Qiskit 개발자 자격 시험

Inho Choi

Qiskit Advocate

Qiskit

구성

Qiskit

- Lecture 1: 게이트와 양자 회로
- Lecture 2: 양자 회로의 측정과 OpenQasm
- Lecture 3: 양자 백엔드에 양자회로 실행하기
- Lecture 4: 양자 회로 및 회로의 정보와 실행결과를 해석하기
- Lecture 5: 유용한 기능들

Lecture 4: 양자 회로 및 회로의 정보와 실행결과를 해석하기

1. 양자 회로 시각화
2. 카운트 시각화
3. 상태 시각화
4. 백엔드 시각화

Qiskit

회로 시각화

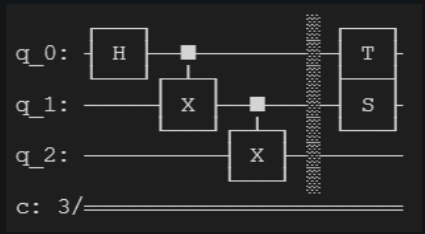
QuantumCircuit.draw

Qiskit

- text: 콘솔로 나타낼 수 있는 텍스트의 형태로 양자 회로를 출력합니다
- mpl: 파이썬에 색이 있는 이미지로 matplotlib을 사용해 출력합니다
- latex: latex의 형태로 높은 수준의 이미지를 컴파일 하여 출력합니다
- latex_source: 컴파일 되지 않은 latex를 출력합니다

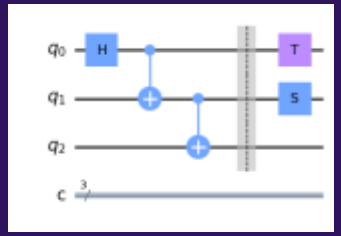
text

```
qc.draw(output='text')
```



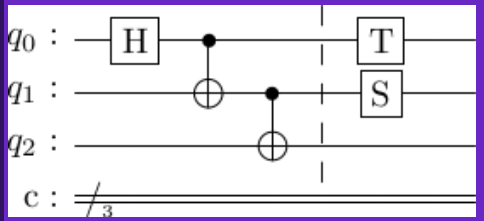
mpl

```
qc.draw(output='mpl')
```



latex

```
qc.draw(output='latex')
```



latex_source

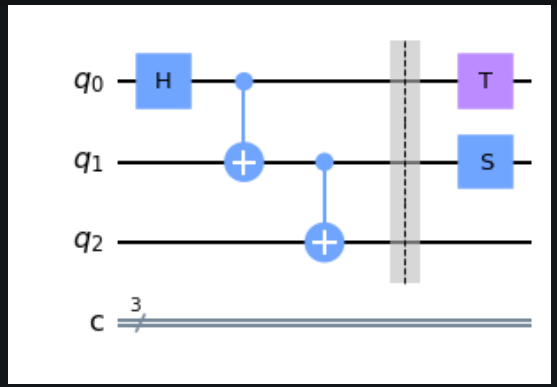
```
qc.draw(output='latex_source')
```

```
'\\documentclass[border=2px]{standa
lone}\n\n\\usepackage[braket,
qm]{qcircuit}\n\\usepackage{graphic
x}\n\n\\begin{document}\n\\scalebox
{1.0}{\n\\Qcircuit @C=1.0em
@R=0.2em @!R { \\\\\n\t
\t\\nghost{{q}_{0} : } &
\\lstick{{q}_{0} : } &
\\gate{\\mathrm{H}} & \\ctrl{1} &
\\qw \\barrier[0em]{2} & \\qw &
\\gate{\\mathrm{T}} & \\qw &
\\qw\\\\\n\t \t\\nghost{{q}_{1} : }
& \\lstick{{q}_{1} : } & \\qw &
\\targ & \\ctrl{1} & \\qw &
\\gate{\\mathrm{S}} & \\qw &
\\qw\\\\\n\t \t\\nghost{{q}_{2} : }
& \\lstick{{q}_{2} : } & \\qw &
\\qw & \\targ & \\qw & \\qw & \\qw
& \\qw\\\\\n\t
\t\\nghost{\\mathrm{{c} : }} &
\\lstick{\\mathrm{{c} : }} &
\\lstick{/_{_{3}}} \\cw & \\cw &
\\cw & \\cw & \\cw & \\cw &
\\cw\\\\\n\\\\ }}\n\\end{document}'
```

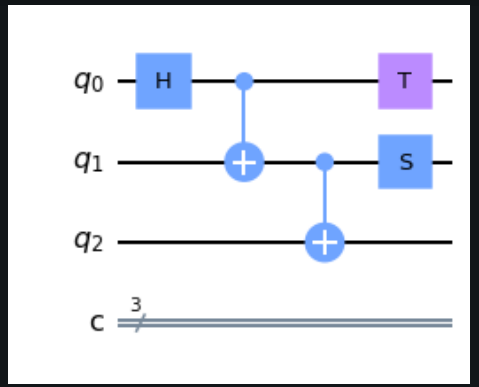
배리어 표기, plot_barriers

Default

```
qc.draw(output='mpl', plot_barriers=True)
```




```
qc.draw(output='mpl', plot_barriers=False)
```

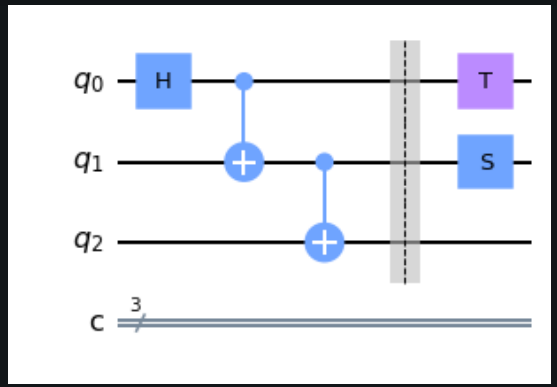


비트 역순 표기, reverse_bits

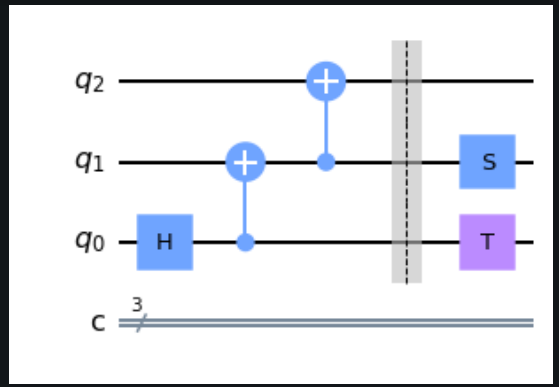
Qiskit

Default

```
qc.draw(output='mpl', reverse_bits=False)
```



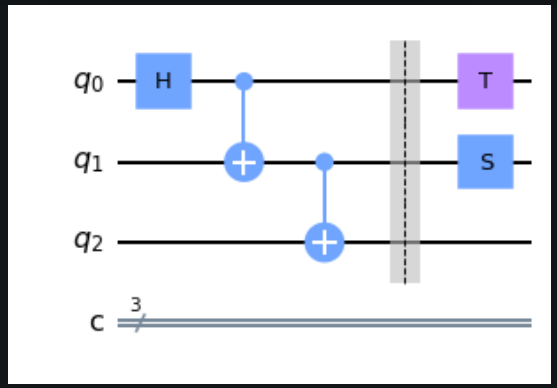

```
qc.draw(output='mpl', reverse_bits=True)
```



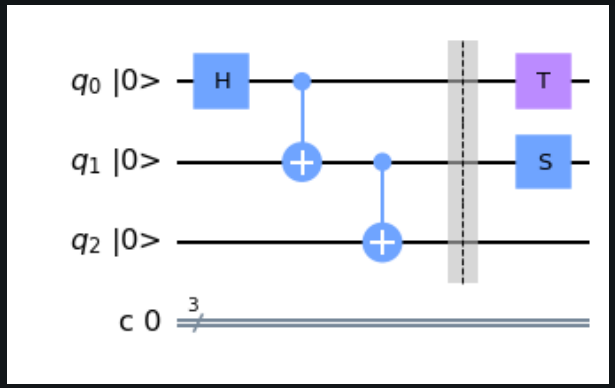
초기 상태 표기, initial_state

Default

```
qc.draw(output='mpl', initial_states=False)
```




```
qc.draw(output='mpl', initial_states=True)
```



Qiskit

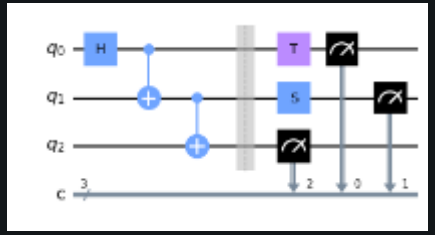
카운트 시각화

카운트 시각화

```
from qiskit.visualization import plot_histogram
```

카운트 시각화

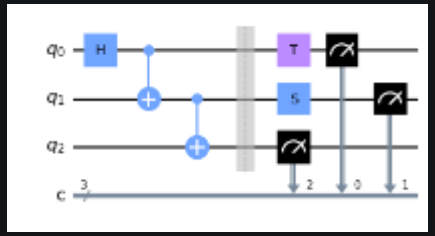
```
from qiskit.visualization import plot_histogram
```



qc

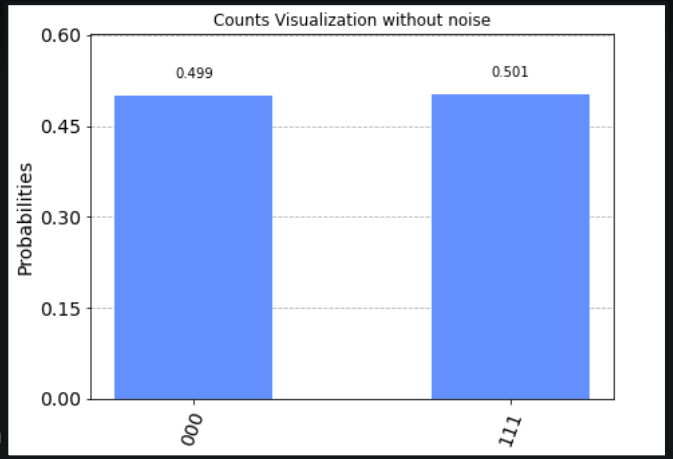
카운트 시각화

```
from qiskit.visualization import plot_histogram
```

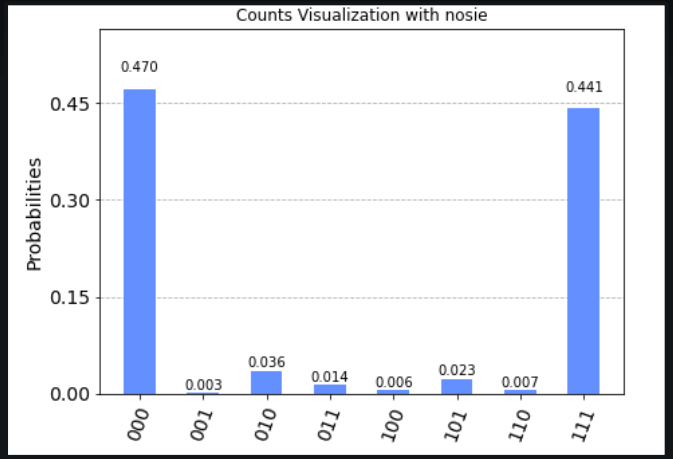


qc

```
plot_histogram(counts,title="Counts Visualization without noise")
```



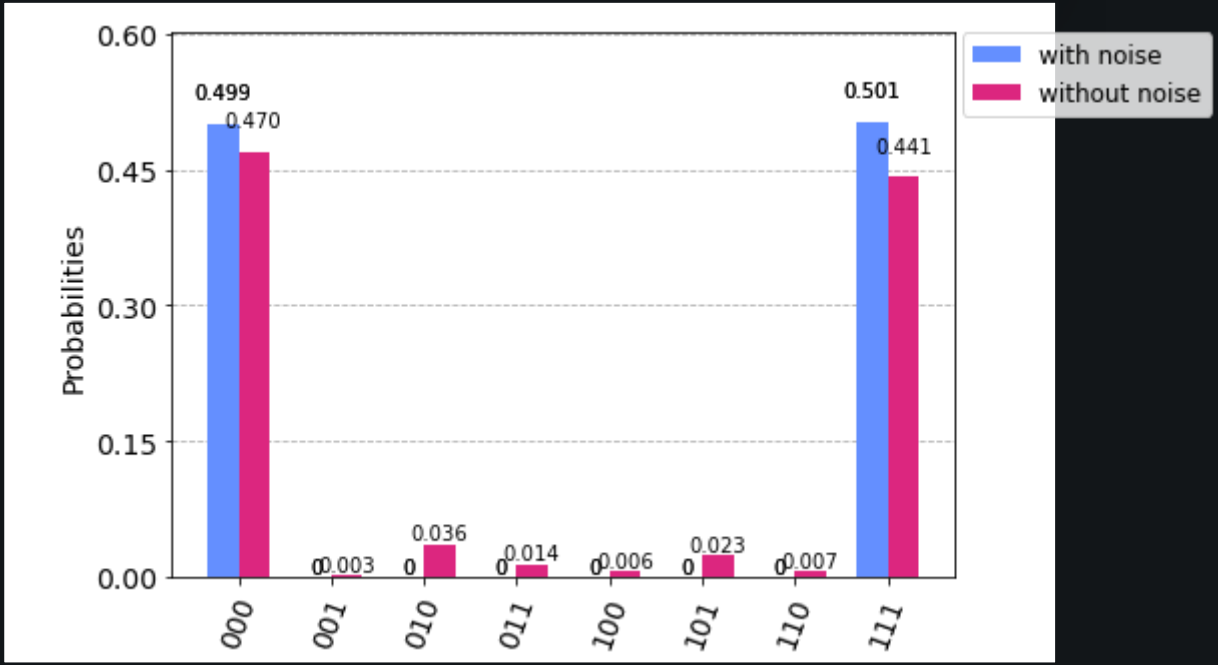

```
plot_histogram(counts2,title="Counts Visualization with nosie")
```



plot_histogram

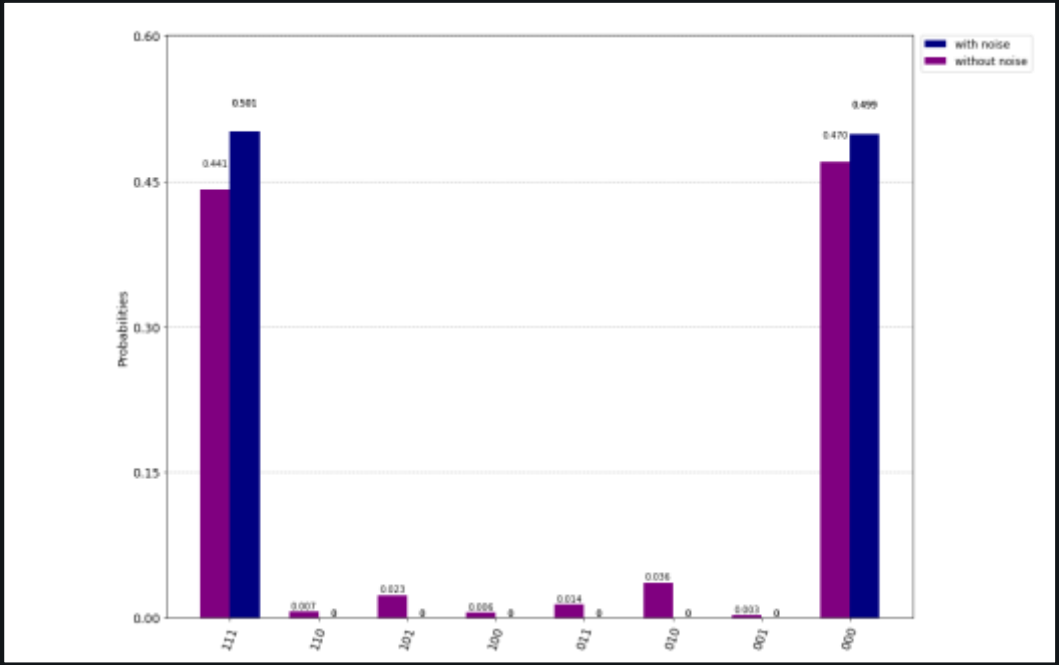
Qiskit

```
plot_histogram([counts,counts2],legend=['with noise', 'without noise'])
```



plot_histogram

```
plot_histogram([counts,counts2],legend=['with noise', 'without noise'], sort='desc', figsize=(15,12), color=['navy', 'purple'])
```



Qiskit

상태 시각화

상태 시각화

Qiskit

큐비트의 상태 벡터를 여러가지의 방법으로 시각화 하여 다양한 정보를 나타냄

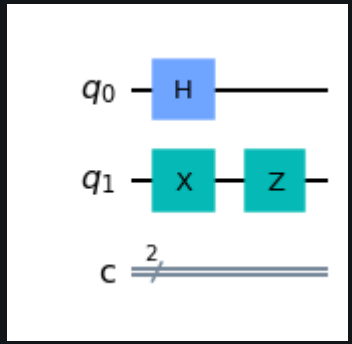
1. `plot_state_city`
2. `plot_state_hinton`
3. `plot_state_qsphere`
4. `plot_state_paulivec`
5. `plot_bloch_multivector`

상태 시각화

Qiskit

큐비트의 상태 벡터를 여러가지의 방법으로 시각화 하여 다양한 정보를 나타냄

1. plot_state_city
2. plot_state_hinton
3. plot_state_qsphere
4. plot_state_paulivec
5. plot_bloch_multivector



qc

```
[ 0.          +0.00000000e+00j  0.          +0.00000000e+00j
 -0.70710678-1.73191211e-16j -0.70710678-1.73191211e-16j]
```

state_vector

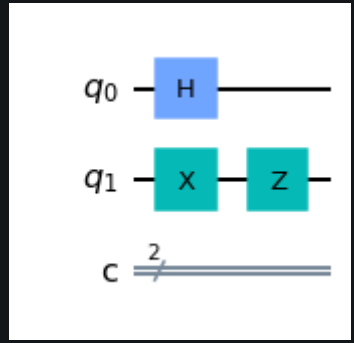
$$-\frac{\sqrt{2}}{2}|10\rangle - \frac{\sqrt{2}}{2}|11\rangle$$

plot_state_city

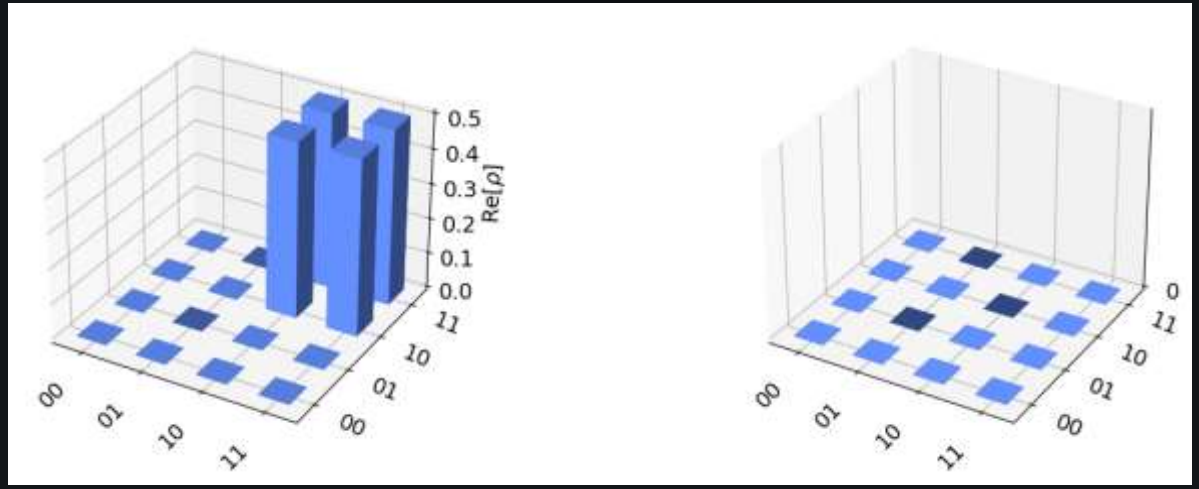
Qiskit

$$-\frac{\sqrt{2}}{2}|10\rangle - \frac{\sqrt{2}}{2}|11\rangle$$

```
plot_state_city(state_vector)
```



qc

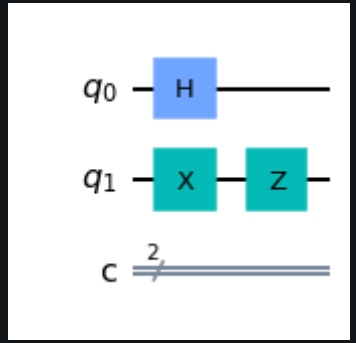


plot_state_hinton

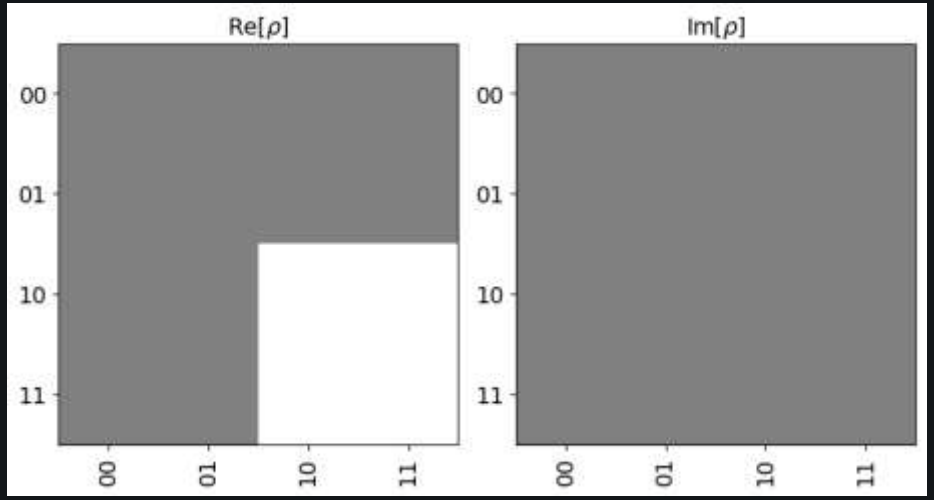
Qiskit

$$-\frac{\sqrt{2}}{2}|10\rangle - \frac{\sqrt{2}}{2}|11\rangle$$

```
plot_state_hinton(state_vector)
```



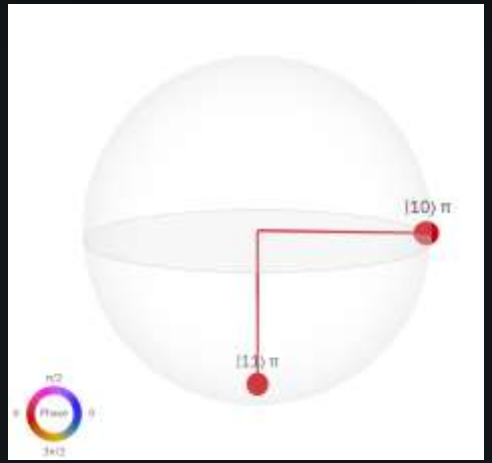
qc

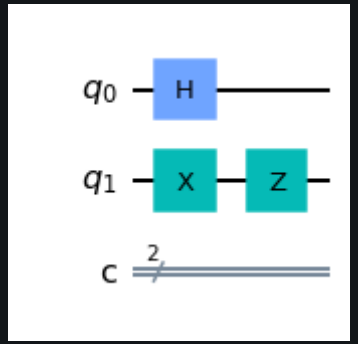


plot_state_hinton

$$-\frac{\sqrt{2}}{2}|10\rangle - \frac{\sqrt{2}}{2}|11\rangle$$

```
plot_state_qsphere(state_vector)
```





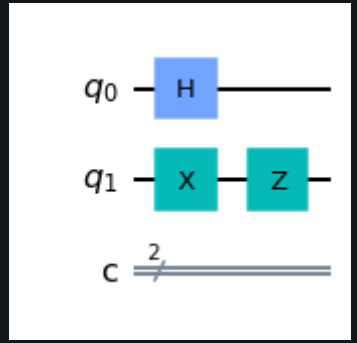
qc

plot_state_paulivec

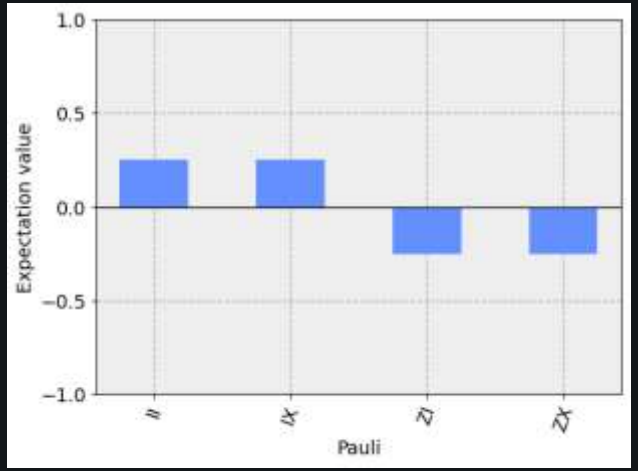
Qiskit

$$-\frac{\sqrt{2}}{2}|10\rangle - \frac{\sqrt{2}}{2}|11\rangle$$

```
plot_state_paulivec(state_vector)
```



qc

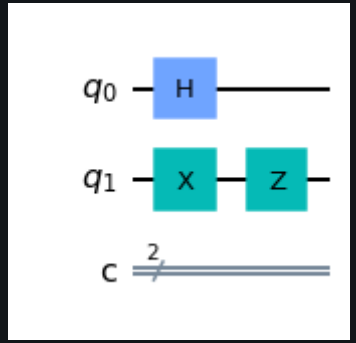


plot_state_multivec

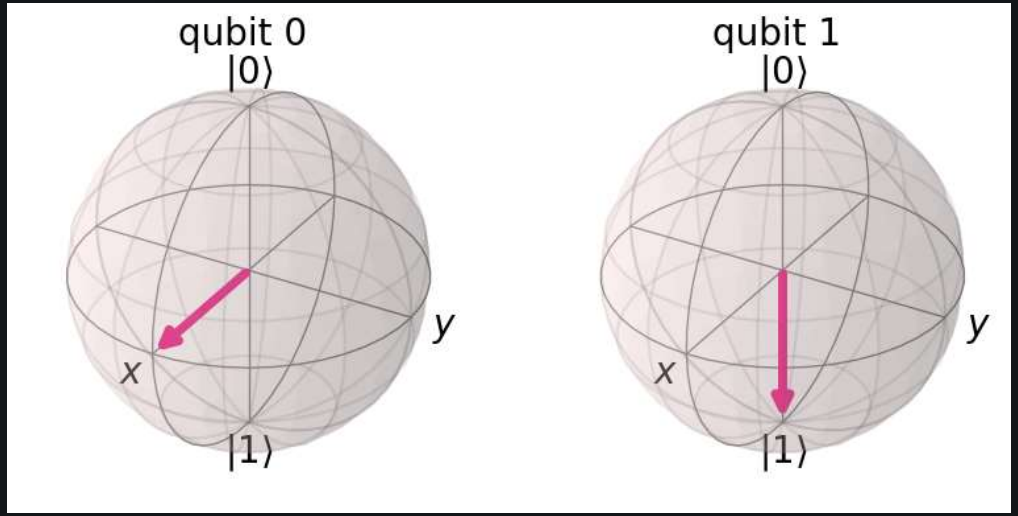
Qiskit

$$-\frac{\sqrt{2}}{2}|10\rangle - \frac{\sqrt{2}}{2}|11\rangle$$

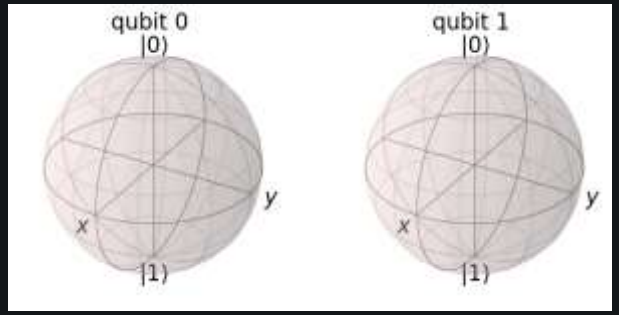
```
plot_bloch_multivector(state_vector)
```

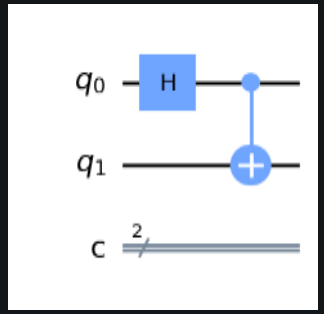


qc



Entangled State



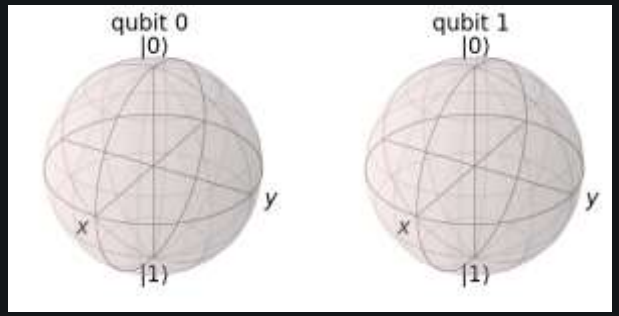


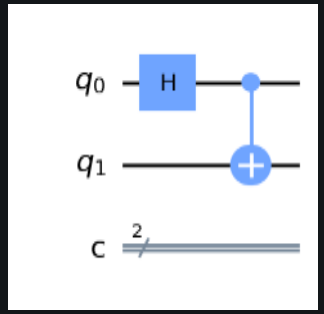
qc

Qiskit

$$\frac{\sqrt{2}}{2}|00\rangle + \frac{\sqrt{2}}{2}|11\rangle$$

Entangled State

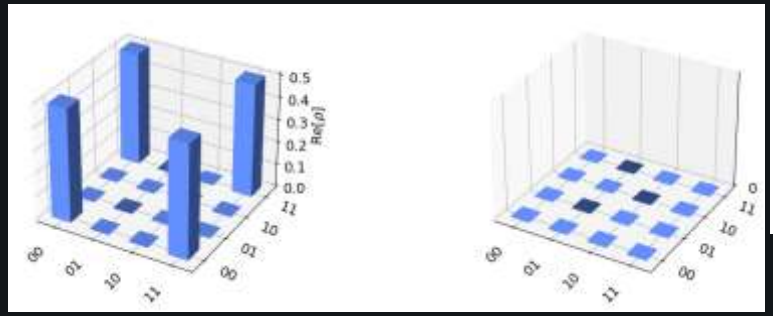


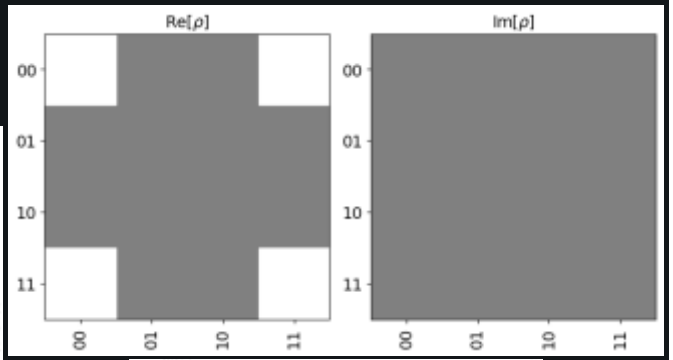


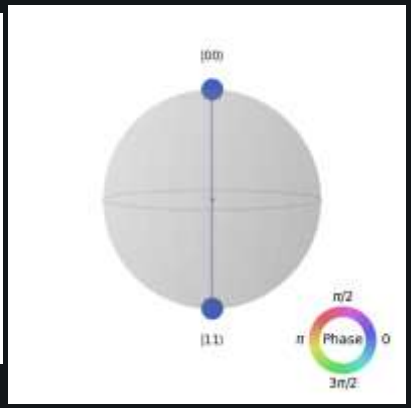
qc

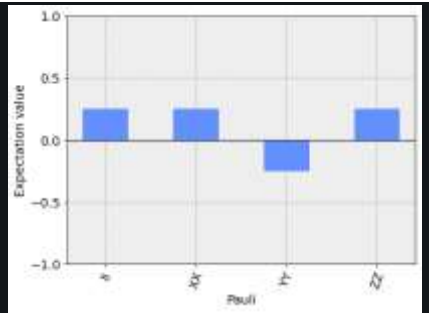
Qiskit

$$\frac{\sqrt{2}}{2}|00\rangle + \frac{\sqrt{2}}{2}|11\rangle$$









Qiskit

백엔드 시각화

백엔드 시각화

- 게이트맵
- 오류맵
- 회로 레이아웃

Qiskit

개이트맵

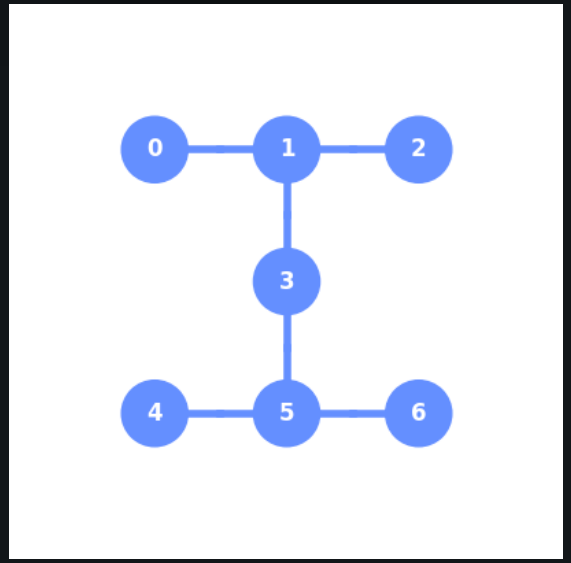
Qiskit

```
provider=IBMQ.load_account()
backend=provider.get_backend("ibm_oslo")
```

개이트맵 & 오류맵

```
provider=IBMQ.load_account()
backend=provider.get_backend("ibm_oslo")
```

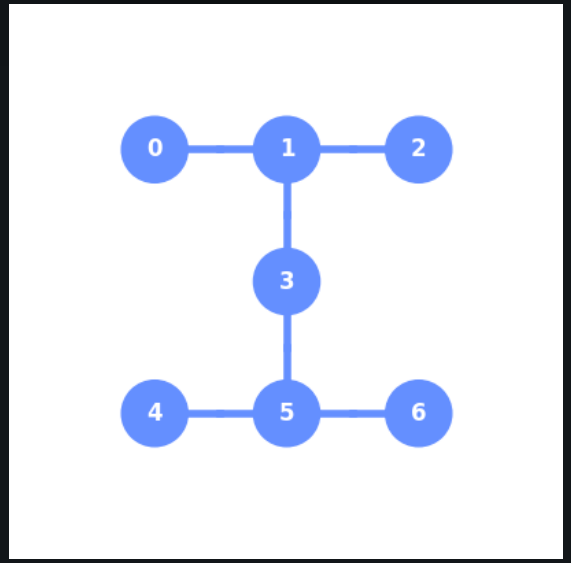
```
plot_gate_map(backend)
```



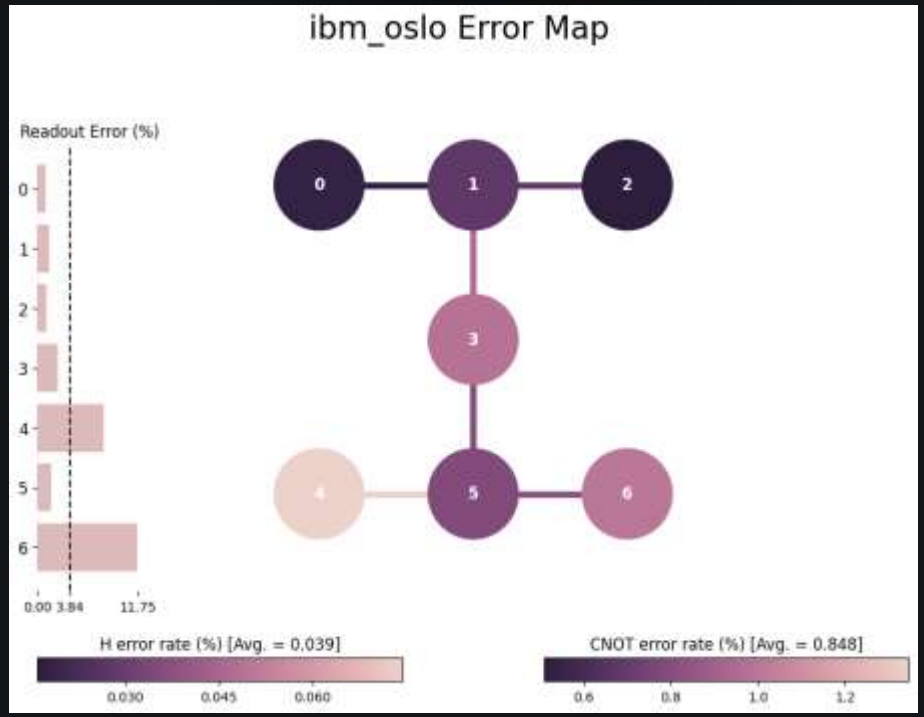
개이트맵 & 오류맵

```
provider=IBMQ.load_account()
backend=provider.get_backend("ibm_oslo")
```

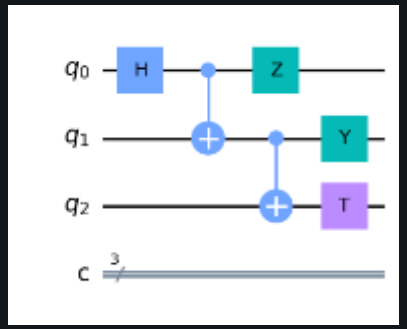
```
plot_gate_map(backend)
```




```
plot_error_map(backend)
```

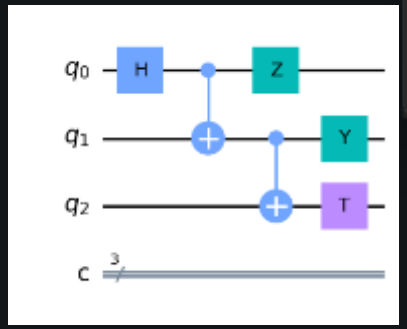


회로 레이아웃



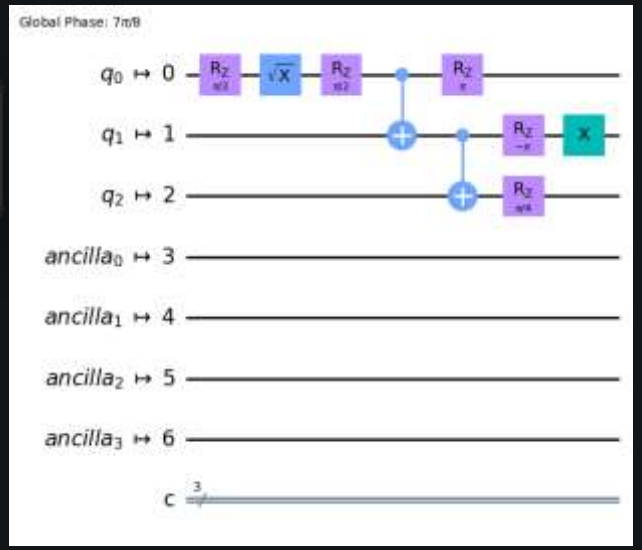
qc

회로 레이아웃



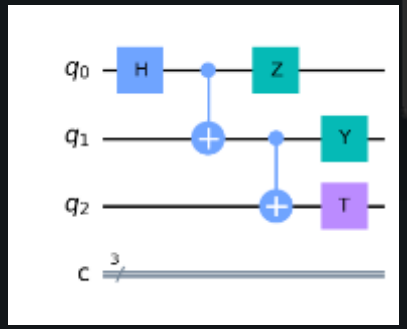
qc

```
qc_transpile=transpile(qc,backend)
```



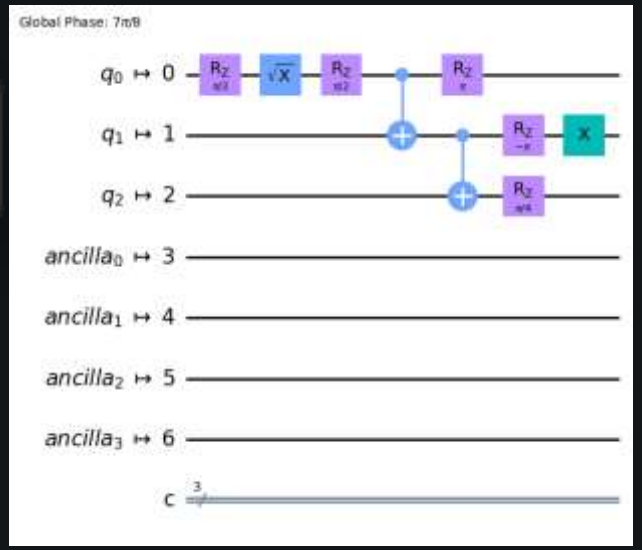
qc_transpile

회로 레이아웃

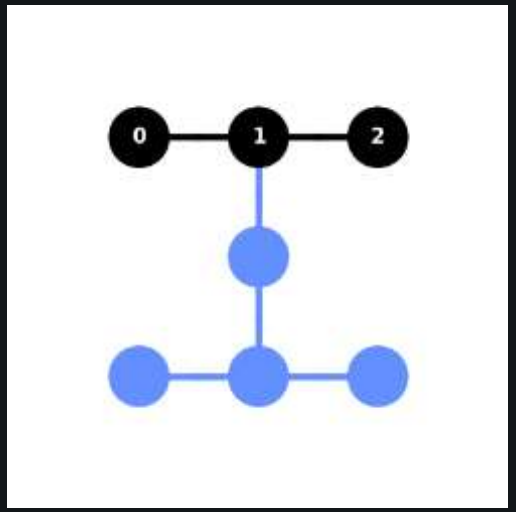


qc

```
qc_transpile=transpile(qc,backend)
```

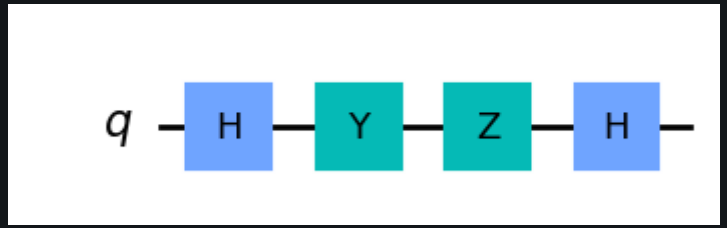


qc_transpile

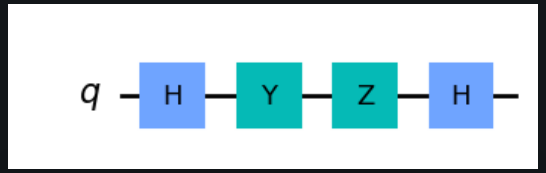



```
plot_circuit_layout(qc_transpile,backend)
```

애니메이션



qc




```
visualize_transition(qc)
```

Qiskit