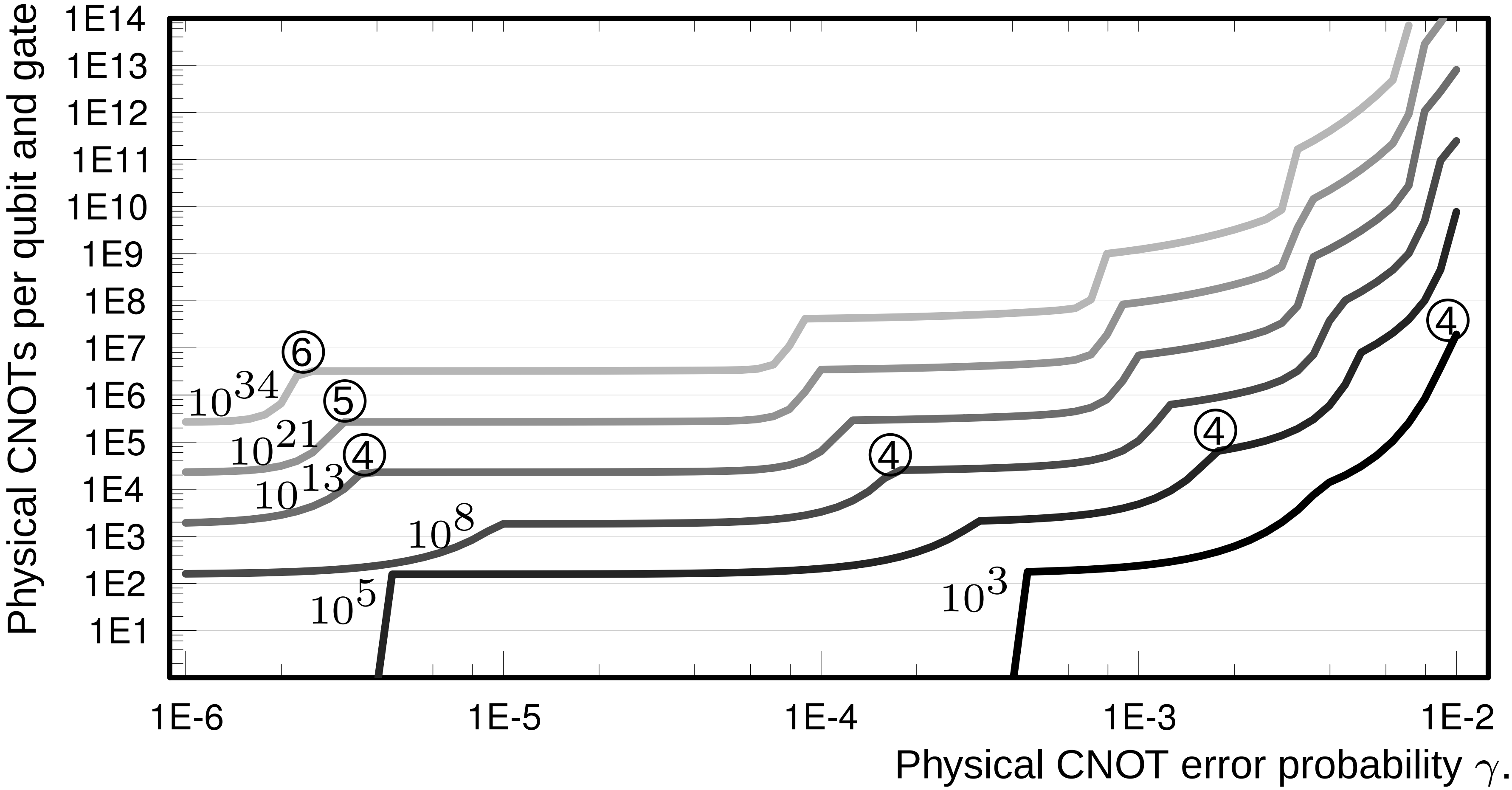
Quantum Computing with Very Noisy Gates

Produced with pdflatex and xfig

- The C_4/C_6 architecture.
- Performance data from simulation.
- Resource requirements.

E. "Manny" Knill: knill@boulder.nist.gov

Typical Resource Requirements



- Resource requirements for the C_4/C_6-architecture and different computation sizes (by simulation and modelling).

The C_4/C_6 Architecture: Features

- Use the simplest error-detecting codes and concatenation.

The C_4/C_6 Architecture: Features

- Use the simplest error-detecting codes and concatenation.
- Exploit error-correcting teleportation.

The C_4/C_6 Architecture: Features

- Use the simplest error-detecting codes and concatenation.
- Exploit error-correcting teleportation.
- Postselected quantum computing for state preparation.

The C_4/C_6 Architecture: Features

- Use the simplest error-detecting codes and concatenation.
- Exploit error-correcting teleportation.
- Postselected quantum computing for state preparation.
- Partial decoding for state preparation.

The C_4/C_6 Architecture: Features

- Use the simplest error-detecting codes and concatenation.
- Exploit error-correcting teleportation.
- Postselected quantum computing for state preparation.
- Partial decoding for state preparation.
- Fault-tolerant implementation of Clifford gates $+\ \delta$ suffices.

The C_4/C_6 Architecture: Features

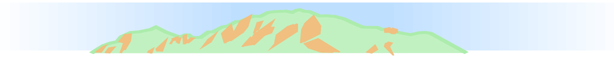
- Use the simplest error-detecting codes and concatenation.
- Exploit error-correcting teleportation.
- Postselected quantum computing for state preparation.
- Partial decoding for state preparation.
- Fault-tolerant implementation of Clifford gates $+ \delta$ suffices.
- Evidence that depolarizing errors $> 3\%$ per gate are ok.

The [[4,2,2]]-Code C_4

- A $[[4,2,2]]$ code: Check ops: $[XXXX], [ZZZZ]$. See: Pauli products.

  Logical ops: $X_L = [XXII], Z_L = [ZIZI]$

  $X_S = [IXIX], Z_S = [IIZZ]$.



The [[4,2,2]]-Code C_4

- A $[[4,2,2]]$ code: Check ops: $[XXXX], [ZZZZ]$. See: Pauli products.
  Logical ops: $X_L = [XXII], Z_L = [ZIZI]$
  $X_S = [IXIX], Z_S = [IIZZ]$.
- State space representation: Logical states

$$\underbrace{|a\rangle_{\mathsf{L}}}_{\text{logical qubit L}} \quad \overbrace{|b\rangle_{\mathsf{S}}}^{\text{logical qubit S}} \quad \underbrace{|s_1 s_2\rangle_{\mathsf{Y}}}_{\text{syndrome bits}}$$

a, b, s_1, s_2 are eigenvalues labels.
E.g. $[XXXX]|ab0s_2\rangle_{\mathsf{LSY}} = 1$,
$[XXXX]|ab1s_2\rangle_{\mathsf{LSY}} = -1$.

The [[4,2,2]]-Code C_4

- A $[[4,2,2]]$ code: Check ops: $[XXXX], [ZZZZ]$. See: Pauli products.

  Logical ops: $X_L = [XXII], Z_L = [ZIZI]$

  $X_S = [IXIX], Z_S = [IIZZ]$.

- State space representation: Logical states

$$\underbrace{|a\rangle_{\mathsf{L}}}_{\text{logical qubit L}} \quad \overbrace{|b\rangle_{\mathsf{S}}}^{\text{logical qubit S}} \quad \underbrace{|s_1 s_2\rangle_{\mathsf{Y}}}_{\text{syndrome bits}}$$

a, b, s_1, s_2 are eigenvalues labels.

E.g. $[XXXX]|ab0s_2\rangle_{\mathsf{LSY}} = 1$,

$[XXXX]|ab1s_2\rangle_{\mathsf{LSY}} = -1$.

- Any single Pauli error changes the syndrome bits: The code is *one-error detecting.*

The [[4,2,2]]-Code C_4

- A $[[4,2,2]]$ code: Check ops: $[XXXX], [ZZZZ]$. See: Pauli products.

  Logical ops: $X_L = [XXII], Z_L = [ZIZI]$

  $X_S = [IXIX], Z_S = [IIZZ]$.

- State space representation: Logical states

$$\underbrace{|a\rangle_{\text{L}}}_{\text{logical qubit L}} \quad \overbrace{|b\rangle_{\text{S}}}^{\text{logical qubit S}} \quad \underbrace{|s_1 s_2\rangle_{\text{Y}}}_{\text{syndrome bits}}$$

a, b, s_1, s_2 are eigenvalues labels.

E.g. $[XXXX]|ab0s_2\rangle_{\text{LSY}} = 1$,

$[XXXX]|ab1s_2\rangle_{\text{LSY}} = -1$.

- Any single Pauli error changes the syndrome bits: The code is *one-error detecting.*

- Some encoded states:

$$\begin{aligned}
|0000\rangle_{\text{LSY}} &= \frac{1}{\sqrt{2}}\left(|0000\rangle_{1234} + |1111\rangle_{1234}\right) \\
|{+}{+}00\rangle_{\text{LSY}} &= \frac{1}{\sqrt{2}}\left(|{+}{+}{+}{+}\rangle_{1234} + |{-}{-}{-}{-}\rangle_{1234}\right) \\
|{+}000\rangle_{\text{LSY}} &= \frac{1}{2}\left(|00\rangle_{12} + |11\rangle_{12}\right)\left(|00\rangle_{34} + |11\rangle_{34}\right)
\end{aligned}$$

The [[3,1,2]]$_4$-Code C_6

- C_4 encodes *qubit pairs*.
- C_6 encodes one qubit pair in three.
  Check ops: $[XIIXXX], [XXXIIX], [ZIIZZZ], [ZZZIIZ]$.
  Logical ops: $X_L = [IXXIII], Z_L = [IIZZIZ]$,
  $X_S = [XIXXII], Z_S = [IIIZZI]$.

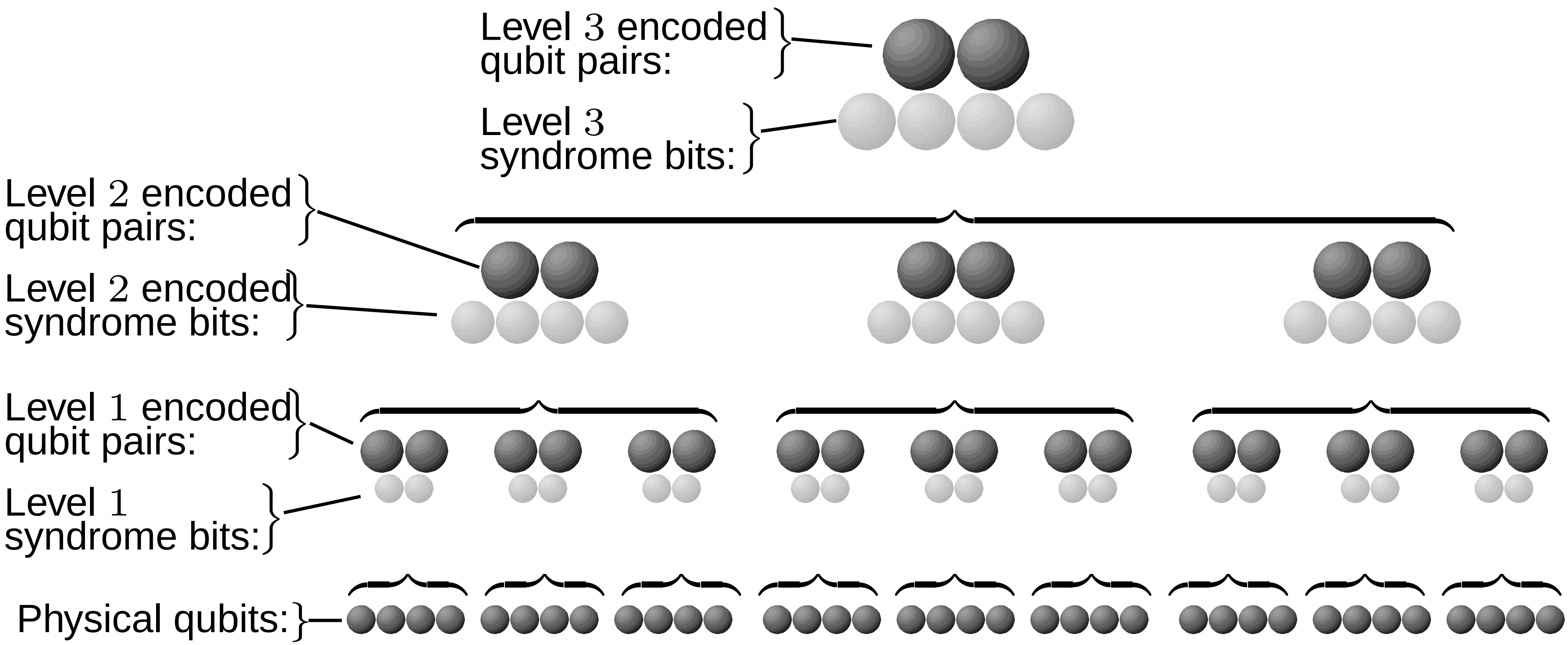
The [[3,1,2]]$_4$-Code C_6

- C_4 encodes *qubit pairs*.
- C_6 encodes one qubit pair in three.
  Check ops: $[XIIXXX], [XXXIIX], [ZIIZZZ], [ZZZIIZ]$.
  Logical ops: $X_L = [IXXIII], Z_L = [IIZZIZ]$,
  $X_S = [XIXXII], Z_S = [IIIZZI]$.
- C_6 is one-qubit-pair-error detecting.

The [[3,1,2]]$_4$-Code C_6

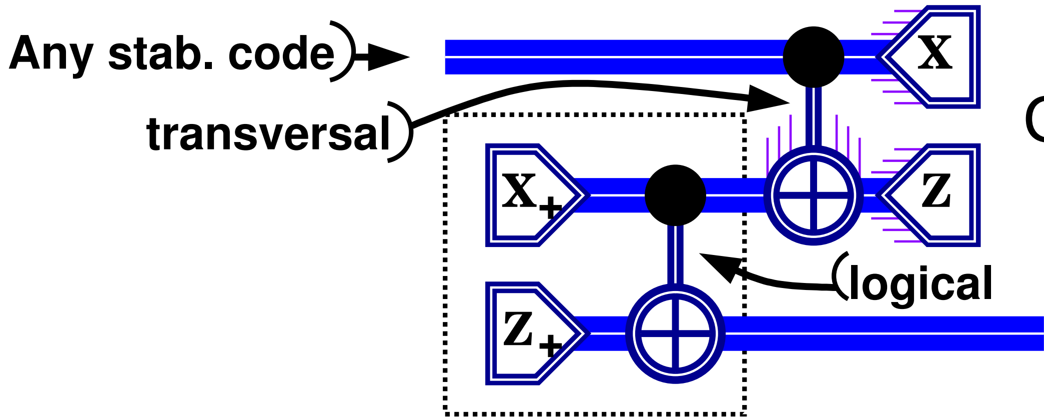
- C_4 encodes *qubit pairs*.
- C_6 encodes one qubit pair in three.
  Check ops: $[XIIXXX], [XXXIIX], [ZIIZZZ], [ZZZIIZ]$.
  Logical ops: $X_L = [IXXIII], Z_L = [IIZZIZ]$,
  $X_S = [XIXXII], Z_S = [IIIZZI]$.
- C_6 is one-qubit-pair-error detecting.
- Encoded states include:
  Local modifications of generalized GHZ states.

C_4/C_6 concatenation hierarchy.



Error-correcting Teleportation

- Syndrome measurement from qubit-wise teleportation.

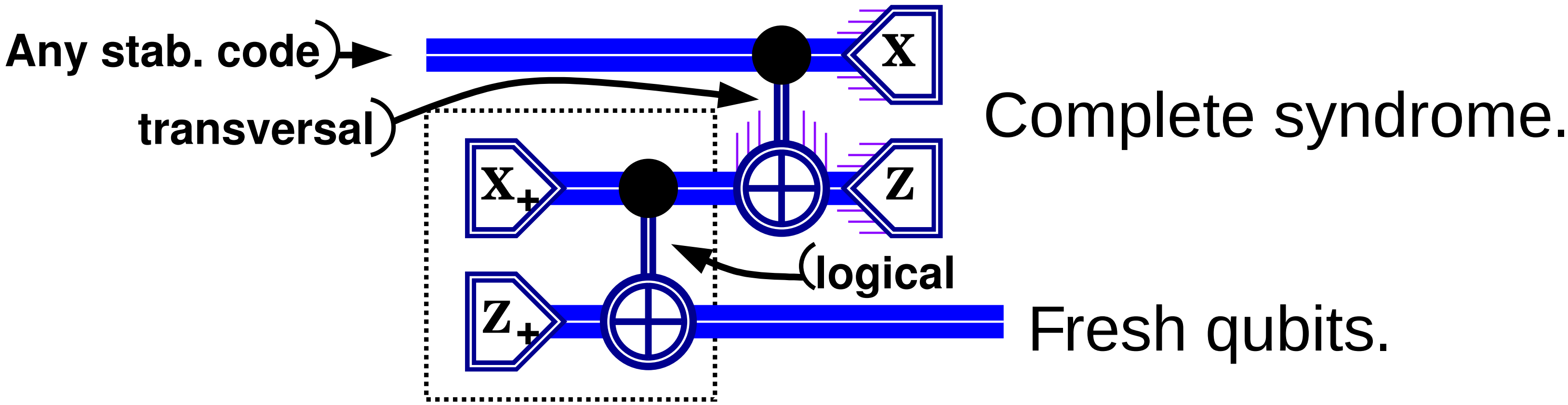


Complete syndrome.

Fresh qubits.

Error-correcting Teleportation

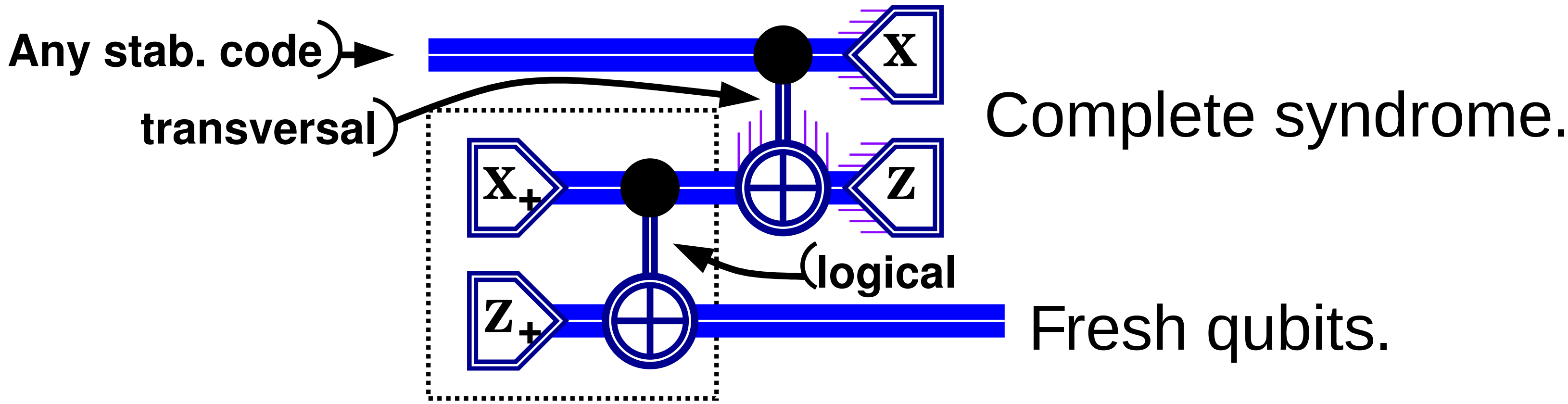
- Syndrome measurement from qubit-wise teleportation.



- Syndrome $\longrightarrow$ error detection, correction, tracking.

Error-correcting Teleportation

- Syndrome measurement from qubit-wise teleportation.



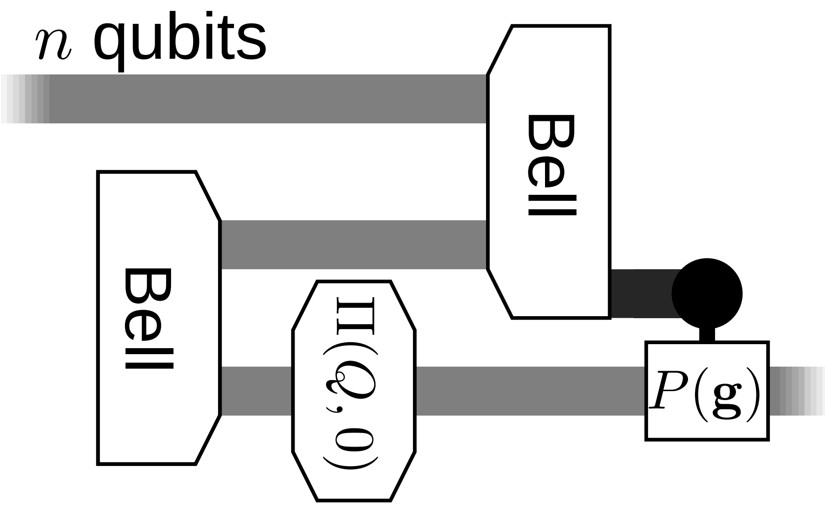
- Syndrome → error detection, correction, tracking.
- Use *Pauli frame* to avoid explicit correction gates.

Error-Correcting Teleportation: Function

- Syndrome is obtained from parities in the Bell measurements.

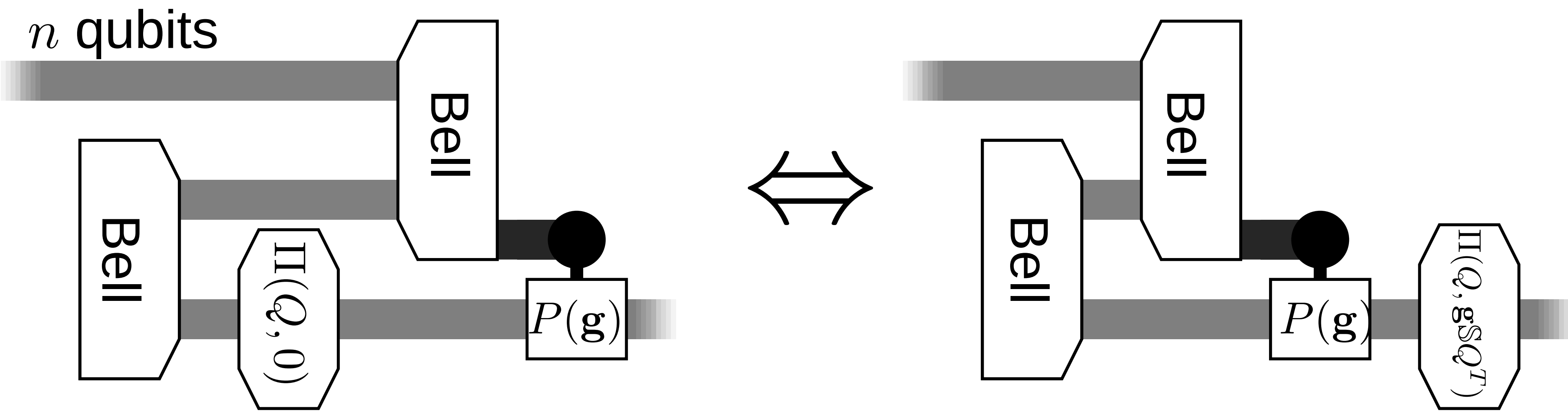
Error-Correcting Teleportation: Function

- Syndrome is obtained from parities in the Bell measurements.



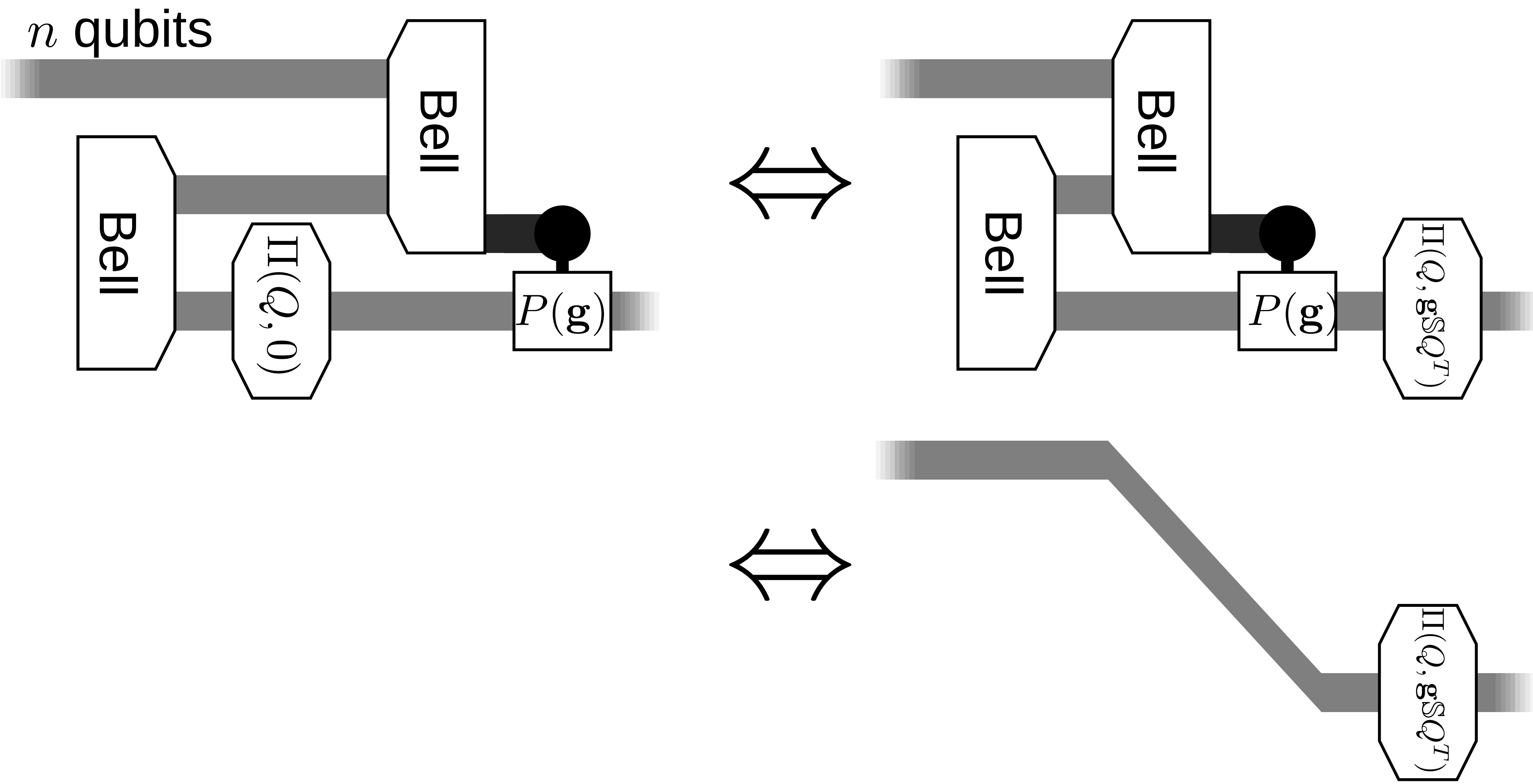
Error-Correcting Teleportation: Function

- Syndrome is obtained from parities in the Bell measurements.

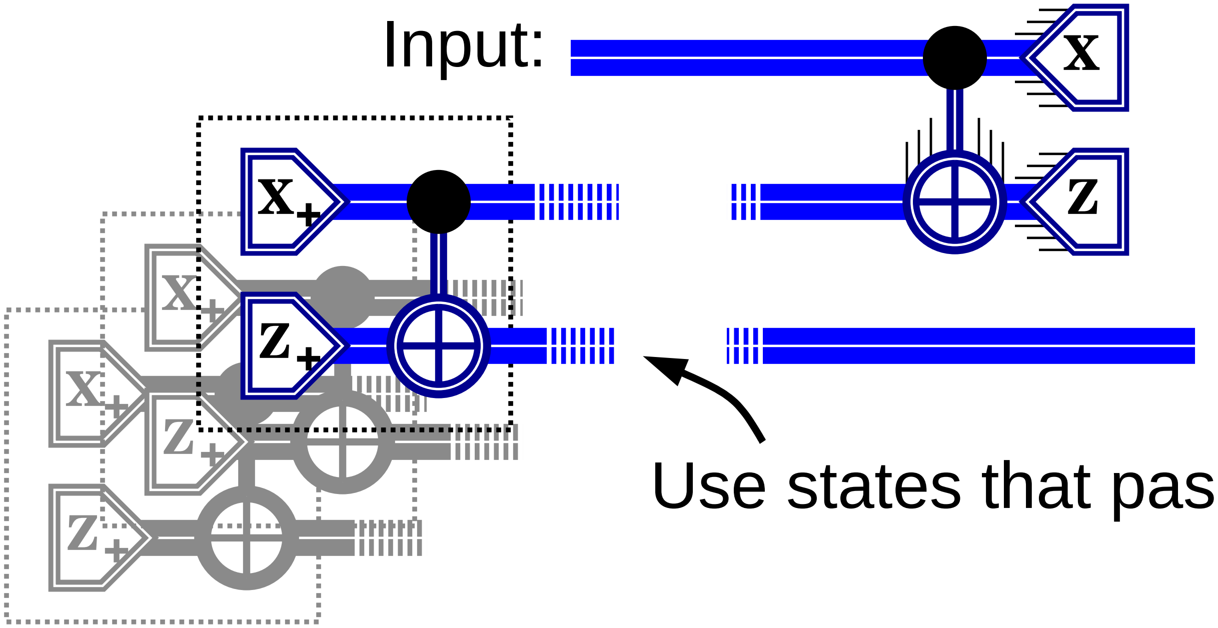


Error-Correcting Teleportation: Function

- Syndrome is obtained from parities in the Bell measurements.

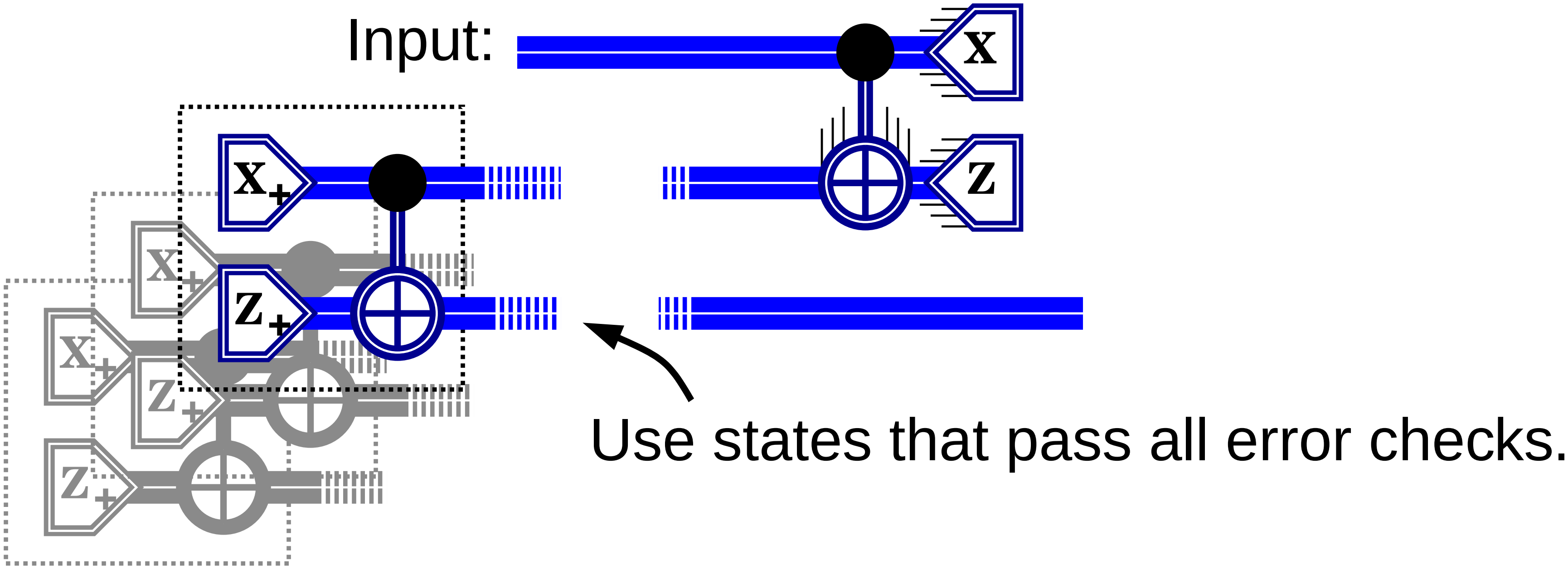


Post-selected State Preparation



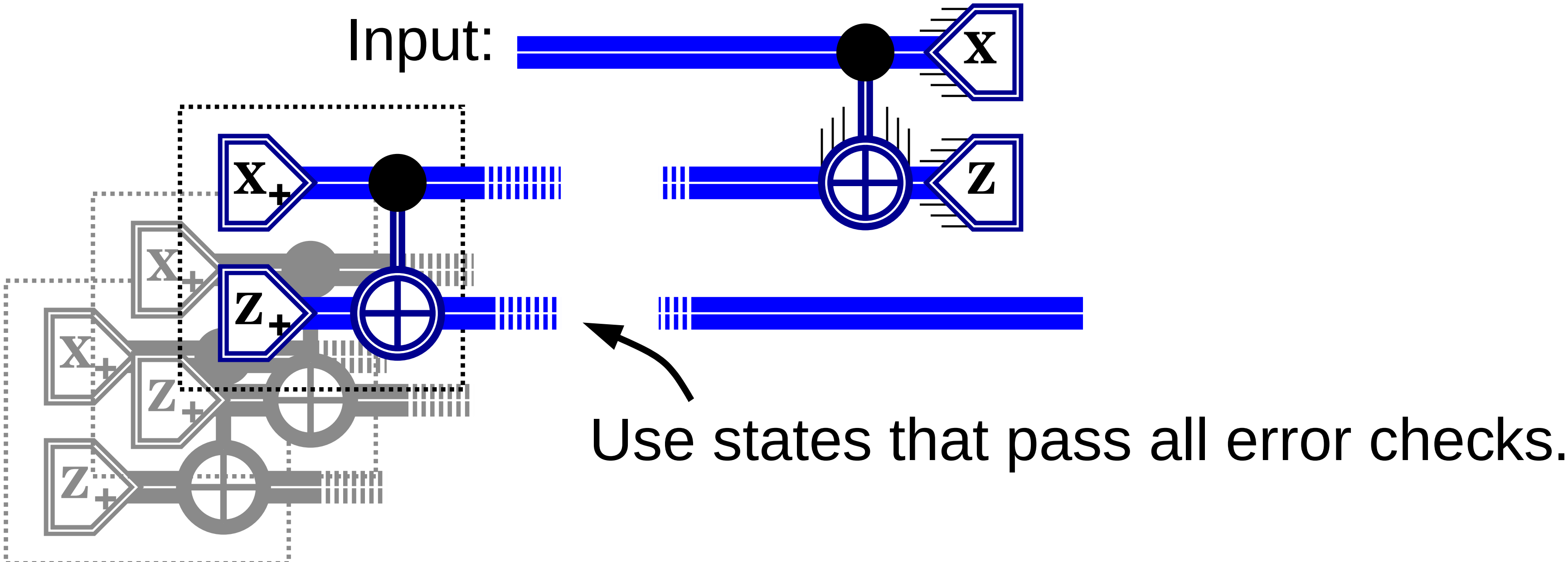
Use states that pass all error checks.

Post-selected State Preparation



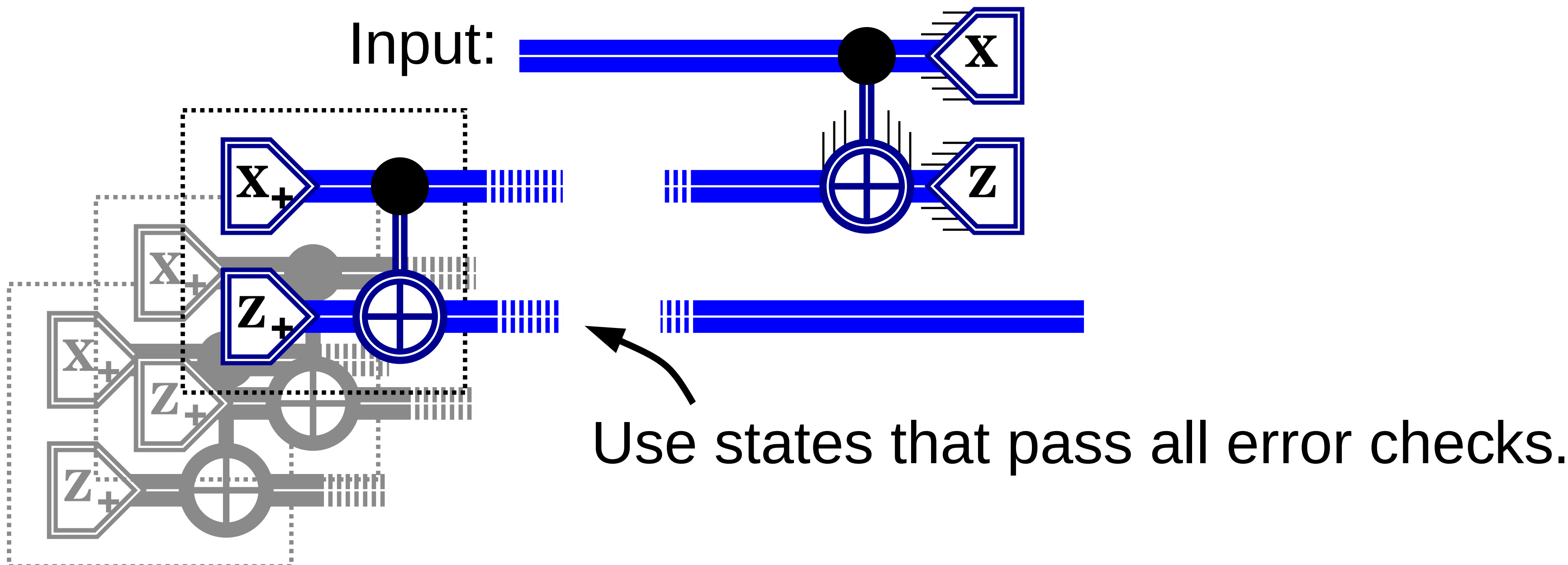
- Error in the prepared state $\equiv$ error in the input state.

Post-selected State Preparation



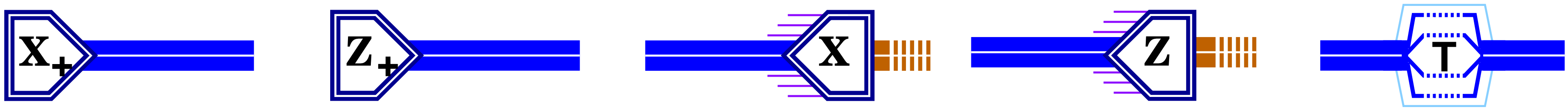
- Error in the prepared state $\equiv$ error in the input state.
- States only need to be "good" conditional on error checks.
  - Residual errors $+$ input $+$ Bell measurement errors must be correctable.

Post-selected State Preparation

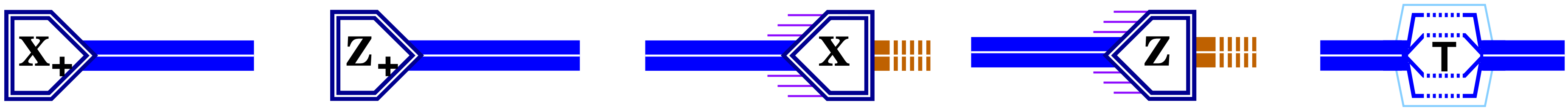


- Error in the prepared state $\equiv$ error in the input state.
- States only need to be "good" conditional on error checks.
  - Residual errors $+$ input $+$ Bell measurement errors must be correctable.
- Can use parallel state preparation factories.

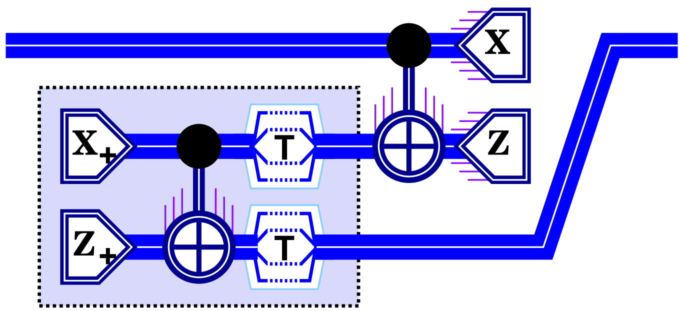
C_4 Bell-State Preparation



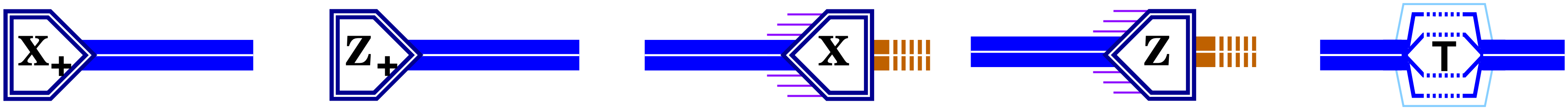
C_4 Bell-State Preparation



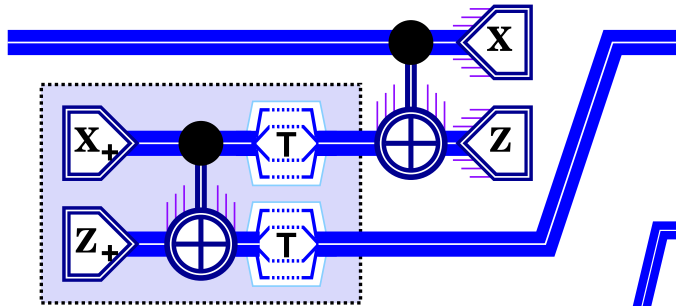
Logical qubits teleportation:



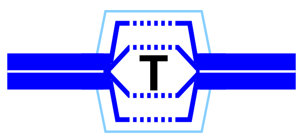
C_4 Bell-State Preparation



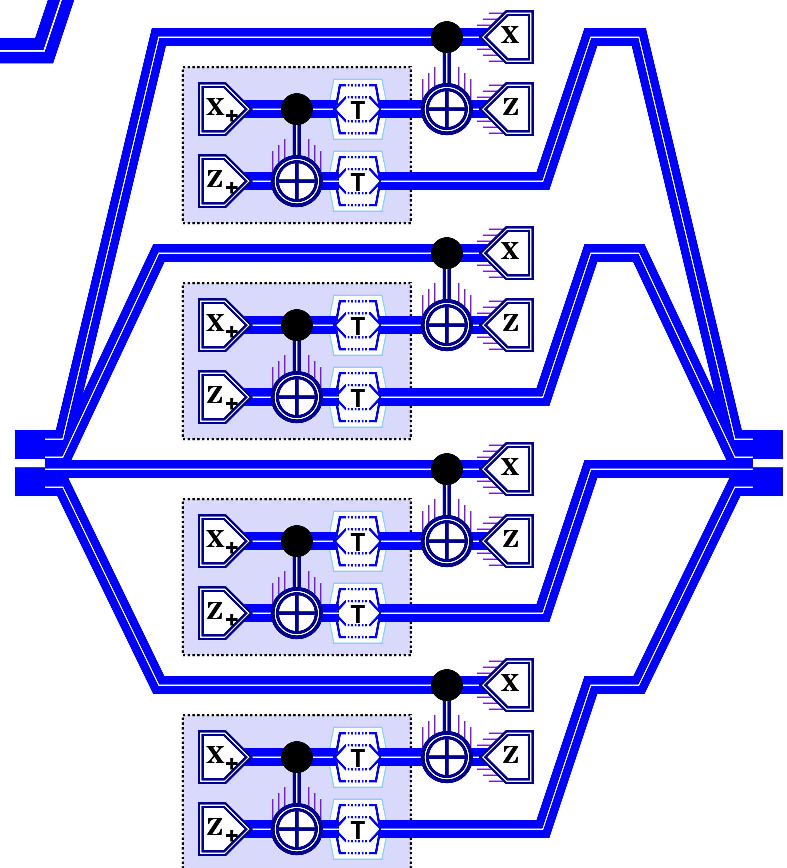
Logical qubits teleportation:



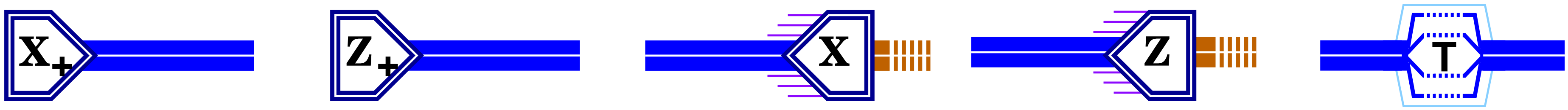
Implementations:



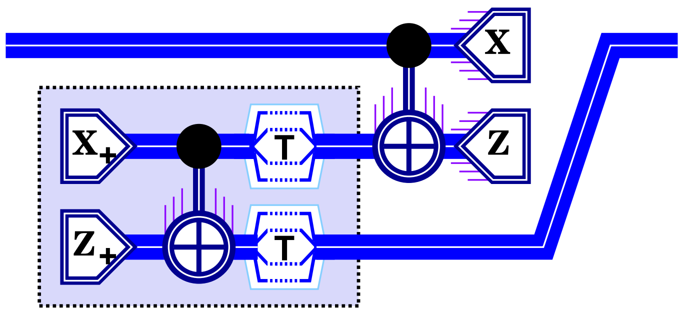
$\Rightarrow$



C_4 Bell-State Preparation

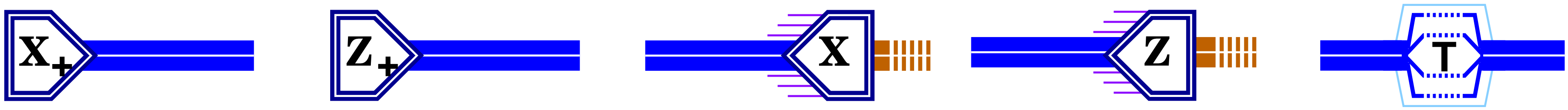


Logical qubits teleportation:

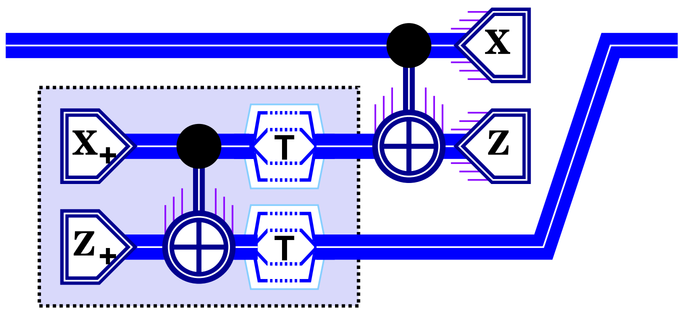


Implementations:

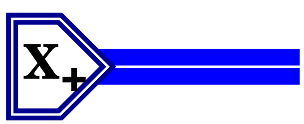
C_4 Bell-State Preparation



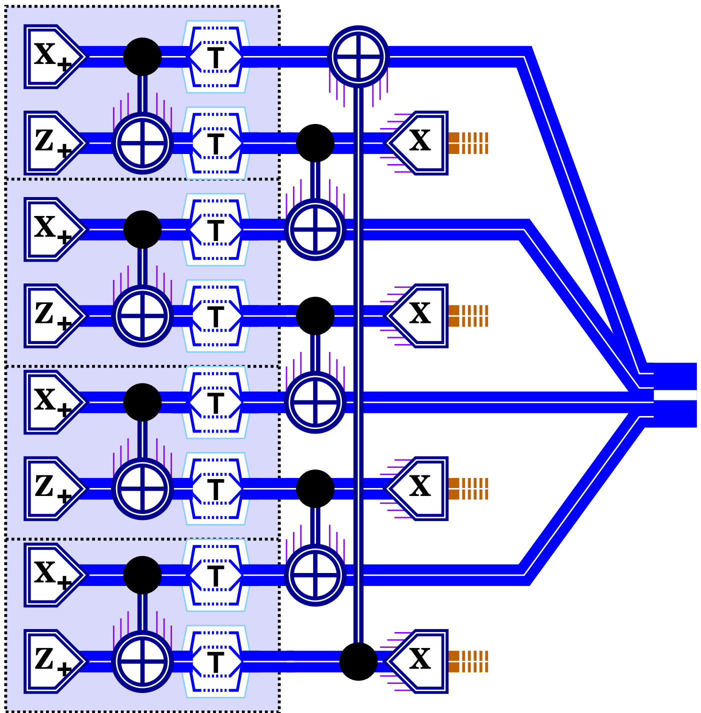
Logical qubits teleportation:



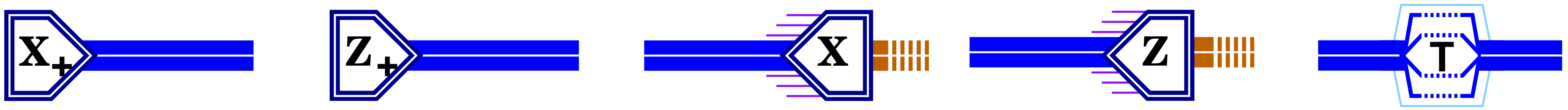
Implementations:



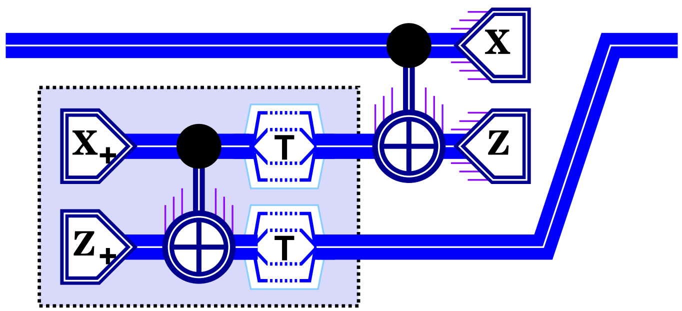
$\Rightarrow$



C_4 Bell-State Preparation

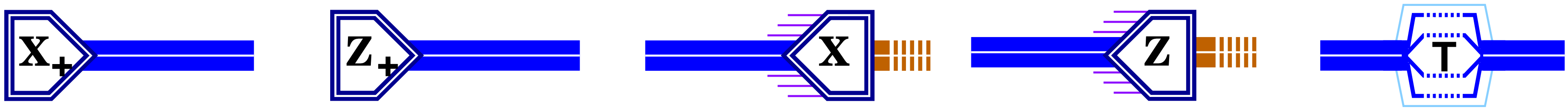


Logical qubits teleportation:

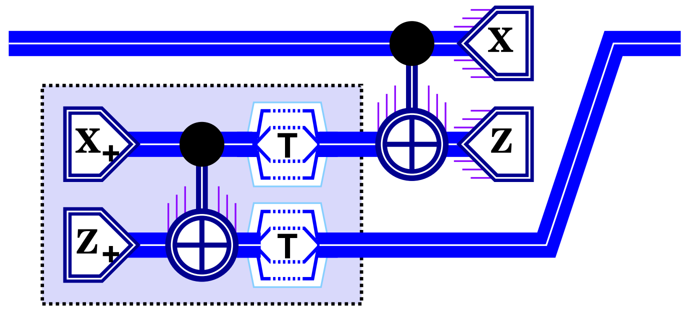


Implementations:

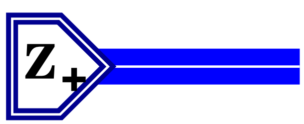
C_4 Bell-State Preparation



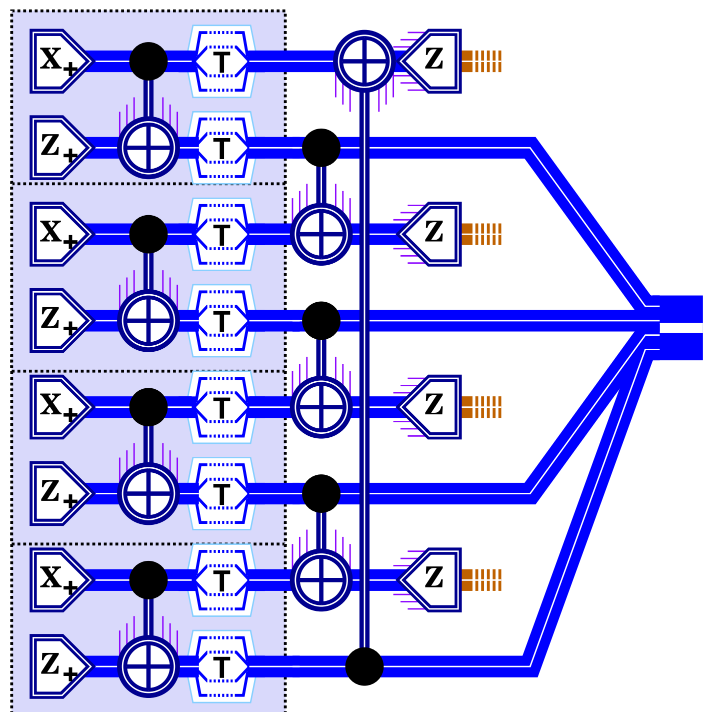
Logical qubits teleportation:



Implementations:



$\Rightarrow$



Postselected Quantum Computing

- Postselected quantum computers.
  - Can execute any of the basic operations, but
  - an operation may fail, possibly destructively.
  - If an operation fails, this is announced.

Postselected Quantum Computing

- Postselected quantum computers.
  - Can execute any of the basic operations, but
  - an operation may fail, possibly destructively.
  - If an operation fails, this is announced.

  . . . exponentially small success probability (not 0) is possible.

Postselected Quantum Computing

- Postselected quantum computers.
  - Can execute any of the basic operations, but
  - an operation may fail, possibly destructively.
  - If an operation fails, this is announced.

  ...exponentially small success probability (not 0) is possible.

- A postselected QC is fault-tolerant if

  success $\longrightarrow$ negligible probability of error.

Postselected Quantum Computing

- Postselected quantum computers.
  - Can execute any of the basic operations, but
  - an operation may fail, possibly destructively.
  - If an operation fails, this is announced.

  ...exponentially small success probability (not 0) is possible.

- A postselected QC is fault-tolerant if

  success $\longrightarrow$ negligible probability of error.

- A postselected f.-t. QC only needs to detect errors.

Postselected Quantum Computing

- Postselected quantum computers.
  - Can execute any of the basic operations, but
  - an operation may fail, possibly destructively.
  - If an operation fails, this is announced.

  ...exponentially small success probability (not 0) is possible.

- A postselected QC is fault-tolerant if

  success $\longrightarrow$ negligible probability of error.

- A postselected f.-t. QC only needs to detect errors.

- Does postselected f.-t. QC imply f.-t. QC?

Postselected Quantum Computing

- Postselected quantum computers.
  - Can execute any of the basic operations, but
  - an operation may fail, possibly destructively.
  - If an operation fails, this is announced.

  ...exponentially small success probability (not 0) is possible.

- A postselected QC is fault-tolerant if

  success $\longrightarrow$ negligible probability of error.

- A postselected f.-t. QC only needs to detect errors.

- Does postselected f.-t. QC imply f.-t. QC?

  ...Nearly: Use postselected f.-t. to prepare key states.

Toward Unconditional Quantum Computing

- Problem: The states needed for f.-t. QC must be disturbed by well-bounded local errors only.

Toward Unconditional Quantum Computing

- Problem: The states needed for f.-t. QC must be disturbed by well-bounded local errors only.

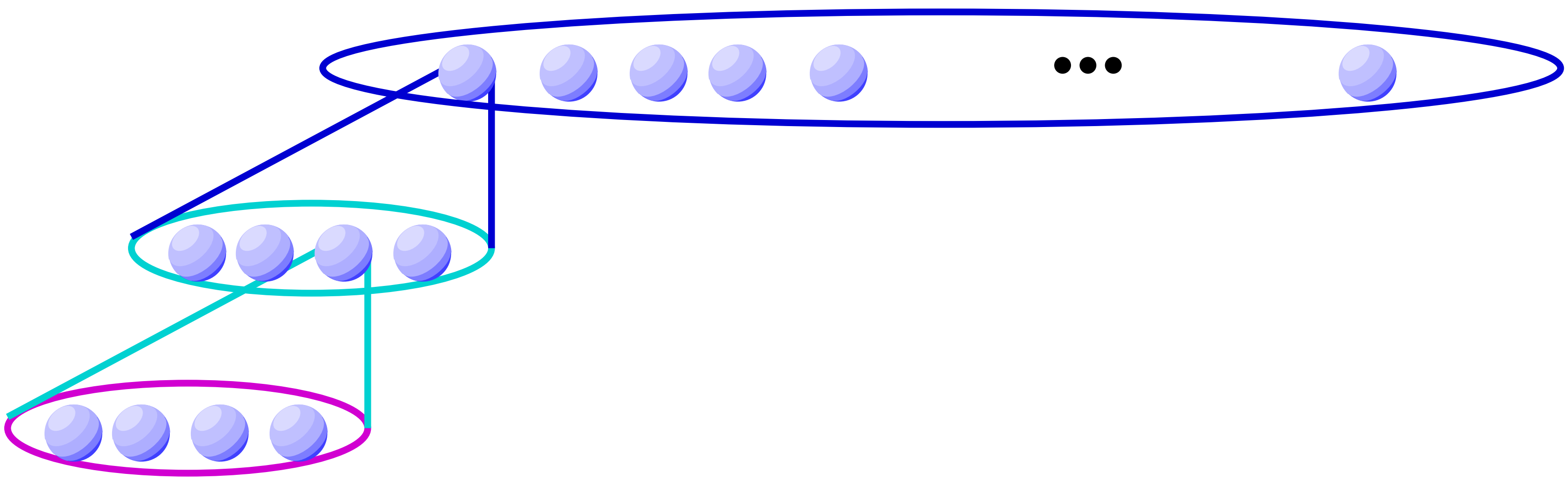
- A solution with postselected f.-t. QC:

Toward Unconditional Quantum Computing

- Problem: The states needed for f.-t. QC must be disturbed by well-bounded local errors only.

- A solution with postselected f.-t. QC:

  1. Implement postselected f.-t. QC with logical qubits based on a small concatenated block code. E.g. C_4/C_6

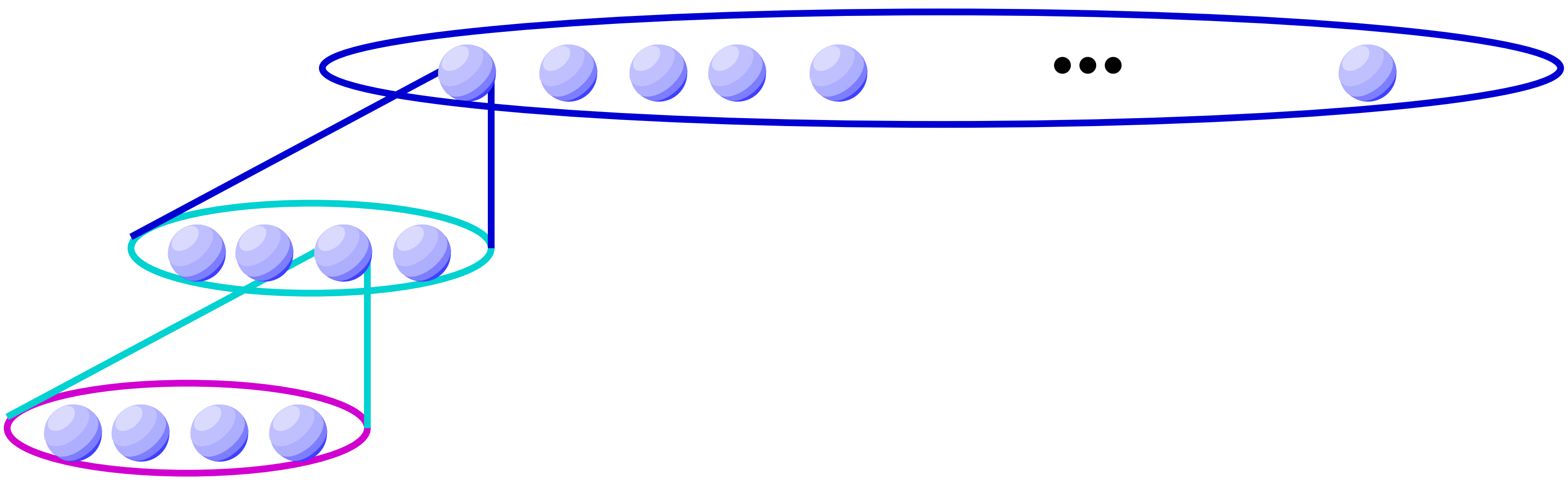


Toward Unconditional Quantum Computing

- Problem: The states needed for f.-t. QC must be disturbed by well-bounded local errors only.

- A solution with postselected f.-t. QC:

    1. Implement postselected f.-t. QC with logical qubits based on a small concatenated block code. E.g. C_4/C_6
    2. Use this to prepare the desired state in encoded form.

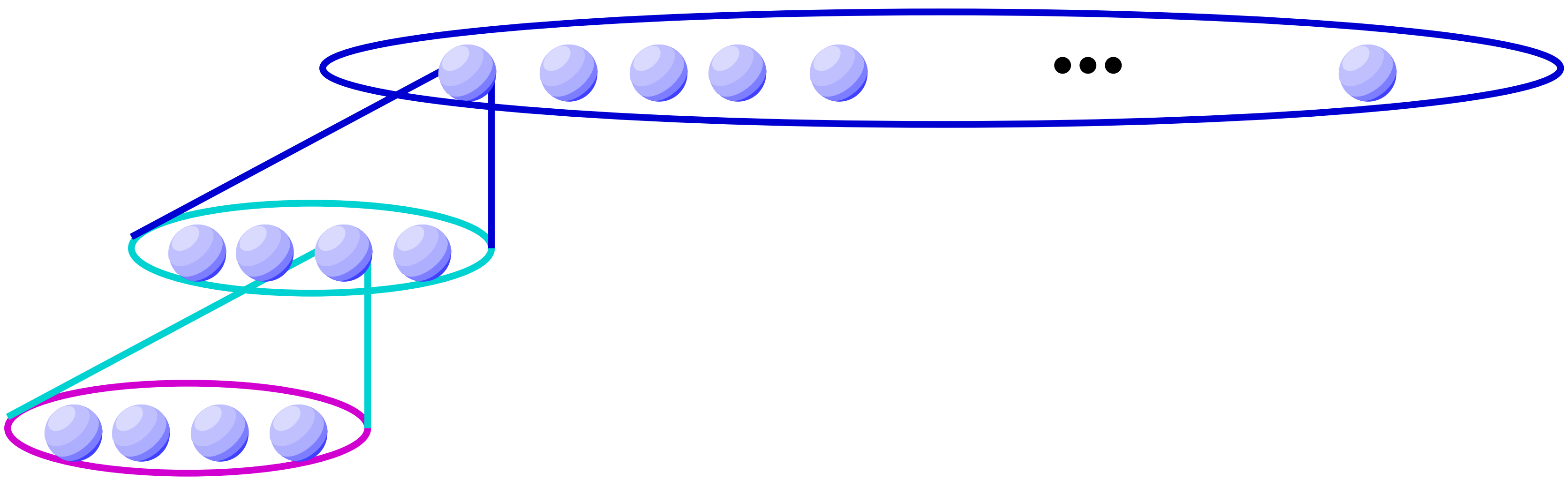


Toward Unconditional Quantum Computing

- Problem: The states needed for f.-t. QC must be disturbed by well-bounded local errors only.

- A solution with postselected f.-t. QC:

  1. Implement postselected f.-t. QC with
     logical qubits based on a small concatenated block code.
     E.g. C_4/C_6
  2. Use this to prepare the desired state in encoded form.
  3. Decode the block code through all levels.

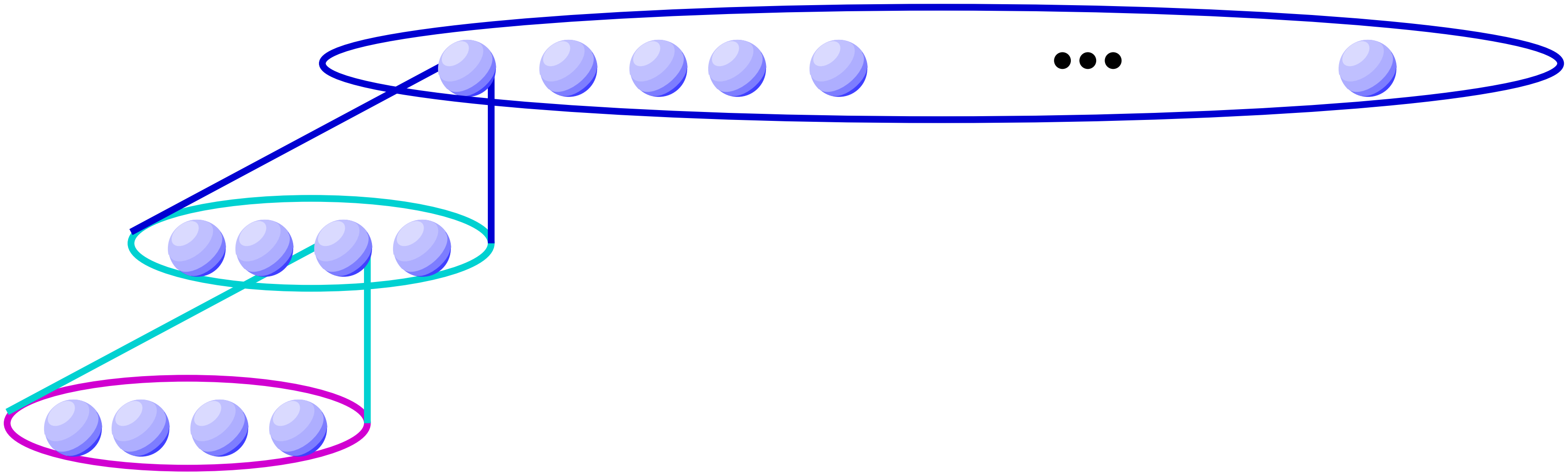


Toward Unconditional Quantum Computing

- Problem: The states needed for f.-t. QC must be disturbed by well-bounded local errors only.

- A solution with postselected f.-t. QC:

  1. Implement postselected f.-t. QC with logical qubits based on a small concatenated block code. E.g. C_4/C_6
  2. Use this to prepare the desired state in encoded form.
  3. Decode the block code through all levels.
  4. Accept the state if no errors are detected in decoding.



Power of Clifford-Pauli Operations

- The CSS operations, $\mathcal{CSS}$:
  Preparation of $|0\rangle$ and $|+\rangle$, **cnot**, Measurement of X and Z.

Power of Clifford-Pauli Operations

- The CSS operations, $\mathcal{CSS}$:
  Preparation of $|0\rangle$ and $|+\rangle$, **cnot**, Measurement of X and Z.
  - CSS operations suffice for encoding/decoding CSS codes.

Power of Clifford-Pauli Operations

- The CSS operations, $\mathcal{CSS}$:
  Preparation of $|0\rangle$ and $|+\rangle$, **cnot**, Measurement of X and Z.
  - CSS operations suffice for encoding/decoding CSS codes.
- Universal quantum computation is possible with $\mathcal{CSS}$, H and $|\pi/8\rangle$-preparation.

$$\text{QC} = \overbrace{\mathcal{CSS} + \underbrace{\text{“}\epsilon\text{”}}_{H}}^{\subseteq\mathcal{N}} + \underbrace{\text{“}\delta\text{”}}_{|\pi/8\rangle}.$$

Power of Clifford-Pauli Operations

- The CSS operations, $\mathcal{CSS}$:
  Preparation of $|0\rangle$ and $|+\rangle$, **cnot**, Measurement of X and Z.
  - CSS operations suffice for encoding/decoding CSS codes.
- Universal quantum computation is possible with $\mathcal{CSS}$, H and $|\pi/8\rangle$-preparation.

$$\mathrm{QC} = \overbrace{\mathcal{CSS} + \underbrace{\text{“}\epsilon\text{”}}_{H}}^{\subseteq \mathcal{N}} + \underbrace{\text{“}\delta\text{”}}_{|\pi/8\rangle}.$$

- A fault-tolerant computation strategy:
  1. Implement a fault tolerant CSS computer, i.e. arbitrarily accurate logical $\mathcal{CSS}$ with feedforward.

Power of Clifford-Pauli Operations

- The CSS operations, $\mathcal{CSS}$:
  Preparation of $|0\rangle$ and $|+\rangle$, **cnot**, Measurement of X and Z.
  - CSS operations suffice for encoding/decoding CSS codes.

- Universal quantum computation is possible with $\mathcal{CSS}$, H and $|\pi/8\rangle$-preparation.

$$\mathrm{QC} = \overbrace{\mathcal{CSS} + \underbrace{\text{“}\epsilon\text{”}}_{H}}^{\subseteq \mathcal{N}} + \underbrace{\text{“}\delta\text{”}}_{|\pi/8\rangle}.$$

- A fault-tolerant computation strategy:
  1. Implement a fault tolerant CSS computer, i.e. arbitrarily accurate logical $\mathcal{CSS}$ with feedforward.
  2. $+\text{“}\epsilon\text{”} + \text{“}\delta\text{”}\ldots$

$+\text{“}\delta\text{”}$: $|\pi/8\rangle$ purification using good $\mathcal{CSS} + \text{”}\epsilon\text{”}$
Bravyi&Kitaev (2004) [15], Knill (2004) [16]

Power of Clifford-Pauli Operations

- The CSS operations, $\mathcal{CSS}$:
  Preparation of $|0\rangle$ and $|+\rangle$, **cnot**, Measurement of X and Z.
  - CSS operations suffice for encoding/decoding CSS codes.

- Universal quantum computation is possible with $\mathcal{CSS}$, H and $|\pi/8\rangle$-preparation.

$$\text{QC} = \overbrace{\mathcal{CSS} + \underbrace{\text{``}\epsilon\text{''}}_{H}}^{\subseteq\mathcal{N}} + \underbrace{\text{``}\delta\text{''}}_{|\pi/8\rangle}.$$

- A fault-tolerant computation strategy:
  1. Implement a fault tolerant CSS computer, i.e. arbitrarily accurate logical $\mathcal{CSS}$ with feedforward.
  2. $+\text{``}\epsilon\text{''} + \text{``}\delta\text{''}\ldots$

  $+\text{``}\delta\text{''}$: $|\pi/8\rangle$ purification using good $\mathcal{CSS} + \text{''}\epsilon\text{''}$
  Bravyi&Kitaev (2004) [15], Knill (2004) [16]

- F.-t. $\mathcal{CSS}$ and ($|\pi/8\rangle$ error) $\leq$ ($|0\rangle$, $|+\rangle$ error) $\Rightarrow$ f.-t. QC?

Simulation of the C_4/C_6 Architecture

- Monte-Carlo simulation to determine C_4/C_6 error behavior up to level 4.
  - Implementation language: Octave.
  - Method: Graph-code normal form.

Simulation of the C_4/C_6 Architecture

- Monte-Carlo simulation to determine C_4/C_6 error behavior up to level 4.
  - Implementation language: Octave.
  - Method: Graph-code normal form.
- Implementation issues:
  - Avoid transients:
    Verify error behavior of the second operation.

Simulation of the C_4/C_6 Architecture

- Monte-Carlo simulation to determine C_4/C_6 error behavior up to level 4.
  - Implementation language: Octave.
  - Method: Graph-code normal form.
- Implementation issues:
  - Avoid transients:
    Verify error behavior of the second operation.
  - Verify full error behavior:
    Operate on one-half of an entangled pair.

Simulation of the C_4/C_6 Architecture

- Monte-Carlo simulation to determine C_4/C_6 error behavior up to level 4.
  - Implementation language: Octave.
  - Method: Graph-code normal form.

- Implementation issues:
  - Avoid transients:
    Verify error behavior of the second operation.
  - Verify full error behavior:
    Operate on one-half of an entangled pair.
  - Verify that errors do not compound:
    Check on a long sequence of operations.

Simulation of the C_4/C_6 Architecture

- Monte-Carlo simulation to determine C_4/C_6 error behavior up to level 4.
  - Implementation language: Octave.
  - Method: Graph-code normal form.

- Implementation issues:
  - Avoid transients:
    Verify error behavior of the second operation.
  - Verify full error behavior:
    Operate on one-half of an entangled pair.
  - Verify that errors do not compound:
    Check on a long sequence of operations.
  - Architecture is not strictly concatenated:
    Full simulations at high levels.

Simulation of the C_4/C_6 Architecture

- Monte-Carlo simulation to determine C_4/C_6 error behavior up to level 4.
  - Implementation language: Octave.
  - Method: Graph-code normal form.

- Implementation issues:
  - Avoid transients:
    Verify error behavior of the second operation.
  - Verify full error behavior:
    Operate on one-half of an entangled pair.
  - Verify that errors do not compound:
    Check on a long sequence of operations.
  - Architecture is not strictly concatenated:
    Full simulations at high levels.
  - Keep track of resources used.

Error Model

- Each operation's errors are uniformly random Pauli errors.

Error Model

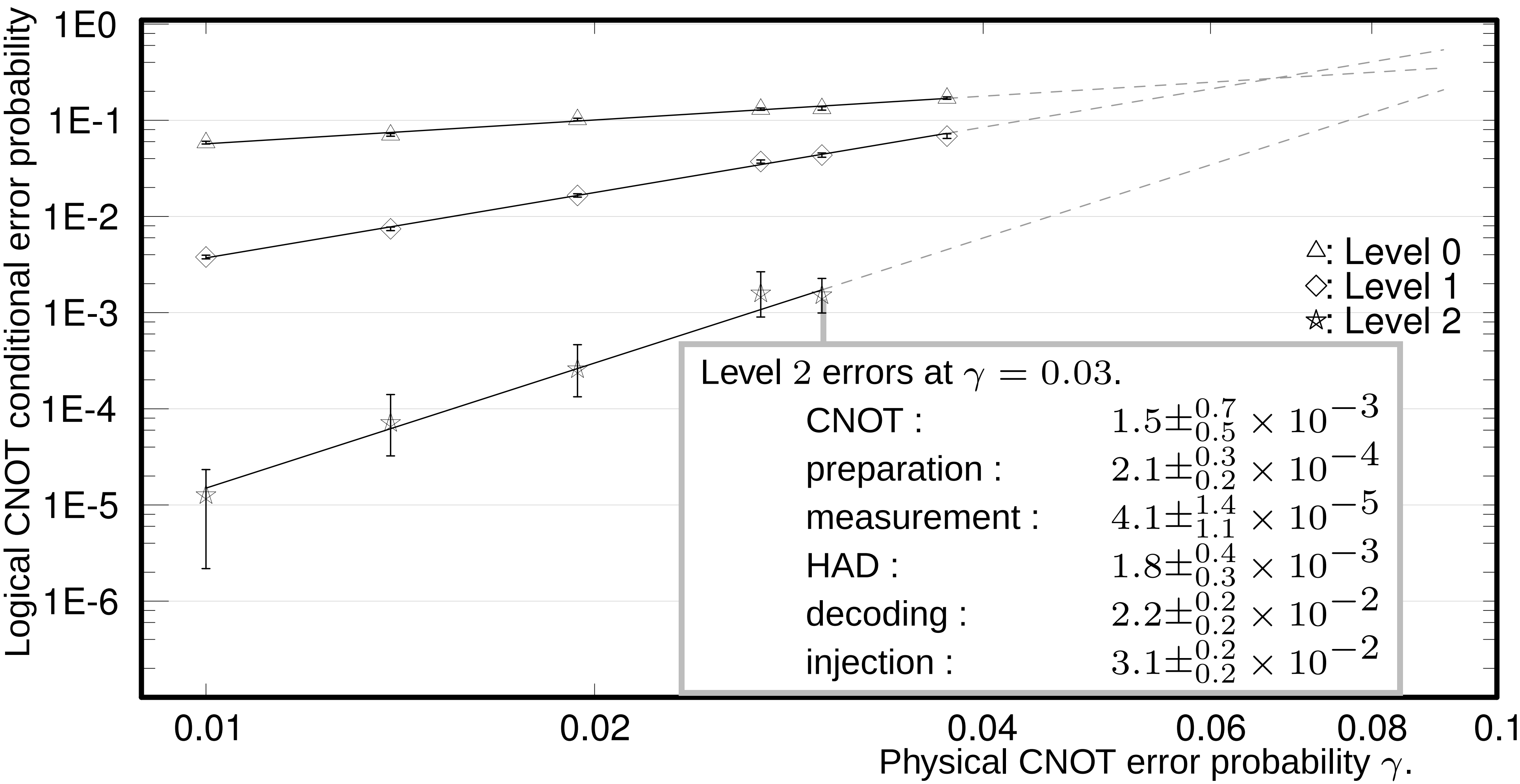
- Each operation's errors are uniformly random Pauli errors.
  - $|\mathfrak{o}\rangle$ preparation with noise:
    Prepared state is $\{(1-e_p) : |\mathfrak{o}\rangle, e_p : \sigma_x|\mathfrak{o}\rangle\}$.
  - Measurement with noise:
    $|\psi\rangle \rightarrow \{(1-e_m) : \mathbb{1}, e_m : \sigma_x\}$ before σ_z measurement.
  - No operation (memory) with noise:
    $\{(1-e_n) : \mathbb{1}, e_n/3 : \sigma_x, e_n/3 : \sigma_y, e_n/3 : \sigma_z\}H$.
  - Hadamard with noise:
    $\{(1-e_h) : \mathbb{1}, e_h/3 : \sigma_x, e_h/3 : \sigma_y, e_h/3 : \sigma_z\}H$.
  - Cnot with noise:
    $\{(1-e_c) : \mathbb{1}, e_c/15 : {\sigma_x}^{(1)}, \ldots, e_c/15 : {\sigma_z}^{(1)}{\sigma_z}^{(2)}\}$**cnot**$^{(12)}$.

Error Model

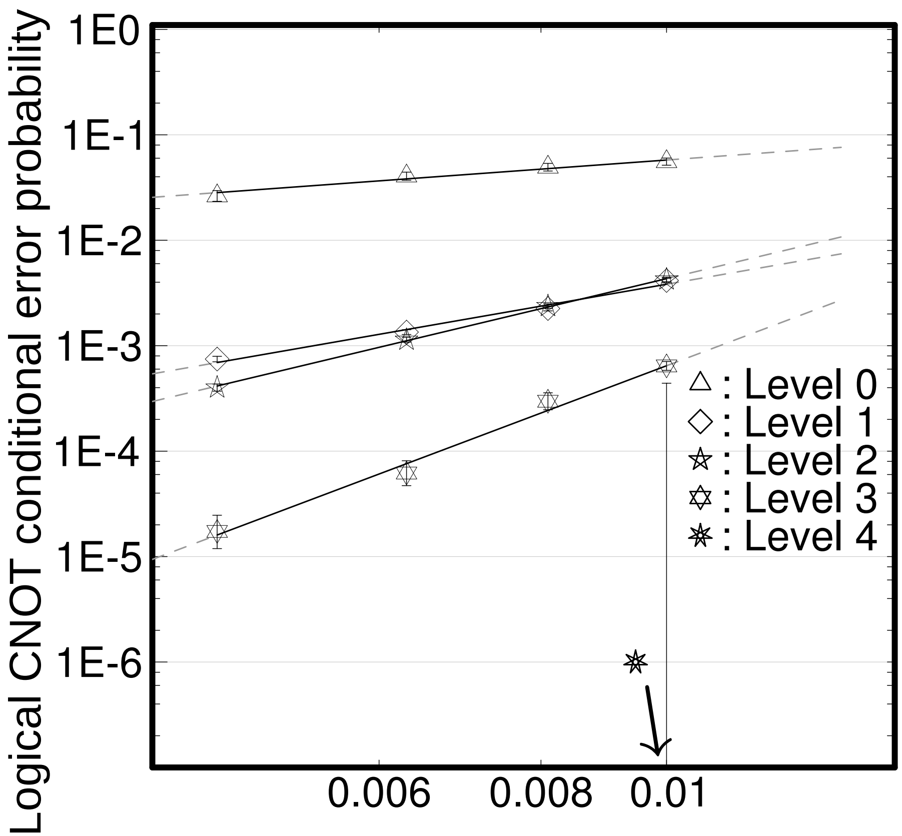
- Each operation's errors are uniformly random Pauli errors.
  - $|\mathfrak{o}\rangle$ preparation with noise:
    Prepared state is $\{(1-e_p) : |\mathfrak{o}\rangle, e_p : \sigma_x|\mathfrak{o}\rangle\}$.
  - Measurement with noise:
    $|\psi\rangle \rightarrow \{(1-e_m) : \mathbb{1}, e_m : \sigma_x\}$ before σ_z measurement.
  - No operation (memory) with noise:
    $\{(1-e_n) : \mathbb{1}, e_n/3 : \sigma_x, e_n/3 : \sigma_y, e_n/3 : \sigma_z\}H$.
  - Hadamard with noise:
    $\{(1-e_h) : \mathbb{1}, e_h/3 : \sigma_x, e_h/3 : \sigma_y, e_h/3 : \sigma_z\}H$.
  - Cnot with noise:
    $\{(1-e_c) : \mathbb{1}, e_c/15 : {\sigma_x}^{(1)}, \ldots, e_c/15 : {\sigma_z}^{(1)}{\sigma_z}^{(2)}\}$**cnot**$^{(12)}$.
- Agnostic choice for $e_{p,m,h,c}$?

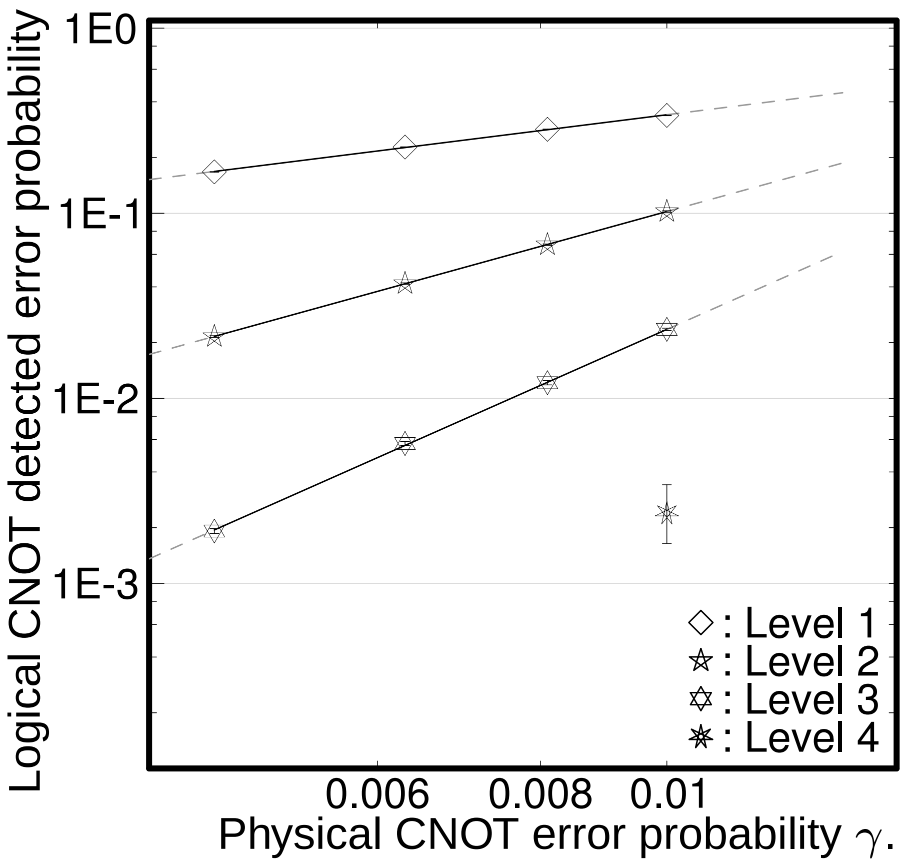
$$e_c = \epsilon, \quad e_h = \frac{4}{5}\epsilon, \quad e_p = \frac{4}{15}\epsilon, \quad e_m = \frac{4}{15}\epsilon, \quad e_n \leq e_h.$$

Conditional Logical Errors with Postselection

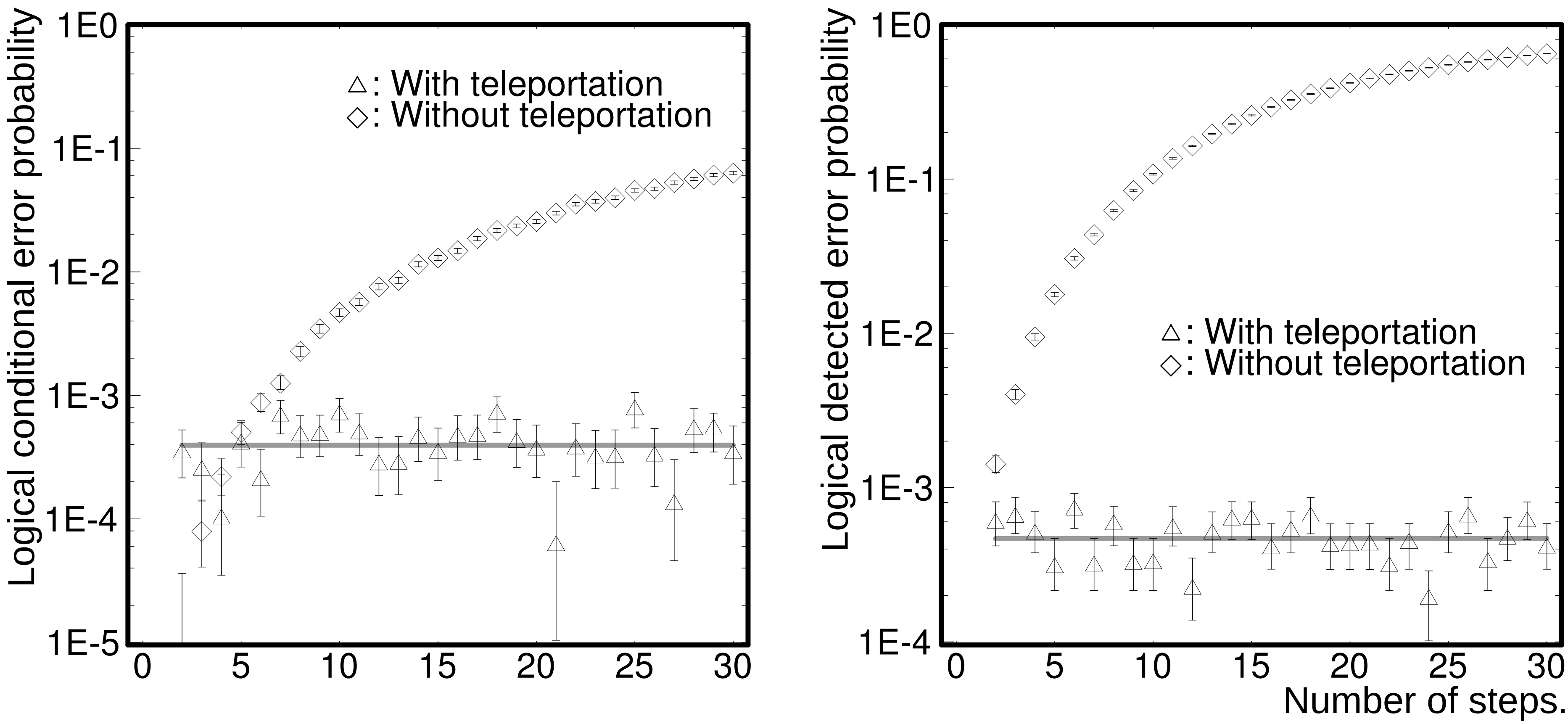


Error Probabilities for Scalable QC



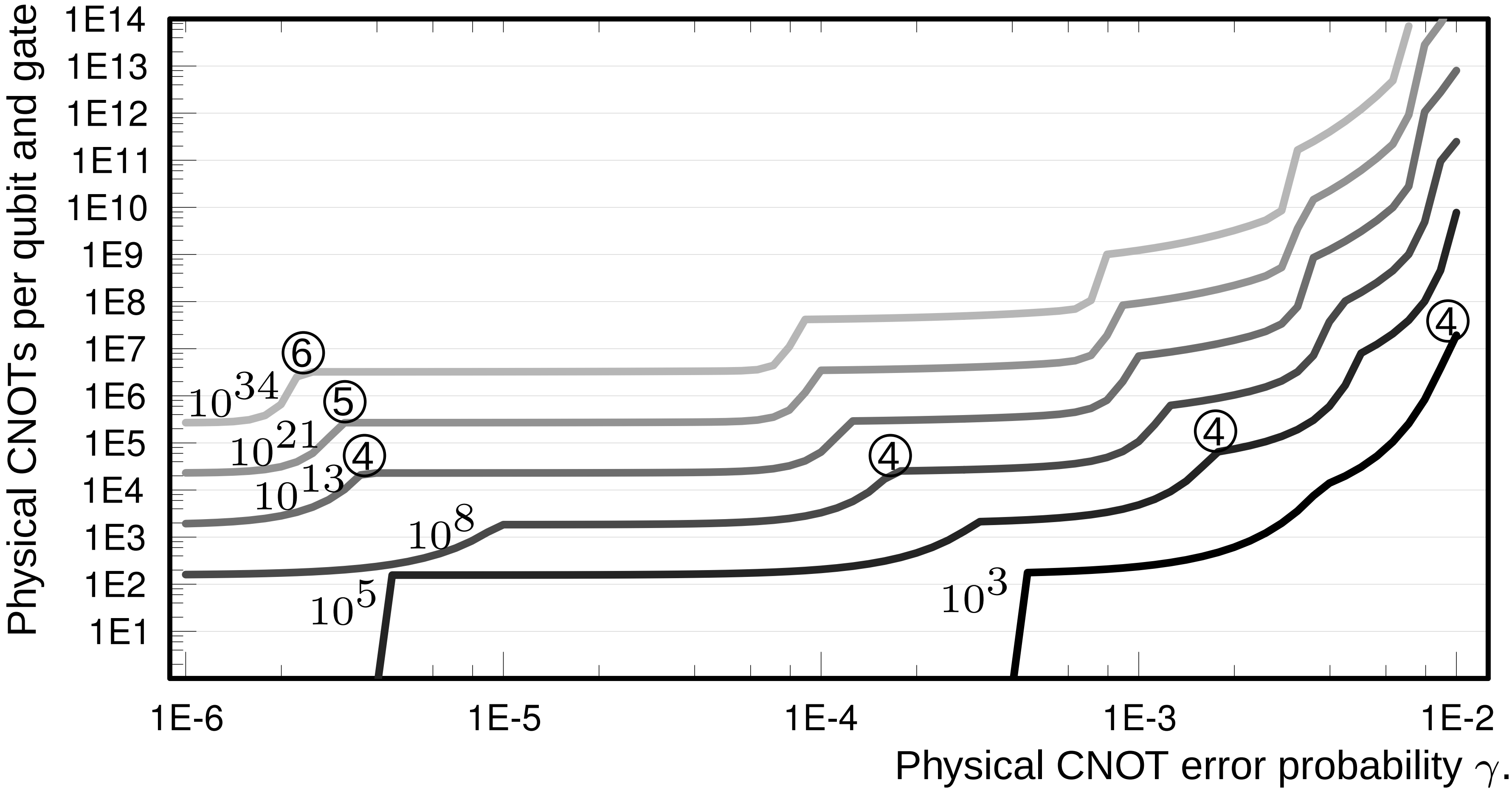


Error Compounding



- 30 applications of the HAD gate at level 3.

Conclusion



Physical CNOTs per qubit and gate
1E14
1E13
1E12
1E11
1E10
1E9
1E8
1E7
1E6
1E5
1E4
1E3
1E2
1E1
1E-6
1E-5
1E-4
1E-3
1E-2
Physical CNOT error probability γ.
10^{34}
6
10^{21}
5
10^{13}
4
10^{8}
10^{5}
4
10^{3}
4
4

The Clifford-Pauli Group

- Pauli matrix notation.

$$I = \mathbb{1}, X = \sigma_x, Y = \sigma_y, Z = \sigma_z$$
$$[IXIYI] = {\sigma_x}^{(2)}{\sigma_y}^{(4)} = \mathbb{1} \otimes \sigma_x \otimes \mathbb{1} \otimes \sigma_y \otimes \mathbb{1}$$

$\mathcal{P}_n$ is the set of $\pm 1 *$Pauli products on n qubits.

Back to the $[[4, 2, 2]]$-code.

The Clifford-Pauli Group

- Pauli matrix notation.

$$I = \mathbb{1}, X = \sigma_x, Y = \sigma_y, Z = \sigma_z$$
$$[IXIYI] = \sigma_x^{(2)} \sigma_y^{(4)} = \mathbb{1} \otimes \sigma_x \otimes \mathbb{1} \otimes \sigma_y \otimes \mathbb{1}$$

  $\mathcal{P}_n$ is the set of $\pm 1 *$Pauli products on n qubits.

- The Clifford-Pauli group:

$$\mathcal{N}_n = \left\{ U \,|\, U\mathcal{P}_n U^\dagger = \mathcal{P}_n, U \text{ is unitary} \right\}$$

Back to the $[[4,2,2]]$-code.

The Clifford-Pauli Group

- Pauli matrix notation.

$$I = \mathbb{1}, X = \sigma_x, Y = \sigma_y, Z = \sigma_z$$
$$[IXIYI] = {\sigma_x}^{(2)}{\sigma_y}^{(4)} = \mathbb{1} \otimes \sigma_x \otimes \mathbb{1} \otimes \sigma_y \otimes \mathbb{1}$$

$\mathcal{P}_n$ is the set of $\pm 1 *$Pauli products on n qubits.

- The Clifford-Pauli group:

$$\mathcal{N}_n = \left\{ U \,\middle|\, U\mathcal{P}_n U^\dagger = \mathcal{P}_n, U \text{ is unitary} \right\}$$

- Generators of $\mathcal{N}_n$: **cnot**, H, $e^{-iZ\pi/4}$

Back to the $[[4,2,2]]$-code.

The Clifford-Pauli Group

- Pauli matrix notation.

$$I = \mathbb{1}, X = \sigma_x, Y = \sigma_y, Z = \sigma_z$$
$$[IXIYI] = {\sigma_x}^{(2)} {\sigma_y}^{(4)} = \mathbb{1} \otimes \sigma_x \otimes \mathbb{1} \otimes \sigma_y \otimes \mathbb{1}$$

$\mathcal{P}_n$ is the set of $\pm 1 *$Pauli products on n qubits.

- The Clifford-Pauli group:

$$\mathcal{N}_n = \left\{ U \,|\, U\mathcal{P}_n U^\dagger = \mathcal{P}_n, U \text{ is unitary} \right\}$$

- Generators of $\mathcal{N}_n$: **cnot**, H, $e^{-iZ\pi/4}$

- **Theorem.** Any quantum computation using Z-eigenstate preparation, operators in $\mathcal{N}_n$, Z-measurements and feedforward can be efficiently classically simulated.

Gottesman (1997) [17]

Back to the $[[4, 2, 2]]$-code.

Fault-Tolerant Quantum Computing I

- **Requirement 3 for scalable QC[a] implementation:**
  Sufficiently low noise affecting physical gates and memory.

DiVincenzo (2000) [1]

[a] Quantum Computing. [b] Fault-Tolerant Quantum Computing

Fault-Tolerant Quantum Computing I

- **Requirement 3 for scalable QC[a] implementation:**
  Sufficiently low noise affecting physical gates and memory.

  DiVincenzo (2000) [1]

- *Error model*: The type of noise affecting a QC implementation.

[a] Quantum Computing. [b] Fault-Tolerant Quantum Computing

Fault-Tolerant Quantum Computing I

- **Requirement 3 for scalable QC[a] implementation:**
  Sufficiently low noise affecting physical gates and memory.

DiVincenzo (2000) [1]

- *Error model*: The type of noise affecting a QC implementation.
- *Fault-tolerant architecture*: A scheme for scalable QC in the presence of noise.

[a] Quantum Computing.

[b] Fault-Tolerant Quantum Computing

Fault-Tolerant Quantum Computing I

- **Requirement 3 for scalable QC[a] implementation:**
  Sufficiently low noise affecting physical gates and memory.

DiVincenzo (2000) [1]

- *Error model*: The type of noise affecting a QC implementation.
- *Fault-tolerant architecture*: A scheme for scalable QC in the presence of noise.

- Fundamental problems of FTQC[b]:
  1. Scalable QC with error model $\mathcal{E}$?

[a] Quantum Computing. [b] Fault-Tolerant Quantum Computing

Fault-Tolerant Quantum Computing I

- **Requirement 3 for scalable QC[a] implementation:**
  Sufficiently low noise affecting physical gates and memory.

DiVincenzo (2000) [1]

- *Error model*: The type of noise affecting a QC implementation.
- *Fault-tolerant architecture*: A scheme for scalable QC in the presence of noise.

- Fundamental problems of FTQC[b]:
  1. Scalable QC with error model $\mathcal{E}$?
  2. Scalable QC with fault-tolerant architecture $\mathcal{A}$ and with $\mathcal{E}$?

[a] Quantum Computing. [b] Fault-Tolerant Quantum Computing

Fault-Tolerant Quantum Computing I

- **Requirement 3 for scalable QC[a] implementation:**
  Sufficiently low noise affecting physical gates and memory.

DiVincenzo (2000) [1]

- *Error model*: The type of noise affecting a QC implementation.
- *Fault-tolerant architecture*: A scheme for scalable QC in the presence of noise.

- Fundamental problems of FTQC[b]:
  1. Scalable QC with error model $\mathcal{E}$?
  2. Scalable QC with fault-tolerant architecture $\mathcal{A}$ and with $\mathcal{E}$?
  3. Resources required to implement QC $\mathcal{C}$ using $\mathcal{A}$ with $\mathcal{E}$?

[a] Quantum Computing.

[b] Fault-Tolerant Quantum Computing

Fault Tolerant Quantum Computing II

Fault-Tolerance Threshold Theorem: *Given: Noisy qubits and gates. If the error rates are sufficiently low, then it is possible to efficiently process quantum information arbitrarily accurately.*

Shor (1995) [2, 3], Kitaev (1996) [4], Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6], . . . Gottesman&Preskill (1999), . . . Steane (2002) [7], . . . Knill (2004) [8, 9], Reichardt (2004) [10]

Fault Tolerant Quantum Computing II

Fault-Tolerance Threshold Theorem: *Given: Noisy qubits and gates. If the error rates are sufficiently low, then it is possible to efficiently process quantum information arbitrarily accurately.*

Shor (1995) [2, 3], Kitaev (1996) [4], Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6], . . . Gottesman&Preskill (1999), . . . Steane (2002) [7], . . . Knill (2004) [8, 9], Reichardt (2004) [10]

- Error model: What is required of “noisy qubits and gates”?

Fault Tolerant Quantum Computing II

Fault-Tolerance Threshold Theorem: *Given: Noisy qubits and gates. If the error rates are sufficiently low, then it is possible to efficiently process quantum information arbitrarily accurately.*

Shor (1995) [2, 3], Kitaev (1996) [4], Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6], . . . Gottesman&Preskill (1999), . . . Steane (2002) [7], . . . Knill (2004) [8, 9], Reichardt (2004) [10]

- Error model: What is required of “noisy qubits and gates”?
- Error model parameters: What is “sufficiently low”?

Fault Tolerant Quantum Computing II

Fault-Tolerance Threshold Theorem: *Given: Noisy qubits and gates. If the error rates are sufficiently low, then it is possible to efficiently process quantum information arbitrarily accurately.*

Shor (1995) [2, 3], Kitaev (1996) [4], Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6], . . . Gottesman&Preskill (1999), . . . Steane (2002) [7], . . . Knill (2004) [8, 9], Reichardt (2004) [10]

- Error model: What is required of “noisy qubits and gates”?
- Error model parameters: What is “sufficiently low”?
- Architecture: How to implement a quantum computation?

Fault Tolerant Quantum Computing II

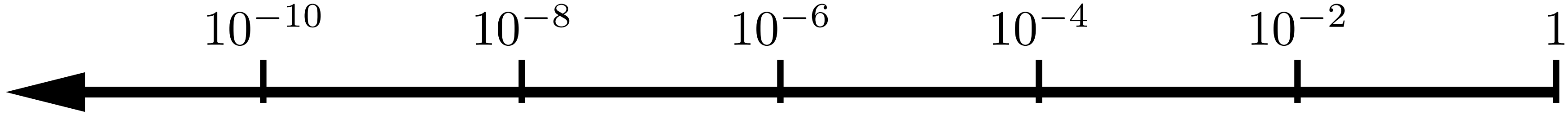
Fault-Tolerance Threshold Theorem: *Given: Noisy qubits and gates. If the error rates are sufficiently low, then it is possible to efficiently process quantum information arbitrarily accurately.*

Shor (1995) [2, 3], Kitaev (1996) [4], Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6], . . . Gottesman&Preskill (1999), . . . Steane (2002) [7], . . . Knill (2004) [8, 9], Reichardt (2004) [10]

- Error model: What is required of “noisy qubits and gates”?
- Error model parameters: What is “sufficiently low”?
- Architecture: How to implement a quantum computation?
- Resource requirements: What is “efficient”?

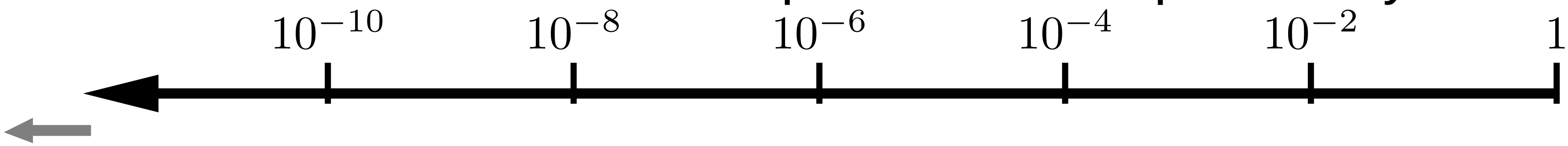
Error Thresholds: Proofs and Estimates

- Error thresholds in model-dependent “error probability”.



Error Thresholds: Proofs and Estimates

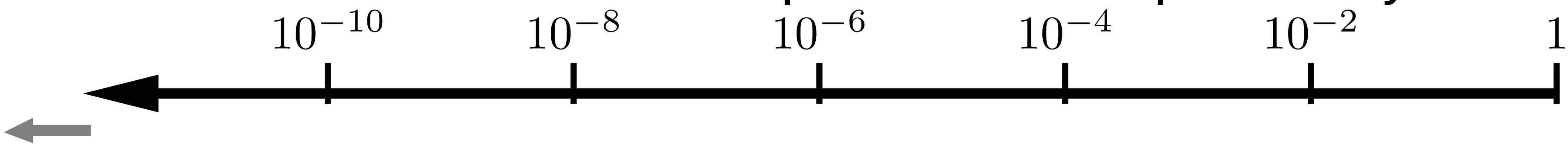
- Error thresholds in model-dependent "error probability".



Correlations $\rightarrow 1$. Clemens&Siddiqui&Gea-Banacloche (2004) [11]

Error Thresholds: Proofs and Estimates

- Error thresholds in model-dependent “error probability”.



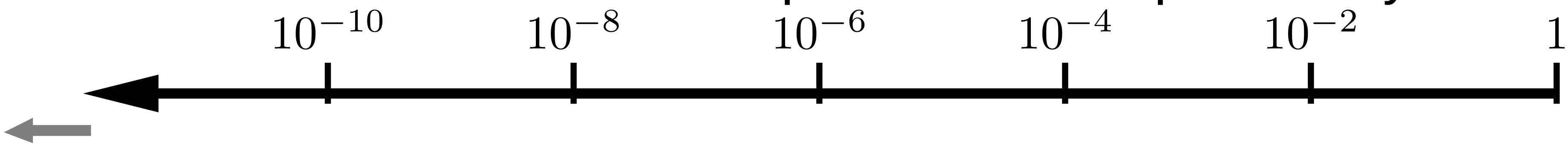
Correlations $\rightarrow 1.$ Clemens&Siddiqui&Gea-Banacloche (2004) [11]

Adversarial, quasi-independent.

Knill&Laflamme&Zurek (1996) [6], Terhal&Burkard (2004) [12], Alicki (2004) [13]

Error Thresholds: Proofs and Estimates

- Error thresholds in model-dependent “error probability”.



Correlations $\rightarrow 1$. Clemens&Siddiqui&Gea-Banacloche (2004) [11]

Adversarial, quasi-independent.

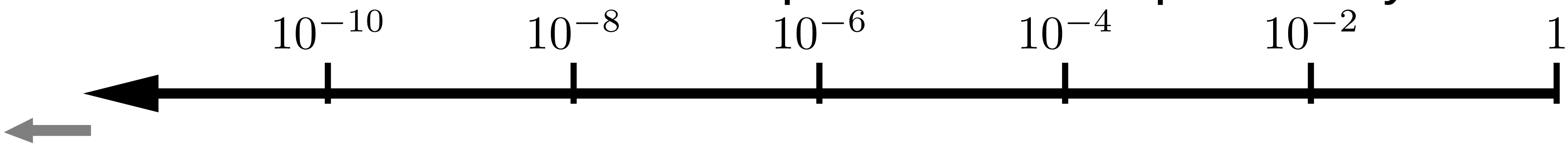
Knill&Laflamme&Zurek (1996) [6], Terhal&Burkard (2004) [12], Alicki (2004) [13]

Adversarial, quasi-independent, probabilistic Pauli.

Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6]

Error Thresholds: Proofs and Estimates

- Error thresholds in model-dependent “error probability”.



Correlations $\rightarrow 1$. Clemens&Siddiqui&Gea-Banacloche (2004) [11]

Adversarial, quasi-independent.

Knill&Laflamme&Zurek (1996) [6], Terhal&Burkard (2004) [12], Alicki (2004) [13]

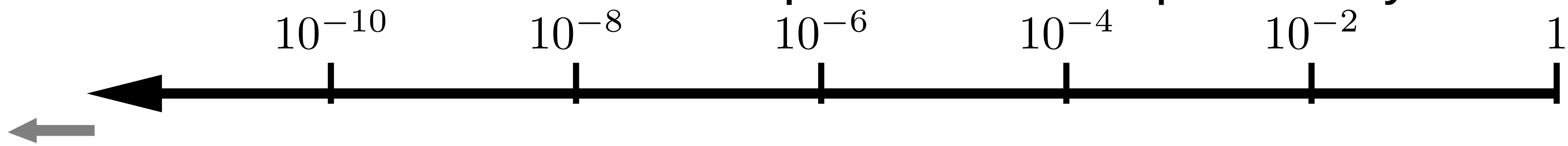
Adversarial, quasi-independent, probabilistic Pauli.

Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6]

Depolarizing errors. Gottesman&Preskill (1999)

Error Thresholds: Proofs and Estimates

- Error thresholds in model-dependent “error probability”.



Correlations $\rightarrow 1$. Clemens&Siddiqui&Gea-Banacloche (2004) [11]

Adversarial, quasi-independent.

Knill&Laflamme&Zurek (1996) [6], Terhal&Burkard (2004) [12], Alicki (2004) [13]

Adversarial, quasi-independent, probabilistic Pauli.

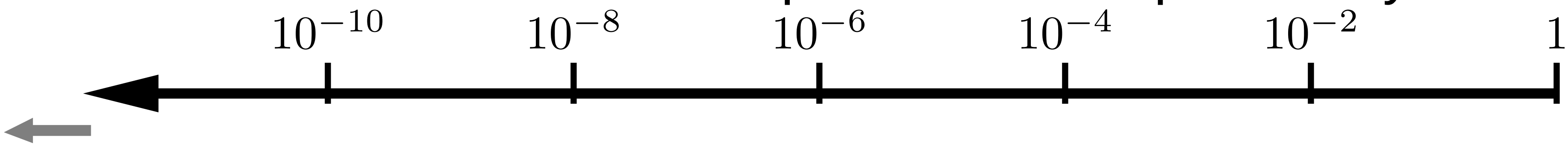
Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6]

Depolarizing errors. Gottesman&Preskill (1999)

Steane (2002) [7]

Error Thresholds: Proofs and Estimates

- Error thresholds in model-dependent “error probability”.



Correlations $\rightarrow 1$. Clemens&Siddiqui&Gea-Banacloche (2004) [11]

Adversarial, quasi-independent.

Knill&Laflamme&Zurek (1996) [6], Terhal&Burkard (2004) [12], Alicki (2004) [13]

Adversarial, quasi-independent, probabilistic Pauli.

Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6]

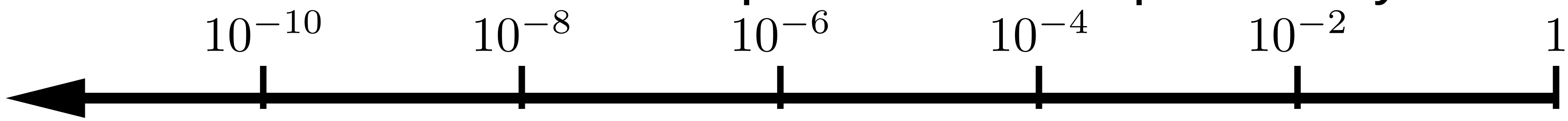
Depolarizing errors. Gottesman&Preskill (1999)

Steane (2002) [7]

Knill (2004) [8, 9], Reichardt (2004) [10]

Error Thresholds: Proofs and Estimates

- Error thresholds in model-dependent "error probability".



Correlations $\rightarrow 1$. Clemens&Siddiqui&Gea-Banacloche (2004) [11]

Adversarial, quasi-independent.

Knill&Laflamme&Zurek (1996) [6], Terhal&Burkard (2004) [12], Alicki (2004) [13]

Adversarial, quasi-independent, probabilistic Pauli.

Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6]

Depolarizing errors. Gottesman&Preskill (1999)

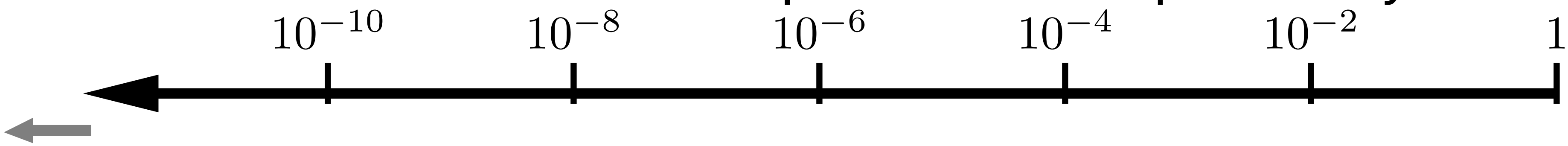
Steane (2002) [7]

Knill (2004) [8, 9], Reichardt (2004) [10]

Detected errors. Knill (2003) [14]

Error Thresholds: Proofs and Estimates

- Error thresholds in model-dependent “error probability”.



Correlations $\to 1$. Clemens&Siddiqui&Gea-Banacloche (2004) [11]

Adversarial, quasi-independent.

Knill&Laflamme&Zurek (1996) [6], Terhal&Burkard (2004) [12], Alicki (2004) [13]

Adversarial, quasi-independent, probabilistic Pauli.

Aharonov&Ben-Or (1996) [5], Knill&Laflamme&Zurek (1996) [6]

Depolarizing errors. Gottesman&Preskill (1999)

Steane (2002) [7]

Knill (2004) [8, 9], Reichardt (2004) [10]

Detected errors. Knill (2003) [14]

Unintended Z-measurements.

Knill (2002)

The Setting

- Physical qubit engineering process.

The Setting

- Physical qubit engineering process.
  - Minimize noise in classical control fields.

    E.g. by proper shielding.

    . . .

The Setting

- Physical qubit engineering process.
  - Minimize noise in classical control fields.
    E.g. by proper shielding.
  - Reduce systematic errors in gates.
    E.g. by self-correcting pulse sequences.
    . . .

The Setting

- Physical qubit engineering process.
  - Minimize noise in classical control fields.
    E.g. by proper shielding.
  - Reduce systematic errors in gates.
    E.g. by self-correcting pulse sequences.
  - Take advantage of available noiseless subsystems.
    E.g. decoherence free subspaces.

  . . .

The Setting

- Physical qubit engineering process.
  - Minimize noise in classical control fields.
    E.g. by proper shielding.
  - Reduce systematic errors in gates.
    E.g. by self-correcting pulse sequences.
  - Take advantage of available noiseless subsystems.
    E.g. decoherence free subspaces.
  - Balance noisy behavior.
    E.g. Improve measurements if **cnot**'s have low noise.
    . . .

The Setting

- Physical qubit engineering process.
  - Minimize noise in classical control fields.
    E.g. by proper shielding.
  - Reduce systematic errors in gates.
    E.g. by self-correcting pulse sequences.
  - Take advantage of available noiseless subsystems.
    E.g. decoherence free subspaces.
  - Balance noisy behavior.
    E.g. Improve measurements if **cnot**'s have low noise.
  - Take advantage of error-detection if possible.
    E.g. by detecting emitted photons.

    ...

The Setting

- Physical qubit engineering process.
  - Minimize noise in classical control fields.
    E.g. by proper shielding.
  - Reduce systematic errors in gates.
    E.g. by self-correcting pulse sequences.
  - Take advantage of available noiseless subsystems.
    E.g. decoherence free subspaces.
  - Balance noisy behavior.
    E.g. Improve measurements if **cnot**'s have low noise.
  - Take advantage of error-detection if possible.
    E.g. by detecting emitted photons.

    ...

- To be considered here: Model-independent methods.

The Setting

- Physical qubit engineering process.
  - Minimize noise in classical control fields.
    E.g. by proper shielding.
  - Reduce systematic errors in gates.
    E.g. by self-correcting pulse sequences.
  - Take advantage of available noiseless subsystems.
    E.g. decoherence free subspaces.
  - Balance noisy behavior.
    E.g. Improve measurements if **cnot**'s have low noise.
  - Take advantage of error-detection if possible.
    E.g. by detecting emitted photons.

    . . .

- To be considered here: Model-independent methods.
  - Further physical engineering is relatively expensive.

The Setting

- Physical qubit engineering process.
  - Minimize noise in classical control fields.
    E.g. by proper shielding.
  - Reduce systematic errors in gates.
    E.g. by self-correcting pulse sequences.
  - Take advantage of available noiseless subsystems.
    E.g. decoherence free subspaces.
  - Balance noisy behavior.
    E.g. Improve measurements if **cnot**'s have low noise.
  - Take advantage of error-detection if possible.
    E.g. by detecting emitted photons.

    ...

- To be considered here: Model-independent methods.
  - Further physical engineering is relatively expensive.
  - Errors are generic, with no known exploitable biases.

Structural Assumptions

Physical resources:

- Arbitrarily many "physical" qubits can be called on.

Structural Assumptions

Physical resources:

- Arbitrarily many “physical” qubits can be called on.

Local control capabilities:

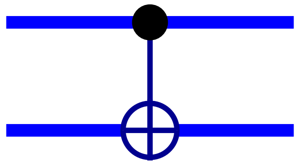
Structural Assumptions

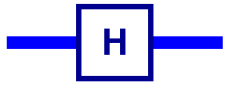
Physical resources:

- Arbitrarily many "physical" qubits can be called on.

Local control capabilities:

- Can apply one and two qubit gates.
  - **cnot**.
  - **H**.





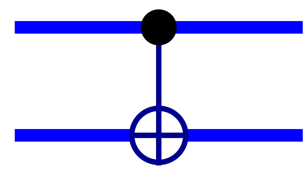
Structural Assumptions

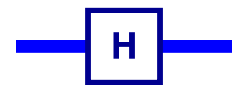
Physical resources:

- Arbitrarily many "physical" qubits can be called on.

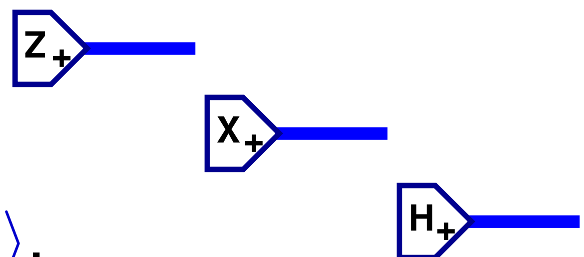
Local control capabilities:

- Can apply one and two qubit gates.
  - **cnot**.
  - **H**.





- At any time, can prepare qubits.
  - $|0\rangle$
  - $|+\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$.
  - $|\pi/8\rangle = \cos(\pi/8)|0\rangle + \sin(\pi/8)|1\rangle$.



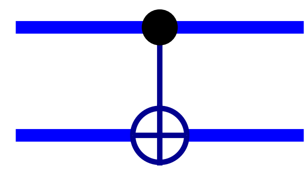
Structural Assumptions

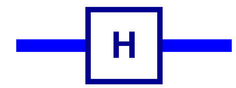
Physical resources:

- Arbitrarily many “physical” qubits can be called on.

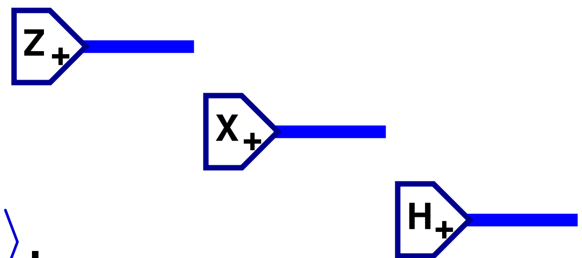
Local control capabilities:

- Can apply one and two qubit gates.
  - **cnot**.
  - **H**.

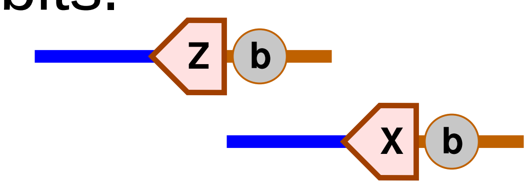




- At any time, can prepare qubits.
  - $|0\rangle$
  - $|+\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)$.
  - $|\pi/8\rangle = \cos(\pi/8)|0\rangle + \sin(\pi/8)|1\rangle$.



- At any time, can measure qubits.
  - $|0\rangle/|1\rangle$ measurement.
  - $|+\rangle/|-\rangle$ measurement.

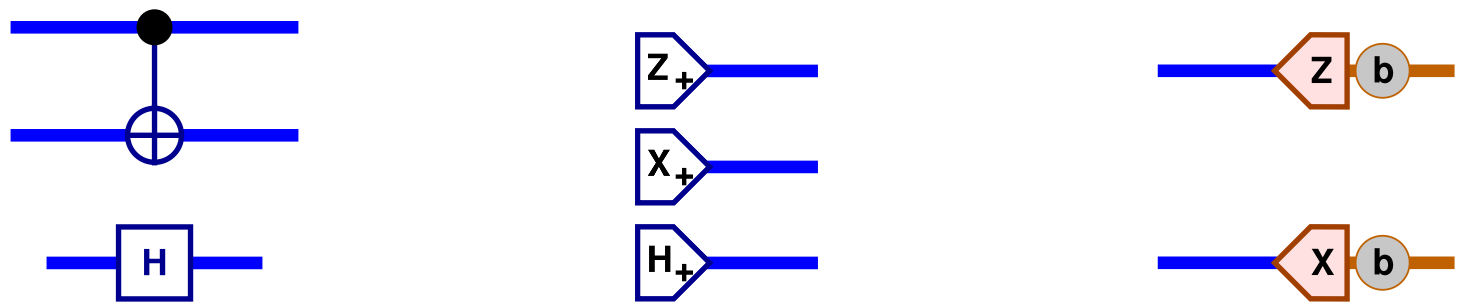


Structural Assumptions

Physical resources:

- Arbitrarily many “physical” qubits can be called on.

Local control capabilities:

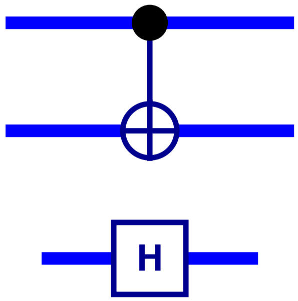


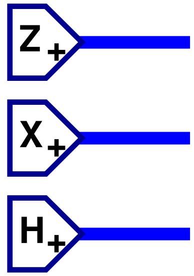
Structural Assumptions

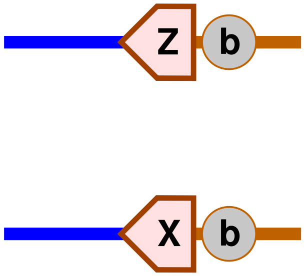
Physical resources:

- Arbitrarily many “physical” qubits can be called on.

Local control capabilities:







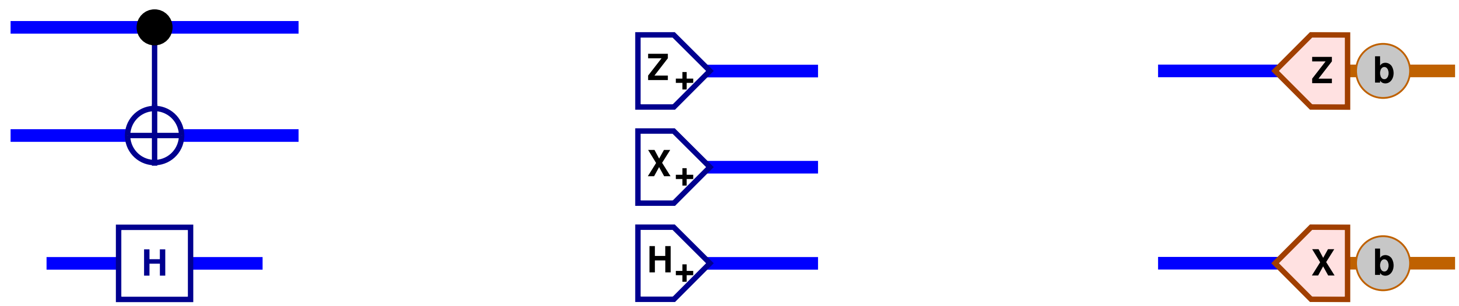
Global control capabilities:

Structural Assumptions

Physical resources:

- Arbitrarily many “physical” qubits can be called on.

Local control capabilities:



Global control capabilities:

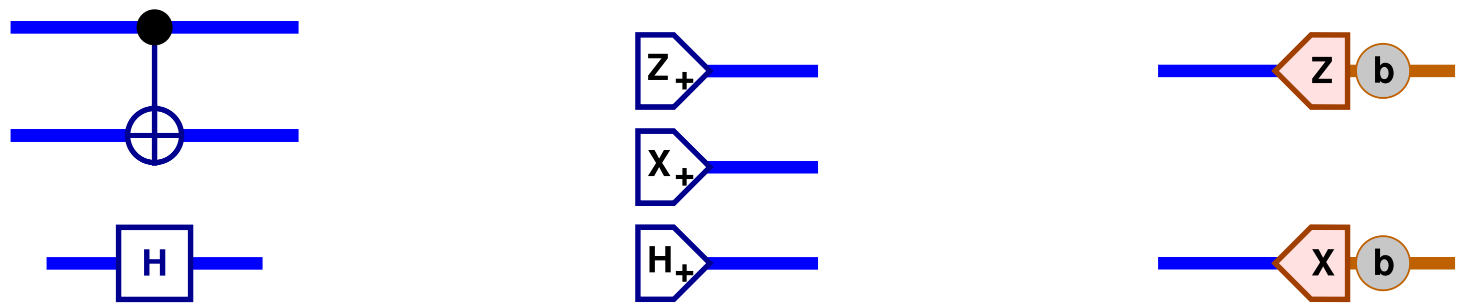
- Massive parallelism or no memory error.

Structural Assumptions

Physical resources:

- Arbitrarily many “physical” qubits can be called on.

Local control capabilities:



Global control capabilities:

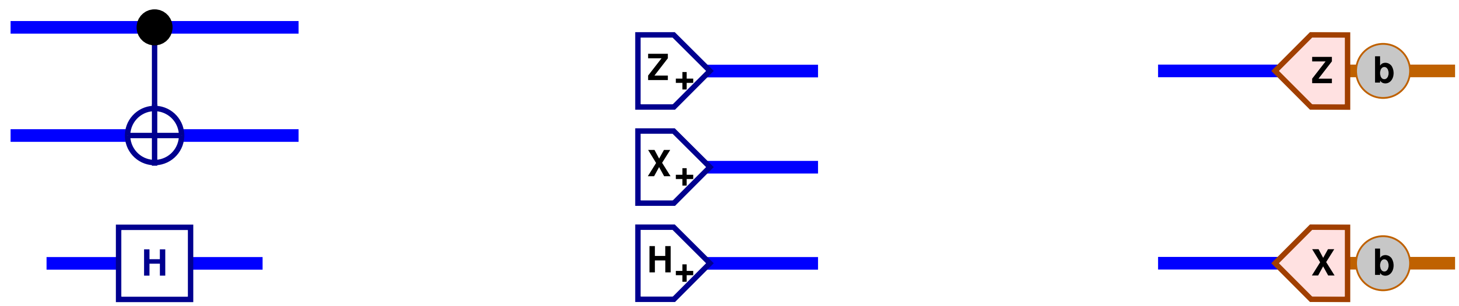
- Massive parallelism or no memory error.
- Negligible classical computation latency.

Structural Assumptions

Physical resources:

- Arbitrarily many “physical” qubits can be called on.

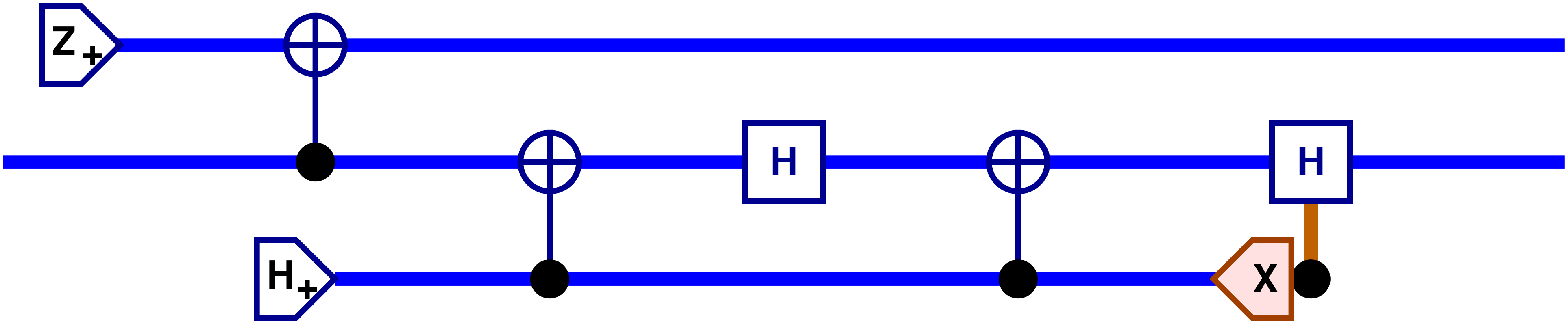
Local control capabilities:



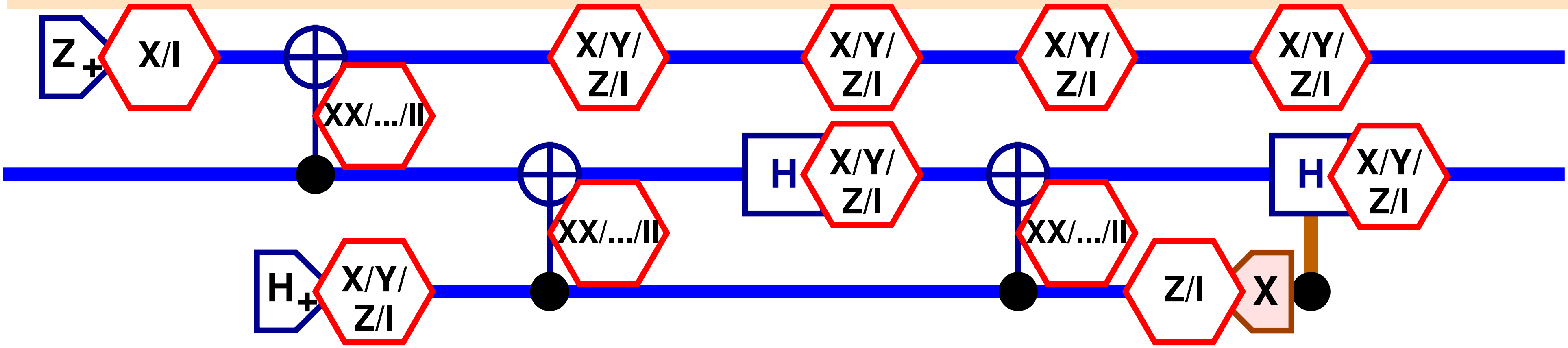
Global control capabilities:

- Massive parallelism or no memory error.
- Negligible classical computation latency.
- Negligible quantum communication latency
  (for non-local two-qubit gates).

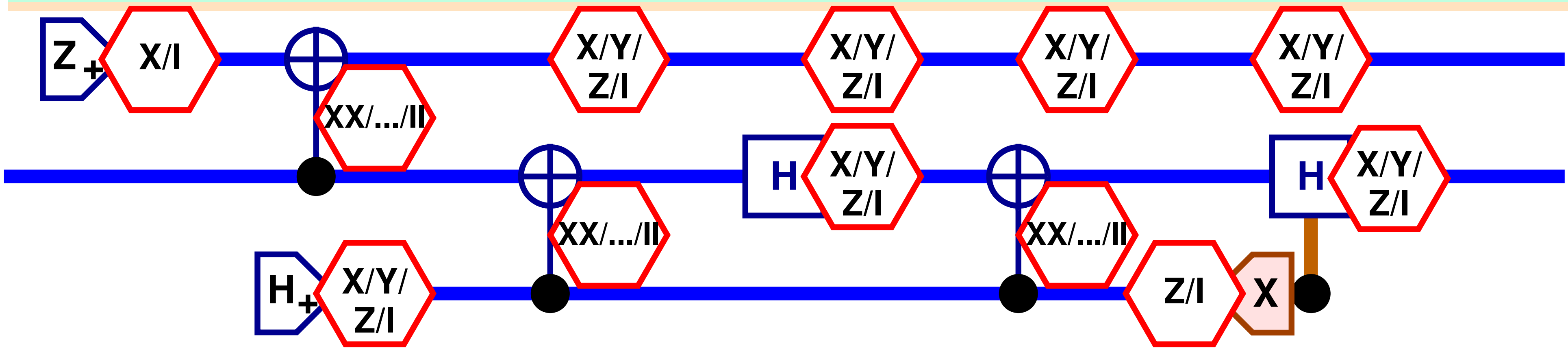
Error Models



Error Models



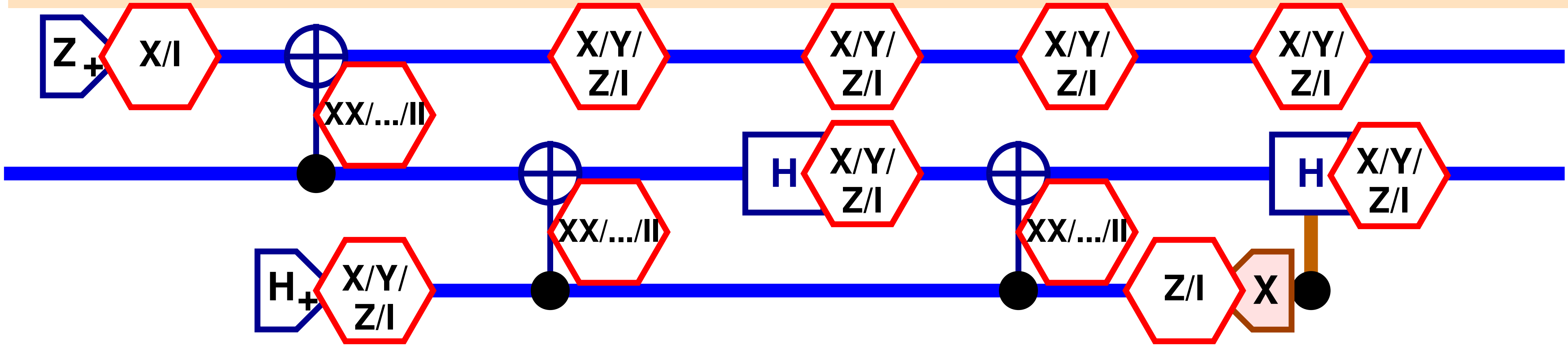
Error Models



- General error expansion:

$$\sum_{e} \sum_{\mathbf{p}=(p_i)_i} \underbrace{|e_{\mathbf{p}}\rangle_{\text{Env}}}_{\text{Unnormalized “environment” state}} \prod_{i} \overbrace{G_i}^{\text{Gate } i} \underbrace{E_i(p_i)}_{\text{Pauli product at location } i}$$

Error Models

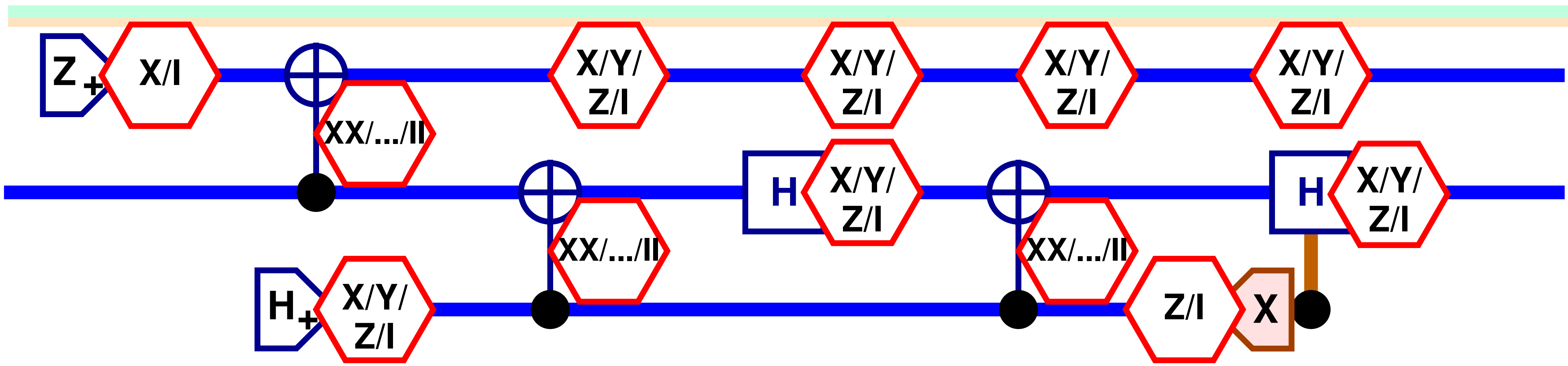


- General error expansion:

$$\sum_{e}\sum_{\mathbf{p}=(p_i)_i} \underbrace{|e_{\mathbf{p}}\rangle_{\text{Env}}}_{\text{Unnormalized “environment” state}} \prod_i \overbrace{G_i}^{\text{Gate } i} \underbrace{E_i(p_i)}_{\text{Pauli product at location } i}$$

- Independent, probabilistic Pauli errors:
  - The $|e_{\mathbf{p}}\rangle$ are orthogonal.

Error Models

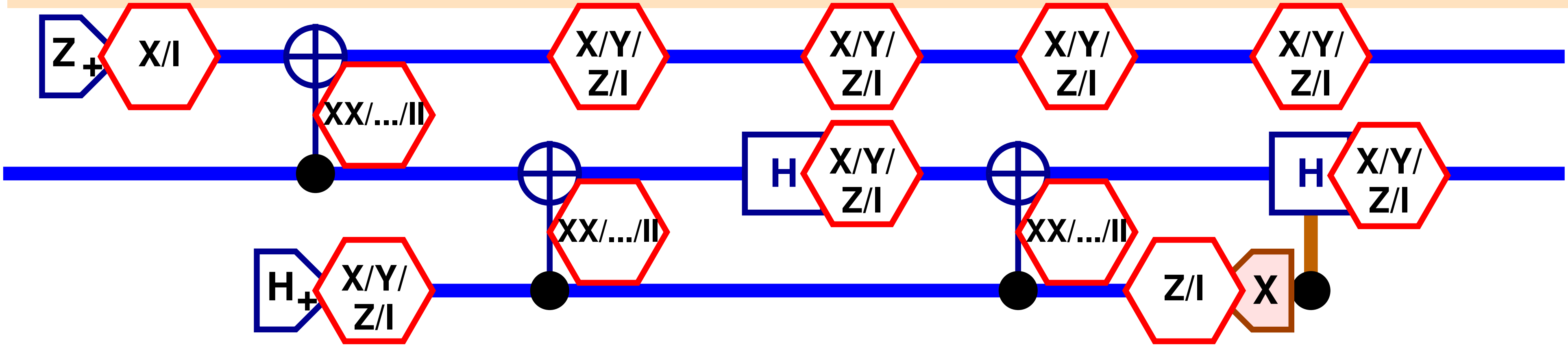


- General error expansion:

$$\sum_e \sum_{\mathbf{p}=(p_i)_i} \underbrace{|e_{\mathbf{p}}\rangle_{\text{Env}}}_{\text{Unnormalized “environment” state}} \prod_i \overbrace{G_i}^{\text{Gate } i} \underbrace{\color{red}{E}_i(p_i)}_{\text{Pauli product at location } i}$$

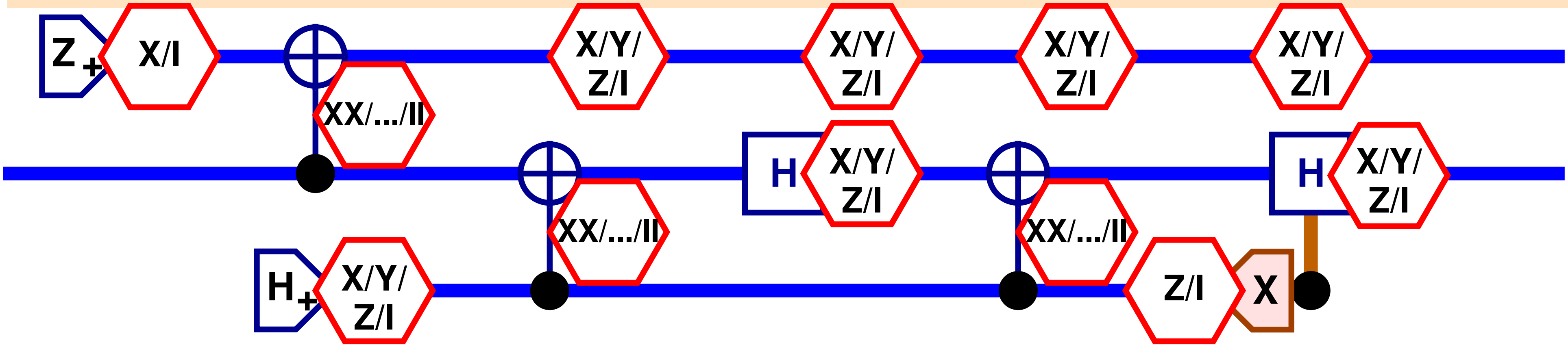
- Independent, probabilistic Pauli errors:
  - The $|e_{\mathbf{p}}\rangle$ are orthogonal.
  - $\||e_{\mathbf{p}}\rangle|^2 = \prod_i e_i(p_i)$, where e_i depends only on the gate type.

Error Models



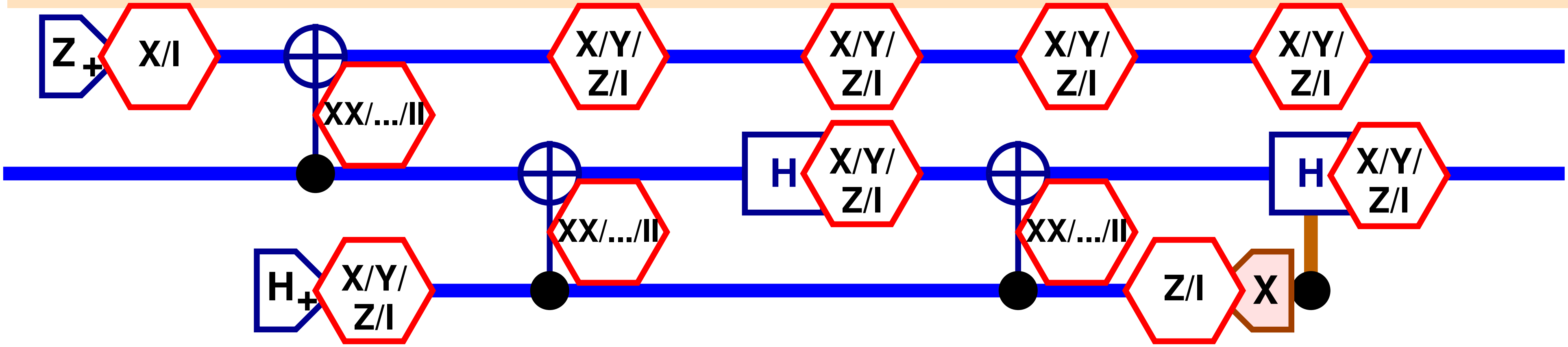
- Independent, probabilistic Pauli errors:
  - The $|e_{\mathbf{p}}\rangle$ are orthogonal.
  - $||e_{\mathbf{p}}\rangle|^2 = \prod_i e_i(p_i)$, where e_i depends only on the gate type.

Error Models



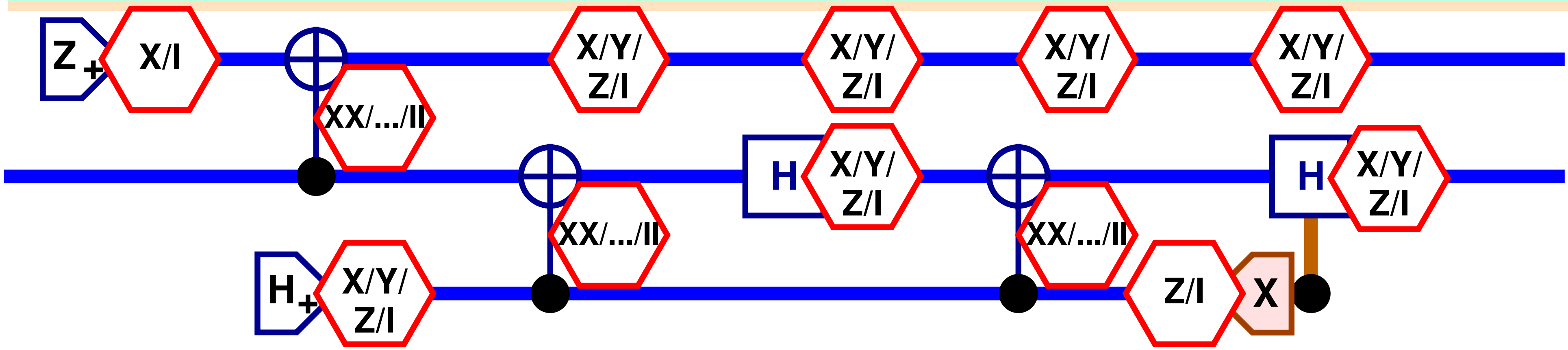
- Independent, probabilistic Pauli errors:
  - The $|e_{\mathbf{p}}\rangle$ are orthogonal.
  - $||e_{\mathbf{p}}\rangle|^2 = \prod_i e_i(p_i)$, where e_i depends only on the gate type.

Error Models



- Independent, probabilistic Pauli errors:
  - The $|e_{\mathbf{p}}\rangle$ are orthogonal.
  - $\||e_{\mathbf{p}}\rangle|^2 = \prod_i e_i(p_i)$, where e_i depends only on the gate type.
- Justification?
  - Stabilizer code implementations imply short lifetimes of unwanted coherences between Pauli errors.

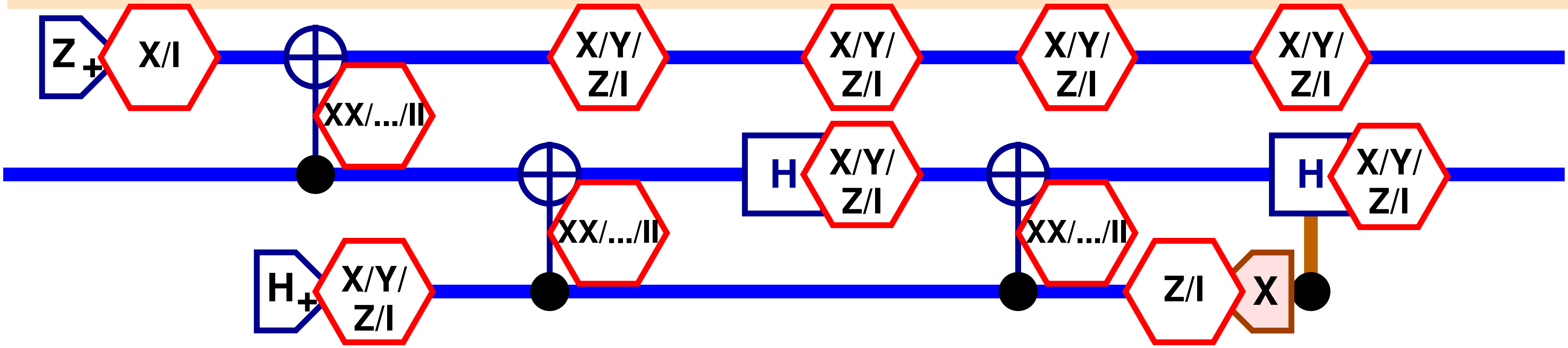
Error Models



- Independent, probabilistic Pauli errors:
  - The $|e_{\mathbf{p}}\rangle$ are orthogonal.
  - $\||e_{\mathbf{p}}\rangle|^2 = \prod_i e_i(p_i)$, where e_i depends only on the gate type.

- Justification?
  - Stabilizer code implementations imply short lifetimes of unwanted coherences between Pauli errors.
  - Random Pauli pulses can further reduce these lifetimes.

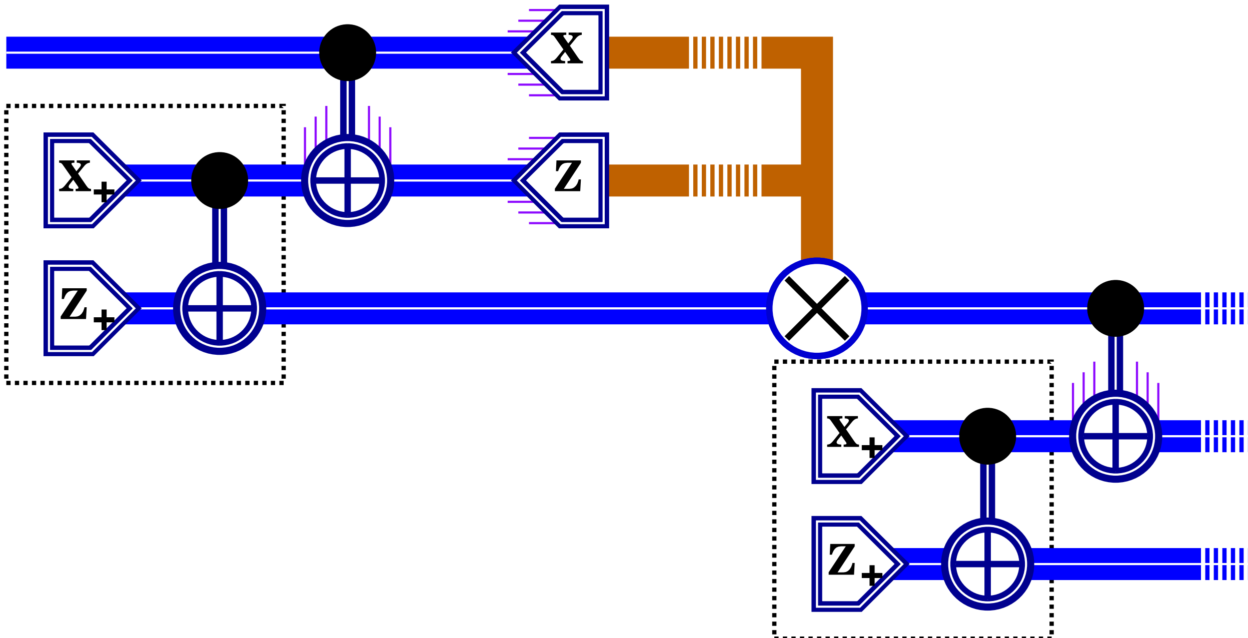
Error Models



- Independent, probabilistic Pauli errors:
  - The $|e_{\mathbf{p}}\rangle$ are orthogonal.
  - $||e_{\mathbf{p}}\rangle|^2 = \prod_i e_i(p_i)$, where e_i depends only on the gate type.
- Justification?
  - Stabilizer code implementations imply short lifetimes of unwanted coherences between Pauli errors.
  - Random Pauli pulses can further reduce these lifetimes.
  - Correlations are usually local.

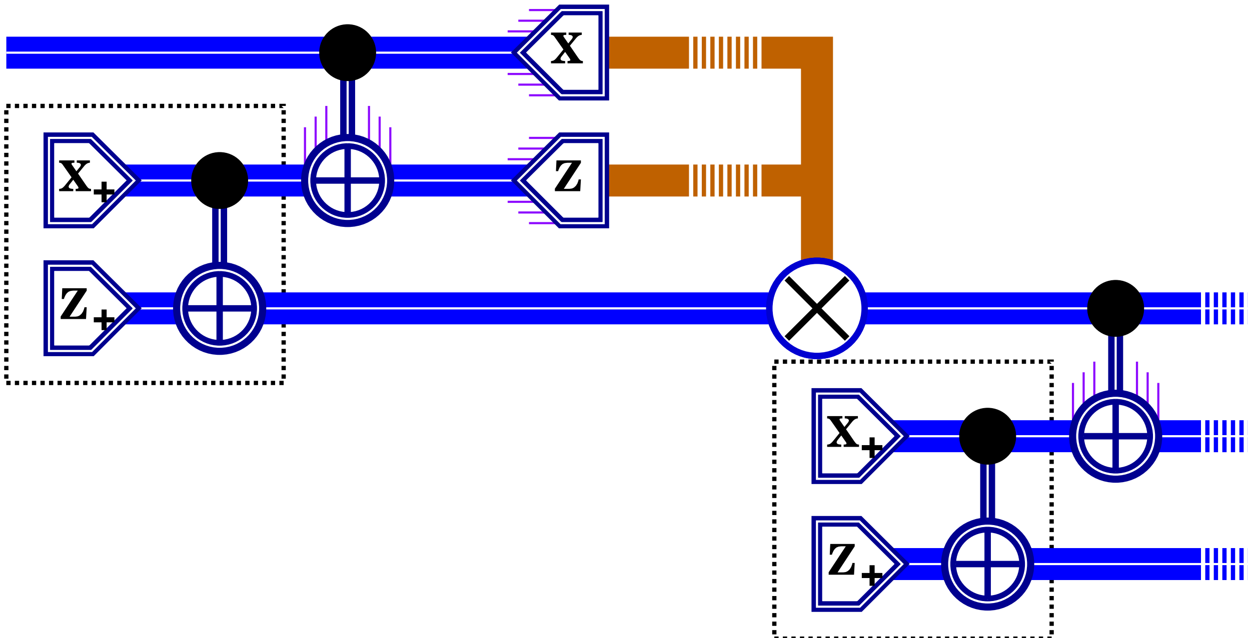
Memory, Measurement, Parallelism Tradeoffs

- Problem: Long measurements req. good quantum memory.



Memory, Measurement, Parallelism Tradeoffs

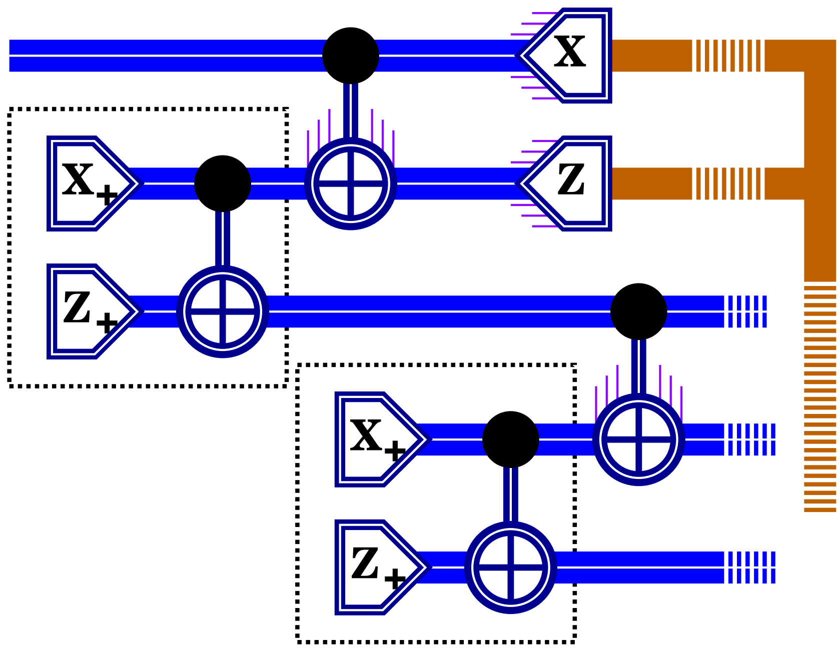
- Problem: Long measurements req. good quantum memory.



- Delay feedforward. Cost: More qubits, parallelism.

Memory, Measurement, Parallelism Tradeoffs

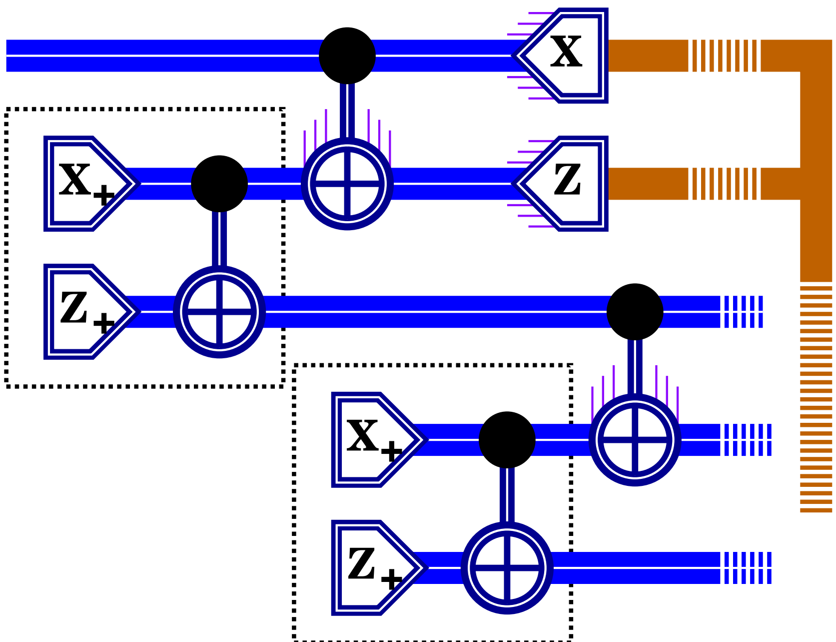
- Problem: Long measurements req. good quantum memory.



- Delay feedforward. Cost: More qubits, parallelism.

Memory, Measurement, Parallelism Tradeoffs

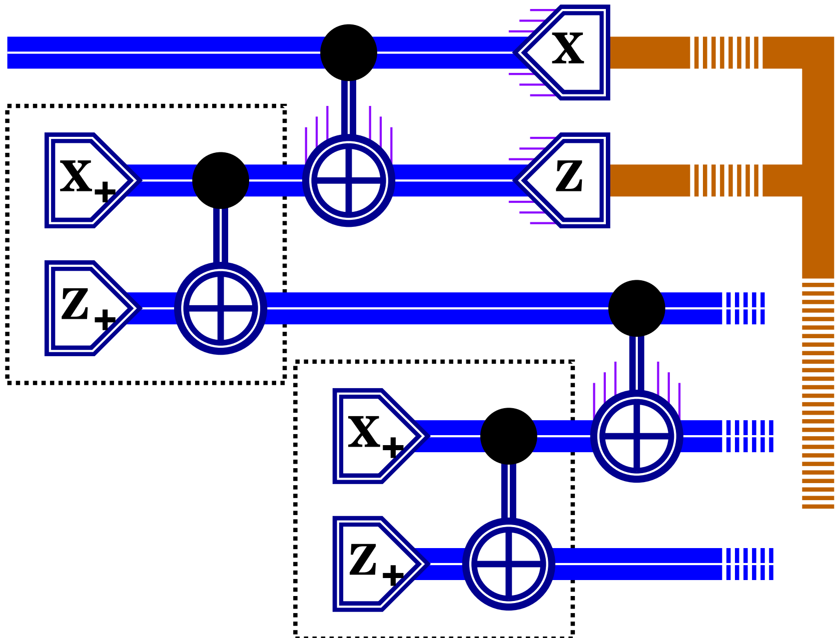
- Problem: Long measurements req. good quantum memory.



- Delay feedforward. Cost: More qubits, parallelism.
- But: Have to wait for goodness check after state preparation.

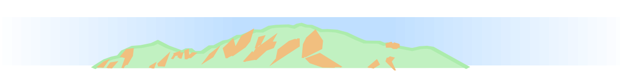
Memory, Measurement, Parallelism Tradeoffs

- Problem: Long measurements req. good quantum memory.



- Delay feedforward. Cost: More qubits, parallelism.
- But: Have to wait for goodness check after state preparation.
- Requirement:

  (memory error rate)$*$(measurement delay) $=$ small.



Contents

References

[1] D.P. DiVincenzo. The physical implementation of quantum computation. *Fort. Phys.*, 48:771–783, 2000.

[2] P. W. Shor. Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A*, 52:2493–2496, 1995.

[3] P. W. Shor. Fault-tolerant quantum computation. In *Proceedings of the 37th Symposium on the Foundations of Computer Science (FOCS)*, pages 56–65, Los Alamitos, California, 1996. IEEE press.

[4] A. Yu. Kitaev. Quantum computations: Algorithms and error correction. *Russian Math. Surveys*, 52:1191–1249, 1997.

[5] D. Aharonov and M. Ben-Or. Fault-tolerant quantum computation with constant error. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computation (STOC)*, pages 176–188, New York, New York, 1996. ACM Press.

[6] E. Knill, R. Laflamme, and W. H. Zurek. Resilient quantum computation. *Science*, 279:342–345, 1998.

[7] A. M. Steane. Overhead and noise threshold of fault-tolerant quantum error correction. *Phys. Rev. A*, 68:042322/1–19, 2003.

[8] E. Knill. Fault-tolerant postselected quantum computation: Threshold analysis. quant-ph/0404104, 2004.

[9] E. Knill. Quantum computing with realistically noisy devices. *Nature*, 434:39–44, 2005.

[10] B. W. Reichardt. Improved ancilla preparation scheme increases fault tolerant threshold. quant-ph/0406025, 2004.

[11] J. P. Clemens, S. Siddiqui, and J. Gea-Banacloche. Quantum error-correction against correlated noise. preprint, 2004.

[12] B. M. Terhal and G. Burkard. Fault-tolerant quantum computation for local non-markovian noise. quant-ph/04020104, 2004.

[13] R. Alicki. Comments on ”fault-tolerant computation for local non-markovian noise. quant-ph/0402139, 2004.

[14] E. Knill. Scalable quantum computation in the presence of large detected-error rates. *Phys. Rev. A*, 71:042322/1–7, 2005.

[15] S. Bravyi and A. Kitaev. Universal quantum computation based on a magic states distillation. quant-ph/0403025, 2004.

[16] E. Knill. Fault-tolerant postselected quantum computation: Schemes. quant-ph/0402171, 2004.

[17] D. Gottesman. *Stabilizer Codes and Quantum Error Correction*. PhD thesis, Calif. Inst. Tech, Pasadena, California, 1997.