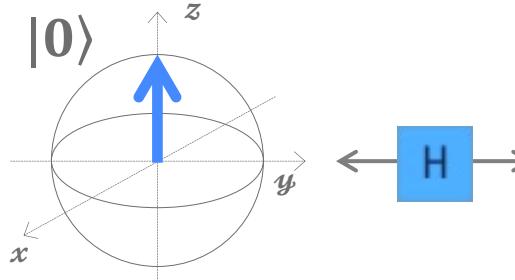
量子コンピューターの計算方法 まとめ



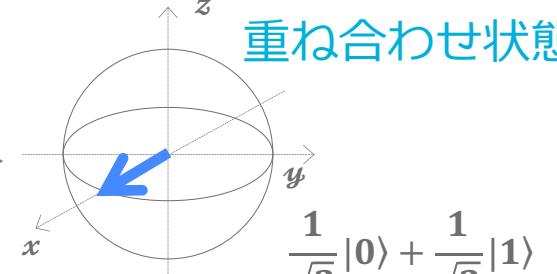
ブロツ木球



北極 : 0の確率が100%

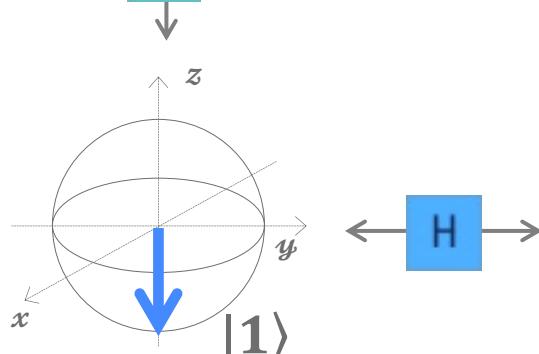


赤道 : 0と1が50%ずつの
重ね合わせ状態

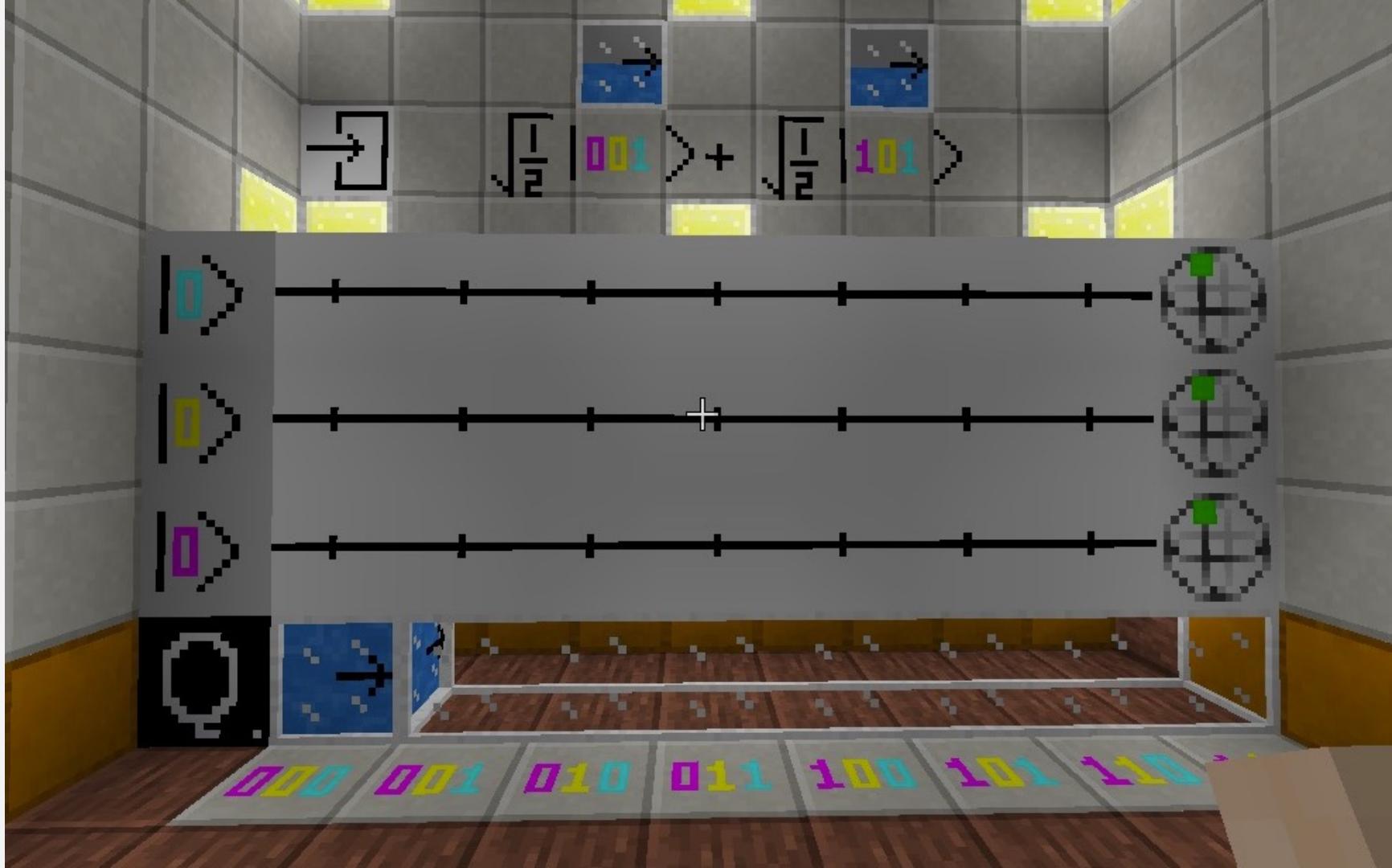


$$\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

南極 : 1の確率が100%



$$\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$





$$\sum_{i=000}^{111}$$

$$\sqrt{\frac{1}{8}}$$

|>

|>

|>

|>

+



000 001 010 011 100 101 110



\sum は、和の記号です。

例) $\sum_{i=0}^5 i = 0 + 1 + 2 + 3 + 4 + 5$

今回の問題は、

$$\sum_{i=000}^{111} \sqrt{\frac{1}{8}} |i\rangle = \sqrt{\frac{1}{8}} (|000\rangle + |001\rangle + |010\rangle + |011\rangle + |100\rangle + |101\rangle + |110\rangle + |111\rangle)$$

$$\sum_{i=000}^{111}$$

$$\sqrt{\frac{1}{8}} |i\rangle$$

$$= \sqrt{\frac{1}{8}} (|000\rangle + |001\rangle \dots + |110\rangle + |111\rangle)$$

$|0\rangle$

$|1\rangle$

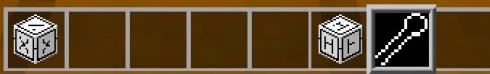
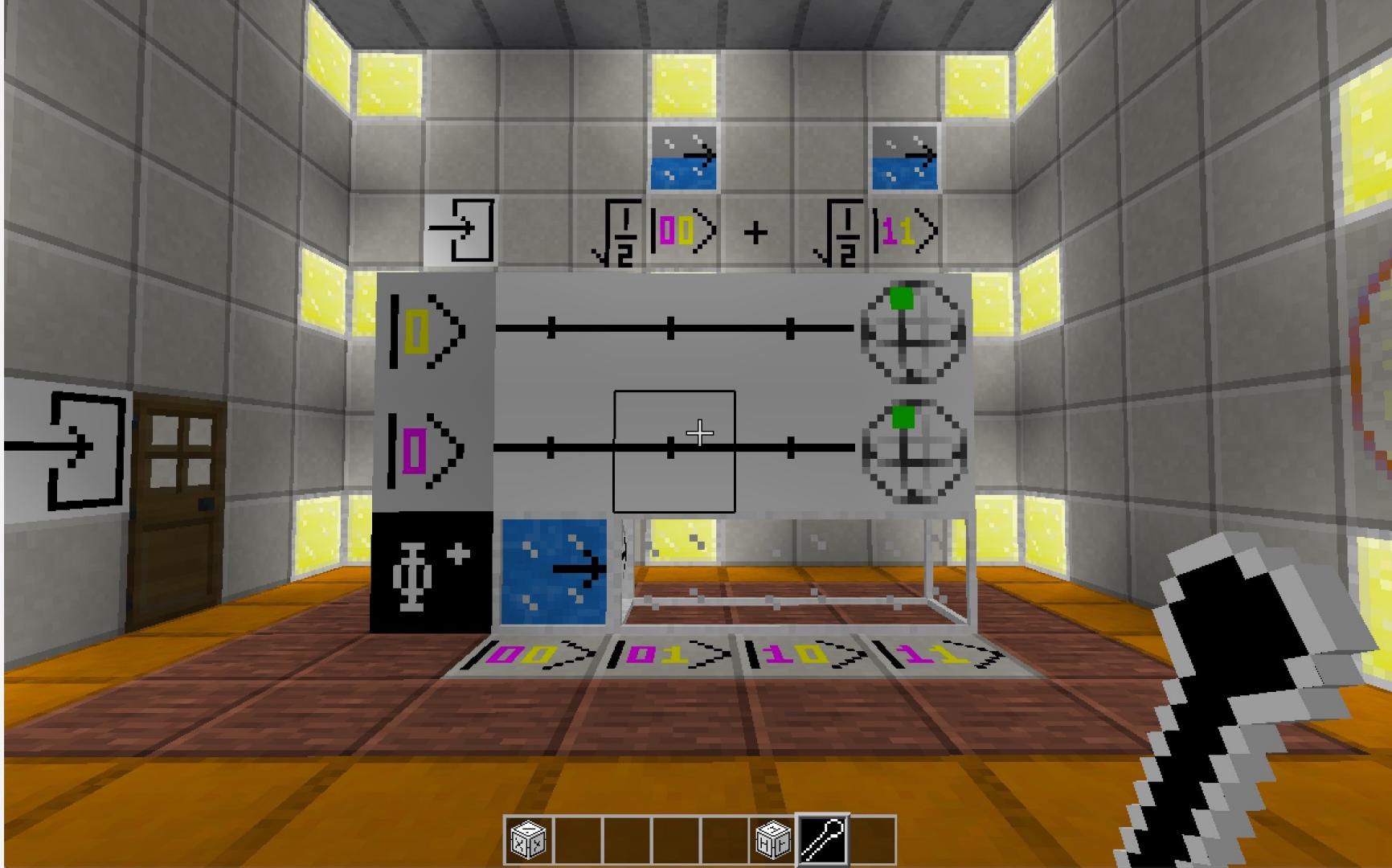
$|2\rangle$

+

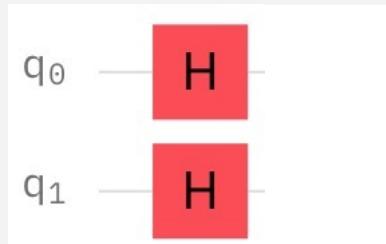


000 001 010 011 100 101 110





量子重ね合わせ



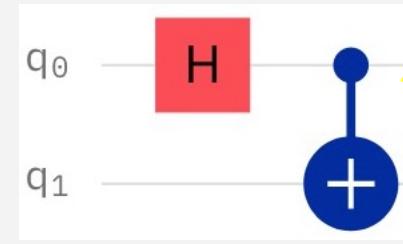
0 0 ... 25%

0 1 ... 25%

1 0 ... 25%

1 1 ... 25%

量子もつれ (エンタングルメント)



0 0 ... 50%

0 1 ... 0%

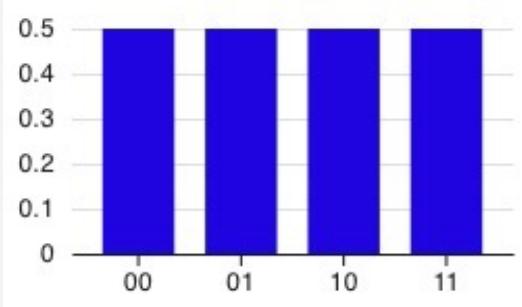
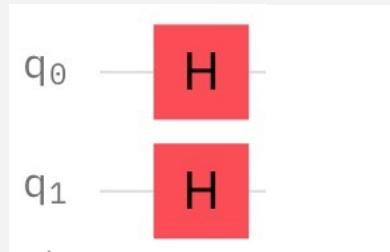
1 0 ... 0%

1 1 ... 50%

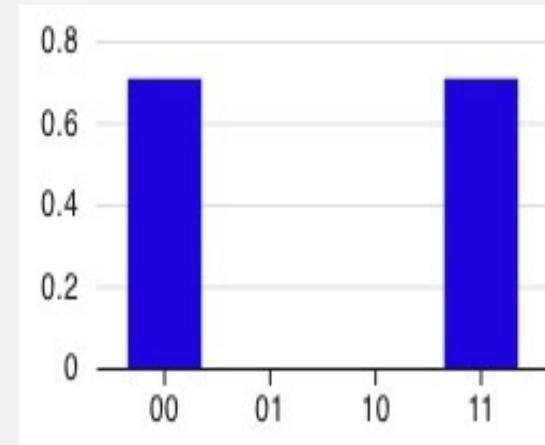
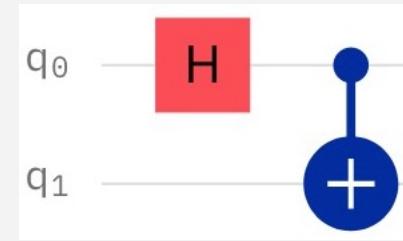
CNOTについては
後ほど詳しく勉強します！

1量子ビット目が0だと分かったら
2量子ビット目も0

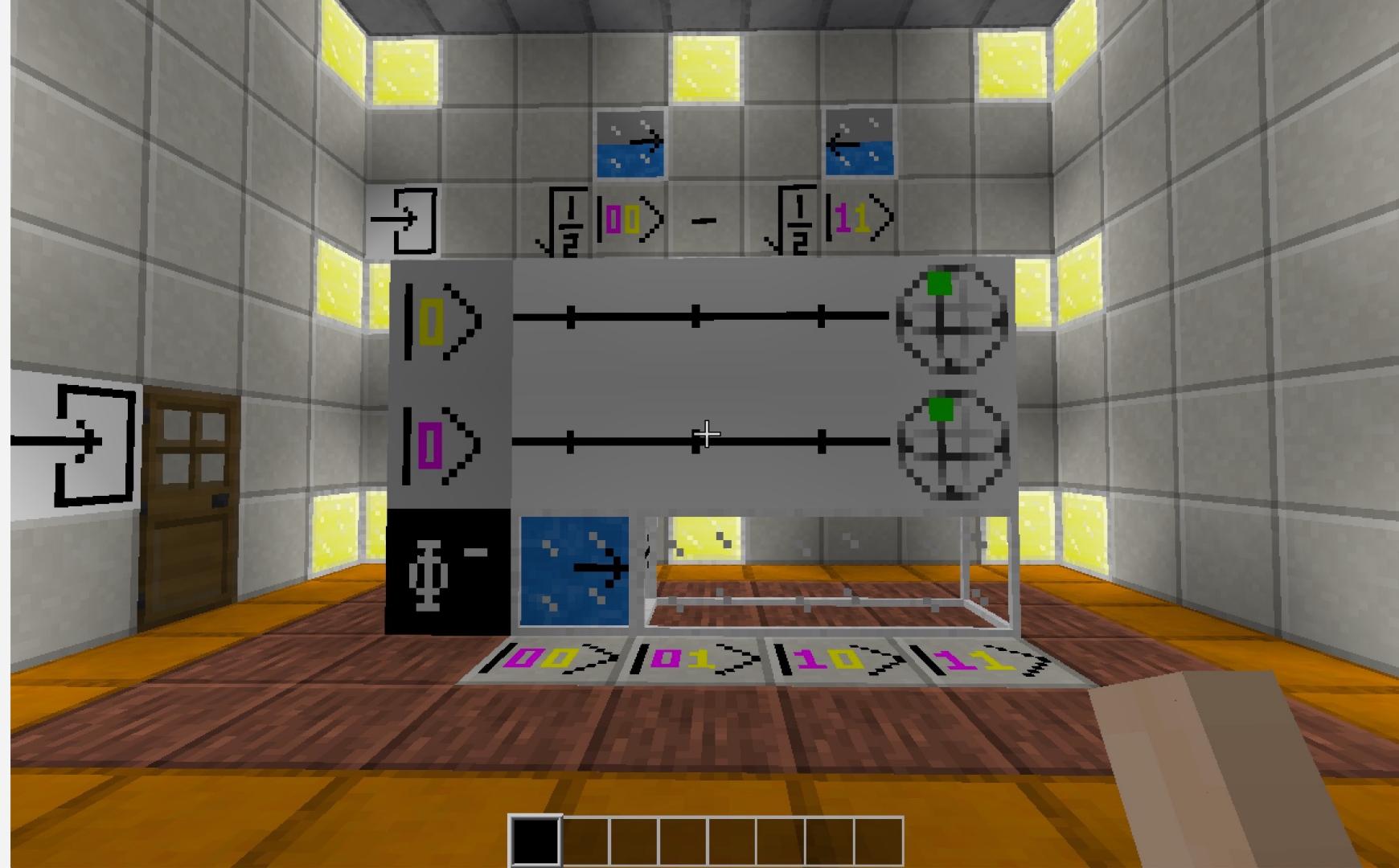
量子重ね合わせ

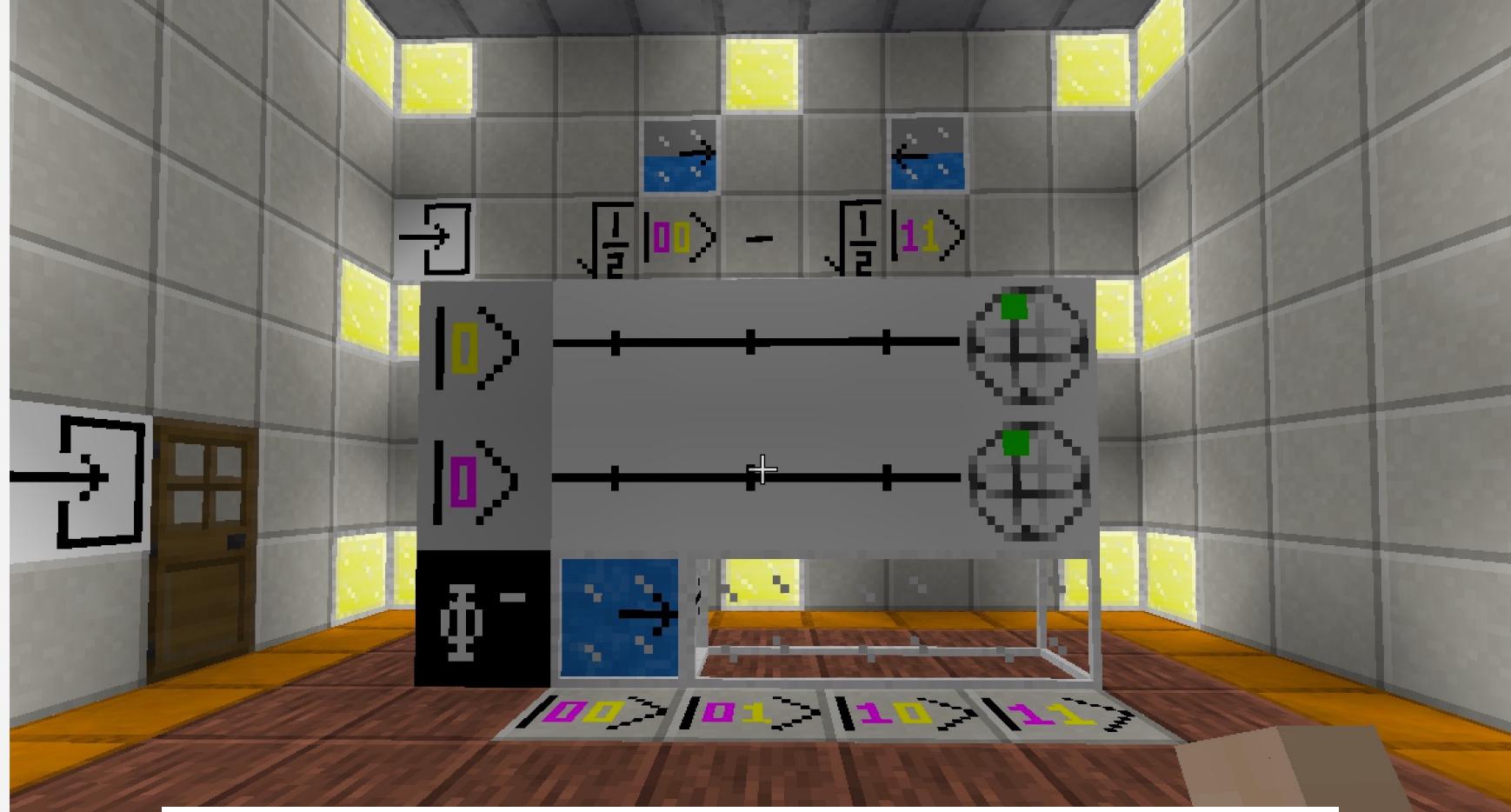


量子もつれ (エンタングルメント)



11

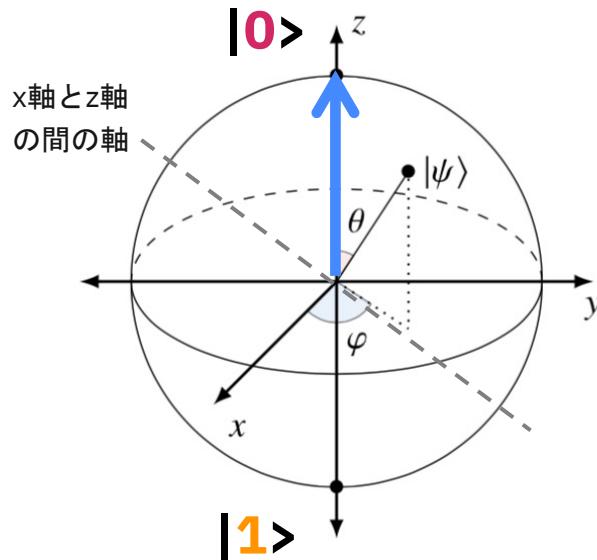




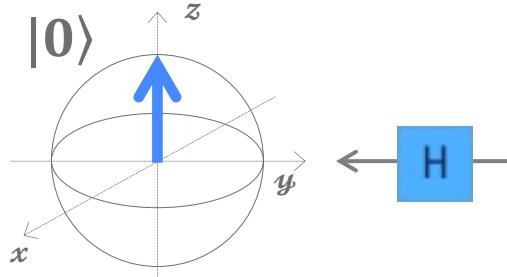
ヒント： $\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$ の重ね合わせにしてからエンタングルを作る

ブロツホ球

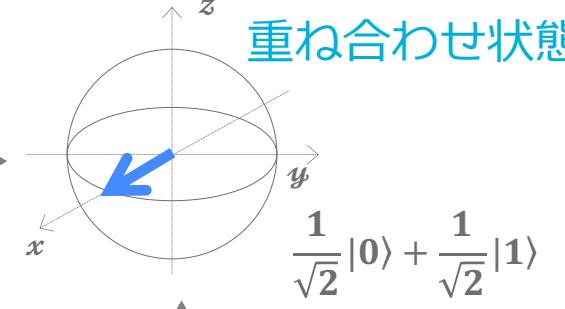
ヒント： $\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$ の重ね合わせにしてからエンタングルを作る



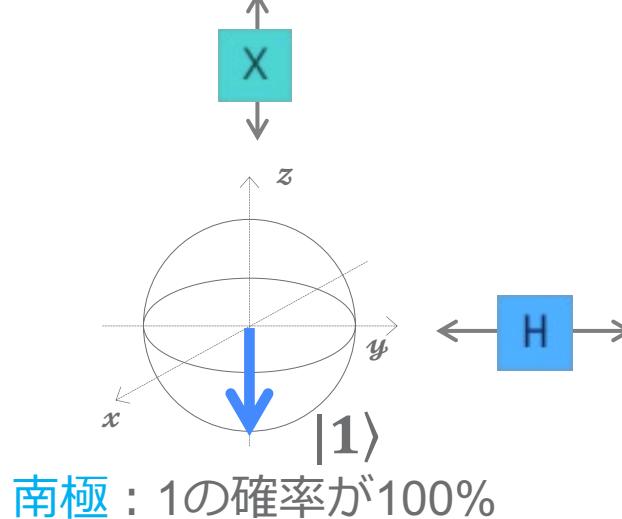
北極：0の確率が100%



赤道：0と1が50%ずつの重ね合わせ状態

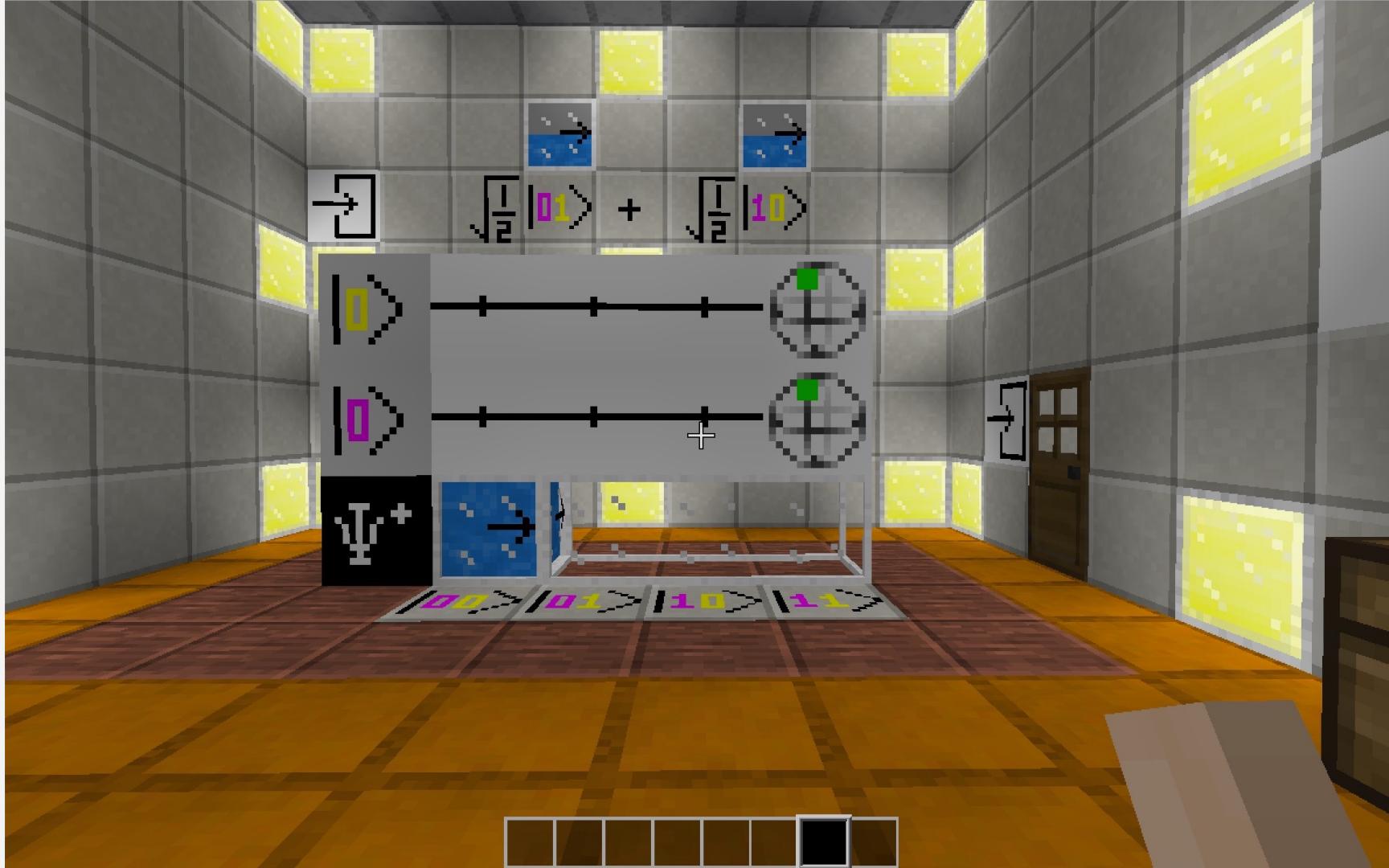


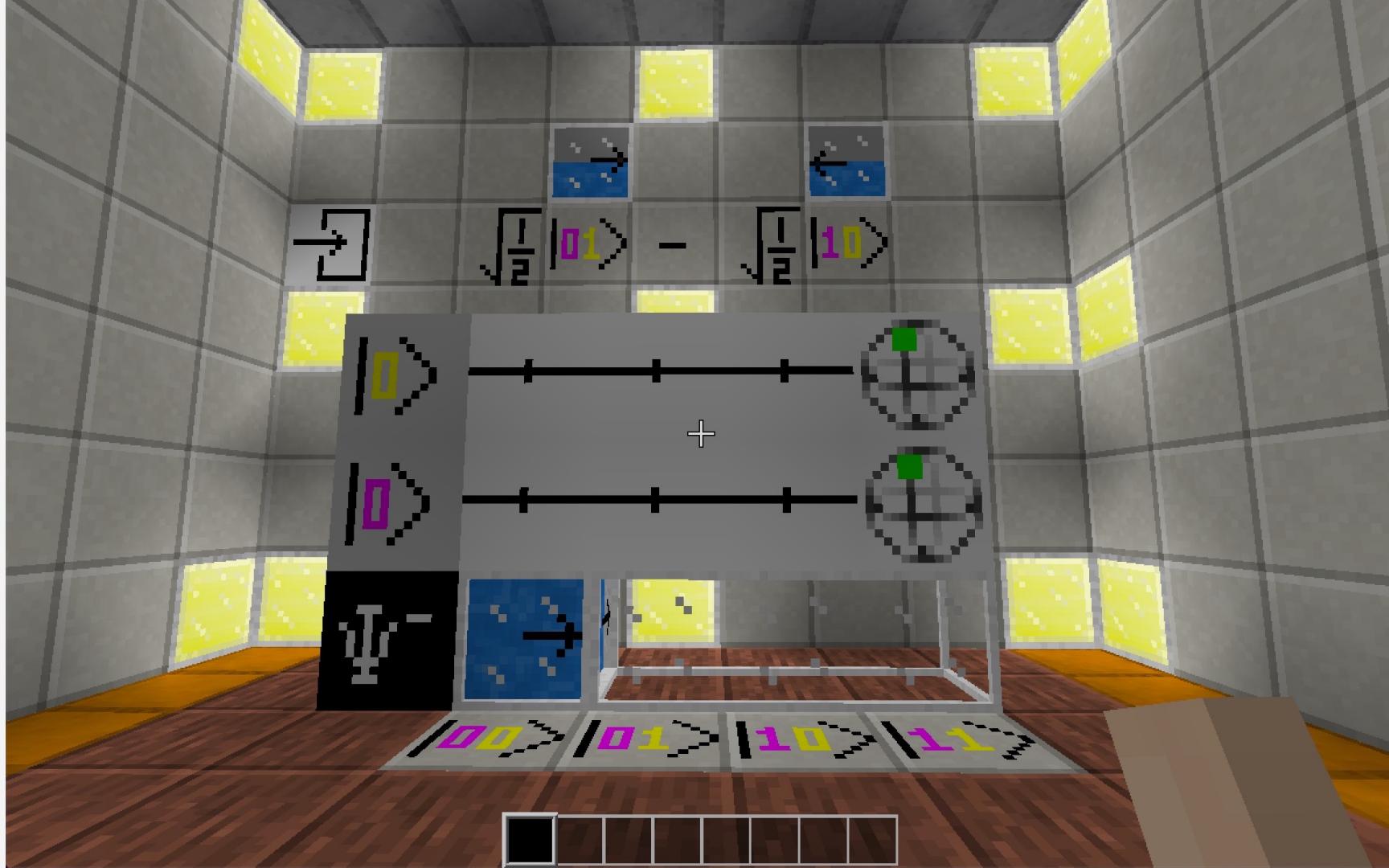
$$\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

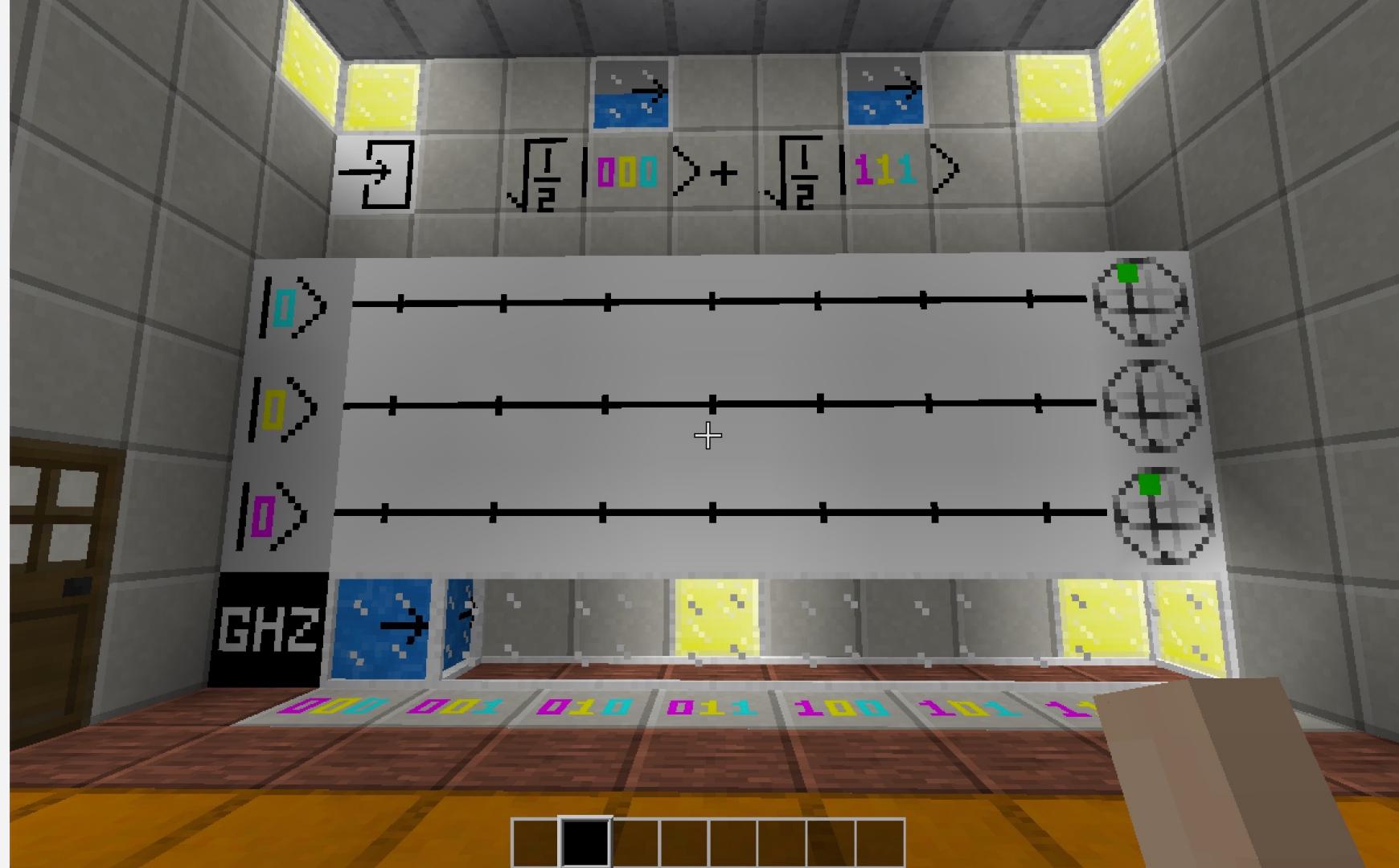


$$\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

南極：1の確率が100%







シェルスクリプトマガジン

A screenshot of a magazine page featuring a quiz game about quantum computers. The top section has a blue header with the text '特別企画 32'. Below it is a grid of icons representing various topics. The main text area says '量子コンピュータの基礎を知る' (Learn the basics of quantum computers). A callout box provides more details:

本物の量子コンピュータを使って誰でも手軽に量子計算の実験ができるようになりました。量子計算は、従来のコンピュータの計算方法とは根本的に異なる原理を用いています。本企画では、量子計算の基礎を紹介した後、量子ゲームを使ってクイズ形式で計算原理を解説します。

日本IBM 沼田史史、小林有里

2016年5月に米IBM社^{※1}が世界で初めてクラウド型の「量子コンピュータ」を発表しました。米国のニューヨークにある研究所に設置された量子コンピュータにクラウド経由で誰でも無料で量子計算の実験ができます。これまで世界中から25万以上の研究者、開発者や学生などからアクセスがあり、本物の量子コンピュータを使って3000回以上の実験が行われています。そして、行われた実験に関する論文は250本以上公開されています(2020年8月時点)。

量子コンピュータの開発自体は、まだ黎明期(ひめいき)であり、その量子比特(ビット)数は数十個程度で、エラー訂正の実装^{※2}はありません。しかし、近い将来実現可能な小規模の量子コンピュータの有効な利用を目的とした実用的なアプリケーションの研究が活発に行われており、特に有望な分野として最適化、量子機械学習、量子化学また金融などでの活用が期待されています。

また、よくある誤解には「量子コンピュータは、どんな問題でもすべて高速に解くことができる」というものがありますが、量子計算の特徴は、量子力学の仕組み、例えば量子重ね合わせや量子もれなどを用いて実現されています。そのため、従来のコンピュータでは煩雑な時間で解くことができなかつた問題を量子コンピュータを用いて効率よく解く方法が実現されています。

※1 米IBM社とは、International Business Machines Corporationを意味します。

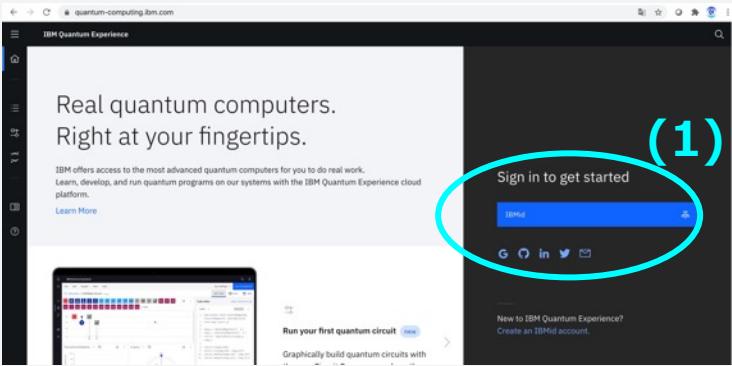
図1 ビット表現の例

$$7 = 1x^2^0 + 1x^2^1 + 1x^2^2$$
$$9838 = 1x^{2^{11}} + 0x^{2^{12}} + 0x^{2^{13}} + 1x^{2^{14}} + 1x^{2^{15}} + 0x^{2^{16}} + 0x^{2^{17}} + 1x^{2^{18}} + 1x^{2^{19}} + 0x^{2^{20}}$$

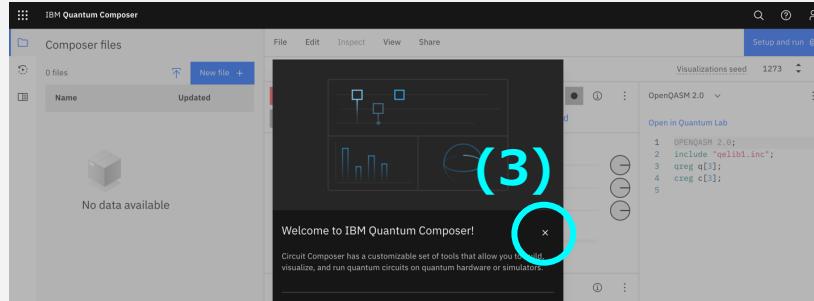
34 シェルスクリプトマガジン 2020 October Vol.68

ハンズオン：いつしょにやってみましょう！

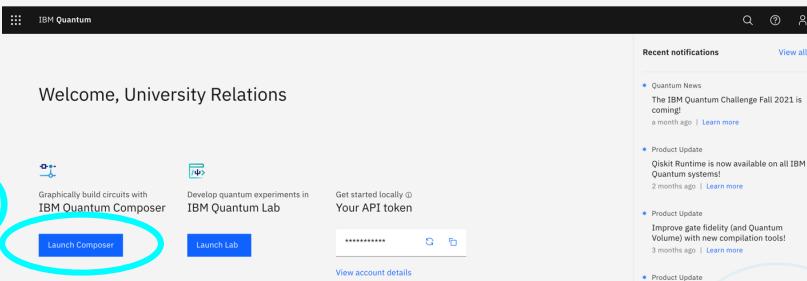
(1) IBM Quantum にログインします。URL:
<https://quantum-computing.ibm.com/>



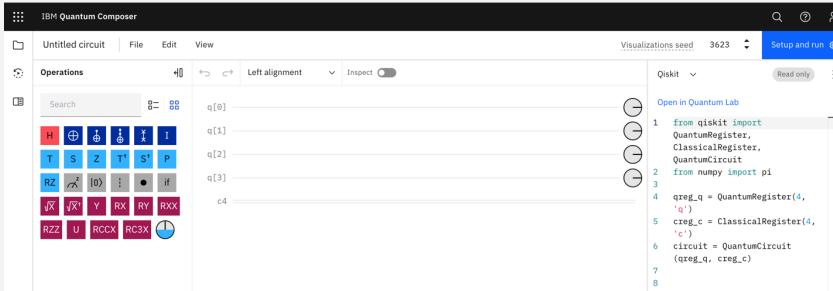
(3) ポップアップウィンドウは「x」をクリックして、閉じます。



(2) 左の青アイコン「Launch Composer」をクリック。



(4) この画面になつたら準備完了です。



1量子ビット回路

The screenshot shows the IBM Quantum Composer interface. At the top, it says "IBM Quantum Composer". Below that is a toolbar with "Untitled circuit", "File", "Edit", "View", and "Visu". The main area is titled "Operations" and has a search bar. On the left is a palette of quantum gates: H, CNOT, Toffoli, Swap, Z, T, S, RZ, RY, RY†, V, P, P†, S†, T†, and If. There are four horizontal lines representing qubits: q[0], q[1], q[2], and q[3]. There is also a vertical line labeled c4 representing a classical bit. A red arrow points to the trash bin icon next to q[1].

マウスでq[1]をクリックするとゴミ箱マークが出てくるので、
クリックして消します。
q[0]だけにして、1量子ビット回路の準備をします。

1量子ビット回路

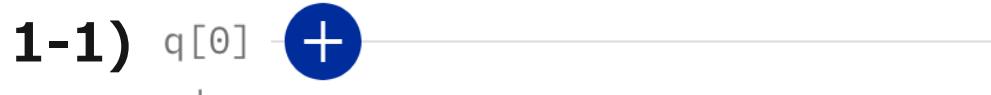
The screenshot shows the IBM Quantum Composer interface. On the left, there's a toolbar labeled "量子ゲート" (Quantum Gates) with various gate icons. A red circle highlights the Hadamard gate (H). A red arrow points from this highlighted gate to the "Operations" section of the main workspace, which contains a search bar and a grid of quantum operations including H, T, RZ, and others. Another red arrow points from the "Operations" section to the workspace itself, where a single-qubit register q[0] and a four-classical-bit register c4 are shown. The workspace has a "Left alignment" button and an "Inspect" toggle. On the right, a large red arrow points from the workspace to a code editor window titled "Qiskit". The code editor displays the following Qiskit code:

```
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
from numpy import pi
qreg_q = QuantumRegister(1, 'q')
creg_c = ClassicalRegister(4, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)
```

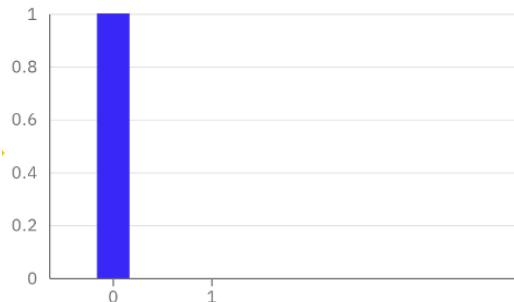
量子ゲートをマウスでドラッグ&ドロップして、
量子回路を作ります。
右側には、Qiskitのコードが自動生成されます。

1. Xゲート(NOTゲート)

図の回路を作つてみてください。下に表示される棒グラフの変化を確認しましょう。



初期状態は $|0\rangle$



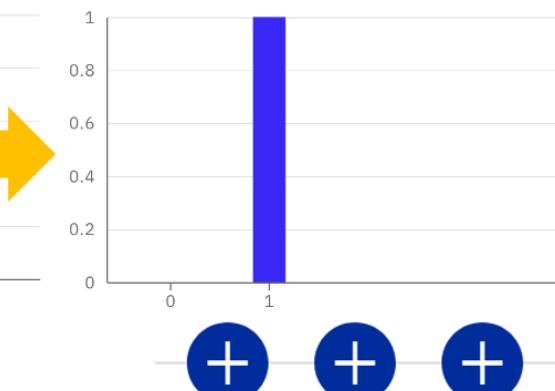
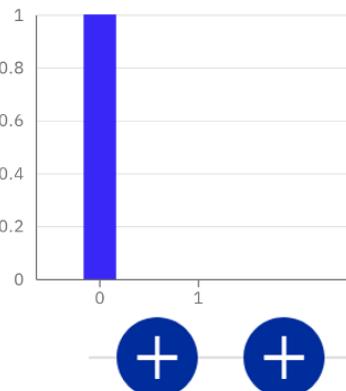
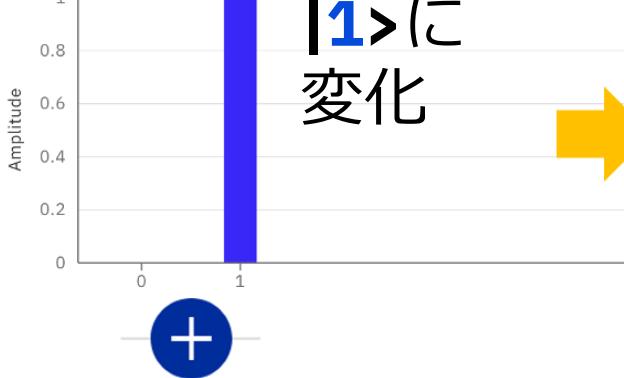
棒グラフ (Statevector 表示) は
量子ビットの状態

$\alpha \times |0\rangle + \beta \times |1\rangle$
の α, β (確率振幅) です。

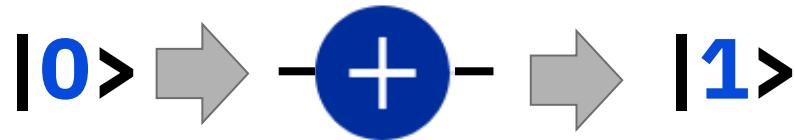
$$|0\rangle = 1 \times |0\rangle + 0 \times |1\rangle$$

$$|1\rangle = 0 \times |0\rangle + 1 \times |1\rangle$$

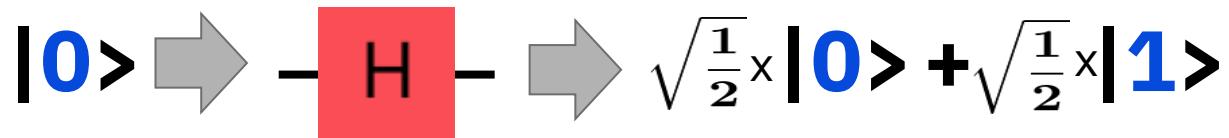
$|1\rangle$ に
変化



量子コンピューターの計算方法

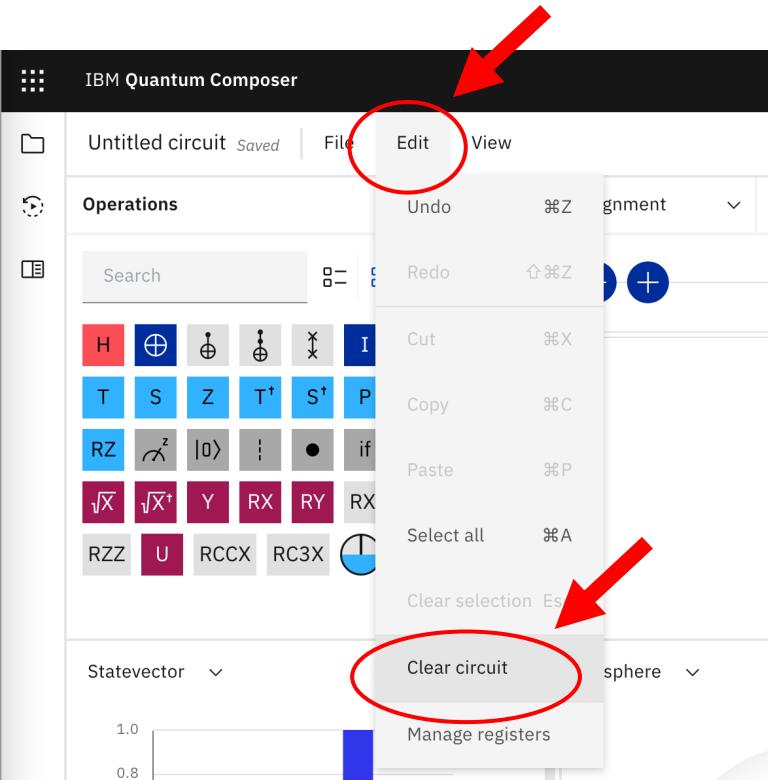


ノット（反転）ゲート

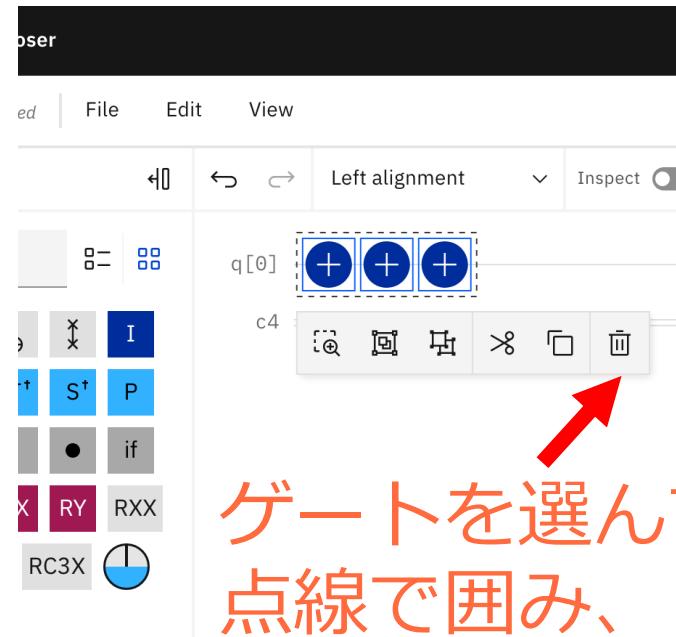


重ね合わせをつくる

置いたゲートを取り除く



または

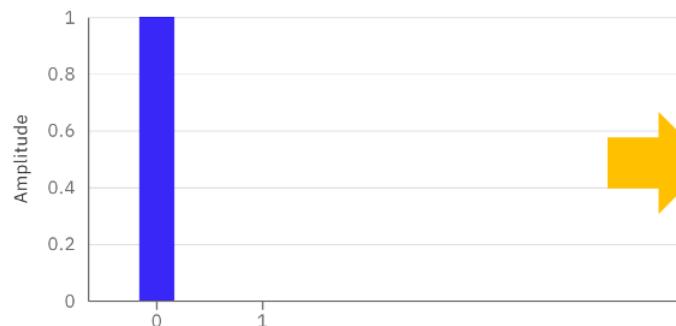
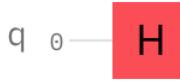


ゲートを選んで
点線で囲み、
ゴミ箱マークを
クリック

2. Hゲート

図の回路を作ってみてください。下に表示される棒グラフの変化を確認しましょう。

2-1)



重ね合わせ



例えば1000回同じ状態を作って、測定すると約500回は0が観測され、約500回は1が観測される状態。

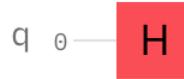
$$|0\rangle = 1 \times |0\rangle + 0 \times |1\rangle$$

$$\begin{aligned} & 0.707 \times |0\rangle + 0.707 \times |1\rangle \\ &= \frac{1}{\sqrt{2}} \times |0\rangle + \frac{1}{\sqrt{2}} \times |1\rangle \end{aligned}$$

2. Hゲート

図の回路を作ってみてください。下に表示される棒グラフの変化を確認しましょう。

2-1)



2-2)



2-3)



量子コンピューターの計算方法

$$|0\rangle \xleftarrow{H} \sqrt{\frac{1}{2}}|0\rangle + \sqrt{\frac{1}{2}}|1\rangle$$



$$|1\rangle \xleftarrow{H} \sqrt{\frac{1}{2}}|0\rangle - \sqrt{\frac{1}{2}}|1\rangle$$

3. Zゲート

図の回路を作ってみてください。下に表示される棒グラフの変化を確認しましょう。

3-1)



3-2)



3-3)



量子コンピューターの計算方法



1 の符号を反転する

量子コンピューターの計算方法

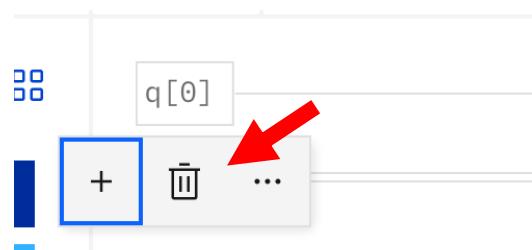
$$|0\rangle \xleftarrow{H} \sqrt{\frac{1}{2}}|0\rangle + \sqrt{\frac{1}{2}}|1\rangle$$

The diagram illustrates the effect of quantum gates on two states. On the left, a blue circle with a white plus sign (+) represents the state $|1\rangle$. An upward arrow is above it and a downward arrow is below it. A red square labeled 'H' is positioned between the two states. To the right of the H gate, the state $\sqrt{\frac{1}{2}}|0\rangle + \sqrt{\frac{1}{2}}|1\rangle$ is shown. On the right, a blue square labeled 'Z' is positioned between the two states. To its right, the state $\sqrt{\frac{1}{2}}|0\rangle - \sqrt{\frac{1}{2}}|1\rangle$ is shown.

$$|1\rangle \xleftarrow{H} \sqrt{\frac{1}{2}}|0\rangle + \sqrt{\frac{1}{2}}|1\rangle$$
$$\xrightarrow{Z} \sqrt{\frac{1}{2}}|0\rangle - \sqrt{\frac{1}{2}}|1\rangle$$

量子ビットを増やす

q[0]をクリックして、さらに「+」マークをクリックして、2量子ビットの回路を準備します。



2量子ビット回路
になります

次に、c4をクリックして、「-」マークをクリックするを2回繰り返して、2古典ビットにします。



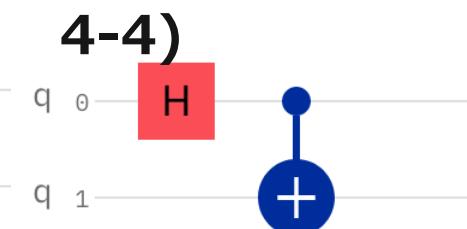
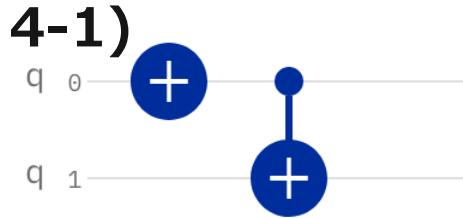
2古典ビットになります

4. CNOTゲート(制御Xゲート)

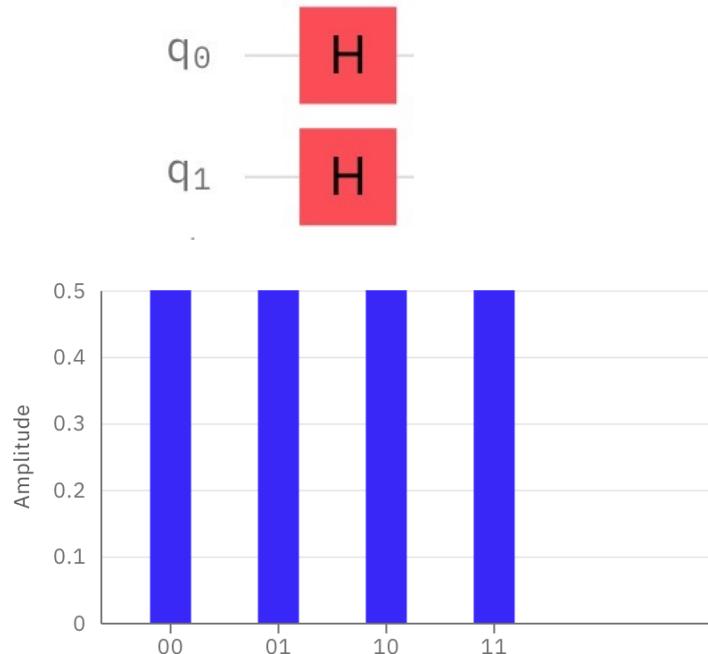
制御ビットが $|1\rangle$ のときのみ、目標ビットを反転（NOT）するゲートです。



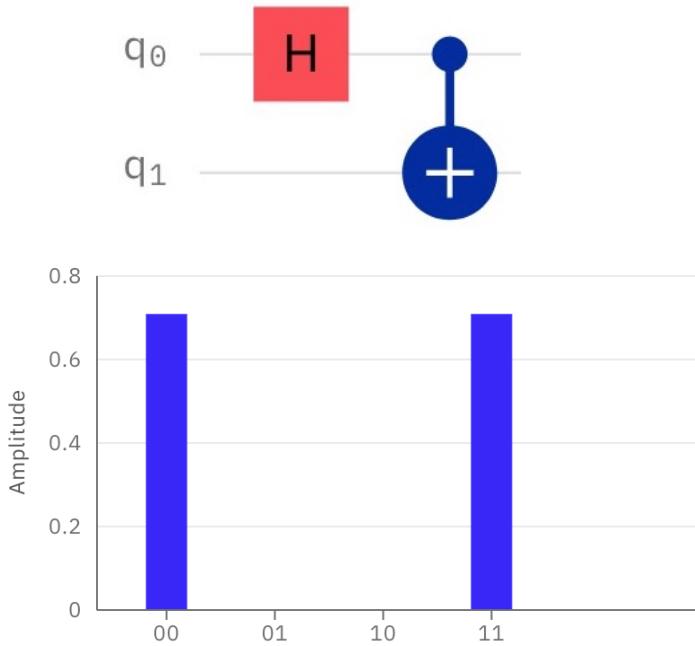
入力		出力	
制御ビット	目標ビット	制御ビット	目標ビット
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0



量子重ね合わせ

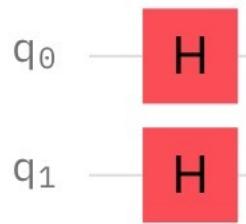


量子もつれ (エンタングルメント)



CNOTゲートは、
エンタングルメントを作ります。

量子重ね合わせ



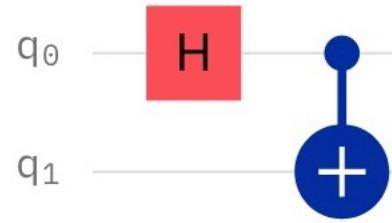
0 0 ... 25%

0 1 ... 25%

1 0 ... 25%

1 1 ... 25%

量子もつれ (エンタングルメント)



0 0 ... 50%

0 1 ... 0%

1 0 ... 0%

1 1 ... 50%

1量子ビット目が0だと分かったら
2量子ビット目も0

5. 測定

ファイル名を変更
(Entanglementなど)

The screenshot shows the IBM Quantum Compose interface. On the left, the 'Operations' panel displays various quantum gates: H, CNOT, T, S, Z, RZ, RX, RY, RXX, RZZ, U, RCCX, RC3X, and M. A red box highlights the 'Untitled circuit' tab in the top navigation bar. A red circle labeled '(2)' is placed over the 'Saved' button. In the center, a quantum circuit is shown with two qubits (q[0] and q[1]) and one classical register (c[2]). A red box labeled '(1)' encloses a measurement gate (M) on q[1]. A red arrow points from this box to the text '測定ゲートを追加'. On the right, the Qiskit editor shows the generated Python code:

```
1  from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
2  from numpy import pi
3
4  qreg_q = QuantumRegister(2, 'q')
5  creg_c = ClassicalRegister(2, 'c')
6  circuit = QuantumCircuit(qreg_q, creg_c)
7
8  circuit.h(qreg_q[0])
9  circuit.cx(qreg_q[0], qreg_q[1])
10 circuit.measure(qreg_q[0], creg_c[0])
11 circuit.measure(qreg_q[1], creg_c[1])
```

A red circle labeled '(3)' is placed over the 'Setup and run' button in the top right corner, with the text 'クリック' (click) written next to it.

まず量子シミュレーターで実験

Set up and run your circuit

Step 1
Choose a system or simulator

Search by system or simulator name

Total pending jobs 1

63 Qubits

ibmq_qasm_simulator See details

Simulator status • Online

Total pending jobs 1

32 Qubits

simulator_statevector See details

Simulator status • Online

Total pending jobs 1

(4) スクロール

Step 2
Choose your settings

Provider: ibm-q-internal/deployed/default

Shots *: 1024

Job name: e.g. Untitled circuit job

(5) ibmq_qasm_simulator を選択

(6) Run on ibmq_qasm_simulator

Close

IBM Quantum Composer

Untitled circuit Saved File Edit View Visualizations seed 3623 Setup and run

Composer jobs

(7) Jobの確認

Search

Operations

q[0] H c2

q[1] + c2

z

Qiskit

Open in Quantum Lab

```
1 from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
2 from numpy import pi
3
4 qreg_q = QuantumRegister(2, 'q')
```



IBM Quantum Composer

Untitled circuit Saved File Edit View Visualizations seed 3623 Setup and run

Composer jobs

View all jobs →

Search jobs

Completed: Jul 21, 2022 8:49 PM ID: 62d93d60aaf3a771... | ibmq_qasm_simulator

(8)

Operations

q[0] H

q[1] +

Qiskit

Open in Quantum Lab

```
1 from qiskit import QuantumRegister, ClassicalRegister,
```

Jobs /
62d93d60aaaf3a771842b9ede

See more details ↗

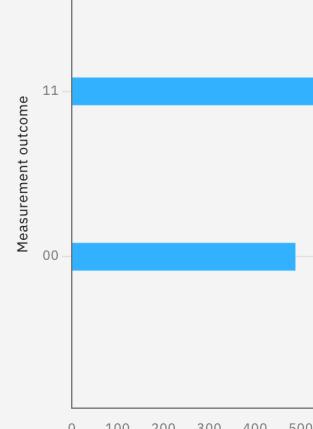
Completed
Jul 21, 2022 8:50 PM (in 8.3s)

Backend
ibmq_qasm_simulator

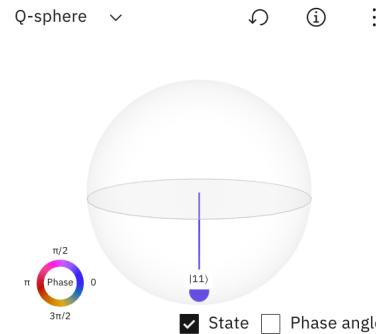
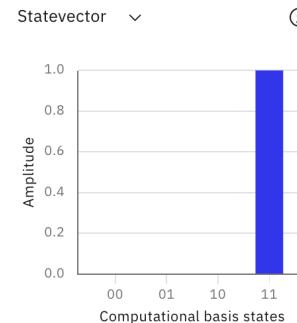
Status timeline Completed

Details

Result - histogram



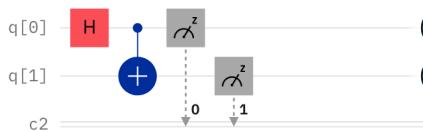
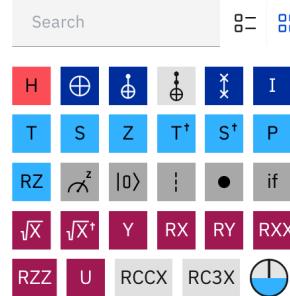
(9) 測定結果



Untitled circuit Saved File Edit View

Operations

Search



Visualizations seed 3623

Setup and run

Read only

(10)

One in Qiskit

```
1 import numpy as np
2 from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
3
4 qreg_q = QuantumRegister(2, 'q')
5 creg_c = ClassicalRegister(2, 'c')
6 circuit = QuantumCircuit(qreg_q, creg_c)
7
8 circuit.h(qreg_q[0])
9 circuit.cx(qreg_q[0], qreg_q[1])
10 circuit.measure(qreg_q[0], creg_c[0])
11 circuit.measure(qreg_q[1], creg_c[1])
```

Set up and run your circuit

実機の量子コンピューターで実験

X

Step 1

Choose a system or simulator

Search bar: Search by system or simulator name

ibmq_quito (selected):
System status: Online
Total pending jobs: 7
5 Qubits 16 QV 2.5K CLOPS

ibmq_belem:
System status: Online

(11)

Step 2

Choose your settings

Provider

ibm-q-education/ibm-3/kawasaki-camp

Shots *

1024

Job limit: None

Optional

Name your job

(12)

Kawasaki-campを選択します

実デバイスを選択します。

- Total pending jobsが少ない

- QVが大きい

などから選んでみましょう。

Close

Run on ibmq_belem

(13)

IBM Quantum Composer

Jobs / (14) Jobの確認

Untitled circuit Saved File Edit View Visualiz

Completed Jul 21, 2022 8:55 PM (in 1m 8s)

Backend ibmq_qasm_simulator

Status timeline Completed

See more details

Operations

Search

H \oplus \ominus \oplus \ominus \otimes I
T S Z T^\dagger S^\dagger P
RZ r_z $|0\rangle$ i ● if

q[0] H r_z
q[1] + r_z
c2 0 1

Qiskit Open in Quantu

```
1 from qiskit import *
2 ClassicalRegister creg
3 from numpy import *
4 qreg_q = QuantumRegister(2)
5 creg_c = ClassicalRegister(2)
6 circuit =
```



IBM Quantum Composer

View all jobs → Composer jobs

Pending: Jul 21, 2022 9:21 PM ID: 62d944d1e59b9100e90919... | ibmq_belem

Completed: Jul 21, 2022 8:59 PM ID: 62d93fbbaaf3a73b512b9ef4 | ibmq_belem

Operations

Search

H \oplus \ominus \oplus \ominus \otimes I
T S Z T^\dagger S^\dagger P
RZ r_z $|0\rangle$ i ● if

q[0] H r_z
q[1] + r_z
c2 0 1

Qiskit Open in Quantu

```
1 from qiskit import *
2 ClassicalRegister creg
3 from numpy import *
4 qreg_q = QuantumRegister(2)
```



IBM Quantum Composer



Jobs /

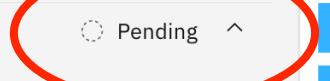
62d944d1e59b9100e90919d8

⋮

待ち時間がある場合

Backend

ibmq_belem

[See more details](#)

Status timeline

- Created: Jul 21, 2022 9:21 PM
- Transpiling: 904ms
- Validating: 892ms
- In queue
- Running
- Completed

Details

Untitled circuit Saved

File

Edit

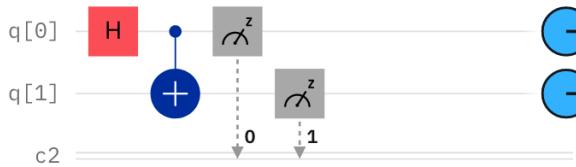
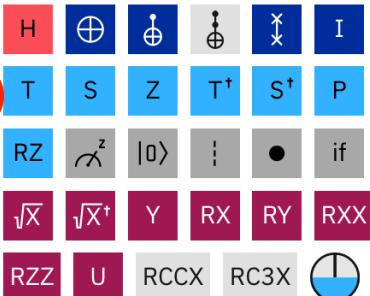
View

Visual

Operations



Search



Qiskit

Open in Quant

```
1 from qis
2 Classics
3 from nu
4 qreg_q :
5 creg_c :
6 circuit
7 circuit
8 circuit
9 circuit
10 circuit
11 circuit
```

[See more details](#)Untitled circuit Saved

Edit

View

Visualizations seed

3623

Setup and run

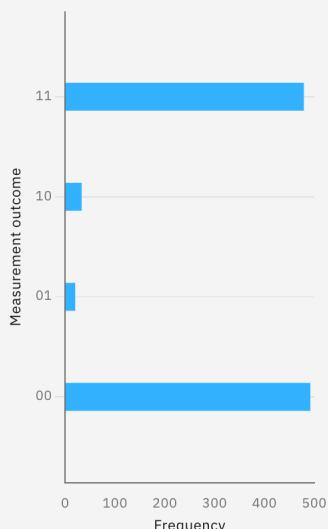
Completed
Jul 21, 2022 9:00:00 PM (in 38.2s)

Backend
ibmq_belem

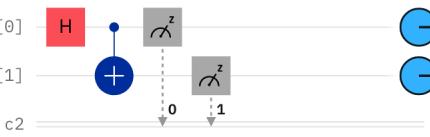
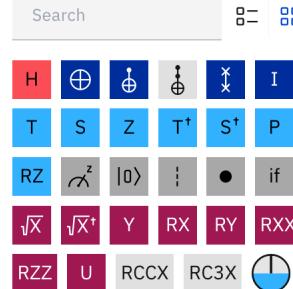
Status timeline Completed

Details

Result - histogram



Operations



Qiskit

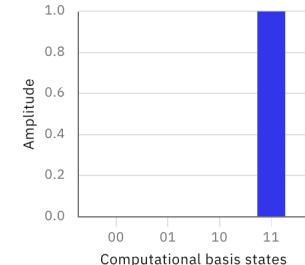
Read only

[Open in Quantum Lab](#)

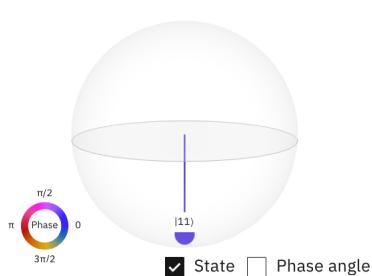
```
1 from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
2 from numpy import pi
3
4 qreg_q = QuantumRegister(2, 'q')
5 creg_c = ClassicalRegister(2, 'c')
6 circuit = QuantumCircuit(qreg_q, creg_c)
7
8 circuit.h(qreg_q[0])
9 circuit.cx(qreg_q[0], qreg_q[1])
10 circuit.measure(qreg_q[0], creg_c[0])
11 circuit.measure(qreg_q[1], creg_c[1])
```

(16) 実機での結果

Statevector



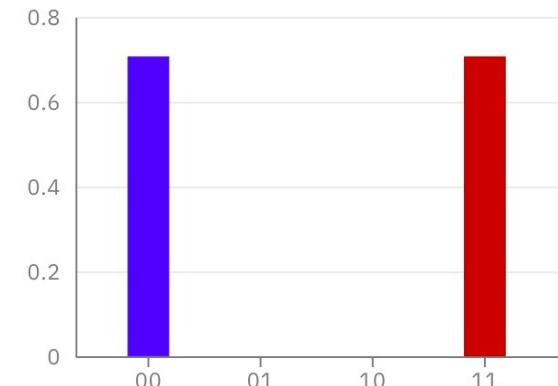
Q-sphere



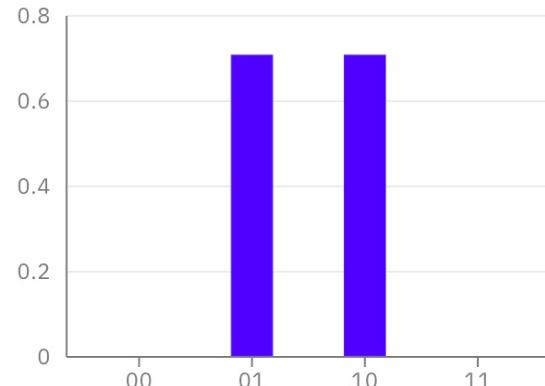
演習問題

2量子ビットのエンタングル状態を作ってみましょう。
答えは一つではないので、どんな作り方でもOKです。

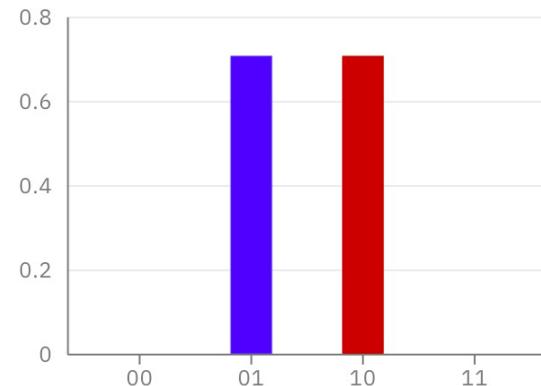
(1) $\frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$



(2) $\frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$



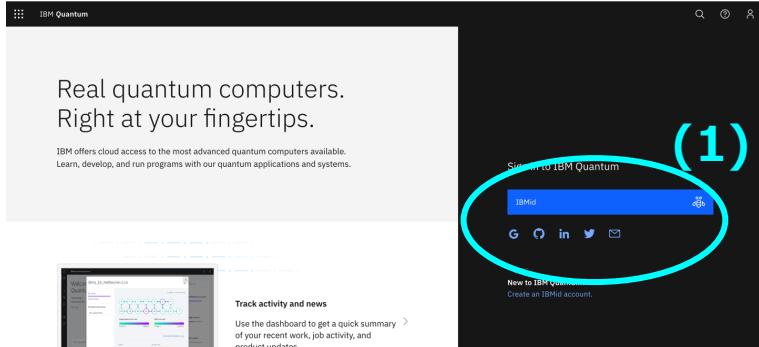
(3) $\frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$



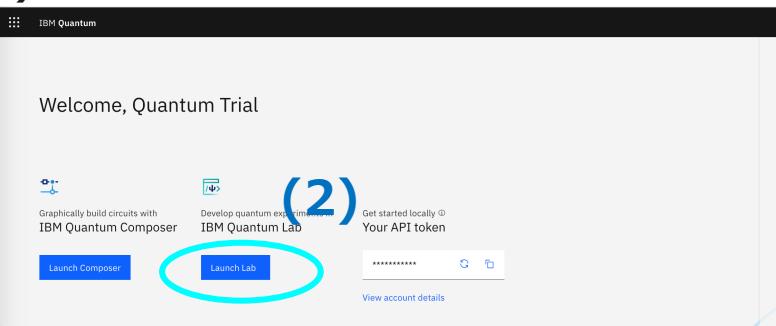
IBM Quantum Labでの実行

(1) IBM Quantumにログインします。

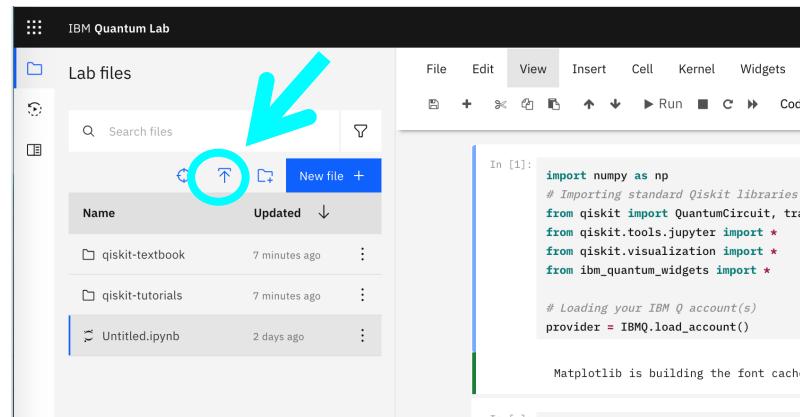
URL: <https://quantum-computing.ibm.com/>



(2) 青の右側「Launch Lab」をクリック。



(3) 左側 下 の「Upload file」から、ご自分のローカルに保存したハンズオンコンテンツ「20220801_gates.ipynb」を探して、アップロードします。



(4) ファイル名「20220801_gates.ipynb」をダブルクリックして開きます。