Qiskit 기초 양자 게이트 가이드 (IBM-Q)

기초 양자 게이트 가이드 (IBM-Q)

1. IBM-Q 기본 게이트

IBM-Q는 **5개 기본 게이트**만 물리적으로 구현함:

게이트	타입	역할
`x`	1-qubit	비트 반전 (NOT)
`sx`	1-qubit	√X, X축 π/2 회전
`rz`	1-qubit	Z축 회전 (위상 조정)
`id`	1-qubit	항등 (지연용)
`ecr`	2-qubit	얽힘 생성

중요: 이 문서의 다른 모든 게이트는 위 5개로 자동 분해됨.

---

2. 단일 큐비트 게이트

2.1 파울리 게이트 (Pauli Gates)

기본적인 비트/위상 반전 게이트.

X 게이트 ⊕ Native

```
qc.x(0)
```

- **행렬**: $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
- **작용**: $|0\rangle \leftrightarrow |1\rangle$ (비트 반전)
- **블로흐 구**: X축 중심 π 회전
- **특징**: 고전 NOT 게이트의 양자 버전

Y 게이트

```
qc.y(0)   # → RZ + SX로 분해
```

- **행렬**: $\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$
- **블로흐 구**: Y축 중심 π 회전
- **분해**: $Y = iXZ$

Z 게이트

```
qc.z(0)  # → RZ(π)로 분해
```

- **행렬**: $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$
- **작용**: $|1\rangle \to -|1\rangle$ (위상 반전)
- **블로흐 구**: Z축 중심 π 회전

2.2 하다마드 게이트 (Hadamard)

H 게이트

```
qc.h(0)  # → RZ + SX + RZ로 분해
```

- **행렬**:

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

- **작용**:
  - $|0\rangle \to |+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$
  - $|1\rangle \to |-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$
- **블로흐 구**: $(X + Z)/\sqrt{2}$ 축 중심 π 회전
- **핵심 용도**: 중첩 상태 생성, 기저 변환

예제: 중첩 생성

```
qc = QuantumCircuit(1)
qc.h(0)
qc.measure_all()
# 결과: '0'과 '1'이 50:50 확률
```

2.3 회전 게이트 (Rotation Gates)

임의 각도 회전.

RZ 게이트 ⊕ Native

```
qc.rz(theta, 0)
```

- **행렬**: $\begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix}$
- **블로흐 구**: Z축 중심 θ 회전
- **특징**: 위상만 변경, 측정 확률 불변

SX 게이트 ⊕ Native

```
qc.sx(0)
```

- **행렬**: $\frac{1}{2}\begin{pmatrix} 1+i & 1-i \\ 1-i & 1+i \end{pmatrix}$
- **블로흐 구**: X축 중심 π/2 회전
- **특징**: $SX \cdot SX = X$ (X의 제곱근)

RX 게이트

```
qc.rx(theta, 0)  # → RZ + SX로 분해
```

- **블로흐 구**: X축 중심 θ 회전

RY 게이트

```
qc.ry(theta, 0)  # → RZ + SX로 분해
```

- **블로흐 구**: Y축 중심 θ 회전
- **특징**: 측정 확률을 직접 조정 (위상 변화 없음)

2.4 위상 게이트 (Phase Gates)

Z축 회전의 특수 케이스.

게이트	회전 각도	코드	관계
S	π/2	`qc.s(0)`	$\sqrt{Z}$
S†	-π/2	`qc.sdg(0)`	S^{-1}

게이트	회전 각도	코드	관계
T	π/4	`qc.t(0)`	$\sqrt{S}$
T†	-π/4	`qc.tdg(0)`	T^{-1}

- **행렬 형태**: $\begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}$
- **블로흐 구**: 모두 Z축 회전
- **모두 RZ로 분해됨**

2.5 범용 게이트 (Universal Gate)

U 게이트

```
qc.u(theta, phi, lam, 0)  # → RZ + SX로 분해
```

- **매개변수**: θ (극각), φ, λ (방위각)
- **특징**: 모든 단일 큐비트 유니터리를 표현 가능
- **분해**: $U = R_Z(\phi) \cdot R_Y(\theta) \cdot R_Z(\lambda)$

---

3. 다중 큐비트 게이트

3.1 ECR 게이트 ⊕ Native

```
qc.ecr(0, 1)
```

- **행렬**:

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

- **블로흐 구 (타겟 큐비트 기준)**:
  - 제어 = |0⟩: 타겟을 $(1, -1, 0)$ 축 중심으로 π/2 회전
  - 제어 = |1⟩: 타겟을 $(1, 1, 0)$ 축 중심으로 π/2 회전
- **타입**: 2-qubit 얽힘 게이트
- **특징**: IBM-Q의 유일한 물리적 2-qubit 게이트
- **역할**: 모든 2-qubit 게이트의 기반

3.2 CNOT 게이트

```
qc.cx(0, 1)  # 0=제어, 1=타겟
```

- **행렬**: $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
- **작용**:
  - 제어 비트 = 0 → 타겟 불변
  - 제어 비트 = 1 → 타겟 반전
- **분해**: ECR + 단일 큐비트 게이트

예제: 벨 상태 생성

```
qc = QuantumCircuit(2)
qc.h(0)          # 중첩 생성
qc.cx(0, 1)      # 얽힘 생성
# 결과: (|00⟩ + |11⟩)/√2
```

3.3 제어 게이트 (Controlled Gates)

제어 큐비트 = 1일 때만 타겟에 게이트 적용.

기본 제어 게이트

```
qc.cz(0, 1)       # Controlled-Z
qc.cy(0, 1)       # Controlled-Y
qc.ch(0, 1)       # Controlled-H
qc.cp(theta, 0, 1)  # Controlled-Phase
```

제어 회전 게이트

```
qc.crx(theta, 0, 1)  # Controlled-RX
qc.cry(theta, 0, 1)  # Controlled-RY
qc.crz(theta, 0, 1)  # Controlled-RZ
```

특징: 모두 ECR과 기본 게이트로 분해됨.

3.4 SWAP 게이트

```
qc.swap(0, 1)
```

- **작용**: 두 큐비트의 상태 교환
- **예**: $|01\rangle \leftrightarrow |10\rangle$
- **분해**: 3개의 CNOT으로 구성
    - $SWAP = CNOT_{0,1} \cdot CNOT_{1,0} \cdot CNOT_{0,1}$

3.5 토폴리 게이트 (Toffoli, CCX)

```
qc.ccx(0, 1, 2)  # 0,1=제어, 2=타겟
```

- **작용**: 제어 비트 2개가 모두 1일 때만 타겟 반전
- **예**: $|110\rangle \rightarrow |111\rangle$
- **용도**:
    - 고전 AND 게이트 구현
    - 가역 논리 연산
    - 전가산기 구성

전가산기 예제

```
qc = QuantumCircuit(4)  # a, b, carry_in, carry_out
qc.ccx(0, 1, 3)  # carry = a AND b
qc.cx(0, 1)      # temp = a XOR b
qc.ccx(1, 2, 3)  # carry |= temp AND carry_in
qc.cx(1, 2)      # sum = temp XOR carry_in
```

3.6 프레드킨 게이트 (Fredkin, CSWAP)

```
qc.cswap(0, 1, 2)  # 0=제어, 1↔2 교환
```

- **작용**: 제어 비트 = 1일 때 타겟 2개 교환
- **용도**: 가역 MUX 구현

---

4. 게이트 분해 (Decomposition)

분해 과정

```
상위 게이트 (H, CNOT, ...)
       ↓ 컴파일러 자동 분해
기본 게이트 (id, rz, sx, x, ecr)
       ↓ 하드웨어 실행
물리적 큐비트 조작
```

분해 확인하기

```python
from qiskit import transpile

qc = QuantumCircuit(1)
qc.h(0)

# 기본 게이트로 분해
transpiled = transpile(qc, basis_gates=['id', 'rz', 'sx', 'x', 'ecr'])
print(transpiled.draw())
# 출력: RZ(π/2) - SX - RZ(π/2)
```

최적화 팁

1. **기본 게이트 우선 사용**: RZ, SX 직접 사용으로 분해 비용 감소
2. **게이트 합치기**: 연속된 RZ → 단일 RZ로 통합
3. **회로 깊이 최소화**: 게이트 수 ↓ → 오류 ↓
4. **Transpile 활용**: 자동 최적화 수행

```python
optimized_qc = transpile(qc,
                         basis_gates=['id', 'rz', 'sx', 'x', 'ecr'],
                         optimization_level=3)
```

---

5. 블로흐 구 요약

회전 축별 게이트

축	게이트 예시	효과
X축	X, SX, RX, H	비트 반전, 측정 확률 변경
Y축	Y, RY	비트+위상 동시 변경
Z축	Z, S, T, RZ	위상만 변경 (확률 불변)

주요 상태

```
         |0⟩ (북극)
          ↑
    +----|----+ (적도: |+⟩, |-⟩, |+i⟩, |-i⟩)
          ↓
         |1⟩ (남극)
```

게이트별 회전 정리

- **X**: X축 π 회전 → 북극↔남극
- **H**: (X+Z)/√2축 π 회전 → 북극↔적도(|+⟩)
- **S**: Z축 π/2 회전 → 적도 면에서 90° 위상 이동
- **T**: Z축 π/4 회전 → 적도 면에서 45° 위상 이동
- **RY(θ)**: Y축 θ 회전 → 북극에서 남극으로 경로 조정

---

6. 빠른 참조

Native 게이트만 사용하기

```
# ✅ 효율적 (직접 실행)
qc.x(0)
qc.sx(0)
qc.rz(np.pi/4, 0)
qc.ecr(0, 1)

# ⚠ 비효율적 (분해 필요)
qc.h(0)       # → RZ + SX + RZ
qc.cnot(0,1) # → ECR + 단일 게이트들
```

자주 쓰는 패턴

```
# 중첩 생성
qc.h(0)

# 벨 상태 (최대 얽힘)
qc.h(0)
qc.cx(0, 1)

# 위상 킥백 (Phase Kickback)
qc.h(1)
```

```python
qc.cx(0, 1)
qc.h(1)  # → CZ와 동등
```

게이트 카운트 줄이기

```python
# Before: 2 게이트
qc.rz(np.pi/4, 0)
qc.rz(np.pi/3, 0)

# After: 1 게이트
qc.rz(np.pi/4 + np.pi/3, 0)
```