

Quantum Computing – 2 Qbits

Bill Celmaster

June 18, 2026

Overview

- ▶ 2-Qbit product states
- ▶ 2-Qbit general states
- ▶ 2×2 Quantum Gates (pictures, words), unitary matrices and quantum circuits
- ▶ A simple quantum circuit and algorithm for IBM's computer
 - ▶ Introduction to IBM's environment
 - ▶ Some Python scripts (for non-Python people)
 - ▶ Translating those scripts to pictures and gates
 - ▶ Predicting the quantum behavior of the quantum circuit
 - ▶ Execution of the algorithm on actual hardware

2-Qbit **product** states

- ▶ In vector notation $s_i = \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix}$
- ▶ Write $|s_1\rangle|s_2\rangle$ or $|s_1s_2\rangle = (\alpha_1|0\rangle + \beta_1|1\rangle)(\alpha_2|0\rangle + \beta_2|1\rangle)$.
- ▶ 4 basis states are $|00\rangle, |01\rangle, |10\rangle, |11\rangle$
- ▶ $|s_1s_2\rangle = \alpha_1\alpha_2|00\rangle + \alpha_1\beta_2|01\rangle + \beta_1\alpha_2|10\rangle + \beta_1\beta_2|11\rangle$
- ▶ Vector notation: $s_1 \otimes s_2 = \begin{pmatrix} \alpha_1\alpha_2 \\ \alpha_1\beta_2 \\ \beta_1\alpha_2 \\ \beta_1\beta_2 \end{pmatrix}$
- ▶ Let $\langle s_3s_4|s_1s_2\rangle \equiv \alpha_3^*\alpha_4^*\alpha_1\alpha_2 + \alpha_3^*\alpha_4^*\alpha_1\beta_2 + \beta_3^*\alpha_4^*\beta_1\alpha_2 + \beta_3^*\alpha_4^*\beta_1\beta_2$
 - ▶ Vector notation: $(s_3 \otimes s_4, s_1 \otimes s_2)$
- ▶ Prob. of measuring $|s_3s_4\rangle$ if system is in state $|s_1s_2\rangle$, is $|\langle s_3s_4|s_1s_2\rangle|^2$
 - ▶ $= |\langle s_3|s_1\rangle|^2 |\langle s_4|s_2\rangle|^2$

2-Qbit **general** states

- ▶ (Normalized) 2-Qbit general states superpose the 4 basis states.

- ▶ $|\psi\rangle = \alpha_1|00\rangle + \alpha_2|01\rangle + \alpha_3|10\rangle + \alpha_4|11\rangle$
 $|\phi\rangle = \beta_1|00\rangle + \beta_2|01\rangle + \beta_3|10\rangle + \beta_4|11\rangle$
where $\sum_{i=1}^4 |\alpha_i|^2 = \sum_{i=1}^4 |\beta_i|^2 = 1$

- ▶ Vector notation: $\psi = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{pmatrix}$, etc.

- ▶ **In general**, $|\psi\rangle$, $|\phi\rangle$ are **not** product states

- ▶ General inner product: $\langle\phi|\psi\rangle \equiv \sum_{i=1}^4 \beta_i^* \alpha_i$

- ▶ Vector notation: (ϕ^*, ψ)

The EPR state – an example of a 2-Qbit general state

- ▶ The EPR state is $|EPR\rangle = \frac{1}{\sqrt{2}}(|0\rangle|0\rangle + |1\rangle|1\rangle)$.
 - ▶ If the first Qbit is $|0\rangle$, so is the second Qbit, etc.
 - ▶ $P(|00\rangle : |EPR\rangle) = P(|11\rangle : |EPR\rangle) = \frac{1}{2}$
 $P(|01\rangle : |EPR\rangle) = P(|10\rangle : |EPR\rangle) = 0$
 - ▶ Therefore we can't factorize $|EPR\rangle$ as $|s_1 s_2\rangle$
 - ▶ Otherwise $0 = P(|01\rangle : |EPR\rangle) = P(|0\rangle : s_1)P(|1\rangle : s_1)$
 - ▶ Then either $P(|0\rangle : s_1) = 0$ or $P(|1\rangle : s_1) = 0$
 - ▶ **Implying either $P(|00\rangle : |EPR\rangle) = 0$ or $P(|11\rangle : |EPR\rangle) = 0$**
 - ▶ We say that the EPR state is correlated.

- ▶ Vector representation of the EPR state is $\begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$.

Physical implementation of a 2-Qbit EPR state

- ▶ Consider the decay of a π^0 -meson into two photons $\pi^0 \rightarrow \gamma\gamma$.
- ▶ If the pion has a total angular momentum $\mathbf{J} = 0$

$$|\Psi_{\gamma\gamma}\rangle = \frac{1}{\sqrt{2}} (|H\rangle_1 |H\rangle_2 + |V\rangle_1 |V\rangle_2),$$

where $|H\rangle$, $|V\rangle$ are horizontally and vertically polarized photons.

- ▶ If photon-1 is horizontally polarized, so is photon-2, etc.

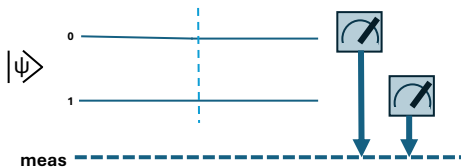
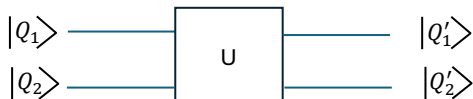


Figure: If the input state is $|\Psi_{\gamma\gamma}\rangle$ then the theoretical measurement is probabilities: $\{HH: 50\%, VV: 50\%\}$

2-Qbit **product** gates – pictorial representation

- ▶ Think of this as 2 **uncorrelated** stationary electrons with spin¹
- ▶ Each electron is in its own state, $|Q_i\rangle = \alpha_i|0\rangle + \beta_i|1\rangle$.
- ▶ Analogy with a Boolean gate - $|Q_i\rangle$ and $|Q'_i\rangle$ are either $|0\rangle$ or $|1\rangle$



¹By forced separation, we avoid Fermi statistics.

2-Qbit matrix representation of “product” gates

► $|Q_1 Q_2\rangle \rightarrow |v, w\rangle = \mathbf{v} \otimes \mathbf{w}$ and $U \rightarrow \mathbf{A} \otimes \mathbf{B}$. $U|Q_1 Q_2\rangle \rightarrow |\mathbf{A}v, \mathbf{B}w\rangle$

► Then

$$\begin{aligned}\begin{pmatrix} v'_1 \\ v'_2 \end{pmatrix} \otimes \begin{pmatrix} w'_1 \\ w'_2 \end{pmatrix} &= \mathbf{A} \otimes \mathbf{B} \left(\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \otimes \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \right) \\ &= \left(\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \otimes \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \right)\end{aligned}$$

► With the 4 basis states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$

$$\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \otimes \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \rightarrow \begin{pmatrix} v_1 w_1 \\ v_1 w_2 \\ v_2 w_1 \\ v_2 w_2 \end{pmatrix}$$

► and $\mathbf{A} \otimes \mathbf{B}$ becomes

$$\begin{pmatrix} A_{11} & A_{12} & 0 & 0 \\ 0 & 0 & A_{11} & A_{12} \\ A_{21} & A_{22} & 0 & 0 \\ 0 & 0 & A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} & 0 & 0 \\ 0 & 0 & B_{11} & B_{12} \\ B_{21} & B_{22} & 0 & 0 \\ 0 & 0 & B_{21} & B_{22} \end{pmatrix} = \begin{pmatrix} A_{11}B_{11} & A_{11}B_{12} & A_{12}B_{11} & A_{12}B_{12} \\ A_{11}B_{21} & A_{11}B_{22} & A_{12}B_{21} & A_{12}B_{22} \\ A_{21}B_{11} & A_{21}B_{12} & A_{22}B_{11} & A_{22}B_{12} \\ A_{21}B_{21} & A_{21}B_{22} & A_{22}B_{21} & A_{22}B_{22} \end{pmatrix}$$

Probability-Factorization and Product Gates

- ▶ Recall:
Prob. of measuring $|s_3 s_4\rangle$ if system is in state $|s_1 s_2\rangle$, is $|\langle s_3 s_4 | s_1 s_2 \rangle|^2$
- ▶ Interpret by saying the two Qbits are uncorrelated:
 - ▶ $P(|s_3 s_4\rangle : |s_2 s_1\rangle) = P(|s_3\rangle : |s_1\rangle)P(|s_4\rangle : |s_2\rangle)$
- ▶ What if a product gate $\mathbf{M} = \mathbf{A} \otimes \mathbf{B}$ is applied to $|s_1 s_2\rangle$?
- ▶ $P_{\mathbf{M}}(|s_3 s_4\rangle : |s_2 s_1\rangle) = \text{prob. of } |s_3 s_4\rangle \text{ after applying } \mathbf{M} \text{ to } |s_1 s_2\rangle.$

$$\begin{aligned} P_{\mathbf{M}}(|s_3 s_4\rangle : |s_2 s_1\rangle) &= (|\langle s_3 | \mathbf{A} | s_1 \rangle|^2) (|\langle s_4 | \mathbf{B} | s_2 \rangle|^2) \\ &= P_{\mathbf{A}}(|s_3\rangle : |s_1\rangle) P_{\mathbf{B}}(|s_4\rangle : |s_2\rangle) \end{aligned}$$

- ▶ This is interpreted as following:
 - ▶ Assume two *uncorrelated* input Qbits.
 - ▶ Then a product gate produces uncorrelated output Qbits.
 - ▶ Think of the product gate acting independently on each Qbit.

2-Qbit general states with 2-Qbit product gates

- ▶ Pictorial representations sometimes “hard to picture”.
- ▶ Example: 2-photon EPR state

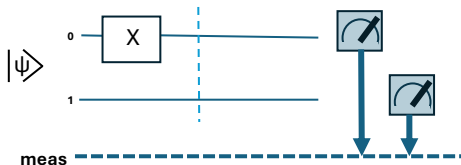


Figure: If the input state is $|\Psi_{\gamma\gamma}\rangle$ then the theoretical measurement is probabilities: $\{HV: 50\%, VH: 50\%\}$

- ▶ Matrix representations: $|H\rangle \rightarrow |0\rangle$, $|V\rangle \rightarrow |1\rangle$; $|\psi\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$;

- ▶
$$\mathbf{U}_{\mathbf{x} \otimes \mathbf{I}_2} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

2-Qbit **general** gates – Pictorial representation

- ▶ Most general input and output (no Boolean counterpart)

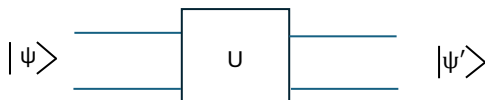


Figure: $|\psi\rangle = \alpha_1|00\rangle + \alpha_2|01\rangle + \alpha_3|10\rangle + \alpha_4|11\rangle$
 $|\psi'\rangle = \beta_1|00\rangle + \beta_2|01\rangle + \beta_3|10\rangle + \beta_4|11\rangle$

- ▶ Alternative picture for product states

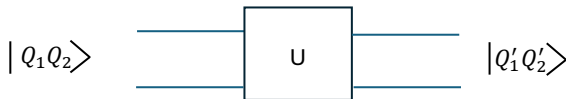


Figure: $|Q_1Q_2\rangle = (\alpha_1|0\rangle + \beta_1|1\rangle) \otimes (\alpha_2|0\rangle + \beta_2|1\rangle)$
 $|Q'_1Q'_2\rangle = (\alpha'_1|0\rangle + \beta'_1|1\rangle) \otimes (\alpha'_2|0\rangle + \beta'_2|1\rangle)$

2-Qbit matrix representation of "general" gates

- ▶ **Less ambiguous than the pictorial representation**
- ▶ Write $|\psi'\rangle = \mathbf{U}|\psi\rangle$.

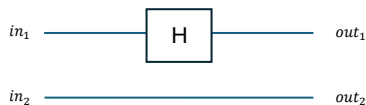
- ▶ In component notation

$$\begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{pmatrix} = \begin{pmatrix} U_{11} & U_{12} & U_{13} & U_{14} \\ U_{21} & U_{22} & U_{23} & U_{24} \\ U_{31} & U_{32} & U_{33} & U_{34} \\ U_{41} & U_{42} & U_{43} & U_{44} \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{pmatrix}$$

- ▶ $\mathbf{U}$ is unitary (i.e. $\mathbf{U}^{-1} = \mathbf{U}^\dagger$)
- ▶ In general, $\mathbf{U}$ does not have the form of a product gate.

Another product gate: The Hadamard $\otimes$ Identity gate

$$\mathbf{U}_{\text{Had} \otimes \mathbf{I}_2} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix}$$



$$\mathbf{v}_{out} = (\mathbf{U}_{\text{Had} \otimes \mathbf{I}_2}) \mathbf{v}_{in}$$

EXAMPLE

$$\mathbf{v}_{in} = |0\rangle \otimes |0\rangle \rightarrow \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Then

$$\mathbf{v}_{out} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \\ 0 \end{pmatrix} \rightarrow \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \otimes |0\rangle$$



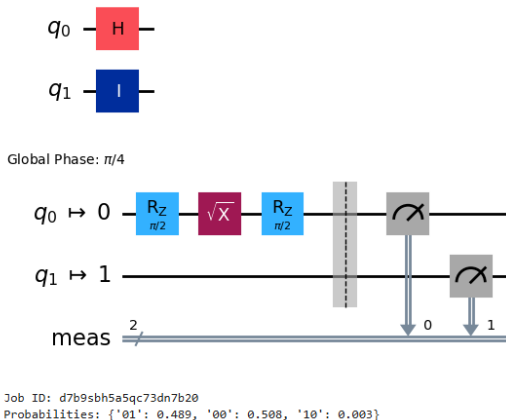
Run the Hadamard $\otimes$ Identity circuit at IBM

- ▶ Create the circuit with input Qbits are $|0\rangle$ and $|0\rangle$
- ▶ Execute 1000 times
- ▶ Print results

```
... PREAMBLE AS BEFORE ...
## First create a circuit locally
# 1. Create a 2-qubit circuit
qc = QuantumCircuit(2)
# 2a. Apply Hadamard gate to qbit 0 [WE CALLED THIS QBIT 1]
qc.h(0)
# 2b. Apply identity gate to qbit 1 [WE CALLED THIS QBIT 2]
qc.id(1)
# 2c. Display this meta-circuit
display(qc.draw("mpl"))
# 3. Measure
qc.measure_all()

...SET UP IBM HARDWARE AS BEFORE AND DRAW THE TRANSPILED CIRCUIT
display(isa_circuit.draw("mpl", idle_wires=False))
## Running and analyzing the circuit
# 8. Create the program to be run on the IBM cloud, which analyzes outputs of quantum circuit
sampler = Sampler(mode=backend)
# 9. Run the circuit with 1000 shots
job = sampler.run([isa_circuit], shots=1000)
print("Job ID:", job.job_id())
# 10. Get results (waits until job finishes)
result = job.result()
# 11. Extract counts
counts = result[0].data.meas.get_counts()
# 12. Change to probabilities and print
total = sum(counts.values())
probs = {k: v / total for k, v in counts.items()}
print("Probabilities:", probs)
```

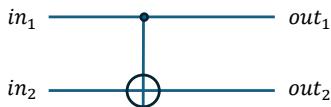
Results from the Hadamard $\otimes$ Identity run



- ▶ '01' : 0.489 means 0.489 is the fraction of outcomes with
Output Qbit #1 has z-value 0 and Output Qbit #2 has z-value 1
- ▶ We see that about half the results are $q_0 = |0\rangle$, half are $q_0 = |1\rangle$.
- ▶ In each of those results, $q_1 = |0\rangle$
- ▶ A **small** fraction (0.003) differ.

A general gate: The CNOT gate

$$\mathbf{U}_{\text{CNOT}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$



“Truth table” is

$$\begin{pmatrix} \mathbf{in_1} & \mathbf{in_2} & \mathbf{out_1} & \mathbf{out_2} \\ |0\rangle & |0\rangle & |0\rangle & |0\rangle \\ |0\rangle & |1\rangle & |0\rangle & |1\rangle \\ |1\rangle & |0\rangle & |1\rangle & |1\rangle \\ |1\rangle & |1\rangle & |1\rangle & |0\rangle \end{pmatrix}$$

Run the CNOT circuit at IBM

- ▶ Create the circuit with input Qbits are $|0\rangle$ and $|0\rangle$
- ▶ Execute 10000 times
- ▶ Print results

```
.....

# Create a 2-qubit circuit
qc = QuantumCircuit(2)

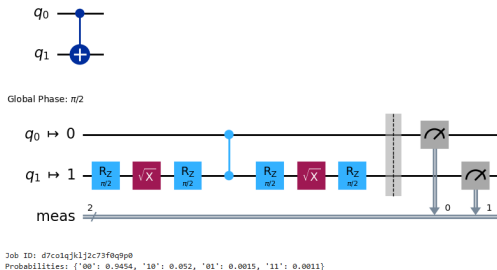
# Perform a CNOT (controlled-X) gate on qubit 1, controlled by qubit 0
qc.cx(0, 1)

.....

# Run the circuit with the number of shots
job = sampler.run([isa_circuit], shots=10000)

.....
```

Results from the CNOT run



- ▶ The transpiled circuit, in matrix notation is

$$U_{\text{CNOT}} = U_{I_1 \otimes \text{Had}} U_{\text{CZ}} U_{I_1 \otimes \text{Had}} = U_{I_1 \otimes \text{Had}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} U_{I_1 \otimes \text{Had}}$$

- ▶ '00' : 0.9454 means 0.9454 is the fraction of outcomes with
Output Qbit #1 has z-value 0 and Output Qbit #2 has z-value 0
- ▶ We see that about 5% of the results come from other Qbit combos.
- ▶ About 5% of results are *wrong*.

Another general gate: 2-Qbit EPR gate (*Bell gate*) – matrix representation

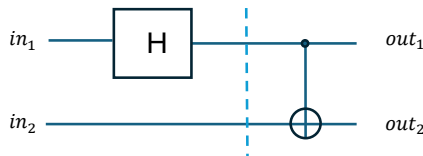
- ▶ Input the product state (2 independent Qbits) $\mathbf{v}_{in} = |0\rangle|0\rangle \rightarrow \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$.
- ▶ Construct gate with matrix $\mathbf{U} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{pmatrix}$.
- ▶ Apply the gate to the input-state: $\mathbf{v}_{out} = \mathbf{U}\mathbf{v}_{in}$.
- ▶ $\mathbf{v}_{out} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \rightarrow \frac{1}{\sqrt{2}} (|0\rangle|0\rangle + |1\rangle|1\rangle)$.
- ▶ This is the EPR state $|EPR\rangle$ (Sometimes called the (*Bell state*))

The EPR gate – pictorial representation

- ▶ There are **not** single physical components corresponding to each possible gate ($\mathbf{U}$).
- ▶ Instead, we build gates from standard components.
 - ▶ One standard gate followed by another, by another, looks like (in matrix representation) $\mathbf{U} = \mathbf{U}_1 \mathbf{U}_2 \dots \mathbf{U}_N$

$$\mathbf{U}_{\text{EPR}} = \mathbf{U}_{\text{CNOT}} \mathbf{U}_{\text{Had} \otimes \mathbf{I}_2}$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{pmatrix}$$



We previously saw that $U_{\text{EPR}} : |0\rangle|0\rangle \rightarrow \frac{1}{\sqrt{2}} (|0\rangle|0\rangle + |1\rangle|1\rangle)$.

Run the EPR circuit at IBM

- ▶ Create the circuit with input Qbits are $|0\rangle$ and $|0\rangle$
- ▶ Execute 10000 times
- ▶ Print results

.....

```
## First create a circuit locally
# Create a new circuit with two qubits
qc = QuantumCircuit(2)

# Add a Hadamard gate to qubit 0
qc.h(0)

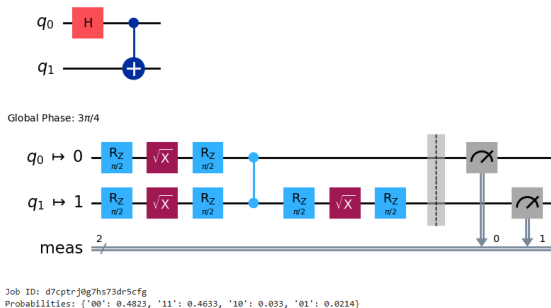
# Perform a controlled-X gate on qubit 1, controlled by qubit 0
qc.cx(0, 1)
```

.....

```
# Run the circuit with the number of shots
job = sampler.run([isa_circuit], shots=10000)
```

.....

Results from the EPR run



- ▶ '00' : .4823, '11': .4633 means *0.4823 is fraction of outcomes with Output Qbit #1 has z-value 0 and Output Qbit #2 has z-value 0 and 0.4633 is the fraction of outcomes with Output Qbit #1 has z-value 1 and Output Qbit #2 has z-value 1*
- ▶ The states $|0\rangle|0\rangle$ and $|1\rangle|1\rangle$ are about equally likely
- ▶ We see that about 5% of the results come from other Qbit combos.
- ▶ About 5% of results are *wrong*.

EPR with rotation

► Recall

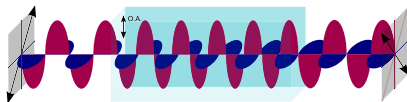


Figure: Half-wave plate. Different refractive indices for vertical and horizontal polarizations.

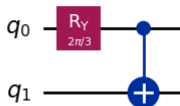
In code, this is `qc.z(q)` , operating on qbit q .

The matrix representation is $\mathbf{T} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

- Now rotate by π/x : $U = \mathbf{R}(\pi/x)\mathbf{T}\mathbf{R}(-\pi/x)$
`qc.ry(2*np.pi/x,0)  $\rightarrow (\cos(\frac{\pi}{x})|0\rangle - \sin(\frac{\pi}{x})|1\rangle)$ .`

EPR with rotation – results

- ▶ Hadamard gate $\rightarrow$ rotation gate; don't print transpiled circuit
Add a rotation gate to qubit 0
"np.pi" is the Python library value of pi
`qc.ry(2*np.pi/3,0)`
- ▶ New circuit should produce $(\cos(\frac{\pi}{3})|0\rangle|0\rangle - \sin(\frac{\pi}{3})|1\rangle|1\rangle)$



Job ID: d7e5sc95a5qc73dqig8g

Probabilities: {'11': 0.7205, '00': 0.257, '01': 0.0175, '10': 0.005}

- ▶ '00' : .257, '11' : .7205 means *0.257 is fraction of outcomes with Output Qbit #1 has z-value 0 and Output Qbit #2 has z-value 0 and 0.7205 is the fraction of outcomes with Output Qbit #1 has z-value 1 and Output Qbit #2 has z-value 1*
- ▶ Prob. of $|0\rangle|0\rangle$ is 0.25; Prob. of $|1\rangle|1\rangle$ is 0.75
- ▶ We see that about 2% of the results come from other Qbit combos.