NISQ+: Boosting quantum computing power by approximating quantum error correction (ISCA'20)

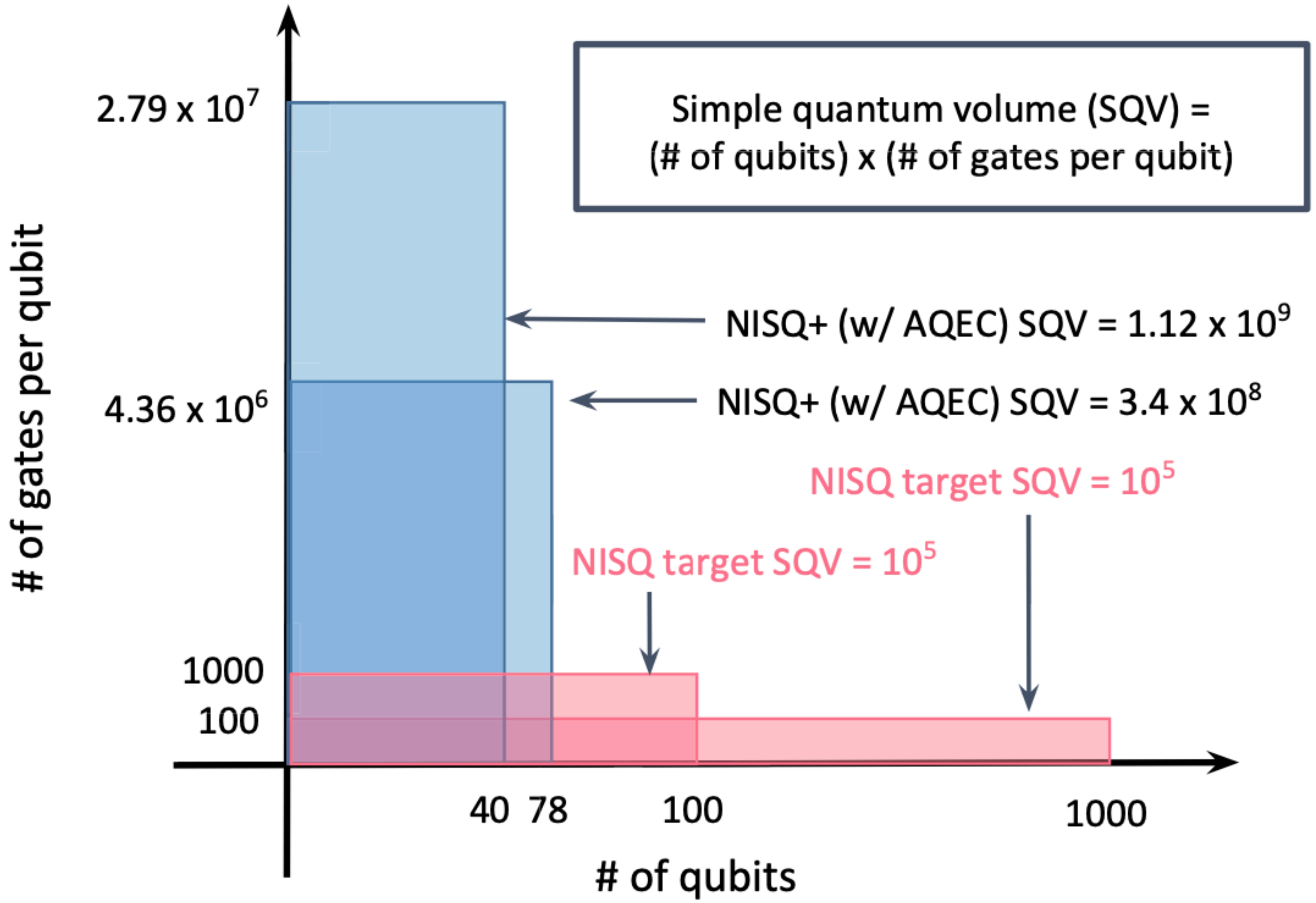
Presenter: Jinyoung Choi

Why Quantum Computing is hard?

- Now we can build NISQ(Noisy Intermediate-Scale Quantum)
  - We have **imperfect control** over qubits
    - **Fragile state (to 0> state)**
    - **Device noise**
  - Qubits are error-prone
  - Intermediate-scale qubits (50- few hundreds)
  - Simulation cost by brute force

Extreme Storage & Computer Architecture Lab

Hard to scale up qubits



1. SQV is limited when we have more cubits
2. AQEC(= error correction) can boost up SQV

Error correction in classical computing

- Initial bit — suppose you have one bit to send
  - 0 & 1
- The channel has noise to disturb the data (bit)
- This can be **mitigated** by repeating bits by encoding
  - 0 -> 000
  - 1 -> 111
- We can correct one-bit error by **majority vote**
  - 000 -> 000 (no error)
  - 001 -> 000 (error on 1st bit / bit flip 0->1)
  - 010 -> 000 (error on 2nd bit / bit flip 0->1)
  - 011 -> 111 (error on 1st bit / bit flip 1->0)
  - 100 -> 000 (error on 1st bit / bit flip 0->1)
  - 101 -> 111 (error on 2nd bit / bit flip 1-> 0)
  - 110 -> 111 (error on 1st bit / bit flip 1->0)
  - 111 -> 111 (no error)

prob. for bit error $p < 1$:

multi-bit error prob. $= 3p^2(1-p)+p^3=3p^2-2p^3$

$< p$ when $p < 0.5$

ESCAL Extreme Storage & Computer Architecture Lab

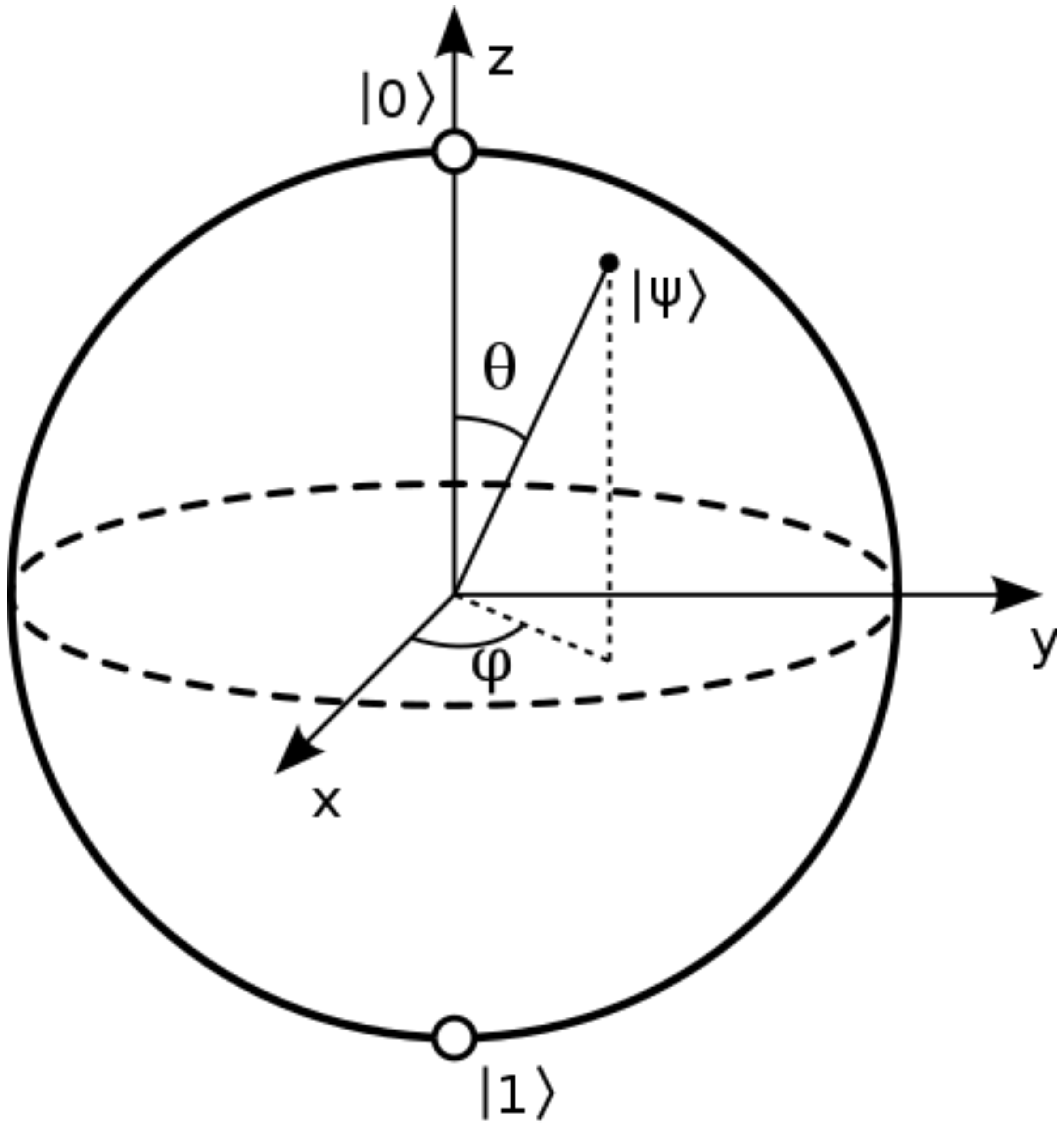
UC RIVERSIDE

Error correction in classical computing

- n,k,d = [3,1,3]
  - 3 bit
  - 1 ecoding bit
  - 3bit can be corrected with code distance 1 (d= 2t+1)

Extreme Storage & Computer Architecture Lab

General Qubit state



Bloch sphere representation of a qubit. The probability amplitudes for the superposition state, $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,$ are given by

$$\alpha = \cos\left(\frac{\theta}{2}\right) \text{ and } \beta = e^{i\phi}\sin\left(\frac{\theta}{2}\right)$$

Extreme Storage & Computer Architecture Lab

Errors in Qubits

- Bit flip
  - Error channel where (p = without error)
  - a|0> + b|1> —> b|0> + a|1>

Bit flip channel

$$E_0 = \sqrt{p}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, E_1 = \sqrt{1-p}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Extreme Storage & Computer Architecture Lab

UC RIVERSIDE

Errors in Qubits

- Phase flip
  - Error channel where (p = without error)
  - a|0> + b|1> —> a|0> - b|1>

Phase flip channel

$$E_0 = \sqrt{p}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, E_1 = \sqrt{1-p}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

ESCAL Extreme Storage & Computer Architecture Lab

UC RIVERSIDE

Errors in Qubits

- Bit-phase flip
  - Error channel where (p = without error)
  - a|0> + b|1> —> -bi|0> + ai|1>

Bit-phase flip channel

$$E_0 = \sqrt{p}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, E_1 = \sqrt{1-p}Y = \sqrt{1-p}\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

Pauli equation

These matrices are usually represented as

$$X = \sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$
$$Y = \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix},$$
$$Z = \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

The Pauli matrices are involutory, meaning that the square of a Pauli matrix is the identity matrix.

$$I^2 = X^2 = Y^2 = Z^2 = -iXYZ = I$$

$$\mathbb{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{6}$$

The single-qubit coherent error process described in equation (5) can be expanded in the above basis as follows

$$U(\delta\theta, \delta\phi)\,|\psi\rangle = \alpha_I \mathbb{1}\,|\psi\rangle + \alpha_X X\,|\psi\rangle + \alpha_Z Z\,|\psi\rangle + \alpha_Y Y\,|\psi\rangle \tag{7}$$

where $\alpha_{I,X,Y,Z}$ are the expansion coefficients. By noting that the Pauli Y-matrix is equivalent (up to a phase) to the product XZ, this expression can be further simplified to

$$U(\delta\theta, \delta\phi)\,|\psi\rangle = \alpha_I \mathbb{1}\,|\psi\rangle + \alpha_X X\,|\psi\rangle + \alpha_Z Z\,|\psi\rangle + \alpha_{XZ} XZ\,|\psi\rangle\,. \tag{8}$$

Quantum error types

As a result of the digitisation of the error there are two fundamental quantum error-types that need to be accounted for by quantum codes. Pauli X-type errors can be thought of as quantum bit-flips that map $X|0\rangle = |1\rangle$ and $X|1\rangle = |0\rangle$. The action of an X-error on the general qubit state is

$$X|\psi\rangle = \alpha X|0\rangle + \beta X|1\rangle = \alpha|1\rangle + \beta|0\rangle\,. \tag{9}$$

The second quantum error type, the Z-error, is often referred to as a phase-flip and has no classical analogue. Phase-flips map the qubit basis states $Z|0\rangle = |0\rangle$ and $Z|1\rangle = -|1\rangle$, and therefore have the following action on the general qubit state

$$Z|\psi\rangle = \alpha Z|0\rangle + \beta Z|1\rangle = \alpha|0\rangle - \beta|1\rangle\,. \tag{10}$$

How to correct bit flip error?

$$|0\rangle_L = |000\rangle$$
$$|1\rangle_L = |111\rangle$$

This works against bit flips: $\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

$$\sigma_x(2): \begin{array}{lcl} |000\rangle_L & = & |010\rangle \\ |111\rangle_L & = & |101\rangle \end{array}$$

Can measure "which bit is different?" This measurement is a projection onto one of four 2-d spaces, generated by the vectors:

$$\{|000\rangle, |111\rangle\} \qquad \{|100\rangle, |011\rangle\}$$
$$\{|010\rangle, |101\rangle\} \qquad \{|001\rangle, |110\rangle\}$$

Possible answers: none, bit 1, bit 2, bit 3.
Applying σ_x to incorrect bit corrects error.

$$|0\rangle_L = |000\rangle$$
$$|1\rangle_L = |111\rangle$$

We can encode superpositions using this code

$$\alpha|0\rangle_L + \beta|1\rangle_L = \alpha|000\rangle + \beta|111\rangle$$

Suppose we have a bit error $\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ in the second bit:

$$\sigma_x(2): \alpha|000\rangle + \beta|111\rangle \rightarrow \alpha|010\rangle + \beta|101\rangle$$

When this is measured, the result is "bit 2 is flipped," and since the measurement gives the same answer for both elements of the superposition, the superposition is not destroyed.
Thus, bit 2 can now be corrected by applying $\sigma_x(2)$.

ESCAL Extreme Storage & Computer Architecture Lab

UC RIVERSIDE

How to correct phase flip?

The unitary transformation

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

takes phase flips to bit flips and vice versa:

$$H\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}H^\dagger = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Suppose we apply H to the 3 encoding qubits and the encoded qubit. What does this do to our code?

Applying H, we get a new code

$$\begin{aligned} |0\rangle_L &= \frac{1}{2}(|000\rangle + |011\rangle + |101\rangle + |110\rangle) \\ |1\rangle_L &= \frac{1}{2}(|100\rangle + |010\rangle + |001\rangle + |111\rangle) \end{aligned}$$

$|0\rangle$ encoded by superposition of states with an odd number of 0's;
$|1\rangle$ by superpositions of states with an even number of 0's.

A bit flip on any qubit exchanges 0 and 1, so it takes a logical 0 to a logical 1. Thus bit flips are three times as likely.

$$\begin{aligned} |0\rangle_L &= \frac{1}{2}(|000\rangle + |011\rangle + |101\rangle + |110\rangle) \\ |1\rangle_L &= \frac{1}{2}(|100\rangle + |010\rangle + |001\rangle + |111\rangle) \end{aligned}$$

A phase flip on any qubit is correctable.
E.g., $\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ on bit 3.

$$\sigma_z(3)\,|0\rangle_L = \frac{1}{2}(|000\rangle - |011\rangle - |101\rangle + |110\rangle)$$

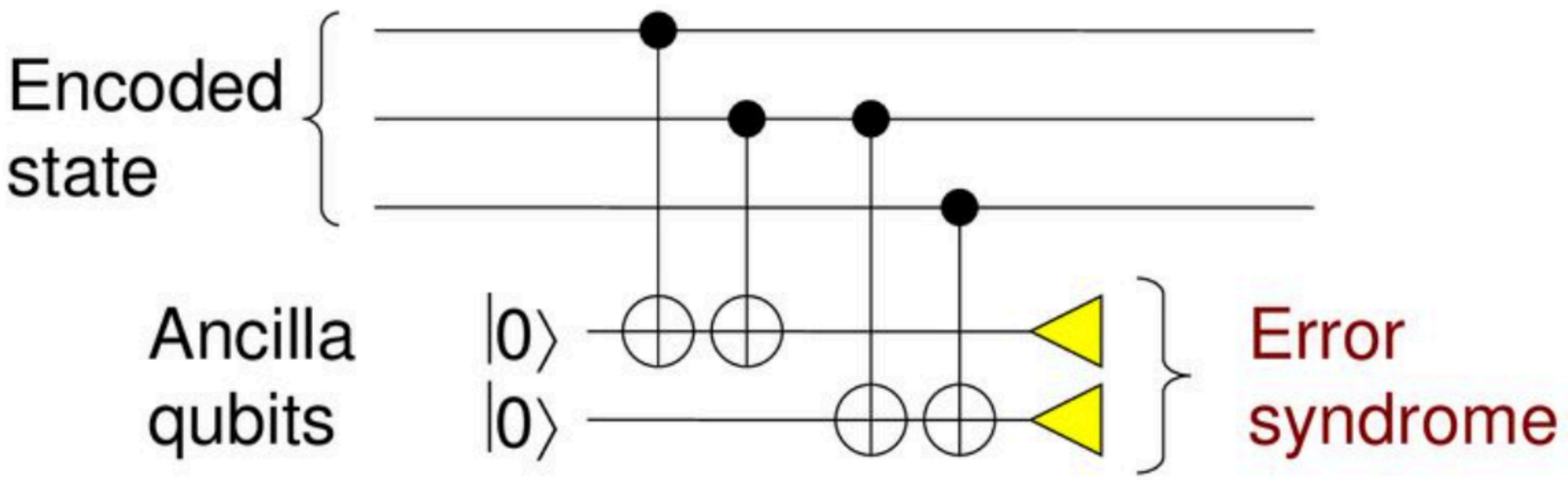
This is orthogonal to $\sigma_z(a)\,|b\rangle_L$ unless $a = 3,\ b = 0$.
So we can measure "which qubit has a phase flip?" and then correct this qubit.

Constraints of classical error correction

- Measurement loses Qubit-state
- Non-cloning theorem
  - It is impossible to create an independent and identical copy of unknown qubit states
- Sign-flip should be covered
  - (classical coding can only detect/correct bit-flip error)

Measure the error, not the data

Use this circuit:



1st bit of error syndrome says whether the first two bits of the state are the same or different.

2nd bit of error syndrome says whether the second two bits of the state are the same or different.

Ancilla qubits	1st	Out1	2nd	Out2
|0>	|0>	|0>	|0>	|0>
		|0>	|1>	|1>
|0>	|1>	|1>	|0>	|1>
		|1>	|1>	|0>

Redundancy, not repetition

This encoding does not violate the no-cloning theorem:

$$\alpha|0\rangle + \beta|1\rangle \rightarrow \alpha|000\rangle + \beta|111\rangle$$
$$\neq (\alpha|0\rangle + \beta|1\rangle)^{\otimes 3}$$

We have repeated the state only in the computational basis; superposition states are spread out (redundant encoding), but not repeated (which would violate no-cloning).

Correcting phase error

Hadamard transform H exchanges bit flip and phase errors:

$$H(\alpha|0\rangle + \beta|1\rangle) = \alpha|+\rangle + \beta|-\rangle$$

$X|+\rangle = |+\rangle$, $X|-\rangle = -|-\rangle$ (acts like phase flip)

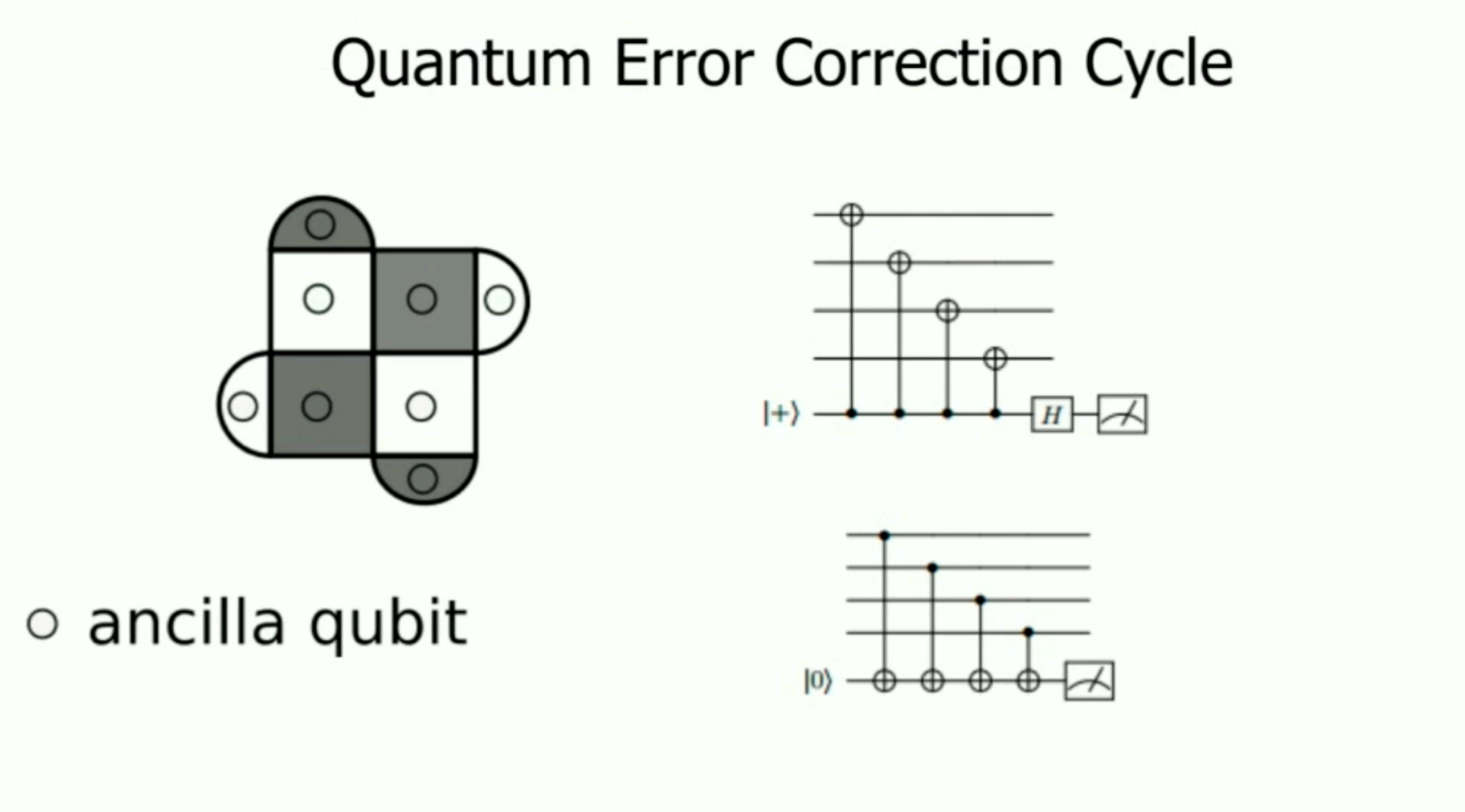
$Z|+\rangle = |-\rangle$, $Z|-\rangle = |+\rangle$ (acts like bit flip)

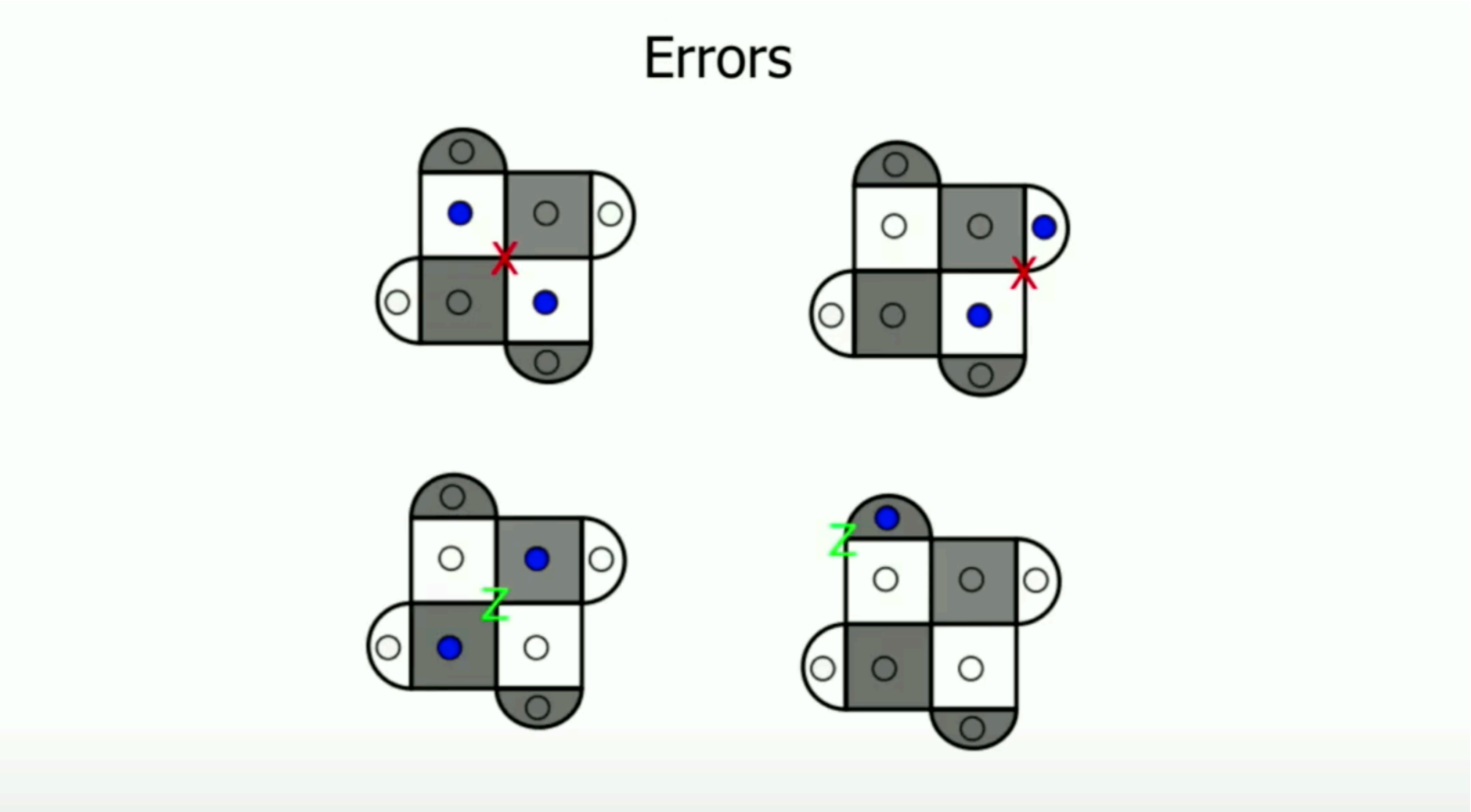
Repetition code corrects a bit flip error

⇒ Repetition code in Hadamard basis corrects a phase error.

$$\alpha|+\rangle + \beta|-\rangle \rightarrow \alpha|{+}{+}{+}\rangle + \beta|{-}{-}{-}\rangle$$

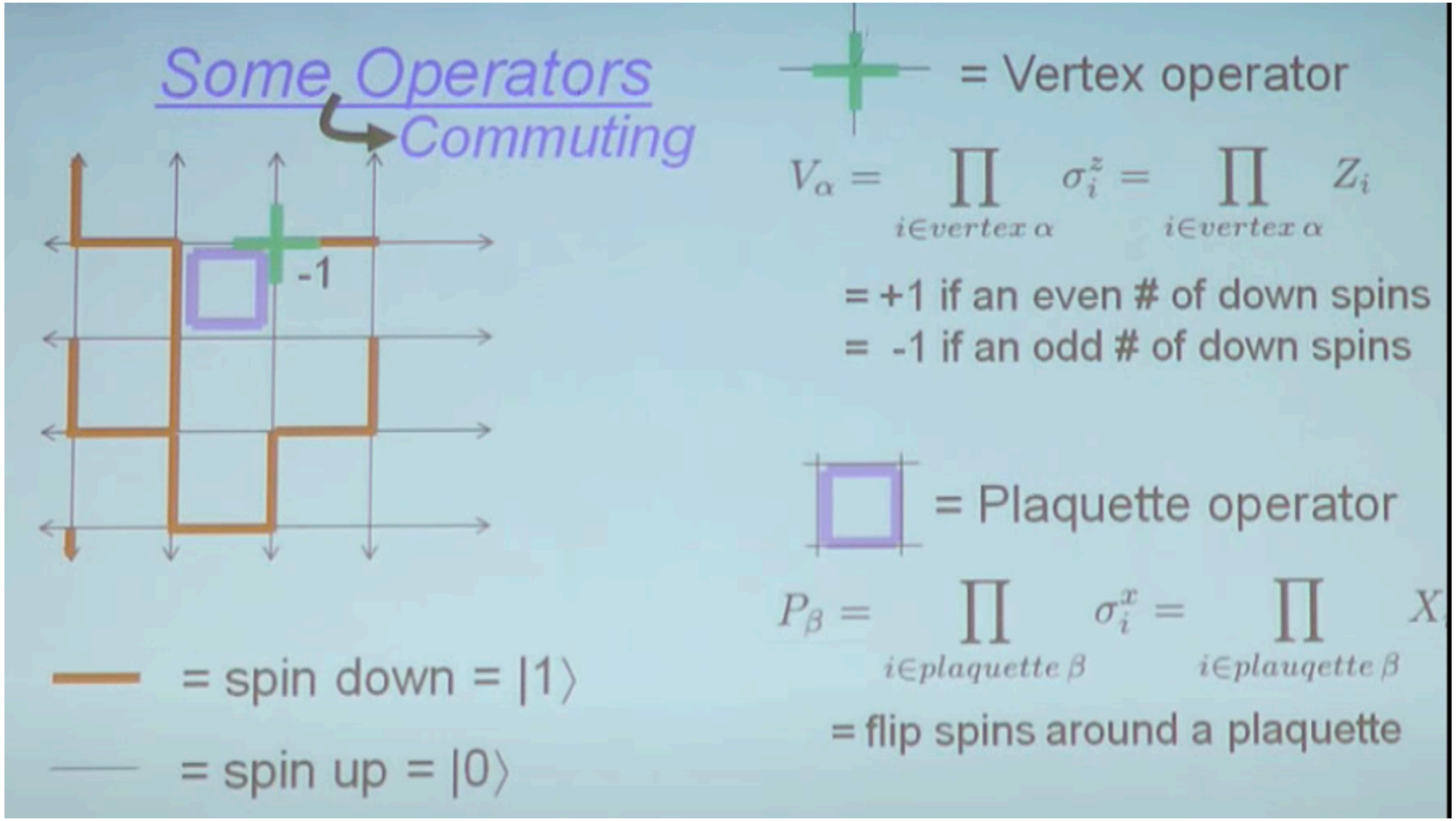
Stabilizer example in 9-Qubit



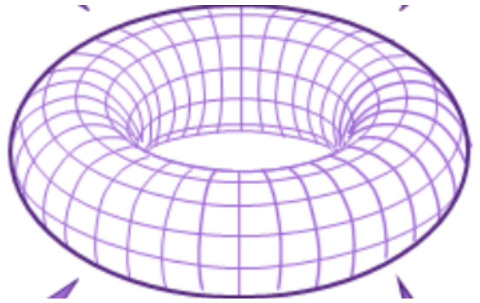


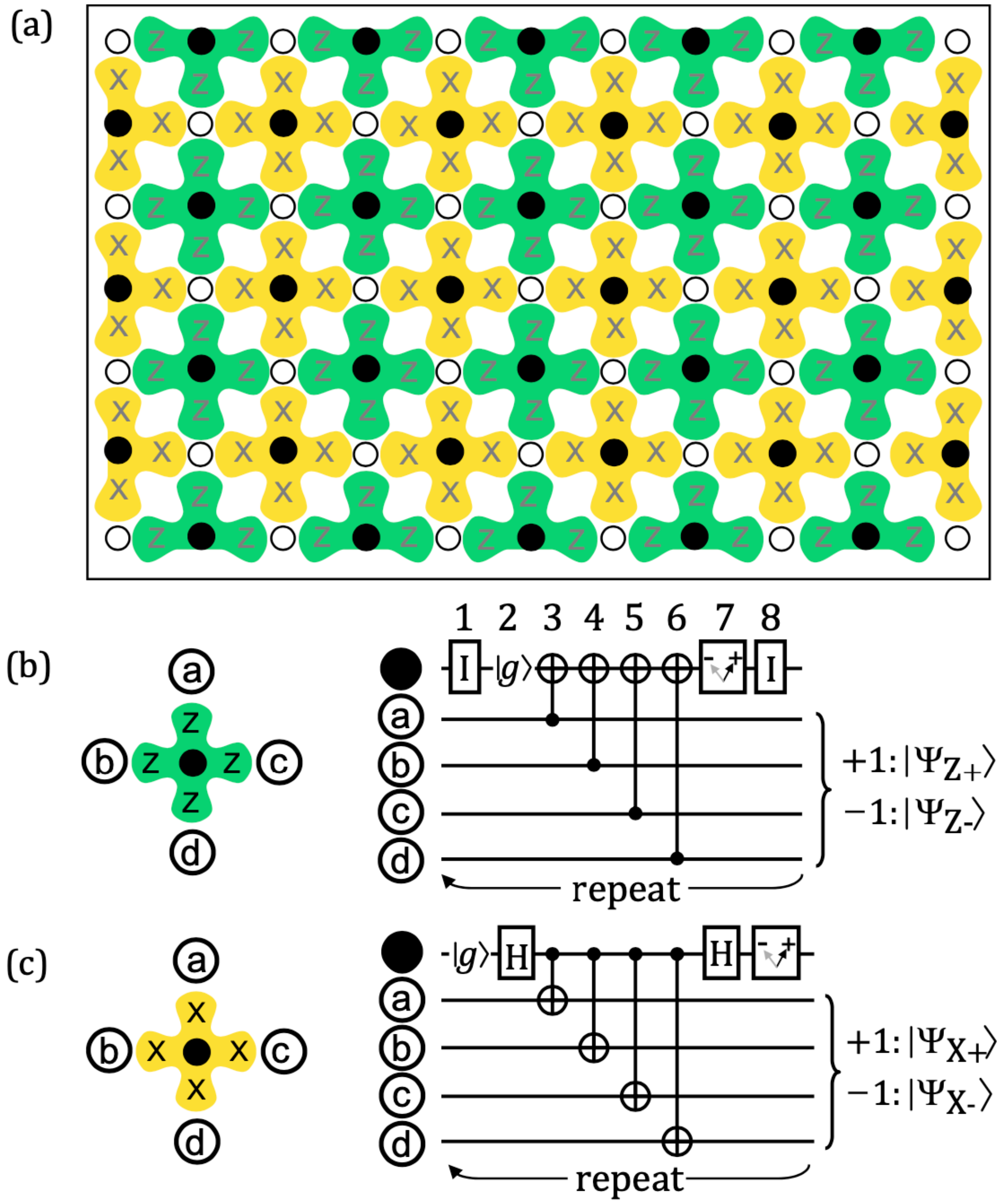
Extreme Storage & Computer Architecture Lab

Operation in toric code

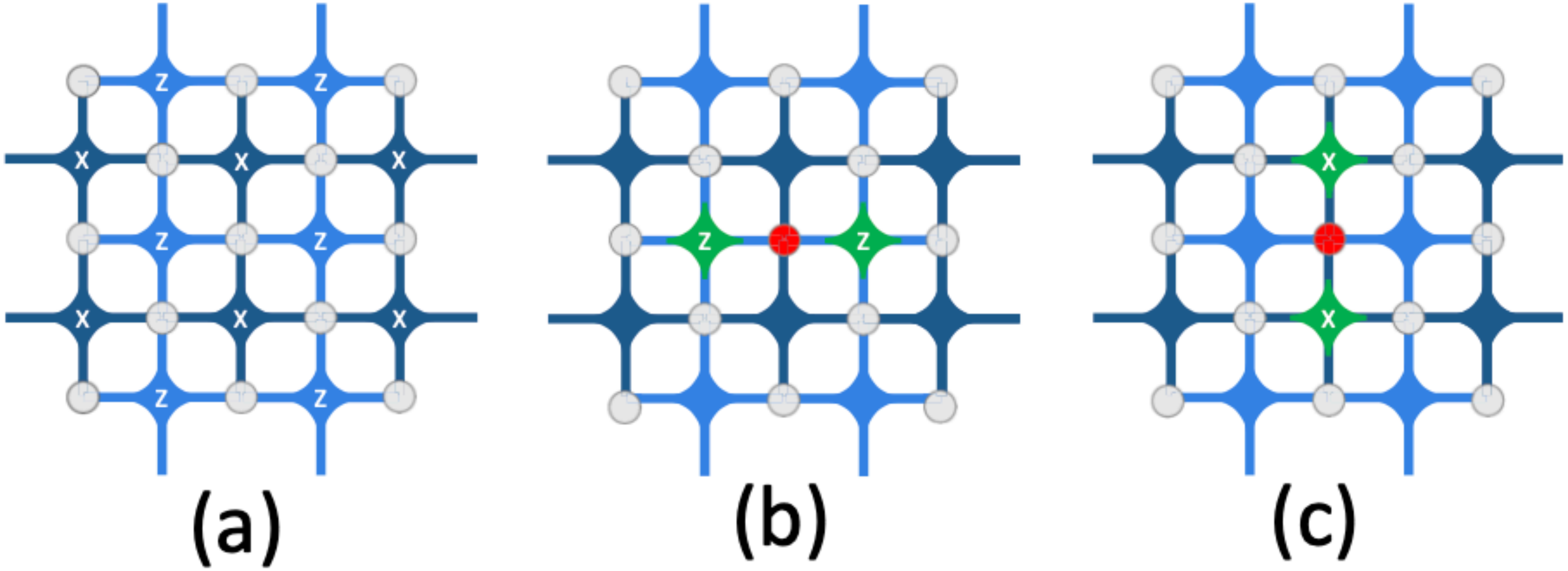


Surface code

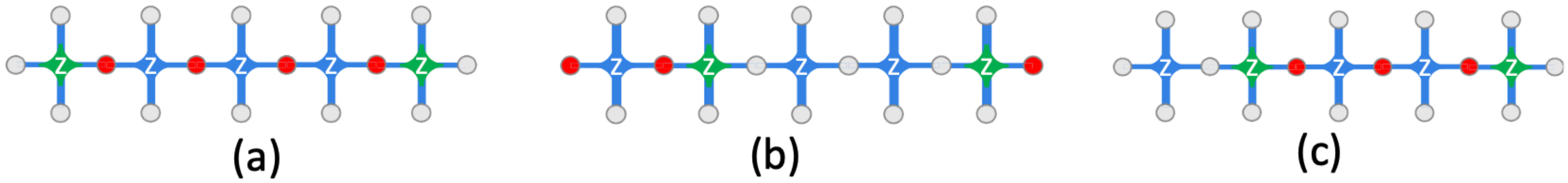




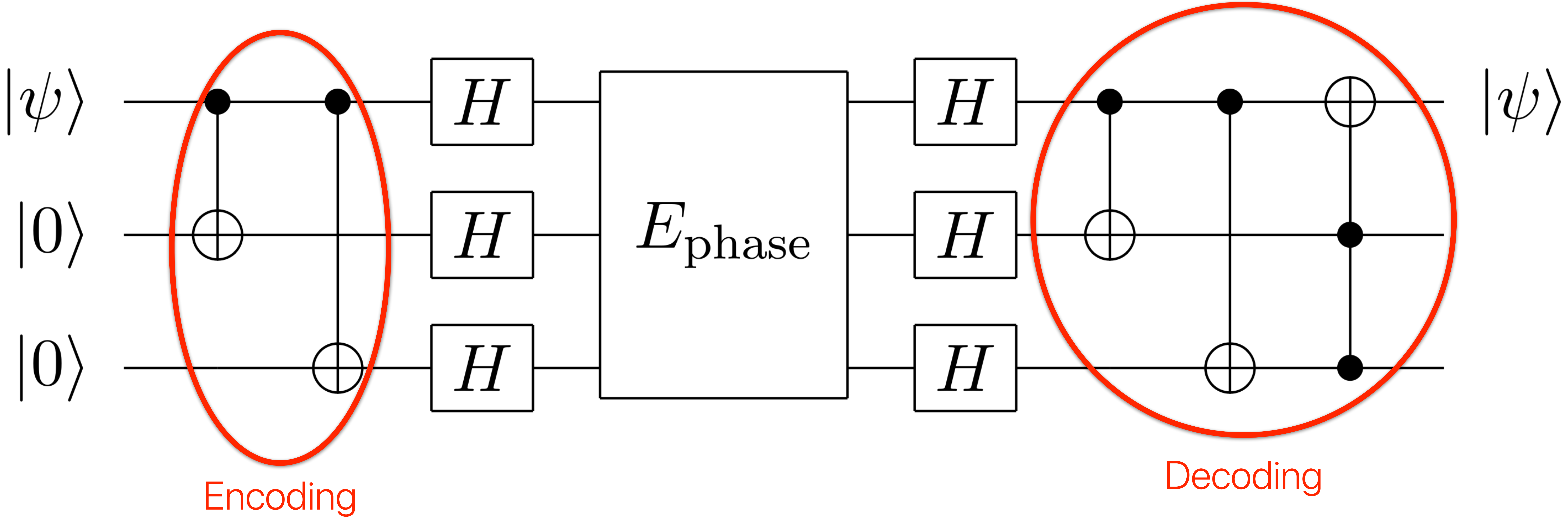
Error detection



Qubit error pattern

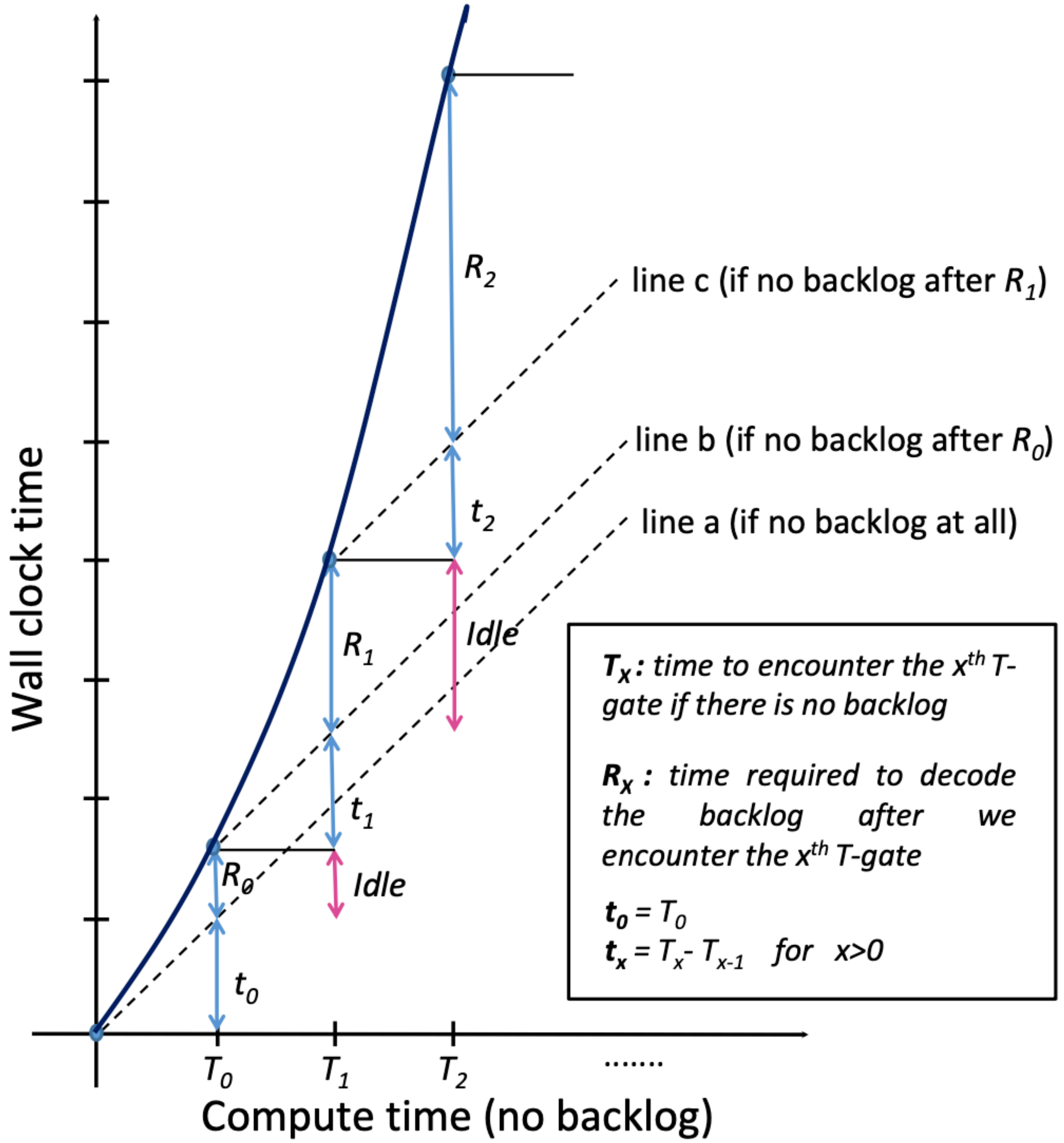


Quantum error correction

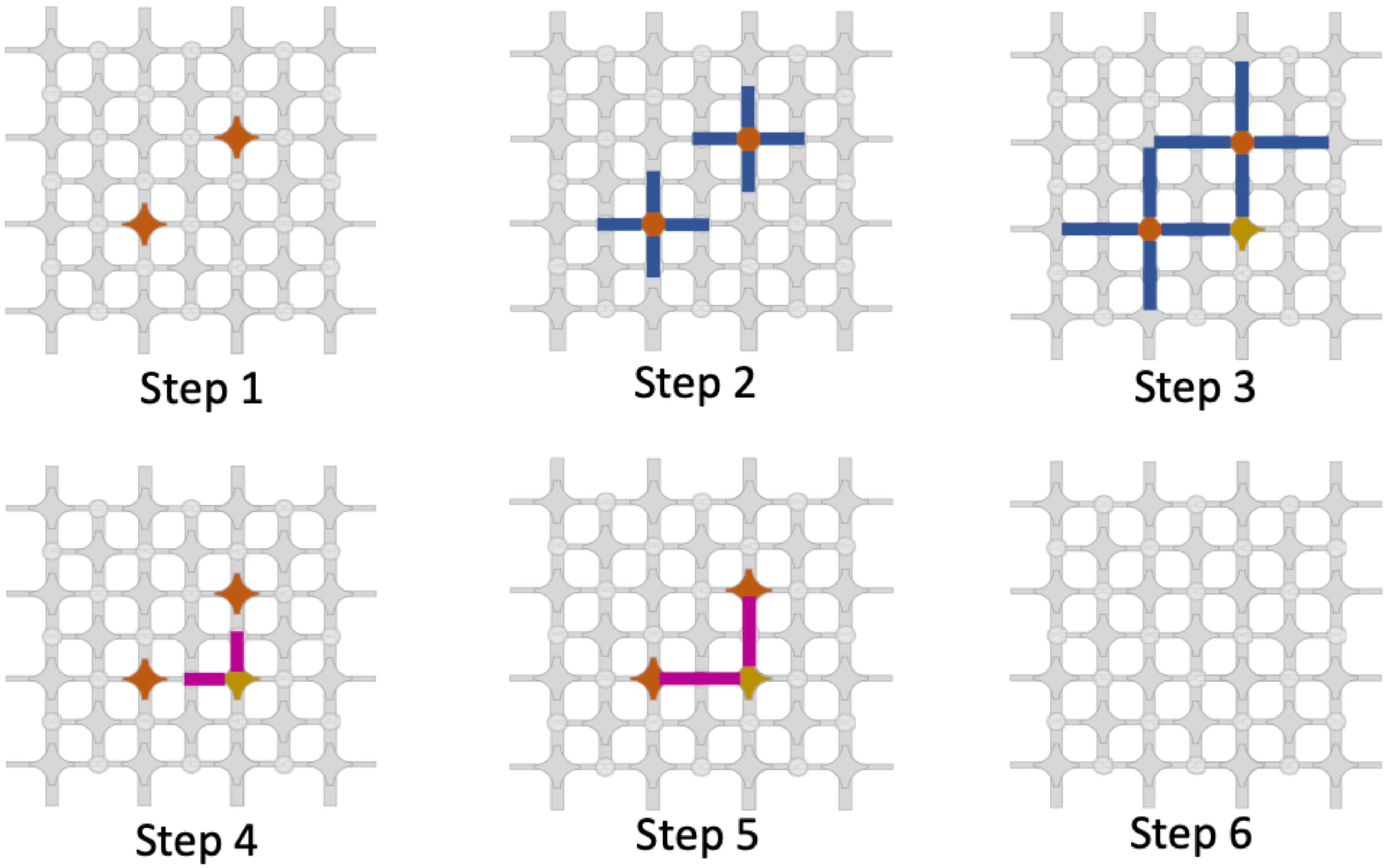


Decoding should be fast

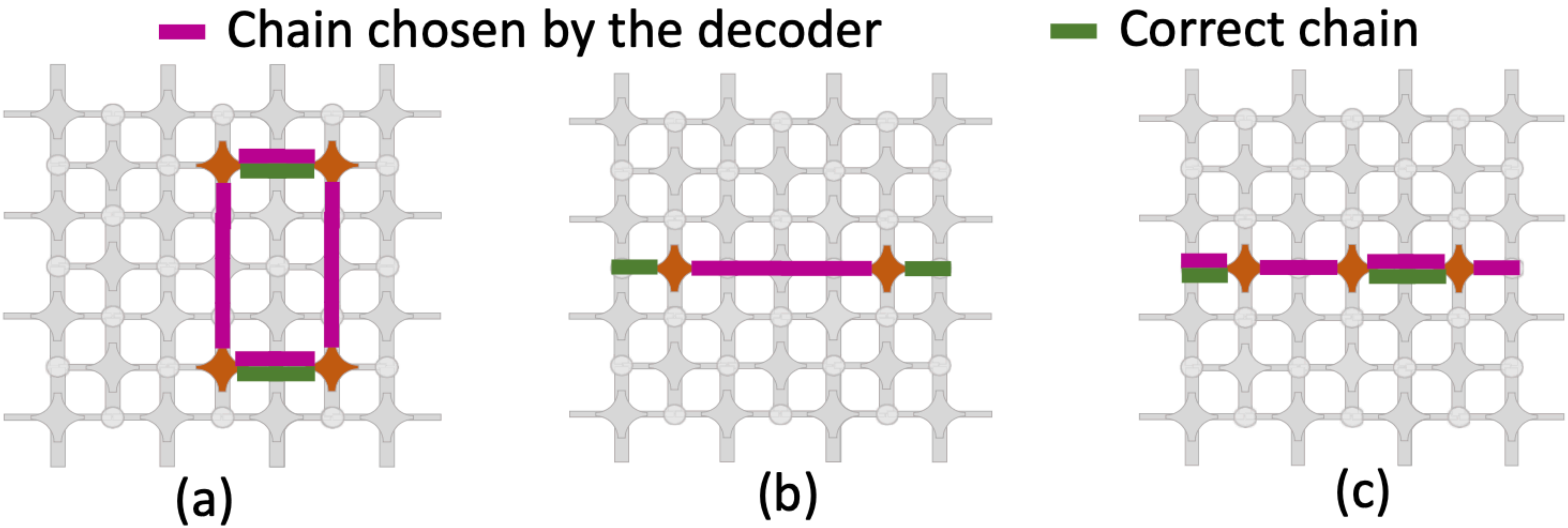
especially considering the cryogenic environment of typical quantum computing systems. If decoding occurs slower than error information is generated, the system will generate a backlog of information as it waits for decoding to complete, introducing an exponential time overhead that will kill any quantum advantage (see Section III). A hardware solution proposed here results in the ability to perform logical gates



Baseline error correction



Constraints of baseline method



Reset

Boundary

Equal distance

Thank you

- Any questions?