

10. Accurate computation out of noisy components (NC 10.5-10.6)

Holy grail:

- (c) The threshold theorem
(good enough implies arbitrarily good)

Elaborating what is good enough:

- (b) Principles of fault-tolerant quantum computation (don't make a mess)

How to achieve (b)?

- (d)-(f) Fault-tolerant logical Pauli & Clifford gates
- (g) Fault-tolerant $\pi/8$ gate, 1-bit teleportation
- (h) Fault-tolerant measurements
- (i) Overhead and assumptions

From last lecture:

Lemma: If a unitary U on 2-dim satisfies the conditions

$$U X U^\dagger = Z, \quad U Z U^\dagger = X$$

then, $U = H$ (the Hadamard gate).

From last lecture:

Lemma: If a unitary U on 2-dim satisfies the conditions

$$U X U^\dagger = Z, \quad U Z U^\dagger = X$$

then, $U = H$ (the Hadamard gate).

If a unitary U on 4-dim satisfies the conditions

$$U X I U^\dagger = X X, \quad U Z I U^\dagger = Z I$$

$$U I X U^\dagger = I X, \quad U I Z U^\dagger = Z Z$$

then, $U = \text{CNOT}$.

Both results are up to an overall phase of U .

Encoded operations can be characterized similarly.

Lemma: fix a stabilizer code C .

If U is a logical operation, $UX_LU^\dagger = Z_L$, $UZ_LU^\dagger = X_L$,
then U acts as H_L on C .

NB U is a logical operation if it conjugates each generator into a product of generators.

Lemma: fix a stabilizer code C .

If U is a logical operation, $UX_LU^\dagger = Z_L$, $UZ_LU^\dagger = X_L$,
then U acts as H_L on C .

Similarly, if U is a logical operation,

$$\begin{aligned} UX_LI_LU^\dagger &= X_LX_L, & UZ_LI_LU^\dagger &= Z_LI_L \\ UI_LX_LU^\dagger &= I_LX_L, & UI_LZ_LU^\dagger &= Z_LZ_L \end{aligned}$$

then U acts as $CNOT_L$ on C .

NB U is a logical operation if it conjugates each generator into a product of generators.

For the 5-qubit code:

$$G1 = XZZXI$$

$$G2 = IXZZX$$

$$G3 = XIXZZ$$

$$G4 = ZXIXZ$$

we have XXXXX, ZZZZZ as logical X, Z.

For the 5-qubit code:

$$G1 = XZZXI$$

$$G2 = IXZZX$$

$$G3 = XIXZZ$$

$$G4 = ZXIXZ$$

we have XXXXX, ZZZZZ as logical X, Z.

The natural choice for H_L is HHHHH.

since $H^{\otimes 5} X_L H^{\otimes 5} = Z_L$, $H^{\otimes 5} Z_L H^{\otimes 5} = X_L$

For the 5-qubit code:

$$G1 = XZZXI$$

$$G2 = IXZZX$$

$$G3 = XIXZZ$$

$$G4 = ZXIXZ$$

we have XXXXX, ZZZZZ as logical X, Z.

The natural choice for H_L is HHHHH.

since $H^{\otimes 5} X_L H^{\otimes 5} = Z_L$, $H^{\otimes 5} Z_L H^{\otimes 5} = X_L$

Qn: is HHHHH an encoded operation?

In A5 Q1b, you will show that it is NOT ...

7-qubit Steane code:
a QECC with many transversal encoded operations

The 7-qubit code has stabilizer generators:

$$G_1 = | \ | \ | \ X \ X \ X \ X$$

$$G_2 = | \ X \ X \ | \ | \ X \ X$$

$$G_3 = X \ | \ X \ | \ X \ | \ X$$

$$G_4 = | \ | \ | \ Z \ Z \ Z \ Z$$

$$G_5 = | \ Z \ Z \ | \ | \ Z \ Z$$

$$G_6 = Z \ | \ Z \ | \ Z \ | \ Z$$

7-qubit Steane code:
 a QECC with many transversal encoded operations

The 7-qubit code has stabilizer generators:

$$\begin{array}{ll}
 G_1 = | & | & | & X & X & X & X & \\
 G_2 = | & X & X & | & | & X & X & \\
 G_3 = X & | & X & | & X & | & X & \\
 G_4 = | & | & | & Z & Z & Z & Z & \\
 G_5 = | & Z & Z & | & | & Z & Z & \\
 G_6 = Z & | & Z & | & Z & | & Z &
 \end{array}$$

Properties of the generators:

1. G_1, G_2, G_3 (called the X-generators) come from the parity check matrix of the classical binary [7,1,3]

Hamming code:

$$\begin{array}{ccccccc}
 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
 1 & 0 & 1 & 0 & 1 & 0 & 1
 \end{array}$$

7-qubit Steane code:

a QECC with many transversal encoded operations

The 7-qubit code has stabilizer generators:

$$G_1 = | \quad | \quad | \quad X \quad X \quad X \quad X$$

$$G_2 = | \quad X \quad X \quad | \quad | \quad X \quad X$$

$$G_3 = X \quad | \quad X \quad | \quad X \quad | \quad X$$

$$G_4 = | \quad | \quad | \quad Z \quad Z \quad Z \quad Z$$

$$G_5 = | \quad Z \quad Z \quad | \quad | \quad Z \quad Z$$

$$G_6 = Z \quad | \quad Z \quad | \quad Z \quad | \quad Z$$

Properties of the generators:

1. G_1, G_2, G_3 (called the X-generators) come from the parity check matrix of the classical binary [7,1,3]

Hamming code: $\begin{array}{ccccccc} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{array}$ 2^2 The 3-bits in each column
 2^1 is the binary representation
 2^0 for the column number !

$\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \downarrow$

1 2 3 4 5 6 7

Very useful for decoding ...

2. The classical code is "self-dual" : any two rows of the parity check matrix have inner product 2.

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

2. The classical code is "self-dual" : any two rows of the parity check matrix have inner product 2.

Thus the generators are commuting (note the Z-generators G4, G5, G6 are obtained from the X-generators G1, G2, G3 by turning X's into Z's, any X-generator and Z-generator anticommute in 2 or 4 tensor components).

$$\begin{array}{ll}
 G_1 = | | | X X X X & \xrightarrow{4} G_4 = | | | Z Z Z Z \\
 G_2 = | X X | | X X & \xrightarrow{2} G_5 = | Z Z | | Z Z \\
 G_3 = X | X | X | X & G_6 = Z | Z | Z | Z
 \end{array}$$

2. The classical code is "self-dual" : any two rows of the parity check matrix have inner product 2.

Thus the generators are commuting (note the Z-generators G_4, G_5, G_6 are obtained from the X-generators G_1, G_2, G_3 by turning X's into Z's, any X-generator and Z-generator anticommute in 2 or 4 tensor components).

3. Each generator has weight 4 (code is doubly even).

2. The classical code is "self-dual" : any two rows of the parity check matrix have inner product 2.

Thus the generators are commuting (note the Z-generators G_4, G_5, G_6 are obtained from the X-generators G_1, G_2, G_3 by turning X's into Z's, and any X-generator and Z-generator anticommute in 2 tensor components).

3. Each generator has weight 4 (code is doubly even).

4. The code is an example of a "CSS code" (for Calderbank-Shor-Steane) -- the generators can be chosen to have only X's or Z's but not both.

Properties of the code:

1. Corrects any 0- or 1-qubit Pauli error (thus an arbitrary single-qubit error).

Properties of the code:

1. Corrects any 0- or 1-qubit Pauli error (thus an arbitrary single-qubit error).

Proof: Measure the 6 generators to get a 6-bit syndrome, convert the outcome + to 0, - to 1, the first 3 syndrome bits is the binary rep of which qubit has a Z error, the last 3 bits is the binary rep of which qubit has an X error.

Properties of the code:

1. Corrects any 0- or 1-qubit Pauli error (thus an arbitrary single-qubit error).

Proof: Measure the 6 generators to get a 6-bit syndrome, convert the outcome + to 0, - to 1, the first 3 syndrome bits is the binary rep of which qubit has a Z error, the last 3 bits is the binary rep of which qubit has an X error.

e.g., If error = Y6,
what is the syndrome?

$$G_1 = | | | X X \color{red}{X} X$$

$$G_2 = | X X | | \color{red}{X} X$$

$$G_3 = X | X | X | \color{red}{X}$$

$$G_4 = | | | Z Z \color{red}{Z} Z$$

$$G_5 = | Z Z | | \color{red}{Z} Z$$

$$G_6 = Z | Z | Z | \color{red}{Z}$$

(a) --+ +++

(b) +-+ -+ -+

(c) --+ --+

Properties of the code:

1. Corrects any 0- or 1-qubit Pauli error (thus an arbitrary single-qubit error).

Proof: Measure the 6 generators to get a 6-bit syndrome, convert the outcome + to 0, - to 1, the first 3 syndrome bits is the binary rep of which qubit has a Z error, the last 3 bits is the binary rep of which qubit has an X error.

e.g., If error = Y_6 ,
the syndrome (meas
G1-G6) is --+ --+.
Converting to 110
110, the binary reps
for 6 and 6: the error
is $Z_6 * X_6 = Y_6$.

$$G_1 = | | | X X \color{red}{X} X$$

$$G_2 = | X X | | \color{red}{X} X$$

$$G_3 = X | X | X | \color{red}{X}$$

$$G_4 = | | | Z Z \color{red}{Z} Z$$

$$G_5 = | Z Z | | \color{red}{Z} Z$$

$$G_6 = Z | Z | Z | \color{red}{Z}$$

if the syndrome (meas
G1-G6) is +++ -++ ,
what is the error?

$$G_1 = | | | X X X X$$

$$G_2 = | X X | | X X$$

$$G_3 = X | X | X | X$$

$$G_4 = | | | Z Z Z Z$$

$$G_5 = | Z Z | | Z Z$$

$$G_6 = Z | Z | Z | Z$$

(a) Z1, (b) X1, (c) Z4, (d) X4, (e) Y4?

if the syndrome (meas
G1-G6) is +++ -++ ,
what is the error?

$$G_1 = | | | X X X X$$

$$G_2 = | X X | | X X$$

$$G_3 = X | X | X | X$$

$$G_4 = | | | Z Z Z Z$$

$$G_5 = | Z Z | | Z Z$$

$$G_6 = Z | Z | Z | Z$$

(a) Z1, (b) X1, (c) Z4, (d) X4, (e) Y4?

Method 1:

+1 eigenvalues with all X generators so, cannot be
a Y or Z error.

from the last 3 bits of the syndrome -++ , only the
4th qubit has a Z on G4 and I on G5 G6.

if the syndrome (meas
G1-G6) is +++ -++ ,
what is the error?

$$G_1 = | | | X X X X$$

$$G_2 = | X X | | X X$$

$$G_3 = X | X | X | X$$

$$G_4 = | | | Z Z Z Z$$

$$G_5 = | Z Z | | Z Z$$

$$G_6 = Z | Z | Z | Z$$

(a) Z1, (b) X1, (c) Z4, (d) X4, (e) Y4?

Method 1:

+1 eigenvalues with all X generators so, cannot be a Y or Z error.

from the last 3 bits of the syndrome -++ , only the 4th qubit has a Z on G4 and I on G5 G6.

Method 2:

Converting +++-++ to 000 100, the Z error is in the zero position (no Z error) and the X error is in the 4 position (binary 100). Error = I * X4 = X4.

$$\begin{array}{ll}
 G_1 = | | | X X X X & G_4 = | | | \textcolor{red}{Z} Z Z Z \\
 G_2 = | X X | | X X & G_5 = | Z Z \textcolor{red}{I} | Z Z \\
 G_3 = X | X | X | X & G_6 = Z | Z \textcolor{red}{I} Z | Z
 \end{array}$$

Following the decoding recipe, each of the 22 ($1+7*3$) 0- or 1-qubit Pauli errors have a unique syndrome, so, they can be identified and corrected. By discretization, an arbitrary error on 1 qubit can be corrected.

2. Encoded operations

$$G_1 = | \ | \ | \ X \ X \ X \ X$$

$$G_2 = | \ X \ X \ | \ | \ X \ X$$

$$G_3 = X \ | \ X \ | \ X \ | \ X$$

$$G_4 = | \ | \ | \ Z \ Z \ Z \ Z$$

$$G_5 = | \ Z \ Z \ | \ | \ Z \ Z$$

$$G_6 = Z \ | \ Z \ | \ Z \ | \ Z$$

(a) By inspection, $XXXXXXX$, $ZZZZZZZZ$ commute with G_1, \dots, G_6 , and they anticommute with one another, so, they can be chosen as logical X and Z .

2. Encoded operations

$$G_1 = | \ | \ | \ X \ X \ X \ X$$

$$G_2 = | \ X \ X \ | \ | \ X \ X$$

$$G_3 = X \ | \ X \ | \ X \ | \ X$$

$$G_4 = | \ | \ | \ Z \ Z \ Z \ Z$$

$$G_5 = | \ Z \ Z \ | \ | \ Z \ Z$$

$$G_6 = Z \ | \ Z \ | \ Z \ | \ Z$$

(a) By inspection, XXXXXX, ZZZZZZ commute with G_1, \dots, G_6 , and they anticommute with one another, so, they can be chosen as logical X and Z.

NB. Logical X, Z should be logical operations, and anticommute with one another. A5Q2 just a little more complicated. If we have 2 encoded qubits, what are the commutation or anticommutation relations for logical X, Z on qubits 1, 2 ?

2. Encoded operations

$$G_1 = | \ | \ | \ X \ X \ X \ X$$

$$G_2 = | \ X \ X \ | \ | \ X \ X$$

$$G_3 = X \ | \ X \ | \ X \ | \ X$$

$$G_4 = | \ | \ | \ Z \ Z \ Z \ Z$$

$$G_5 = | \ Z \ Z \ | \ | \ Z \ Z$$

$$G_6 = Z \ | \ Z \ | \ Z \ | \ Z$$

(a) By inspection, XXXXXX, ZZZZZZ commute with G_1, \dots, G_6 , and they anticommute with one another, so, they can be chosen as logical X and Z.

(b) Consider HHHHHH as a candidate for logical H.

(i) checking how it conjugates the logical X and Z:

$$H^{\otimes 7} X_L H^{\otimes 7} = H^{\otimes 7} X^{\otimes 7} H^{\otimes 7} = Z^{\otimes 7} = Z_L$$

2. Encoded operations

$$G_1 = | \ | \ | \ X \ X \ X \ X$$

$$G_2 = | \ X \ X \ | \ | \ X \ X$$

$$G_3 = X \ | \ X \ | \ X \ | \ X$$

$$G_4 = | \ | \ | \ Z \ Z \ Z \ Z$$

$$G_5 = | \ Z \ Z \ | \ | \ Z \ Z$$

$$G_6 = Z \ | \ Z \ | \ Z \ | \ Z$$

(a) By inspection, XXXXXX, ZZZZZZ commute with G_1, \dots, G_6 , and they anticommute with one another, so, they can be chosen as logical X and Z.

(b) Consider HHHHHH as a candidate for logical H.

(i) checking how it conjugates the logical X and Z:

$$H^{\otimes 7} X_L H^{\otimes 7} = H^{\otimes 7} X^{\otimes 7} H^{\otimes 7} = Z^{\otimes 7} = Z_L$$

$$H^{\otimes 7} Z_L H^{\otimes 7} = H^{\otimes 7} Z^{\otimes 7} H^{\otimes 7} = X^{\otimes 7} = X_L$$

correct commutation relations on logical X, Z.

(ii) checking if HHHHHHHH is a logical operation:

$$\text{e.g., } H^{\otimes 7} III XXXX H^{\otimes 7} = III zzzz$$

So, IIIXXXX is conjugated to another generator.

(ii) checking if HHHHHHH is a logical operation:

e.g., $H^{\otimes 7} III XXXX H^{\otimes 7} = III zzzz$

So, $II XXXX$ is conjugated to another generator.

Similarly: $H^{\otimes 7} G_i H^{\otimes 7} = G_{i+3}$ for $i = 1, 2, 3$

$H^{\otimes 7} G_i H^{\otimes 7} = G_{i-3}$ for $i = 4, 5, 6$

$G_1 = III XXXX \longleftrightarrow G_4 = III zzzz$

$G_2 = IXX IIXX \longleftrightarrow G_5 = Izz IIZz$

$G_3 = XIX IXIX \longleftrightarrow G_6 = zIz IZIZ$

(ii) checking if HHHHHHH is a logical operation:

e.g., $H^{\otimes 7} IIIXXXX H^{\otimes 7} = IIIZZZZ$

So, $IIXXXX$ is conjugated to another generator.

Similarly: $H^{\otimes 7} G_i H^{\otimes 7} = G_{i+3}$ for $i = 1, 2, 3$
 $H^{\otimes 7} G_i H^{\otimes 7} = G_{i-3}$ for $i = 4, 5, 6$

$$\begin{array}{lcl} G_1 = IIIXXX & \longleftrightarrow & G_4 = IIIZZZ \\ G_2 = IXXIIX & \longleftrightarrow & G_5 = IZZIIZ \\ G_3 = XIXIXI & \longleftrightarrow & G_6 = ZIZIZI \end{array}$$

(The "bit-wise" Hadamard exchanges each X-generator with a corresponding Z-generator.)

(ii) checking if HHHHHHH is a logical operation:

e.g., $H^{\otimes 7} III XXXX H^{\otimes 7} = III ZZZZ$

So, IIIXXXX is conjugated to another generator.

Similarly: $H^{\otimes 7} G_i H^{\otimes 7} = G_{i+3}$ for $i=1, 2, 3$
 $H^{\otimes 7} G_i H^{\otimes 7} = G_{i-3}$ for $i=4, 5, 6$

So, each generator is conjugated to a product of generators, so HHHHHHH is a logical operation.

Altogether, it acts as the logical Hadamard gate.

(c) for logical CNOT, consider 2 code blocks, 14 qubits encoding 2 qubits. The generators are:

$$G_i I^{\otimes 7}, \quad i = 1, 2, \dots, 6$$

$$I^{\otimes 7} G_i, \quad i = 1, 2, \dots, 6$$

(c) for logical CNOT, consider 2 code blocks, 14 qubits encoding 2 qubits. The generators are:

$$G_i I^{\otimes 7}, \quad i = 1, 2, \dots, 6$$

$$I^{\otimes 7} G_i, \quad i = 1, 2, \dots, 6$$

Here, $X_L I_L = X^{\otimes 7} I^{\otimes 7}$, $Z_L I_L = Z^{\otimes 7} I^{\otimes 7}$ encoded X,
 $I_L X_L = I^{\otimes 7} X^{\otimes 7}$, $I_L Z_L = I^{\otimes 7} Z^{\otimes 7}$ Z for qubit 1
 encoded X,
 Z for qubit 2

(c) for logical CNOT, consider 2 code blocks, 14 qubits encoding 2 qubits. The generators are:

$$G_i I^{\otimes 7}, \quad i = 1, 2, \dots, 6$$

$$I^{\otimes 7} G_i, \quad i = 1, 2, \dots, 6$$

Here, $X_L I_L = X^{\otimes 7} I^{\otimes 7}$, $Z_L I_L = Z^{\otimes 7} I^{\otimes 7}$ encoded X,
 $I_L X_L = I^{\otimes 7} X^{\otimes 7}$, $I_L Z_L = I^{\otimes 7} Z^{\otimes 7}$ Z for qubit 1
 encoded X,
 Z for qubit 2

To show U is a logical CNOT, need to:

(a) check commutation relations

$$U X_L I_L U^\dagger = X_L X_L, \quad U Z_L I_L U^\dagger = Z_L I_L$$

$$U I_L X_L U^\dagger = I_L X_L, \quad U I_L Z_L U^\dagger = Z_L Z_L$$

(b) check U conjugates generators to products of generators

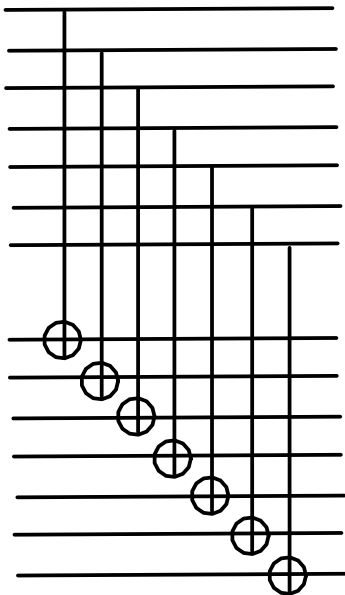
(c) for logical CNOT, consider 2 code blocks, 14 qubits encoding 2 qubits. The generators are:

$$G_i I^{\otimes 7}, \quad i = 1, 2, \dots, 6$$

$$I^{\otimes 7} G_i, \quad i = 1, 2, \dots, 6$$

Here, $X_L I_L = X^{\otimes 7} I^{\otimes 7}$, $Z_L I_L = Z^{\otimes 7} I^{\otimes 7}$ encoded X,
Z for qubit 1
 $I_L X_L = I^{\otimes 7} X^{\otimes 7}$, $I_L Z_L = I^{\otimes 7} Z^{\otimes 7}$ encoded X,
Z for qubit 2

Candidate logical CNOT (U):



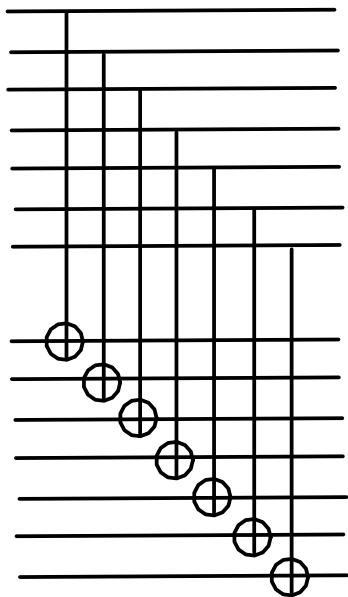
(c) for logical CNOT, consider 2 code blocks, 14 qubits encoding 2 qubits. The generators are:

$$G_i I^{\otimes 7}, \quad i = 1, 2, \dots, 6$$

$$I^{\otimes 7} G_i, \quad i = 1, 2, \dots, 6$$

Here, $X_L I_L = X^{\otimes 7} I^{\otimes 7}$, $Z_L I_L = Z^{\otimes 7} I^{\otimes 7}$ encoded X, Z for qubit 1
 $I_L X_L = I^{\otimes 7} X^{\otimes 7}$, $I_L Z_L = I^{\otimes 7} Z^{\otimes 7}$ encoded X, Z for qubit 2

Candidate logical CNOT (U):



U conjugates X_1 to $X_1 * X_8$,

$$X_i \rightarrow X_i X_{i+7} \text{ for } i = 1, 2, \dots, 7$$

$$(X_i = I^{\overleftarrow{i-1}} X I^{\overleftarrow{14-i}})$$

Checking commutation relations with the logical Paulis:

$$\begin{aligned} \textcircled{1} \quad U(X_L \otimes I_L)U^\dagger &= U(X^{\otimes 7} I^{\otimes 7})U^\dagger \\ &= U X_1 X_2 \dots X_7 U^\dagger \end{aligned}$$

$$(X_i = \overset{\leftarrow 7-1 \rightarrow}{II \dots XI} \overset{\leftarrow 14-2 \rightarrow}{\dots I})$$

Checking commutation relations with the logical Paulis:

$$\begin{aligned} \textcircled{1} \quad U(X_L \otimes I_L)U^\dagger &= U(X^{\otimes 7} I^{\otimes 7})U^\dagger \\ &= U X_1 X_2 \dots X_7 U^\dagger \\ &= U X_1 U^\dagger U X_2 U^\dagger \dots U X_7 U^\dagger \end{aligned}$$

$\begin{matrix} \leftarrow 1-1 \rightarrow & \leftarrow 14-2 \rightarrow \\ (X_i = I I \dots X I \dots I) \end{matrix}$

Checking commutation relations with the logical Paulis:

$$\begin{aligned}
 \textcircled{1} \quad U(X_L \otimes I_L)U^\dagger &= U(X^{\otimes 7} I^{\otimes 7})U^\dagger \\
 &= U X_1 X_2 \cdots X_7 U^\dagger \\
 &= U X_1 U^\dagger U X_2 U^\dagger \cdots U X_7 U^\dagger \\
 &= X_1 X_8 \cdot X_2 X_9 \cdots X_7 X_{14}
 \end{aligned}$$

$\begin{matrix} \leftarrow 7-1 \rightarrow & \leftarrow 14-2 \rightarrow \\ (X_i = I I \cdots X I \cdots I) \end{matrix}$

Checking commutation relations with the logical Paulis:

$$\begin{aligned}
 \textcircled{1} \quad U(X_L \otimes I_L)U^\dagger &= U(X^{\otimes 7} I^{\otimes 7})U^\dagger \\
 &= U X_1 X_2 \cdots X_7 U^\dagger \\
 &= U X_1 U^\dagger U X_2 U^\dagger \cdots U X_7 U^\dagger \\
 &= X_1 X_8 \cdot X_2 X_9 \cdots X_7 X_{14} \\
 &= (X_1 X_2 \cdots X_7) (X_8 X_9 \cdots X_{14}) \\
 &= X^{\otimes 7} \otimes X^{\otimes 7} \\
 &= X_L \otimes X_L
 \end{aligned}$$

$\begin{matrix} \leftarrow 1-1 \rightarrow & \leftarrow 14-1 \rightarrow \\ (X_i = I I \cdots X I \cdots I) \end{matrix}$

Checking commutation relations with the logical Paulis:

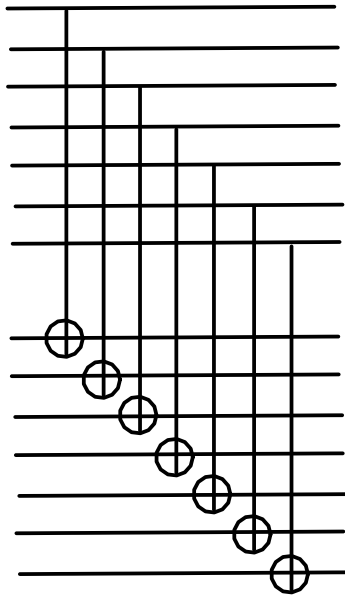
$$\begin{aligned}
 \textcircled{1} \quad U(X_L \otimes I_L)U^\dagger &= U(X^{\otimes 7} I^{\otimes 7})U^\dagger \\
 &= U X_1 X_2 \cdots X_7 U^\dagger \\
 &= U X_1 U^\dagger U X_2 U^\dagger \cdots U X_7 U^\dagger \\
 &= X_1 X_8 \cdot X_2 X_9 \cdots X_7 X_{14} \\
 &= (X_1 X_2 \cdots X_7) (X_8 X_9 \cdots X_{14}) \\
 &= X^{\otimes 7} \otimes X^{\otimes 7} \\
 &= X_L \otimes X_L
 \end{aligned}$$

$\begin{matrix} \leftarrow 1-1 \rightarrow & \leftarrow 14-1 \rightarrow \\ (X_i = I I \cdots X I \cdots I) \end{matrix}$

$\therefore U$ conjugates $X_L I_L$ to $X_L X_L$.

Checking commutation relations with the logical Paulis:

②

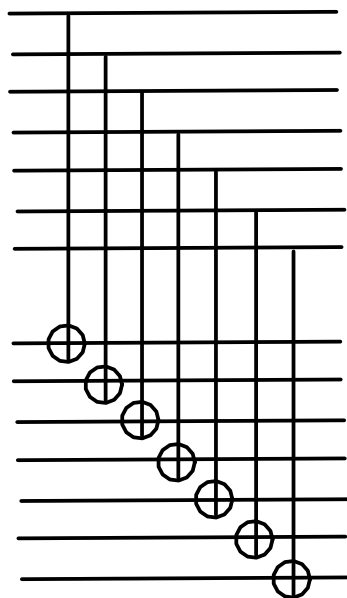


U conjugates X_8 to X_8 ,

X_i to X_i for $i = 8, 9, \dots, 14$

Checking commutation relations with the logical Paulis:

②



U conjugates X_8 to X_8 ,

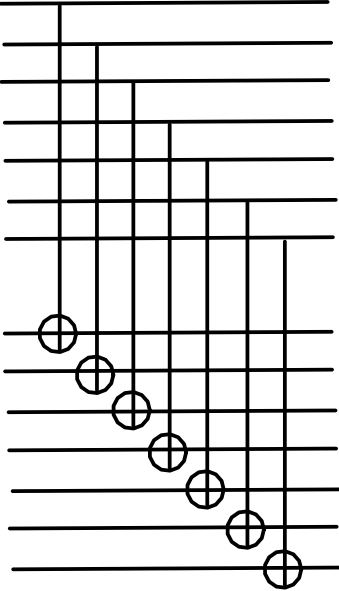
$$X_i \rightarrow X_i \text{ for } i = 8, 9, \dots, 14$$

$$U(I_L \otimes X_L) U^\dagger$$

$$= U(I^{\otimes 7} X^{\otimes 7}) U^\dagger$$

$$= U(X_8 X_9 X_{10} X_{11} X_{12} X_{13} X_{14}) U^\dagger$$

Checking commutation relations with the logical Paulis:

②  U conjugates X_8 to X_8 ,

$X_i \rightarrow X_i$ for $i = 8, 9, \dots, 14$

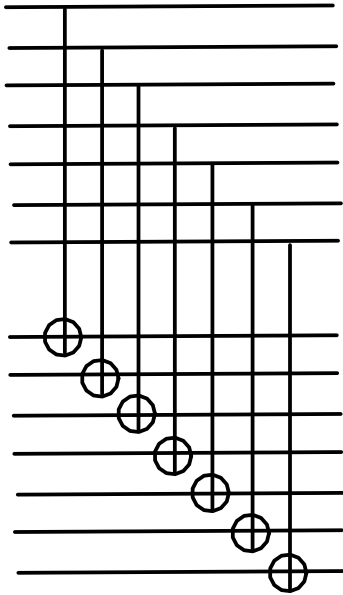
$$\begin{aligned}
 & U(I_L \otimes X_L) U^\dagger \\
 &= U(I^{\otimes 7} X^{\otimes 7}) U^\dagger \\
 &= U(X_8 X_9 X_{10} X_{11} X_{12} X_{13} X_{14}) U^\dagger \\
 &= X_8 X_9 X_{10} X_{11} X_{12} X_{13} X_{14} \\
 &= I_L \otimes X_L
 \end{aligned}$$

\therefore U conjugates $I_L X_L$ to $I_L X_L$.

Checking commutation relations with the logical Paulis:

③

④



Exercise: show that U

conjugates Z_1 to Z_1 ,

$$Z_i \rightarrow Z_i \text{ for } i=1, 2, \dots, 7$$

$$\therefore \dots \dots Z_L I_L \rightarrow Z_L I_L.$$

conjugates Z_8 to $Z_1 * Z_8$,

$$Z_i \rightarrow Z_{i-7} \cdot Z_i \text{ for } i=8, 9, \dots, 14$$

$$\therefore \dots \dots I_L Z_L \rightarrow Z_L Z_L$$

So, the transversal CNOT acts correctly on the encoded X's and Z's.

Next: show that the transversal CNOT is an encoded operation.

Is the transversal CNOT an encoded operation?

e.g., it conjugates $\overset{G_1}{| | |} \overset{\otimes I_L}{XXXX} \overset{\otimes I^{\otimes 7}}{\otimes} I^{\otimes 7}$ to $\overset{G_1}{| | |} \overset{\otimes G_1}{XXXX} \overset{\otimes G_1}{| | |} XXXX$

Is the transversal CNOT an encoded operation?

e.g., it conjugates $G_1 \otimes I_L$ to $G_1 \otimes G_1$

$$\begin{aligned}
 & \text{e.g., it conjugates } |111\rangle\langle 111| \otimes |XXXX\rangle\langle XXXX| \text{ to } |111\rangle\langle 111| \otimes |XXXX\rangle\langle XXXX| \\
 & = (|111\rangle\langle 111| \otimes |XXXX\rangle\langle XXXX|) \cdot (I^{\otimes 7} \otimes |111\rangle\langle 111|) \\
 & \quad G_1 \otimes I_L \quad \cdot \quad I_L \otimes G_1 \\
 & \text{products of generators}
 \end{aligned}$$

Is the transversal CNOT an encoded operation?

e.g., it conjugates $\overset{G_1}{|111\rangle} \otimes \overset{I_L}{|XXXX\rangle} \overset{I^{\otimes 7}}{\rightarrow} \overset{G_1}{|111\rangle} \otimes \overset{G_1}{|XXXX\rangle}$

$$= \left(\overset{G_1}{|111\rangle} \otimes \overset{I_L}{|XXXX\rangle} \right) \cdot \left(\overset{I_L}{|XXXX\rangle} \otimes \overset{G_1}{|111\rangle} \right)$$

products of generators

Ex: prove that each of the 12 generators are conjugated to a product of generators.

So, the transversal CNOT acts as an encoded CNOT
(hard to show without the stabilizer formalism.)

10. Accurate computation out of noisy components (NC 10.5-10.6)

What we are aiming to achieve:

(b) Principles of fault-tolerant quantum computation (don't make a mess)

Benefit of achieving the above

(c) The threshold theorem
(good enough implies arbitrarily good)

How to achieve (b)?

(d)-(f) Fault-tolerant logical Pauli & Clifford gates

(g) Fault-tolerant $\pi/8$ gate, 1-bit teleportation

(h) Fault-tolerant measurements

(i) Overhead and assumptions

Definition: Pauli group on n qubits P_n

Consider $\mathbb{C}^{2^{\otimes n}}$. Let X_t, Z_t be the X, Z Pauli operator acting on the t-th qubit (I on the rest).

The Pauli group is defined to be the group generated multiplicatively by X_t, Z_t , for $t=1,2,\dots,n$, and scalar i .

Let $Y = iXZ = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$.

e.g., $n=2$. Generators: XI, IX, ZI, IZ

Group elements generated multiplicatively:

$$\left\{ \begin{array}{ll} II, & XI, IX, XX, & ZI, IZ, ZZ, \\ YI, XZ, YZ, & ZX, IY, ZY, & YX, XY, YY \end{array} \right\} \times \{1, -1, i, -i\}$$

NB. Focus on the quotient group without the scalar.

What about the logical T gate?

$\{H, \text{CNOT}, R\}$ (where $R=T^2$) multiplicatively generates the "Clifford group", defined as the group of matrices each conjugates the Pauli group (with scalars) to itself.

What about the logical T gate?

$\{H, \text{CNOT}, R\}$ (where $R=T^2$) multiplicatively generates the "Clifford group", defined as the group of matrices each conjugates the Pauli group (with scalars) to itself.

On the 7-qubit code, the encoded Clifford gates are transversal!

Need to show this for H, CNOT, and R.
(See A5Q2 for the encoded R gate.)

What about the logical T gate?

$\{H, \text{CNOT}, R\}$ (where $R=T^2$) multiplicatively generates the "Clifford group", defined as the group of matrices each conjugates the Pauli group (with scalars) to itself.

On the 7-qubit code, the encoded Clifford gates are transversal! (See A5Q2 for the R gate.)

The Clifford group is finite and not universal -- need the T gate, but the 7-qubit code has no transversal encoded T gate.

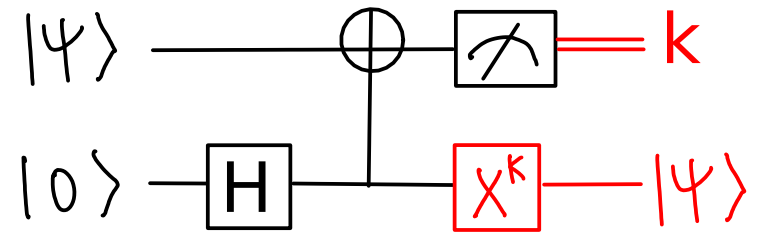
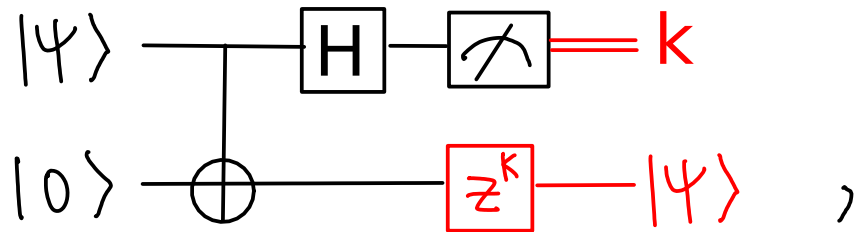
Idea: "teleport" the encoded T gate onto the unknown logical input ! (NC 10.6.2, KLM 5.3)

We start with 1-bit teleportation of quantum states, then use circuit identities to derive a circuit for the T gate (unencoded).

This general idea is then applied to encoded states.

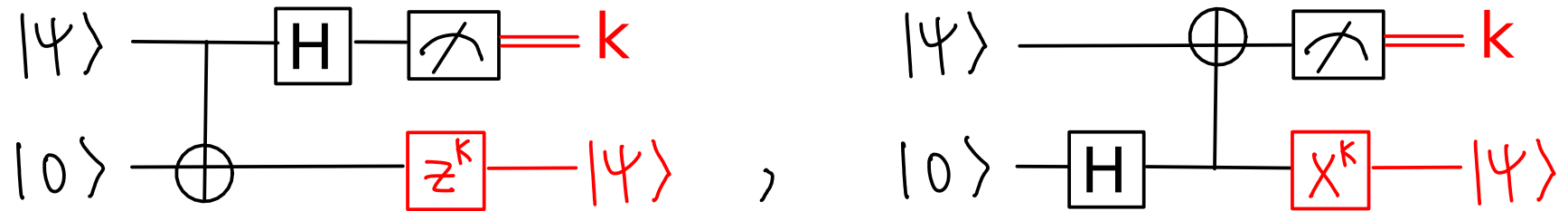
1-bit teleportation:

The following 2 circuits are correct. (Proof: quiz + ex.)

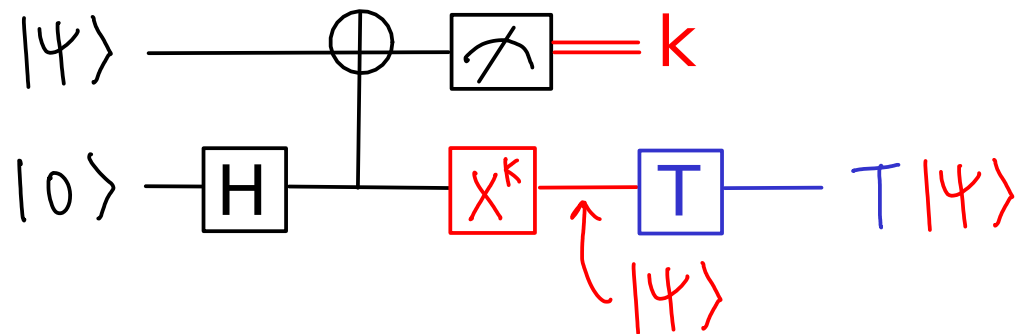


1-bit teleportation:

The following 2 circuits are correct. (Proof: quiz + ex.)

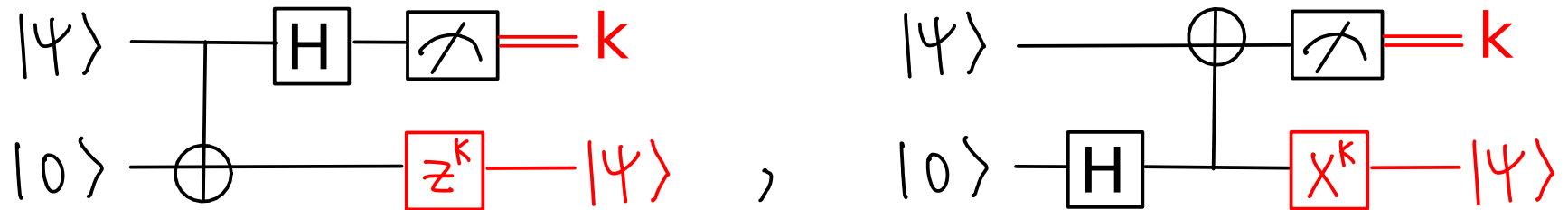


From the 2nd circuit we have:

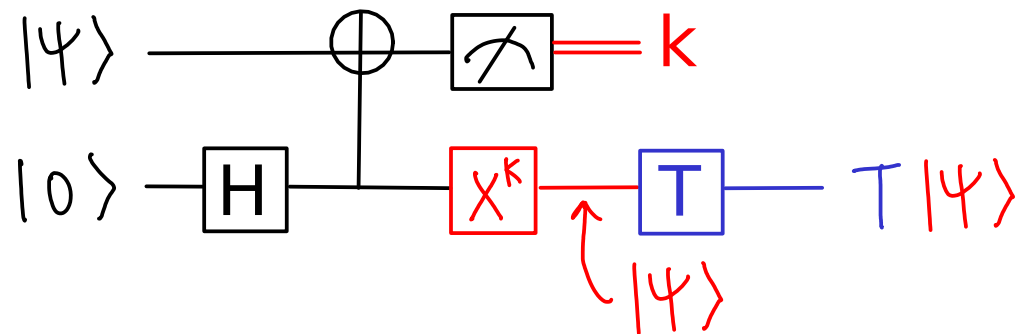


1-bit teleportation:

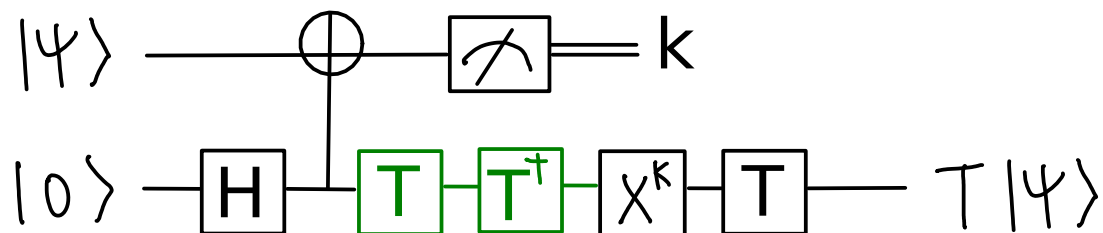
The following 2 circuits are correct. (Proof: quiz + ex.)



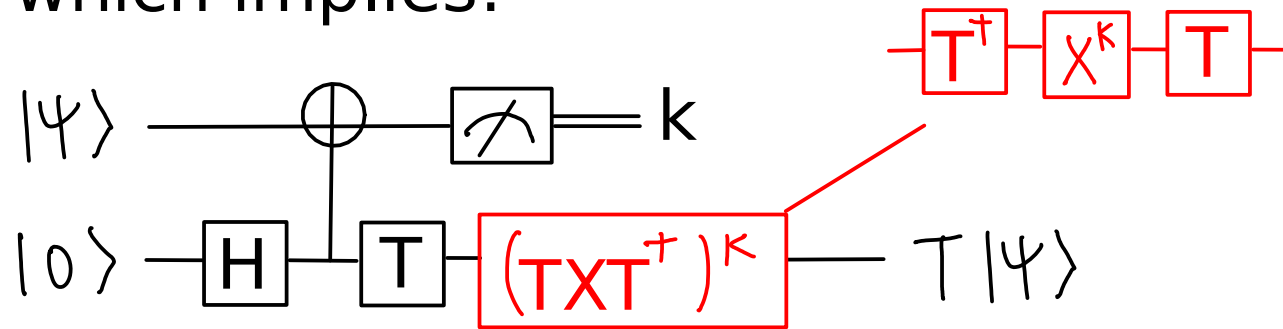
From the 2nd circuit we have:



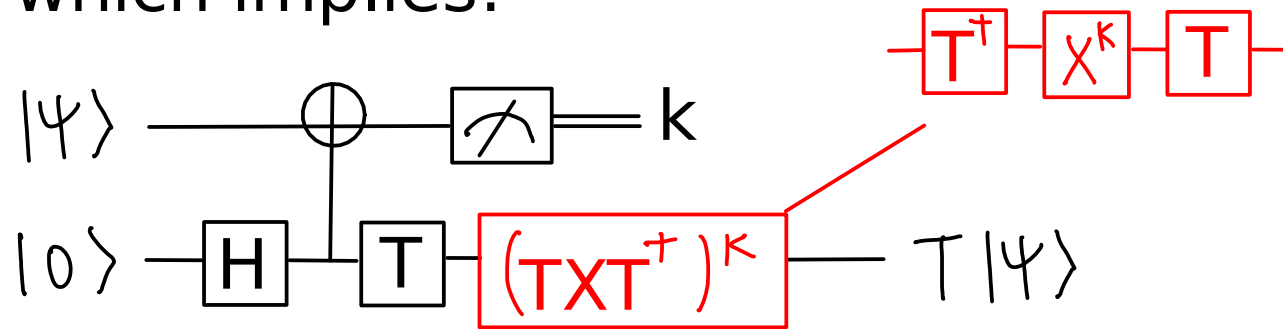
which implies:



which implies:



which implies:

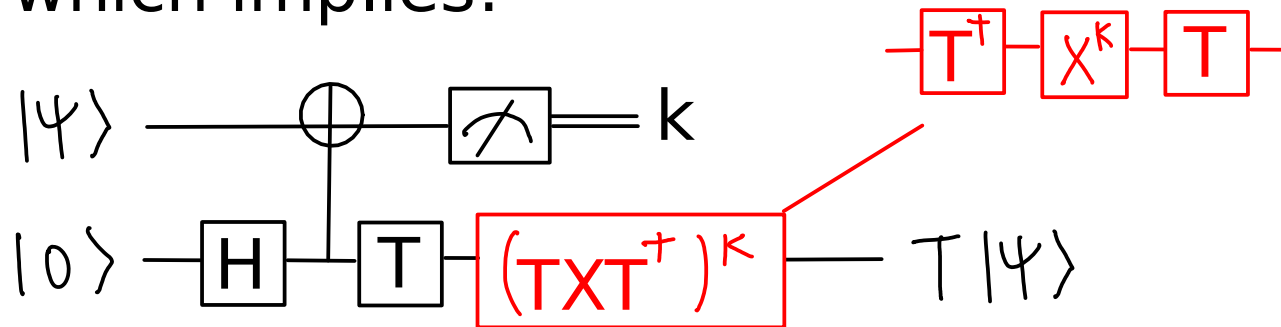


Also TXT^\dagger

$$= TXT^\dagger XX$$

$$= T \quad T \quad X$$

which implies:



Also TXT^\dagger

$$= TXT^\dagger XX$$

$$= T \quad T \quad X$$

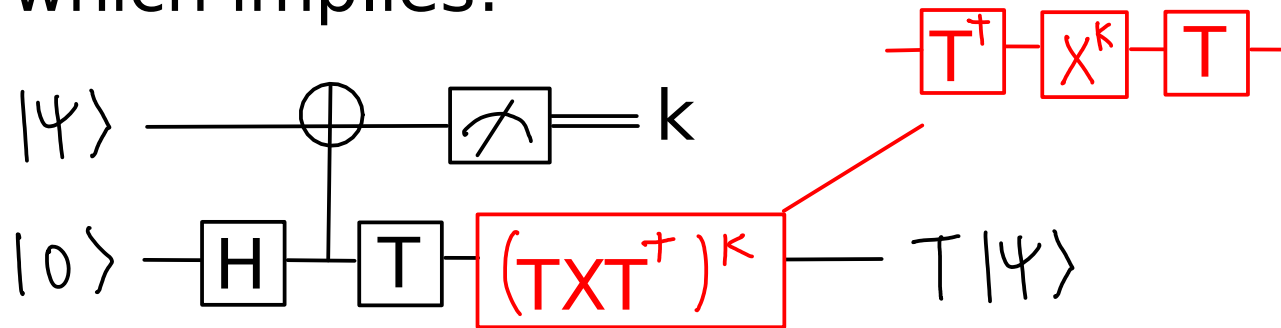
$$T = e^{-i\frac{\pi}{8}z}$$

$$XT^\dagger X = X e^{i\frac{\pi}{8}z} X^\dagger$$

$$= e^{i\frac{\pi}{8}XzX^\dagger} = e^{-i\frac{\pi}{8}z} = T$$

(LA test)

which implies:



Also TXT^\dagger

$$= TXT^\dagger XX$$

$$= T T X$$

$$= RX \quad (R=T^2)$$

up to a phase

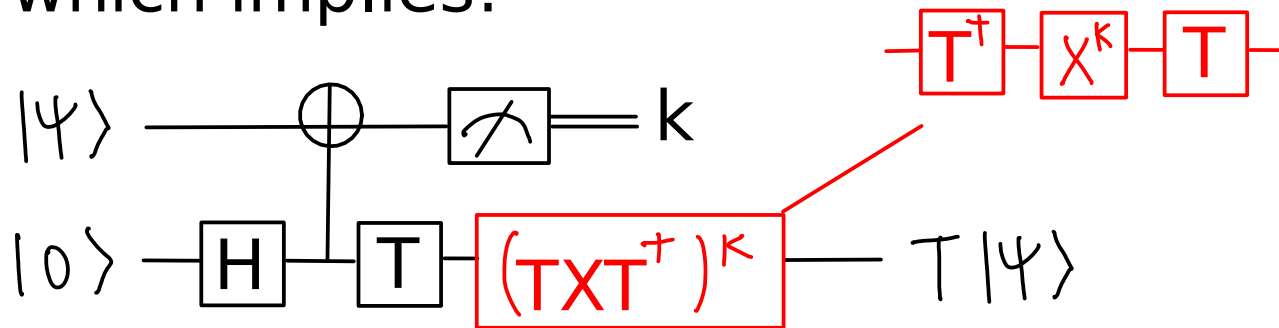
$$T = e^{-i\frac{\pi}{8}Z}$$

$$XT^\dagger X = X e^{i\frac{\pi}{8}Z} X$$

$$= e^{i\frac{\pi}{8}XZX} = e^{-i\frac{\pi}{8}Z} = T$$

(LA test)

which implies:



Also TXT^\dagger

$$= TXT^\dagger XX$$

$$= T T X$$

$$= RX \quad (R=T^2)$$

up to a phase

$$T = e^{-i\frac{\pi}{8}z}$$

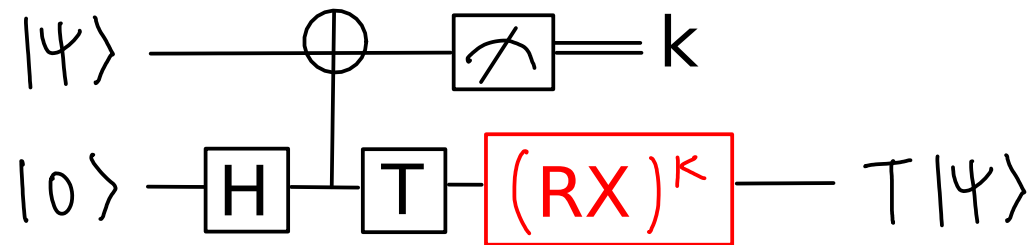
$$XT^\dagger X = X e^{i\frac{\pi}{8}z} X$$

$$= e^{i\frac{\pi}{8}xz} = e^{-i\frac{\pi}{8}z} = T$$

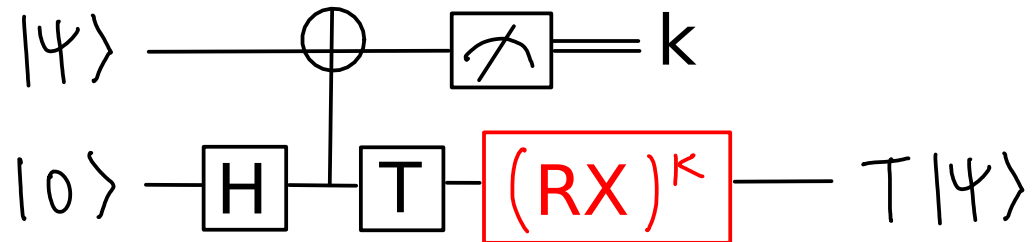
(LA test)

NB. There is no superposition between $k=0,1$ cases, so, overall phase in TXT^\dagger is overall phase in the output.

which implies:



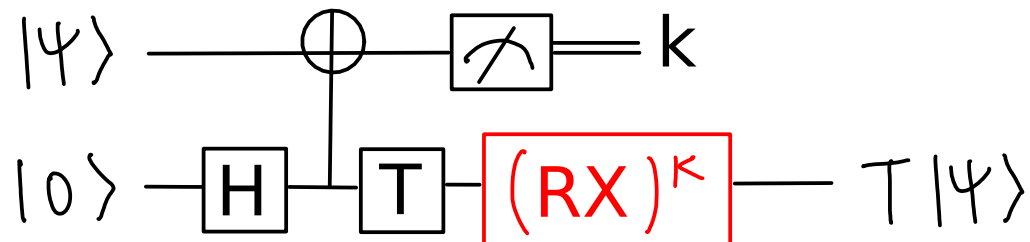
which implies:



$$\text{Also, CNOT} = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X$$

Control Target Control Target

which implies:

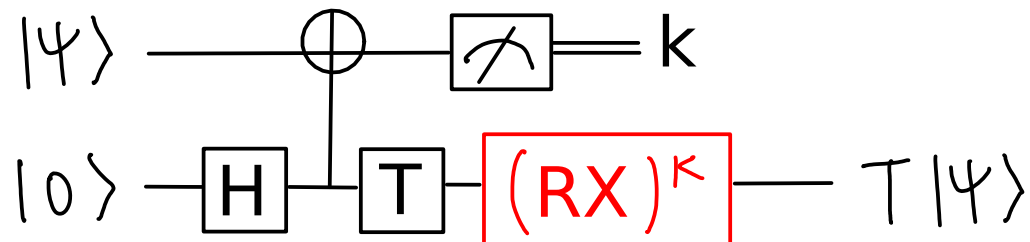


$$\begin{aligned} \text{Also, CNOT} &= |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X \\ &\quad \text{Control} \quad \text{Target} \quad \text{Control} \quad \text{Target} \\ &= \left(\frac{I + Z}{2} \right) \otimes I + \left(\frac{I - Z}{2} \right) \otimes X \end{aligned}$$

which commutes with any Z-rotation on control-qubit

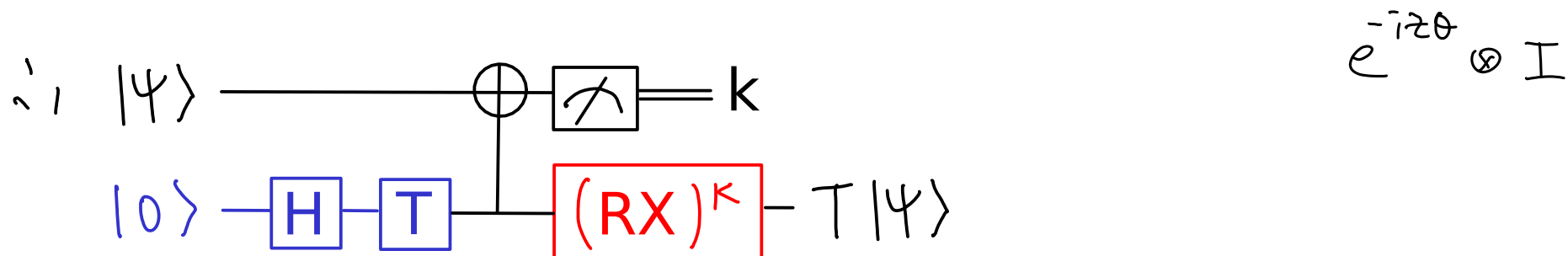
$$e^{-iZ\theta} \otimes I$$

which implies:

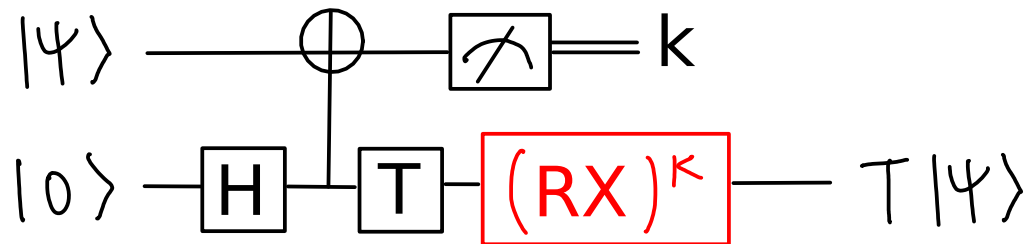


$$\begin{aligned} \text{Also, CNOT} &= |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X \\ &\quad \text{Control} \quad \text{Target} \quad \text{Control} \quad \text{Target} \\ &= \left(\frac{I+\sigma_z}{2}\right) \otimes I + \left(\frac{I-\sigma_z}{2}\right) \otimes X \end{aligned}$$

which commutes with any Z-rotation on control-qubit

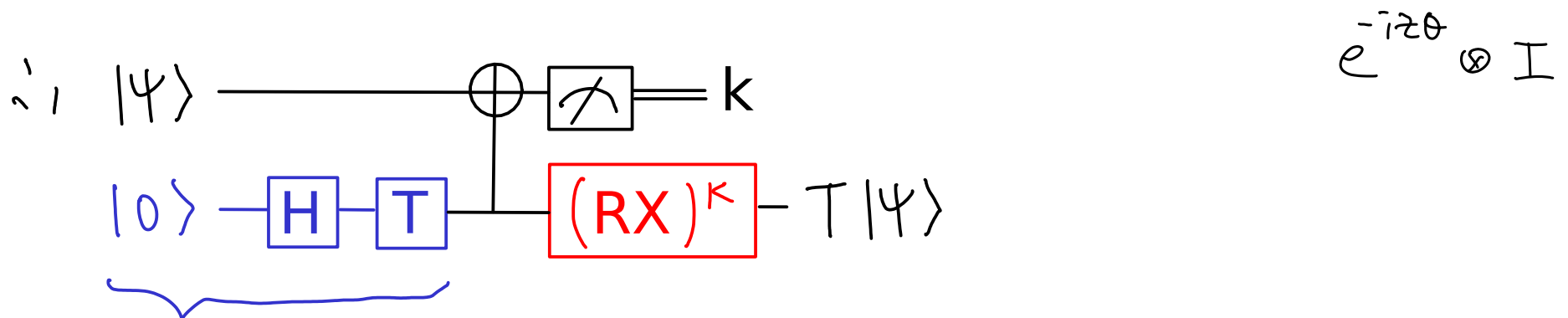


which implies:



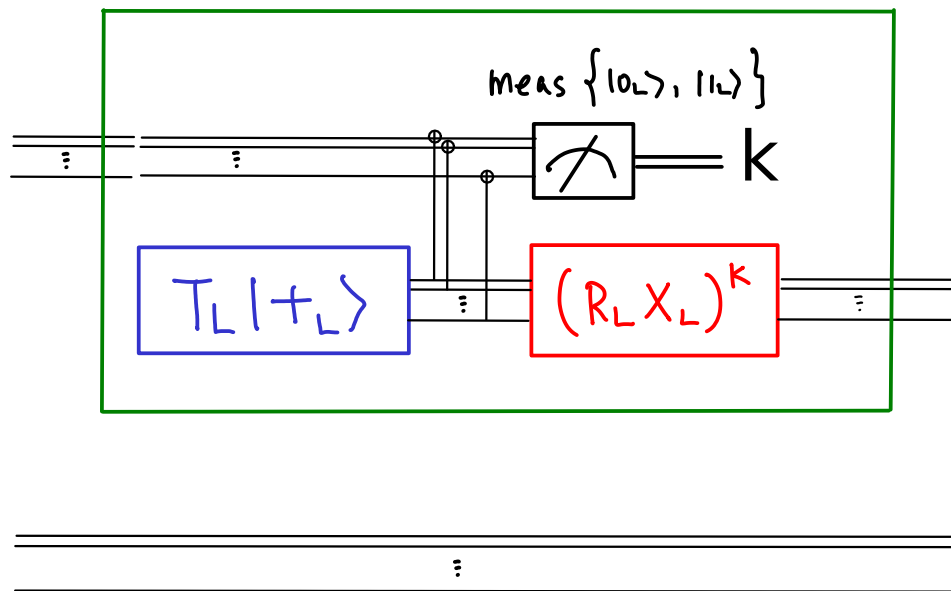
$$\begin{aligned} \text{Also, CNOT} &= |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X \\ &\quad \text{control target} \quad \text{control target} \\ &= \left(\frac{I+z}{2}\right) \otimes I + \left(\frac{I-z}{2}\right) \otimes X \end{aligned}$$

which commutes with any Z-rotation on control-qubit



fixed ancilla $TH|0\rangle = T|+\rangle$
(will see how to prepare ancilla later ...)

Encoded T gate on 7-qubit code:

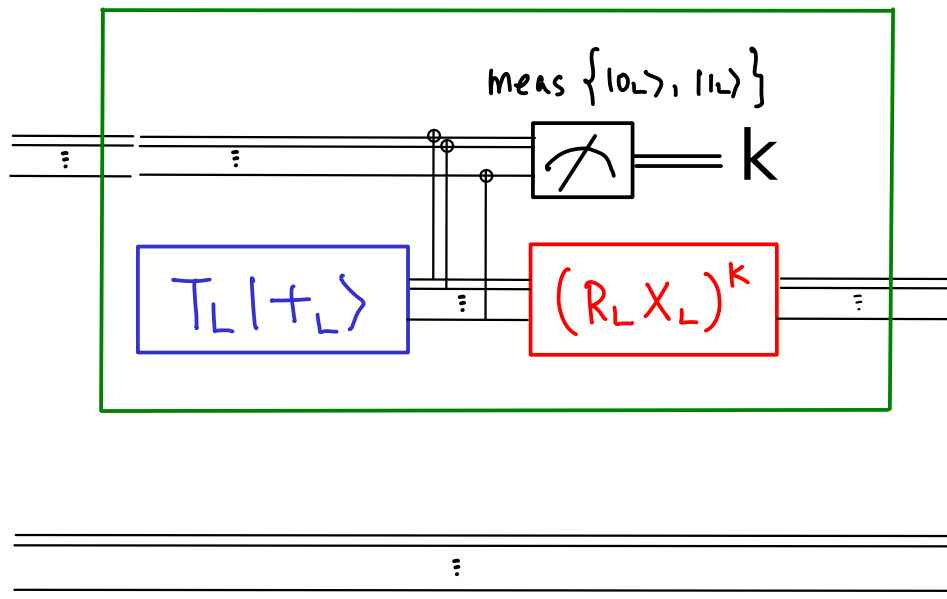


encoded T gate
on 1 code block C

R : system not
being acted on

$$\forall \text{ input } |\Psi_L\rangle_{CR}, \text{ output} = T_L \otimes I_L |\Psi_L\rangle_{CR}$$

Encoded T gate on 7-qubit code:



encoded T gate
on 1 code block C

R: system not
being acted on

\forall input $|\Psi_L\rangle_{CR}$, output = $T_L \otimes I_L |\Psi_L\rangle_{CR}$

Takes effort to prepare $T_L |+_L\rangle$.

Can be done offline, with verifications etc before being used in the precious fault-tolerant simulation circuit.

How to prepare $T_L |+\rangle$?

e.g., prepare many copies with non-transversal gate with small physical error rate, "distill" to suppress error.

How to prepare $T_L|+_L\rangle$?

e.g., prepare many copies with non-transversal gate with small physical error rate, "distill" to suppress error.

e.g., $T_L|+_L\rangle$ has stabilizers generators $G1, \dots, G6$, logical
and $TX T^\dagger = RX$.

It can be prepared by measuring $G1, \dots, G6, RX$.
Simplest: only keep state if outcome = ++++++.
(Or "add Pauli's" to map from -1 to +1 eigenspace.)

Will consider fault-tolerant measurements later ...

Up to an overall phase, one 7-qubit state stabilized with 7 generators. But need to recursively prepare for the level required for overall error ϵ .

The C^k hierarchy:

Let $C^1 = \bigcup_n P_n$ (Pauli group)

(Clifford group, generated by CNOT, H, R)

Let $C^2 = \bigcup_n \{U \in U(2^n) : U P_n U^\dagger \in P_n\} = \bigcup_n \{U \in U(2^n) : U P_n U^\dagger \in C^1\}$

Let $C^3 = \bigcup_n \{U \in U(2^n) : U P_n U^\dagger \in C_n\} = \bigcup_n \{U \in U(2^n) : U P_n U^\dagger \in C^2\}$

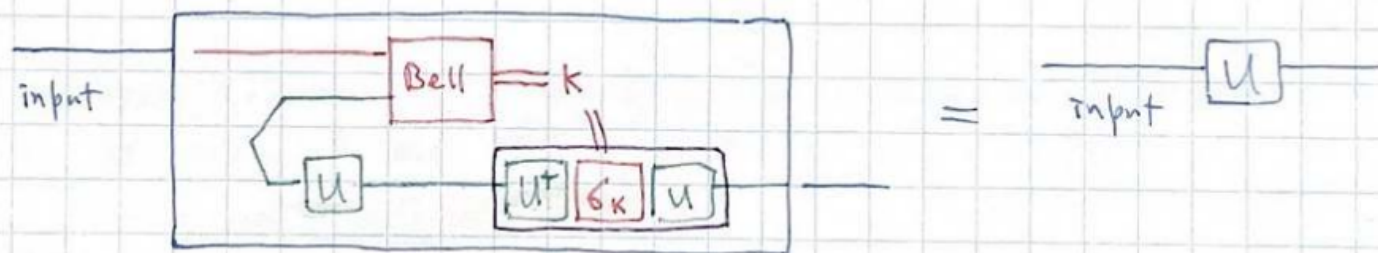
\vdots

C^k

$= \bigcup_n \{U \in U(2^n) : U P_n U^\dagger \in C^{k-1}\}$

Clifford + any C^3 gate universal (Nebe, Rains, Sloane)

Teleporting a C^3 gate:



① This box teleports, then apply U

② This box can be implemented with

(i) State $I \otimes U$ (max entangled state)

← Will learn more in part II

✓ (ii) Bell measurement (XX, ZZ)

✓ (iii) $U C_k U^\dagger$ which is Clifford!

More efficient schemes exist for (NOT, $R_{\pi/8}$, etc (1-bit teleportation))

Gate teleportation summary:

Implements a (difficult to apply, e.g., non-Clifford) gate U on an arbitrary unknown input state, preserving correlations with other systems, by:

Gate teleportation summary:

Implements a (difficult to apply, e.g., non-Clifford) gate U on an arbitrary unknown input state, preserving correlations with other systems, by:

- Performing other simpler gates (e.g., Clifford) and measurements

Gate teleportation summary:

Implements a (difficult to apply, e.g., non-Clifford) gate U on an arbitrary unknown input state, preserving correlations with other systems, by:

- Performing other simpler gates (e.g., Clifford) and measurements
- Preparing a suitable state of the form $U|anc\rangle$ for some standard state $|anc\rangle$. The preparation itself need not involving applying U to $|anc\rangle$.

Historical remarks

The idea to teleport a gate from a state was "folklore," and related to "programmable gate array" (Nielsen and Chuang 1996) in which one "buys the quantum program" (as a quantum state) offline, and convert the state to the ability to perform a gate on a different unknown quantum state later.

Historical remarks

The idea to teleport a gate from a state was "folklore," and related to "programmable gate array" (Nielsen and Chuang 1996) in which one "buys the quantum program" (as a quantum state) offline, and convert the state to the ability to perform a gate on a different unknown quantum state later.

Other applications of this conceptual idea:

- * study of entanglement (entangled state and gates are interconverted to one another using additional classical communication of measurement outcomes)

Historical remarks

The idea to teleport a gate from a state was "folklore," and related to "programmable gate array" (Nielsen and Chuang 1996) in which one "buys the quantum program" (as a quantum state) offline, and convert the state to the ability to perform a gate on a different unknown quantum state later.

Other applications of this conceptual idea:

- * study of entanglement (entangled state and gates are interconverted to one another using additional classical communication of measurement outcomes)
- * proving no-go theorem of improving atomic clocks using entanglement

Historical remarks

The idea to teleport a gate from a state was "folklore," and related to "programmable gate array" (Nielsen and Chuang 1996) in which one "buys the quantum program" (as a quantum state) offline, and convert the state to the ability to perform a gate on a different unknown quantum state later.

Other applications of this conceptual idea:

- * study of entanglement (entangled state and gates are interconverted to one another using additional classical communication of measurement outcomes)
- * proving no-go theorem of improving atomic clocks using entanglement
- * measurement-based quantum computation
- * blind quantum computation
- * verifying untrusted QC with limited quantum capabilities

Further historical remark on FT gates:

The first proposal to implement an encoded operation fault-tolerantly outside of the Clifford group was due to Shor 1996 (surprise!!!) He provided a mysterious and long recipe to perform a Toffoli gate, and mused that it's reminiscent of teleportation.

Further historical remark on FT gates:

The first proposal to implement an encoded operation fault-tolerantly outside of the Clifford group was due to Shor 1996 (surprise!!!) He provided a mysterious and long recipe to perform a Toffoli gate, and mused that it's reminiscent of teleportation.

Preskill (1997) gave a simple circuit explanation of his result. Other circuits similar to what we saw were proposed. Gottesman and Chuang (1999) gave a proposal to teleport gate using (full) teleportation, but the circuits are twice the size of the ad hoc proposals. Chuang, L, Zhou (2000) gave a systematic derivation of many efficient circuits using 1-bit teleportation.

10. Accurate computation out of noisy components (NC 10.5-10.6)

What we are aiming to achieve:

(b) Principles of fault-tolerant quantum computation (don't make a mess)

Benefit of achieving the above

(c) The threshold theorem
(good enough implies arbitrarily good)

How to achieve (b)?

(d)-(f) Fault-tolerant logical Pauli & Clifford gates

(g) Fault-tolerant $\pi/8$ gate, 1-bit teleportation

(h) Fault-tolerant measurements

(i) Overhead and assumptions

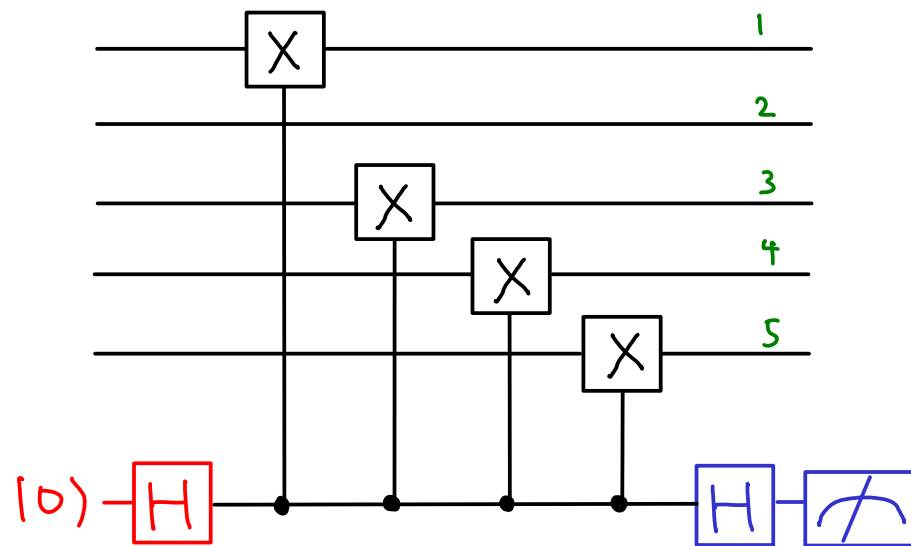
To obtain all the gadgets, we still have to

- prepare simple logical states
- measure logical states
- perform EC

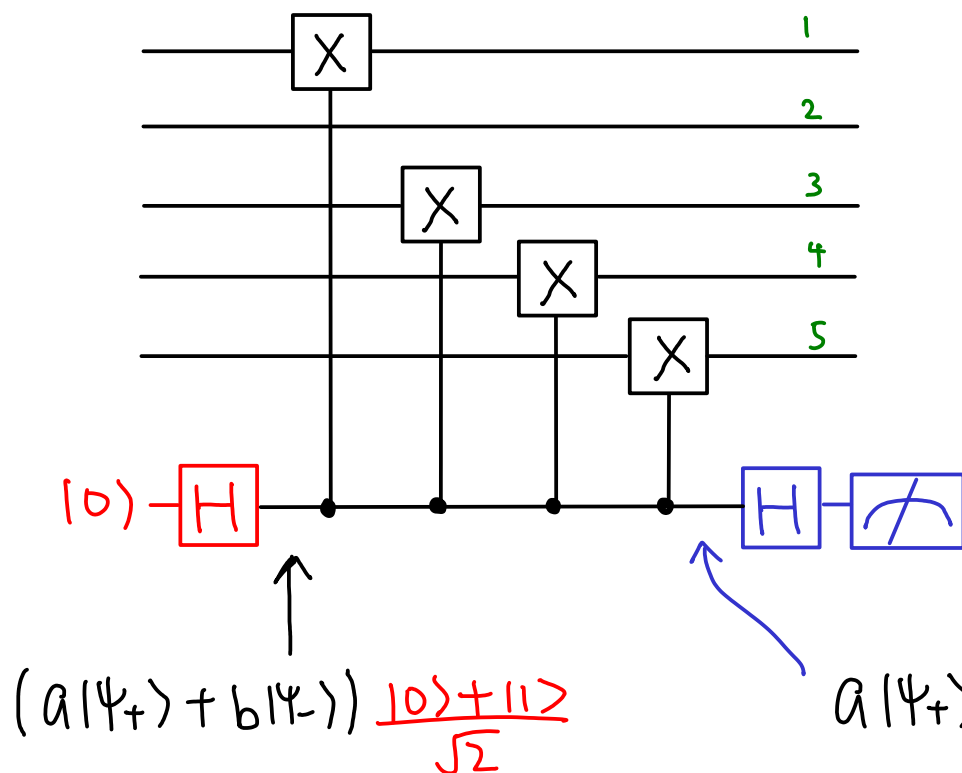
all without spreading errors.

All these tasks rely on measuring the eigenvalue of Pauli matrices.

For example, we can measure XIXXX by:

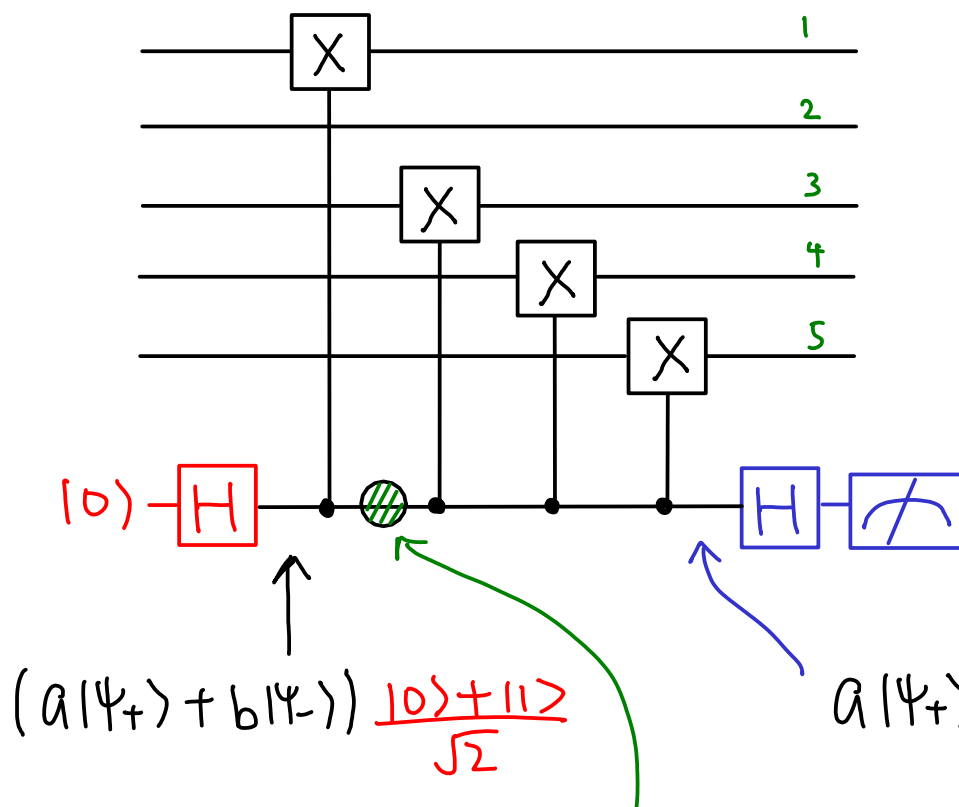


For example, we can measure XIXXX by:



Let the input on qubits 1-5 be $a|\psi_+\rangle + b|\psi_-\rangle$ where $|\psi_{\pm}\rangle$ are ± 1 eigenstates of XIXXX.

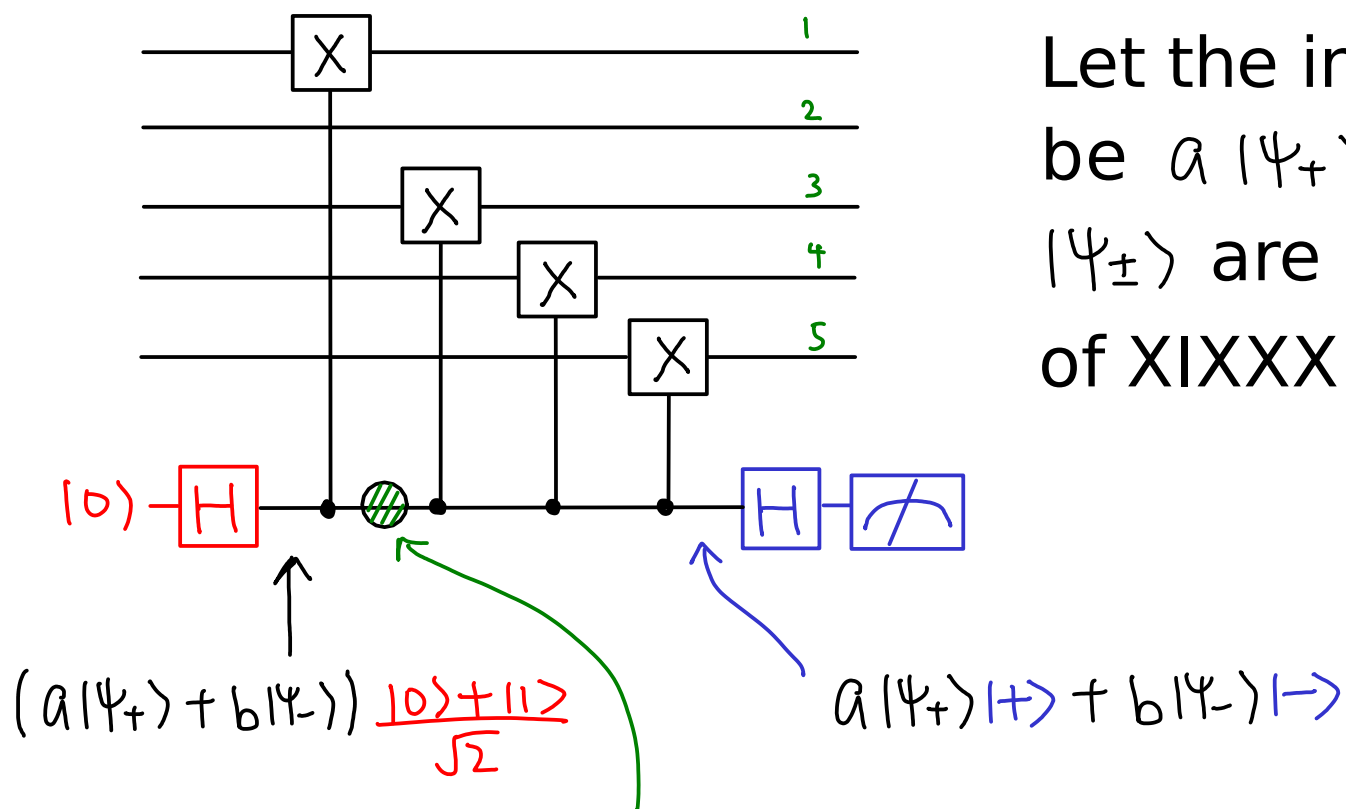
For example, we can measure XIXXX by:



Let the input on qubits 1-5 be $a|\psi_+\rangle + b|\psi_-\rangle$ where $|\psi_{\pm}\rangle$ are ± 1 eigenstates of XIXXX.

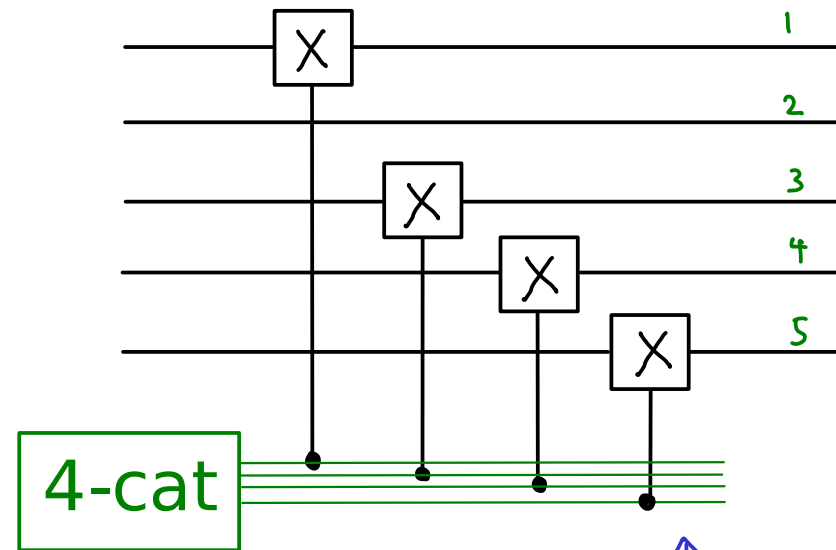
But a single X error here can propagate to qubits 3,4,5 in the code block, so, this measurement is not fault tolerant.

For example, we can measure XIXXX by:



But a single X error here can propagate to qubits 3,4,5 in the code block, so, this measurement is not fault tolerant.

Solution: (1) replace $|+\rangle$ by "4-cat";
one qubit for each controlled-X.

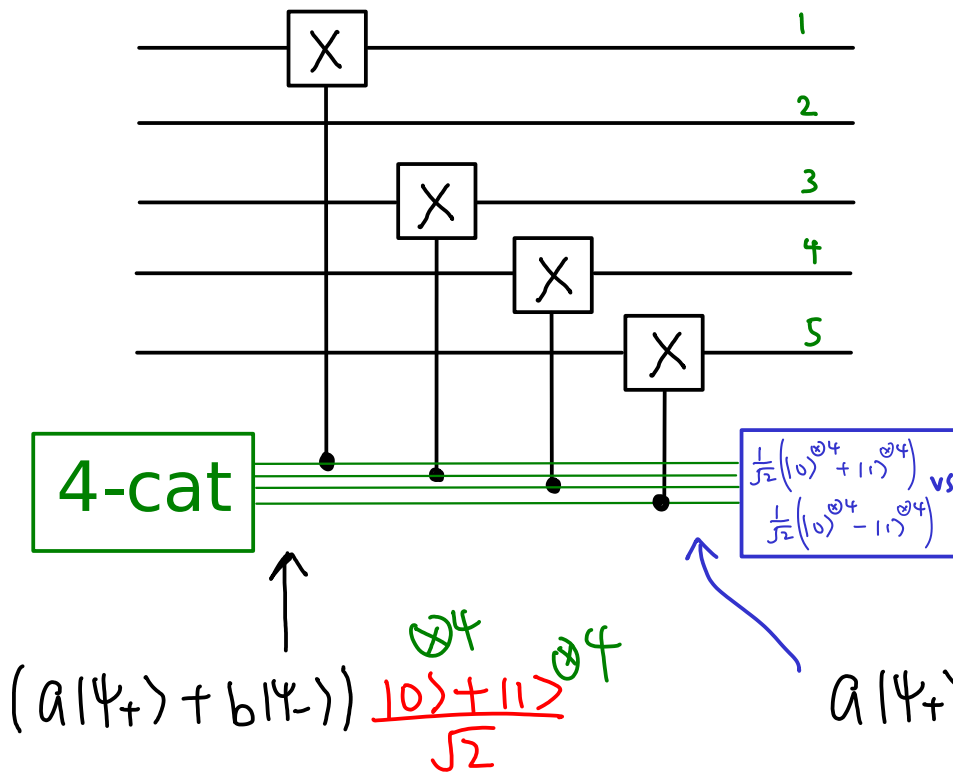


Let the input on qubits 1-5
be $a|\psi_+\rangle + b|\psi_-\rangle$ where
 $|\psi_{\pm}\rangle$ are ± 1 eigenstates
of $XIXXX$.

$$(a|\psi_+\rangle + b|\psi_-\rangle) \frac{10^{\otimes 4} + 11^{\otimes 4}}{\sqrt{2}}$$

$$a|\psi_+\rangle \frac{1}{\sqrt{2}}(|0^{\otimes 4} + 11^{\otimes 4}\rangle) + b|\psi_-\rangle \frac{1}{\sqrt{2}}(|0^{\otimes 4} - 11^{\otimes 4}\rangle)$$

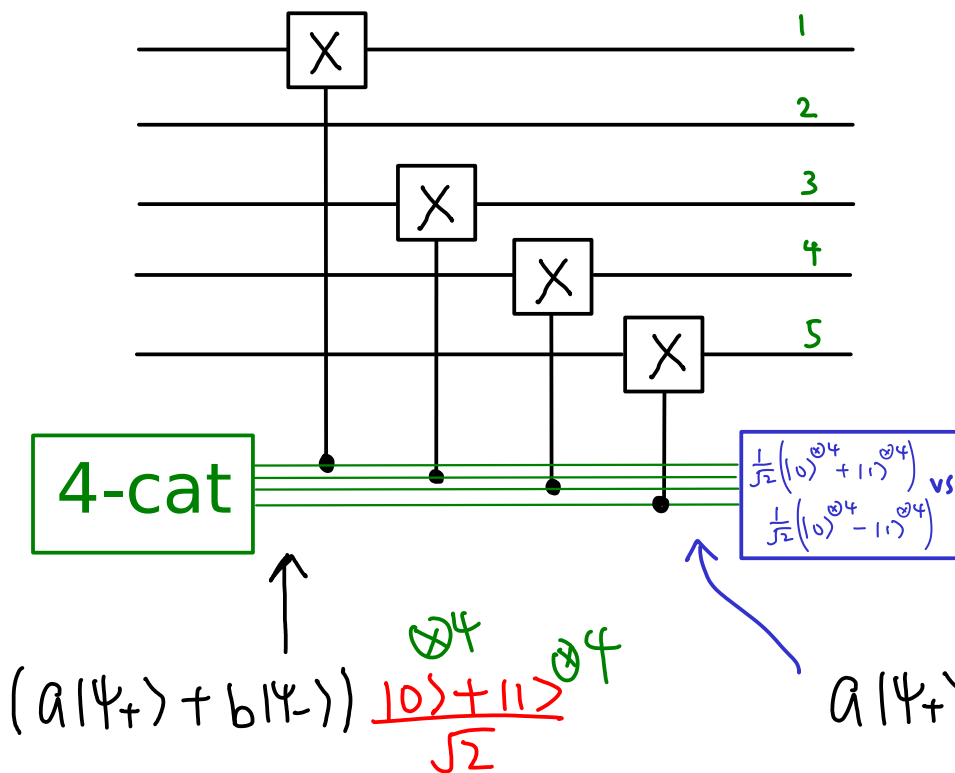
Solution: (1) replace $|+\rangle$ by "4-cat";
one qubit for each controlled-X.



Let the input on qubits 1-5
be $a|\psi_+\rangle + b|\psi_-\rangle$ where
 $|\psi_{\pm}\rangle$ are ± 1 eigenstates
of $XIXXX$.

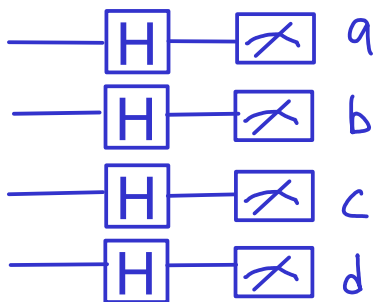
$$a|\psi_+\rangle \frac{1}{\sqrt{2}}(|0\rangle^4 + |1\rangle^4) + b|\psi_-\rangle \frac{1}{\sqrt{2}}(|0\rangle^4 - |1\rangle^4)$$

Solution: (1) replace $|+\rangle$ by "4-cat";
one qubit for each controlled-X.

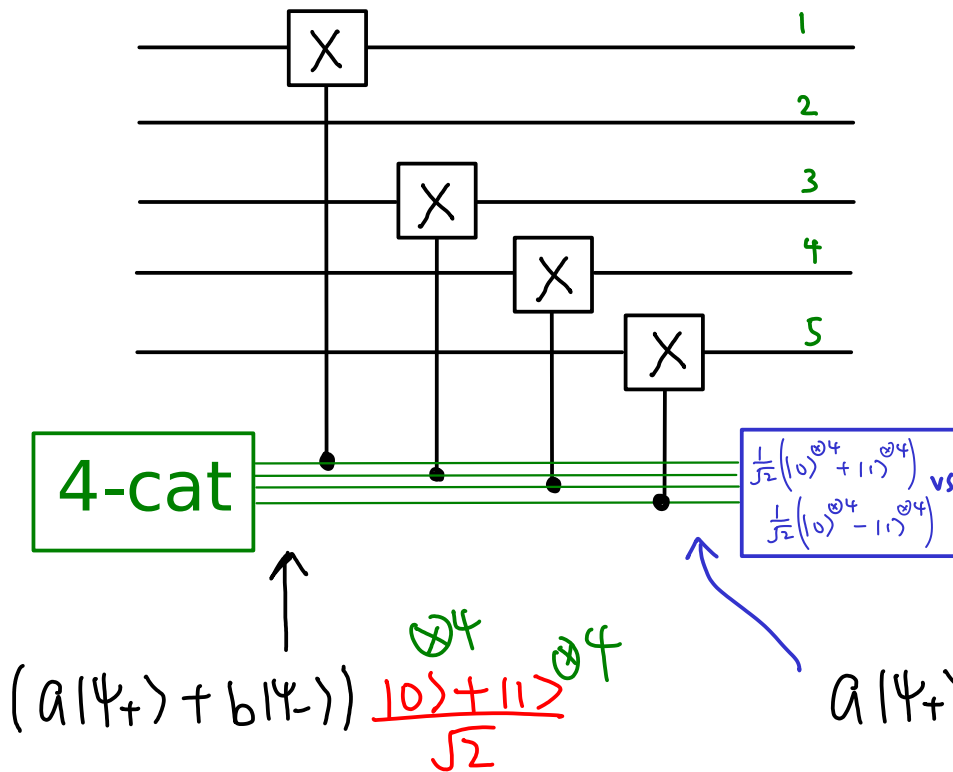


Let the input on qubits 1-5
be $a|\psi_+\rangle + b|\psi_-\rangle$ where
 $|\psi_{\pm}\rangle$ are ± 1 eigenstates
of $XIXXX$.

(2) modify the measurement to be transversal ...

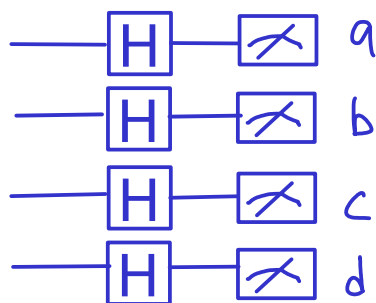


Solution: (1) replace $|+\rangle$ by "4-cat";
one qubit for each controlled-X.



Let the input on qubits 1-5
be $a|\psi_+\rangle + b|\psi_-\rangle$ where
 $|\psi_{\pm}\rangle$ are ± 1 eigenstates
of $XIXXX$.

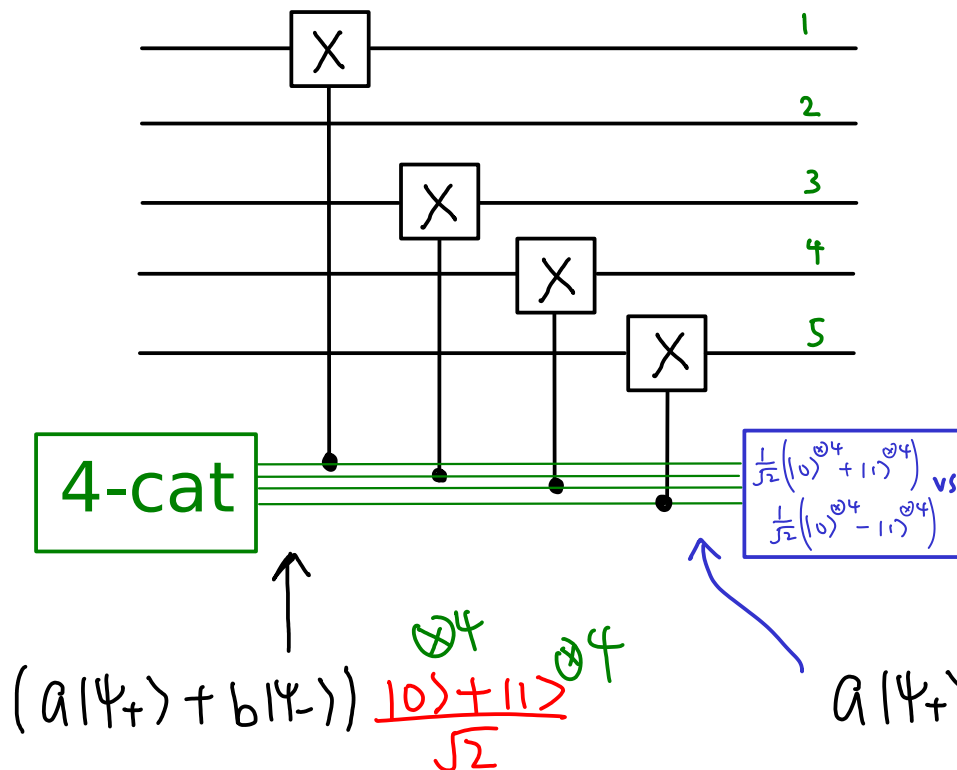
(2) modify the measurement to be transversal ...



Ex: verify that

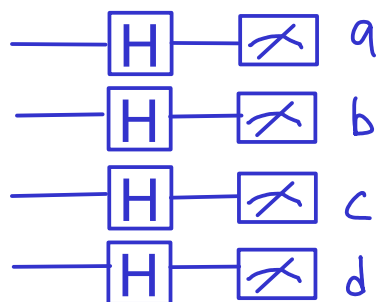
$a+b+c+d = 0 \pmod{2}$ if state is $\frac{1}{\sqrt{2}}(|0\rangle^{\otimes 4} + |1\rangle^{\otimes 4})$
1 $\frac{1}{\sqrt{2}}(|0\rangle^{\otimes 4} - |1\rangle^{\otimes 4})$

Solution: (1) replace $|+\rangle$ by "4-cat";
one qubit for each controlled-X.



Let the input on qubits 1-5 be $a|\psi_+\rangle + b|\psi_-\rangle$ where $|\psi_{\pm}\rangle$ are ± 1 eigenstates of XIXXX.

(2) modify the measurement to be transversal ...



Ex: verify that

$a+b+c+d = 0 \pmod{2}$ if state is $\frac{1}{\sqrt{2}}(|0\rangle^{\otimes 4} + |1\rangle^{\otimes 4})$
1 $\frac{1}{\sqrt{2}}(|0\rangle^{\otimes 4} - |1\rangle^{\otimes 4})$

Derivation (1) for the exercise using stabilizers (out of syllabus W25).
 Idea as in A1Q4(b). Meas $S1 \otimes S2$ by meas $S1 \otimes I$, $I \otimes S2$ separately.

	$ 0000\rangle + 1111\rangle$		vs	$ 0000\rangle - 1111\rangle$	
stabs	zzzz	+	+		
	zzzz	+	+		
	zzzz	+	+		
	zzzz	+	-		
$H^{\otimes 4}$	xxxx	+	+		
	xxxx	+	+		
	xxxx	+	+		
	zzzz	+	-		
				displaced.	
				← preserved.	

meas zzzz, zzzz, zzzz, zzzz

Derivation (2): use Quiz Q3(b)

(b) [1 mark] Show that $\forall s \in \{0, 1\}^n$,

$$H^{\otimes n}|s\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} (-1)^{s \cdot y} |y\rangle$$

where for $s = (s_1, s_2, \dots, s_n)$, $y = (y_1, y_2, \dots, y_n)$, $s \cdot y \equiv s_1 y_1 + s_2 y_2 + \dots + s_n y_n \pmod{2}$, and $|s\rangle = |s_1\rangle |s_2\rangle \cdots |s_n\rangle$, $|y\rangle = |y_1\rangle |y_2\rangle \cdots |y_n\rangle$.

Derivation (2): use Quiz Q3(b)

(b) [1 mark] Show that $\forall s \in \{0, 1\}^n$,

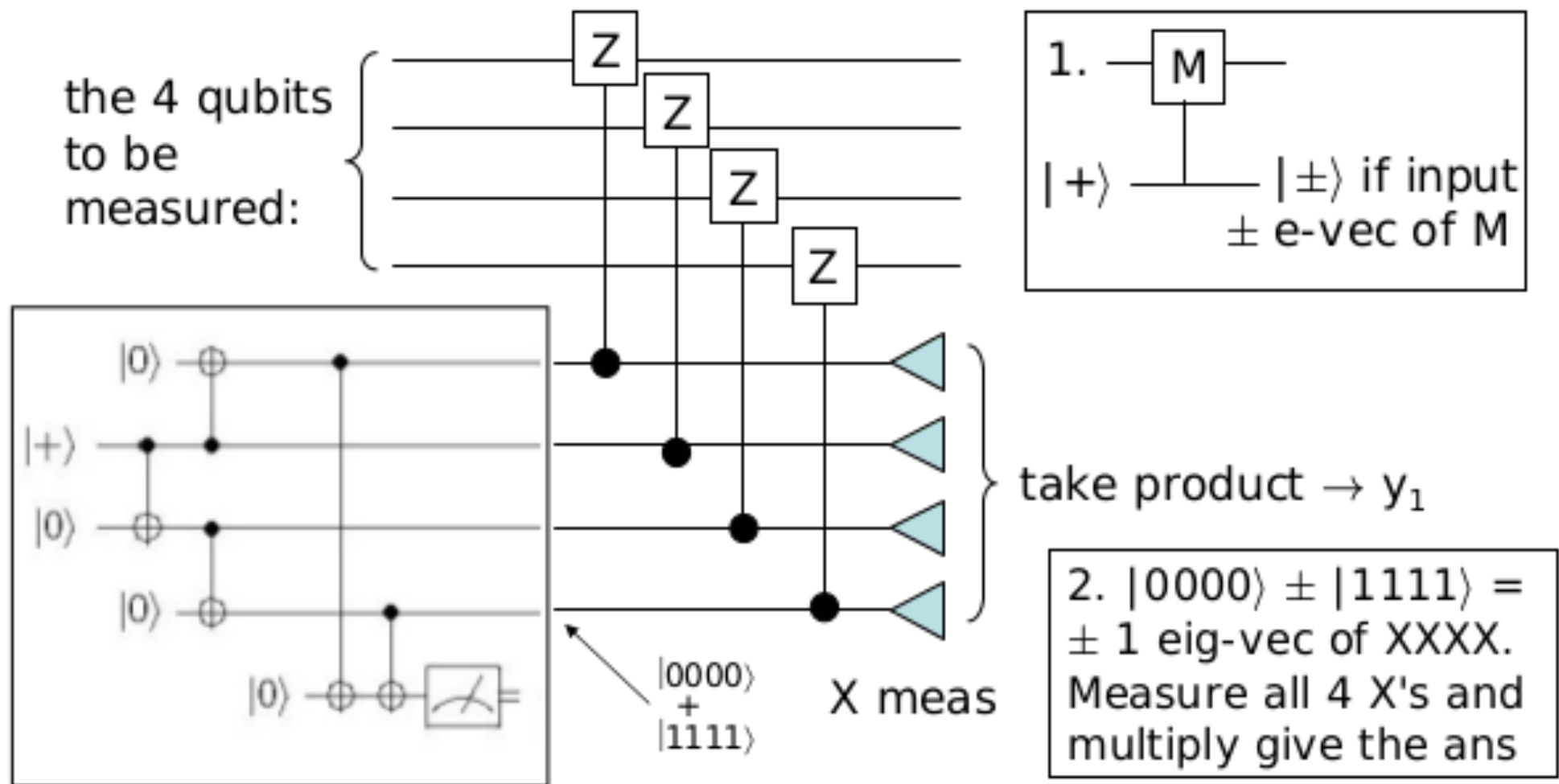
$$H^{\otimes n}|s\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} (-1)^{s \cdot y} |y\rangle$$

where for $s = (s_1, s_2, \dots, s_n)$, $y = (y_1, y_2, \dots, y_n)$, $s \cdot y \equiv s_1 y_1 + s_2 y_2 + \dots + s_n y_n \pmod{2}$, and $|s\rangle = |s_1\rangle |s_2\rangle \dots |s_n\rangle$, $|y\rangle = |y_1\rangle |y_2\rangle \dots |y_n\rangle$.

Apply to $s = 0000$ and 1111 , summing gives only the terms with even Hamming weight (number of 1's in a bitstring). Subtracting gives only the terms with odd Hamming weight. So, measuring the resulting state in computational basis after $H^{\otimes 4}$ can determine $|0000\rangle + |1111\rangle$ vs $|0000\rangle - |1111\rangle$.

Fault tolerant measurements:

e.g. measure $ZZZZ$ on 4 specific qubits in the codeblock:



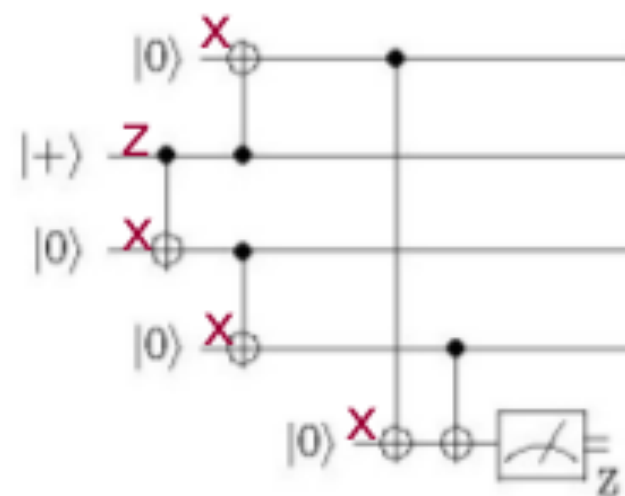
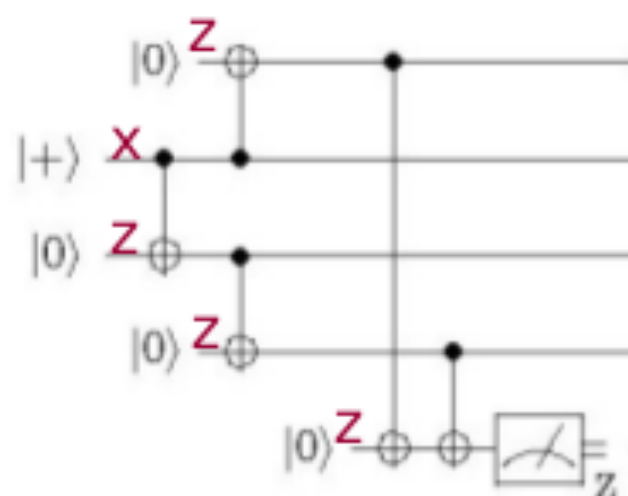
accept only if outcome = ~~1~~ 0

claim: at most 1 X or 1 Z error.

1. If noiseless, this prepares $|0000\rangle + |1111\rangle$
2. Output cannot have more than 2 X errors or 1 Z error.
3. 1 fault cannot result in 2 X errors:

USE $s=1$, that if there is a fault we analyze, no other faults are around.

1 fault at time step 1:



Each of above has no effect.

$Y \sim$ an X and a Z , same as an error in the RHS

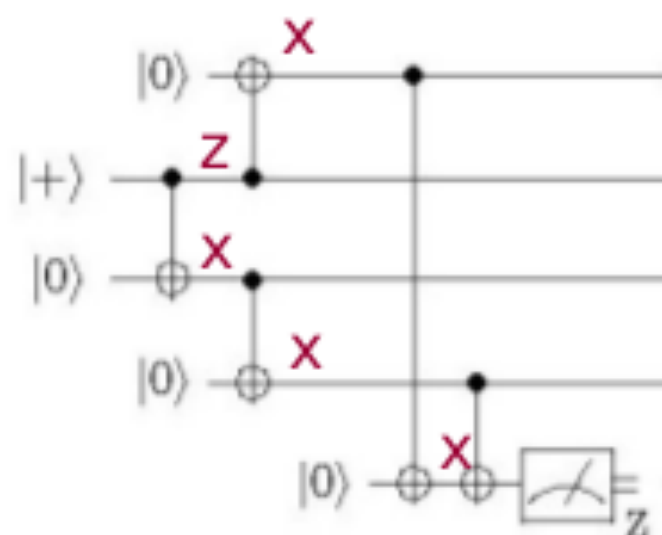
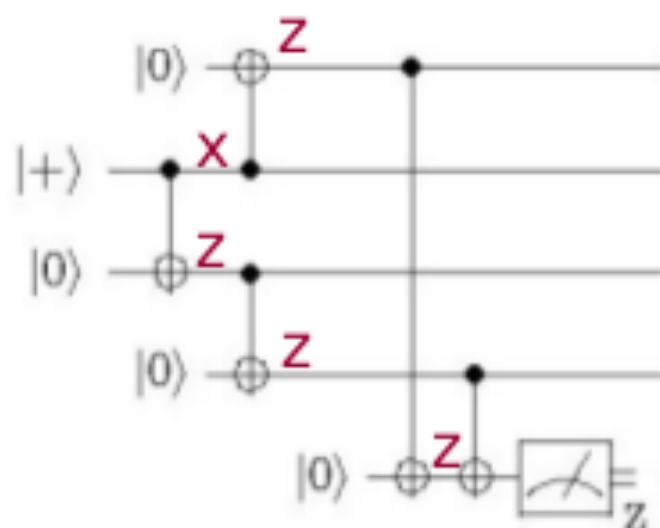
Each of above commutes pass CNOT. So, equivalent to 1 fault in 2nd step.

All case then covered in consideration for step 2.

1. If noiseless, this prepares $|0000\rangle + |1111\rangle$
2. Output cannot have more than 2 X errors or 1 Z error.
3. 1 fault cannot result in 2 X errors:

1 fault in all subsequent steps results in 1 error. Thus [3] holds.

1 fault at time step 2:



Each Z stays and affect the final state, but that's OK.

Right before the verification (last two CNOT's) $X_2 \rightarrow X_1 X_2$.
 $X_3 \rightarrow X_3 X_4$. Any of $X_{1,2,3,4}$ will result in meas=-1 & we reject.
 Same for the X in the verification step. Same for 1 Y. 1

A final error in the 4-cat or one measurement can be interpreted as an error in the circuit with 4 controlled-Z, so, 4-cat and measurements are assumed perfect.

A final error in the 4-cat or one measurement can be interpreted as an error in the circuit with 4 controlled-Z, so, 4-cat and measurements are assumed perfect.

If there is an error in the circuit:

(1) If one of the controlled-Z is faulty, we have at most one error in the code block, one error in the ancilla, and a bad meas outcome.

A final error in the 4-cat or one measurement can be interpreted as an error in the circuit with 4 controlled-Z, so, 4-cat and measurements are assumed perfect.

If there is an error in the circuit:

(1) If one of the controlled-Z is faulty, we have at most one error in the code block, one error in the ancilla, and a bad meas outcome.

(2) If there is one fault in the code block, we have at most one error in the code block, one error in the ancilla, and a bad meas outcome.

A final error in the 4-cat or one measurement can be interpreted as an error in the circuit with 4 controlled-Z, so, 4-cat and measurements are assumed perfect.

If there is an error in the circuit:

(1) If one of the controlled-Z is faulty, we have at most one error in the code block, one error in the ancilla, and a bad meas outcome.

(2) If there is one fault in the code block, we have at most one error in the code block, one error in the ancilla, and a bad meas outcome.

(3) If there is 1 error in the ancilla, there is at most 1 error in the code block, and a bad measurement outcome.

A final error in the 4-cat or one measurement can be interpreted as an error in the circuit with 4 controlled-Z, so, 4-cat and measurements are assumed perfect.

If there is an error in the circuit:

(1) If one of the controlled-Z is faulty, we have at most one error in the code block, one error in the ancilla, and a bad meas outcome.

(2) If there is one fault in the code block, we have at most one error in the code block, one error in the ancilla, and a bad meas outcome.

(3) If there is 1 error in the ancilla, there is at most 1 error in the code block, and a bad measurement outcome.

Repeat measurement 3 times. If at most 1 fault occurs throughout, majority of outcomes will be correct, with at most one residual error in the code block.

The above measurement (with 3 repetitions) can now be used to prepare the logical $|0\rangle$ state, by measuring the stabilizer generators (and accepting only if the outcomes are all +1), and by measuring the logical Z.

The above measurement (with 3 repetitions) can now be used to prepare the logical $|0\rangle$ state, by measuring the stabilizer generators (and accepting only if the outcomes are all +1), and by measuring the logical Z.

Finally, to measure syndrome, one potential problem is an error that comes in between the measurement of different generators. So we repeat not the individual measurements but the entire syndrome measurement 3 times, followed by an appropriate correction. This gives an EC gadget.

Putting state preparation gadgets, encoded gate gadgets, measurement gadgets, and EC gadgets together in a region/rectangle,

Putting state preparation gadgets, encoded gate gadgets, measurement gadgets, and EC gadgets together in a region/rectangle,

if there is a single fault:

- * if it hits early, the rest of the region is correct, so the fault will be corrected by the "trailing" EC gadgets

Putting state preparation gadgets, encoded gate gadgets, measurement gadgets, and EC gadgets together in a region/rectangle,

if there is a single fault:

- * if it hits early, the rest of the region is correct, so the fault will be corrected by the "trailing" EC gadgets
- * if it hits late enough, at the trailing EC gadgets, the region can be interpreted as a correct simulation followed by a fault in the next region ...

Putting state preparation gadgets, encoded gate gadgets, measurement gadgets, and EC gadgets together in a region/rectangle,

if there is a single fault:

- * if it hits early, the rest of the region is correct, so the fault will be corrected by the "trailing" EC gadgets
- * if it hits late enough, at the trailing EC gadgets, the region can be interpreted as a correct simulation followed by a fault in the next region ...

This gives a handwavy argument why it takes 2 faults within a region to make a faulty simulation