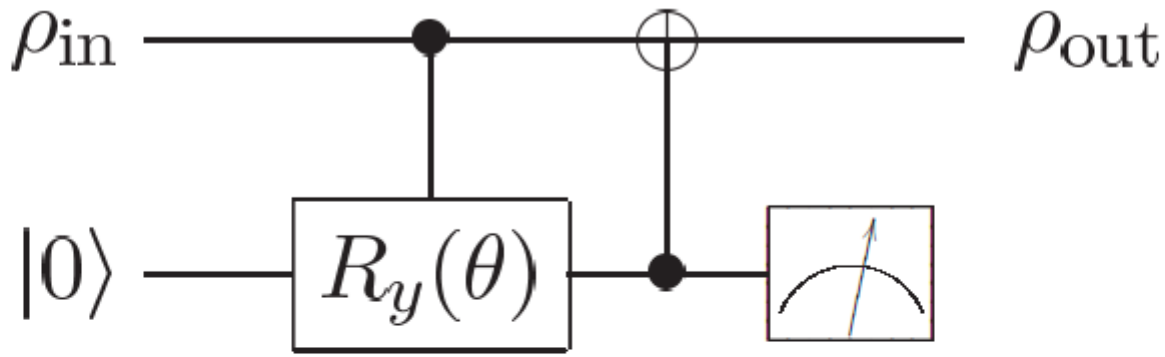


Figure 8.13. Circuit model for amplitude damping

Exercise 8.20: (Circuit model for amplitude damping) Show that the circuit in Figure 8.13 models the amplitude damping quantum operation, with $\sin^2(\theta/2) = \gamma$.

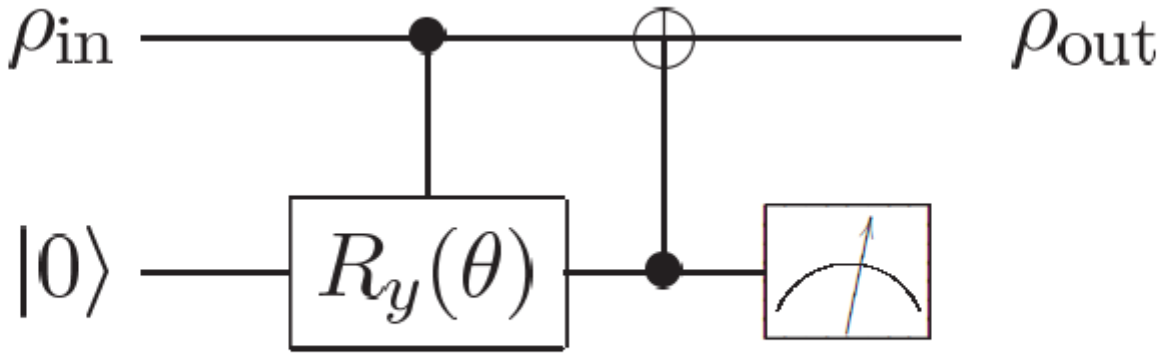


Figure 8.13. Circuit model for amplitude damping

$$P_0 = |0\rangle\langle 0|$$

$$P_1 = |1\rangle\langle 1|$$

$$R_y(\theta) = e^{-i\frac{\theta}{2}\sigma_y} = \cos\left(\frac{\theta}{2}\right) I - i \sin\left(\frac{\theta}{2}\right)\sigma_y$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$\langle i|P_0 = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ +\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$$

$$cR_y^{01}(\theta) = P_0 \otimes I + P_1 \otimes R_y(\theta)$$

$$cX_{10} = I \otimes P_0 + X \otimes P_1$$

$$E_i = (I \otimes \langle i|)(I \otimes P_0 + X \otimes P_1)(P_0 \otimes I + P_1 \otimes R_y(\theta))(I \otimes |0\rangle)$$

$$= (I \otimes \delta_{i0}\langle 0| + X \otimes \delta_{i1}\langle 1|)(P_0 \otimes |0\rangle + P_1 \otimes R_y(\theta)|0\rangle)$$

$$= P_0 \otimes \delta_{i0} + P_1 \otimes \delta_{i0}\langle 0|R_y(\theta)|0\rangle + X P_1 \otimes \delta_{i1}\langle 1|R_y(\theta)|0\rangle$$

$$\langle 0|R_y(\theta)|0\rangle = \cos\left(\frac{\theta}{2}\right) = \begin{pmatrix} 1 & 0 \\ 0 & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$$

$$\langle 1|R_y(\theta)|0\rangle = \pm\sin\left(\frac{\theta}{2}\right) \quad \sqrt{\gamma}$$

$i=0$

$$= P_0 + P_1 \cos\left(\frac{\theta}{2}\right)$$

$i=1$

$$= |0\rangle\langle 1|\left(\pm\sin\left(\frac{\theta}{2}\right)\right) = \begin{pmatrix} 0 & \pm\sin\left(\frac{\theta}{2}\right) \\ 0 & 0 \end{pmatrix}$$

Exercise 8.23: (Amplitude damping of dual-rail qubits) Suppose that a single qubit state is represented by using two qubits, as

$$|\psi\rangle = a\,|01\rangle + b\,|10\rangle\,. \tag{8.113}$$

Show that $\mathcal{E}_{\rm AD} \otimes \mathcal{E}_{\rm AD}$ applied to this state gives a process which can be described by the operation elements

$$E_0^{\rm dr} = \sqrt{1-\gamma}\, I \tag{8.114}$$

$$E_1^{\rm dr} = \sqrt{\gamma}\left[|00\rangle\langle 01| + |00\rangle\langle 10|\right], \tag{8.115}$$

that is, either nothing ($E_0^{\rm dr}$) happens to the qubit, or the qubit is transformed ($E_1^{\rm dr}$) into the state $|00\rangle$, which is orthogonal to $|\psi\rangle$. This is a simple error-detection code, and is also the basis for the robustness of the 'dual-rail' qubit discussed in Section 7.4.

Exercise 8.30: $(T_2 \leq T_1/2)$ The T_2 phase coherence relaxation rate is just the exponential decay rate of the off-diagonal elements in the qubit density matrix, while T_1 is the decay rate of the diagonal elements (see Equation (7.144)). Amplitude damping has *both* nonzero T_1 and T_2 rates; show that for amplitude damping $T_2 = T_1/2$. Also show that if amplitude and phase damping are *both* applied then $T_2 \leq T_1/2$.

$$\begin{bmatrix} a & b \\ b^* & 1-a \end{bmatrix} \rightarrow \begin{bmatrix} (a-a_0)e^{-t/T_1} + a_0 & be^{-t/T_2} \\ b^* e^{-t/T_2} & (a_0-a)e^{-t/T_1} + 1 - a_0 \end{bmatrix}, \qquad (7.144)$$

$$\begin{pmatrix} a_0 & 0 \\ 0 & 1-a_0 \end{pmatrix} = \rho_\infty$$

ibm_washington

Exploratory

Details

127
Qubits

64
QV

850
CLOPS

Status: Online
Total pending jobs: 140 jobs
Processor type: Eagle r1
Version: 1.1.0
Basis gates: CX, ID, RZ, SX, X
Your usage: --

Avg. CNOT Error: 1.000e+0
Avg. Readout Error: 2.826e-2
Avg. T1: 101.21 us
Avg. T2: 96.28 us
Providers with access: --
Supports Qiskit Runtime: Yes

Calibration data

Map view | Graph view | Table view

Qubit:

Frequency (GHz)

Avg 5.064

min 4.767 max 5.292

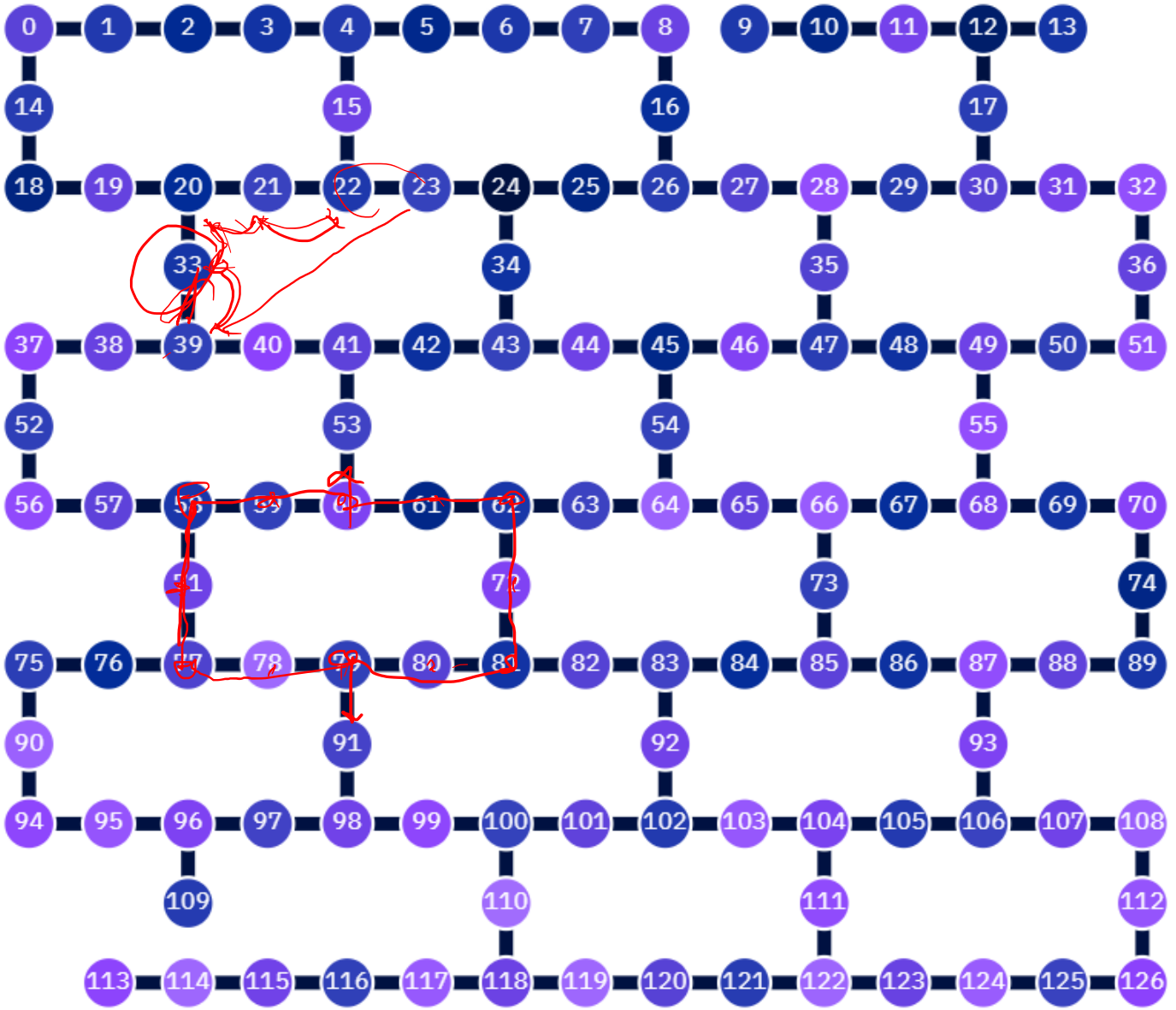
Connection:

CNOT error

Avg 1.000e+0

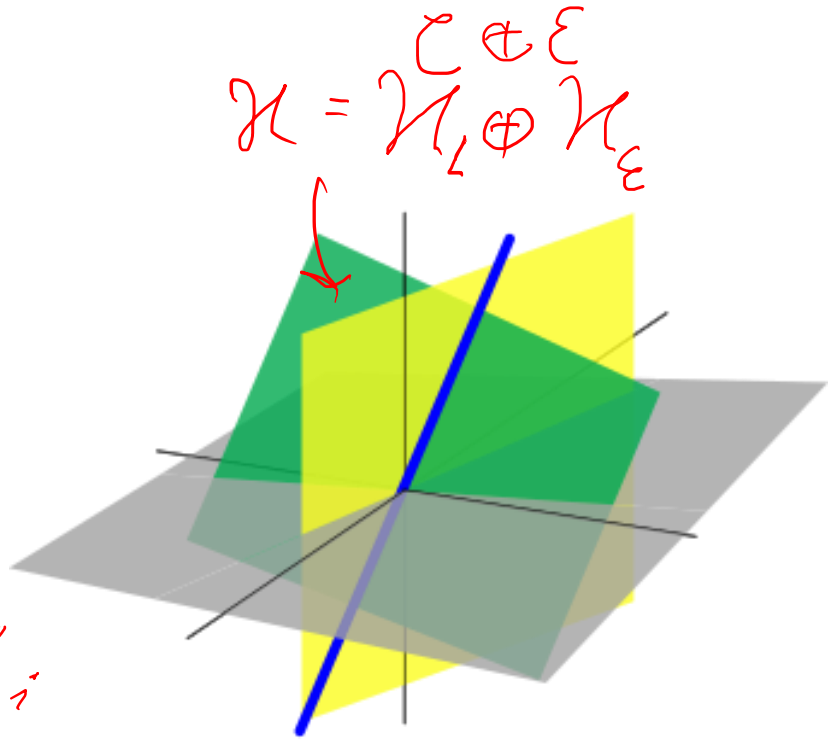
min 1.000e+0 max 1.000e+0

IBM Quantum Washington machine specs

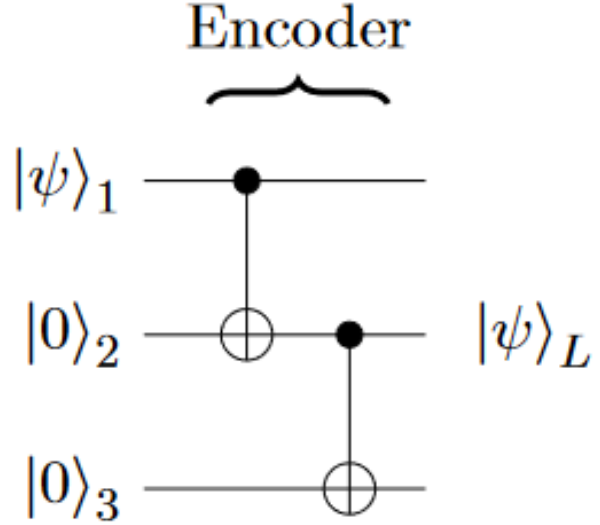


Stabilizer codes

Code subspace and stabilizers

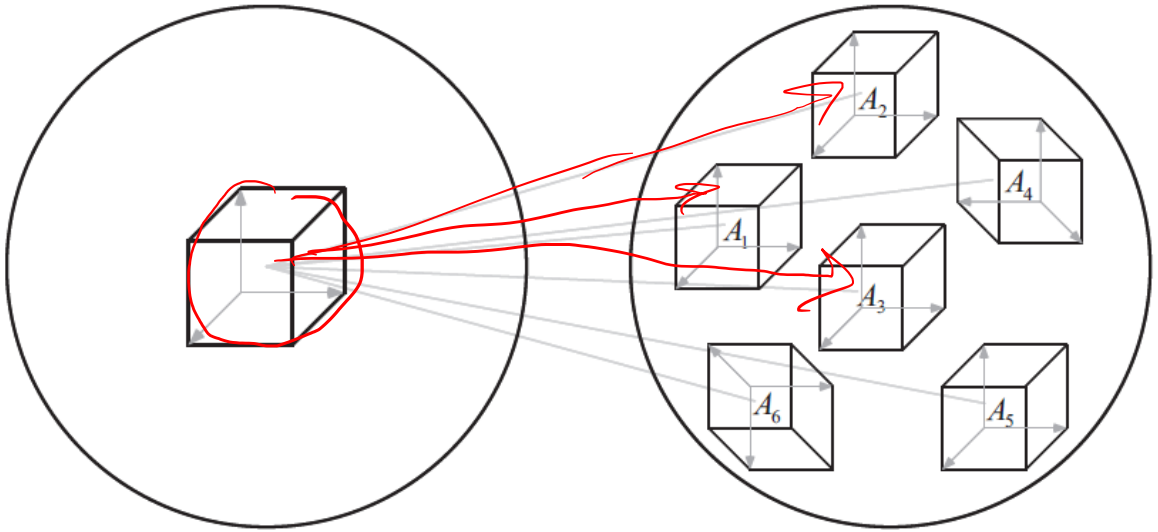


- The Shor code is an example of a **stabilizer code**.
- Correct codestates live in the **code subspace** $\mathcal{C} \subset \mathcal{H}$ of the full Hilbert space.
- E.g., for 3-qubit code subspace $\mathcal{C}$ spanned by $|000\rangle, |111\rangle$. $\quad |\psi\rangle_L = a|000\rangle + b|111\rangle$
- Code subspace $\mathcal{C}$ can be determined by **stabilizers** ("stabilizer codes"): Operators S_i such that
  $S_i|\psi\rangle_L = |\psi\rangle_L \;\; \forall\, |\psi\rangle_L \in \mathcal{C}.$
- In other words, correct codewords are eigenstates of all the stabilizers with eigenvalue 1. **Stabilizers are measured to have value 1 in any correct codestate.**
- E.g. for 3-qubit code $S_1 = Z_1Z_2,\; S_2 = Z_2Z_3.$



Error detection and code distance

- Detectable errors move the state outside the code subspace to an orthogonal subspace. This can be detected by a change in the values of the stabilizers (”parity checks”).
- **Distance** of an ECC: Minimal number of single-qubit errors that transforms one correct codestate to another.
- QECC of distance d with n physical and k logical qubits is denoted by $[[n, k, d]]$.
- A distance d code can identify and correct up to $t = \frac{d-1}{2}$ single-qubit errors.



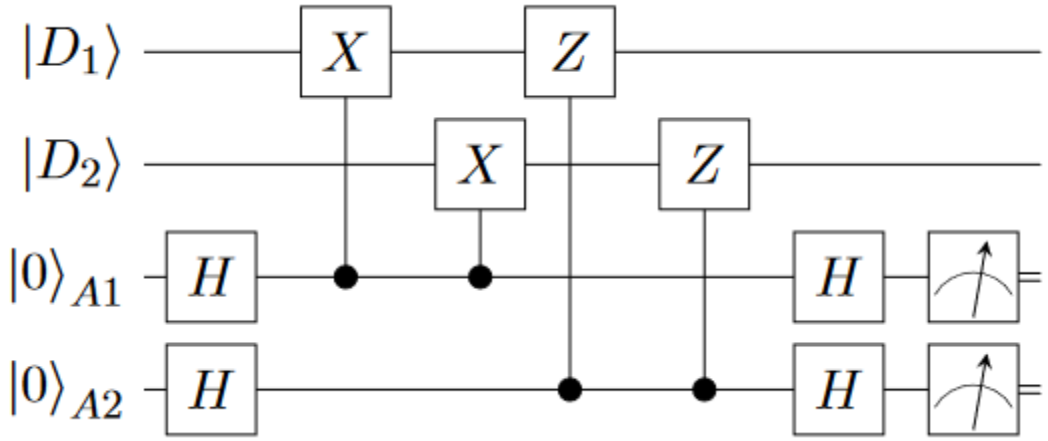
For example:

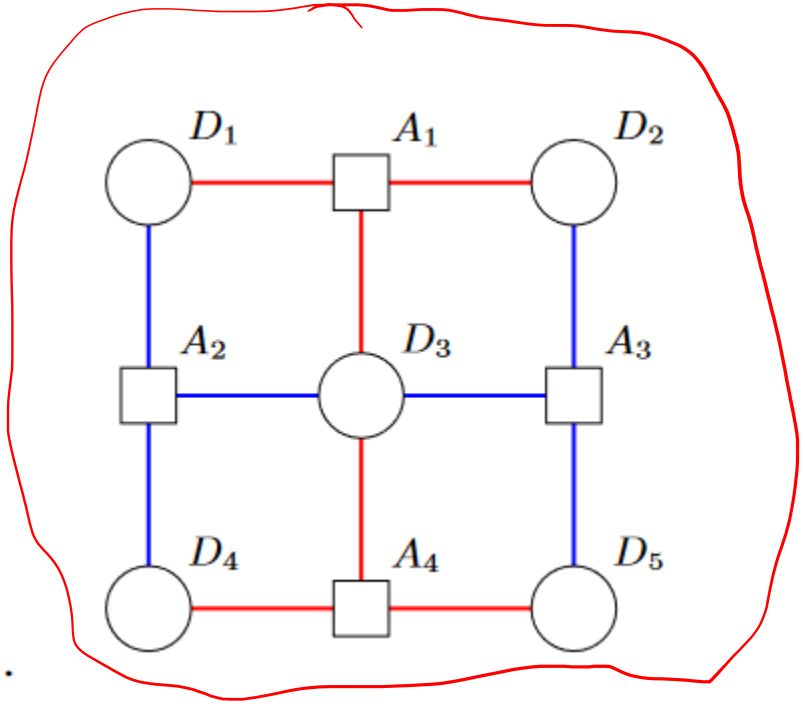
- 3-qubit code is a $[[3,1,1]]$-code. Cannot detect phase (Z) errors at all!
- Shor code is a [[9,1,3]]-code.

Surface codes

[[5,1,2]] surface code

- Each square on a 2d grid has two data and two ancilla qubits. Ancilla qubits are used to perform the stabilizer measurements on the data qubits (red = X measurements, blue = Z measurements).





$$\mathcal{S}_{[[5,1,2]]} = \langle X_{D_1}X_{D_2}X_{D_3},\ Z_{D_1}Z_{D_3}Z_{D_4},\ Z_{D_2}Z_{D_3}Z_{D_5},\ X_{D_3}X_{D_4}X_{D_5}\rangle.$$

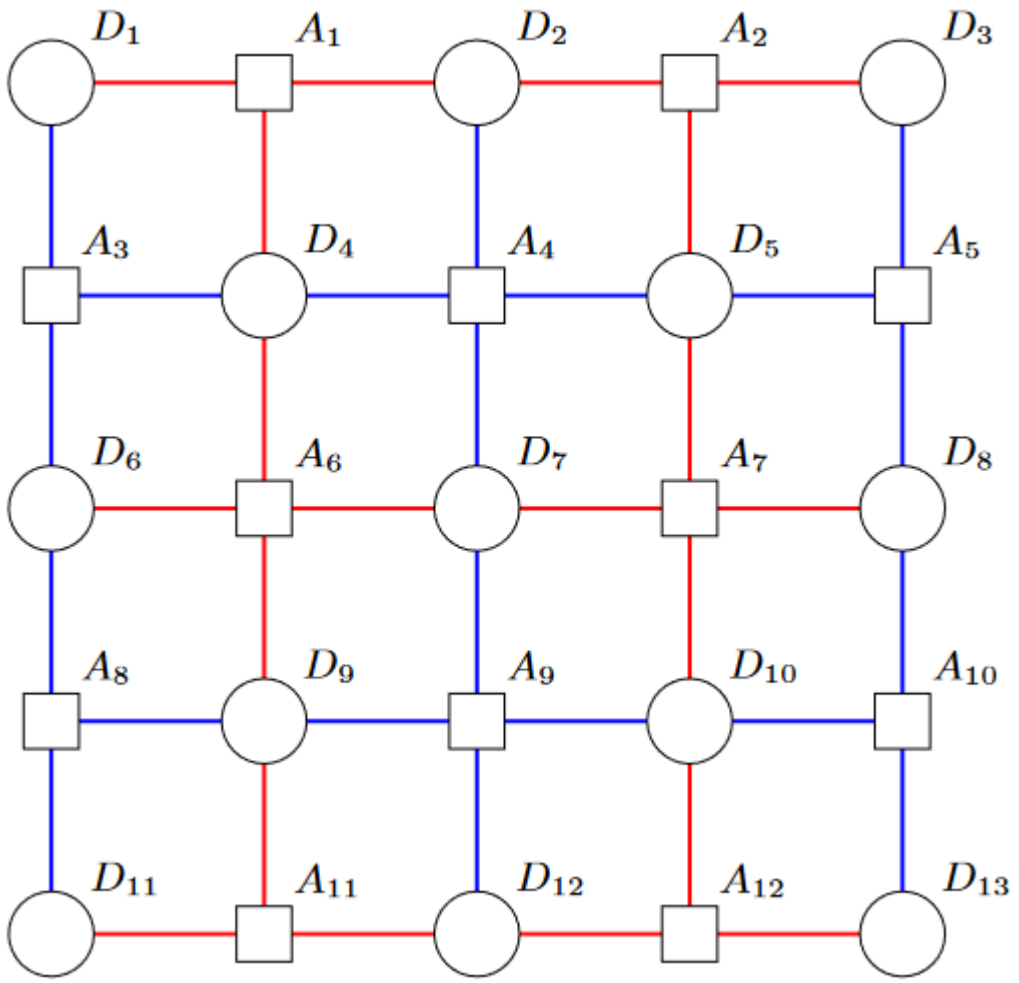
$$X_L = X_1X_4, \qquad Z_L = Z_1Z_2$$

[[13,1,3]] surface code

$\frac{d-1}{2} = 1$

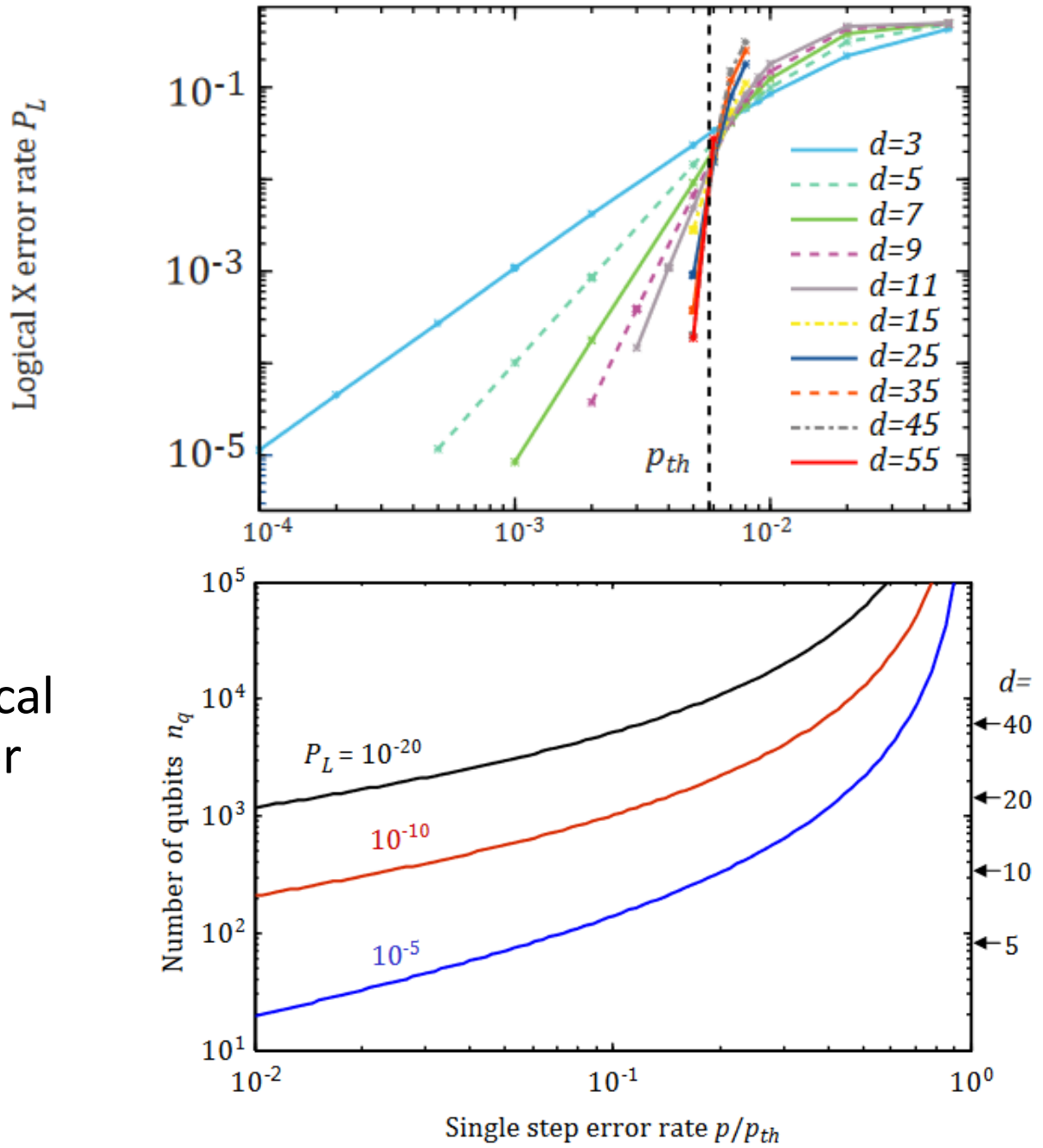
$$X_L = X_1 X_6 X_{11},$$

$$Z_L = Z_1 Z_2 Z_3$$

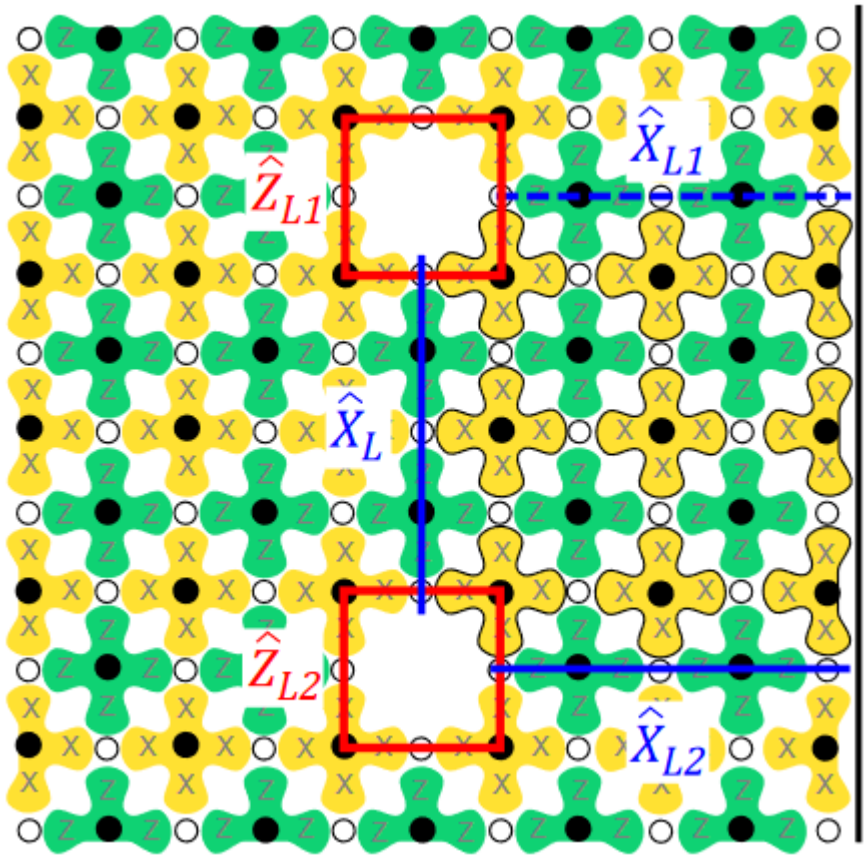


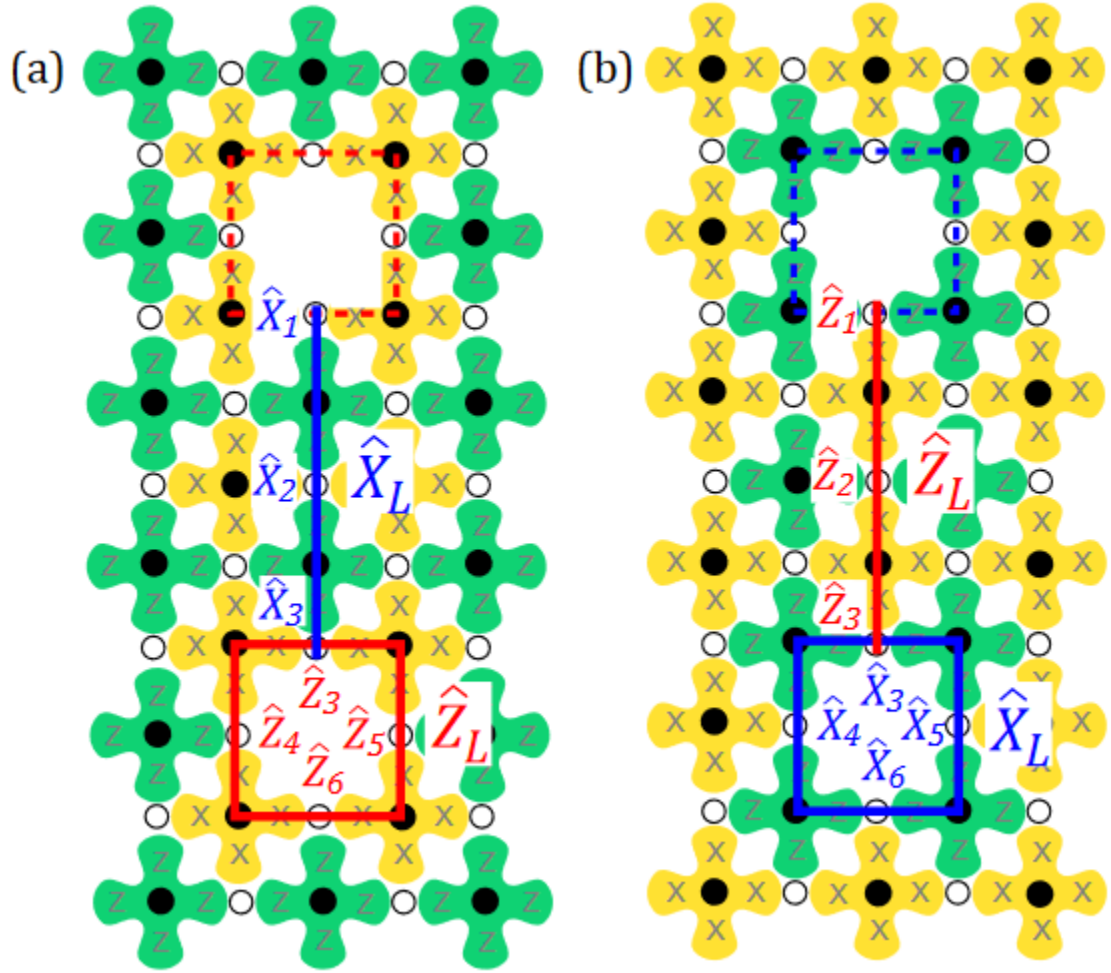
2d surface code qubits

- Surface code construction can be scaled up arbitrarily.
- Only nearest-neighbour qubit connectivity required!
- Code distance and qubit number related as $n = d^2 + (d-1)^2$.
- **Threshold theorem** for stabilizer codes: Increasing code distance will reduce the logical error rate p_L, provided that the physical error rate is below some p_{th}.
- Scaling with distance $p_L \sim (p/p_{th})^{d/2}$.
- For the 2d surface code estimated threshold $p_{th} \approx 1 - 10\%$.



Multiple logical qubits on 2d surface code





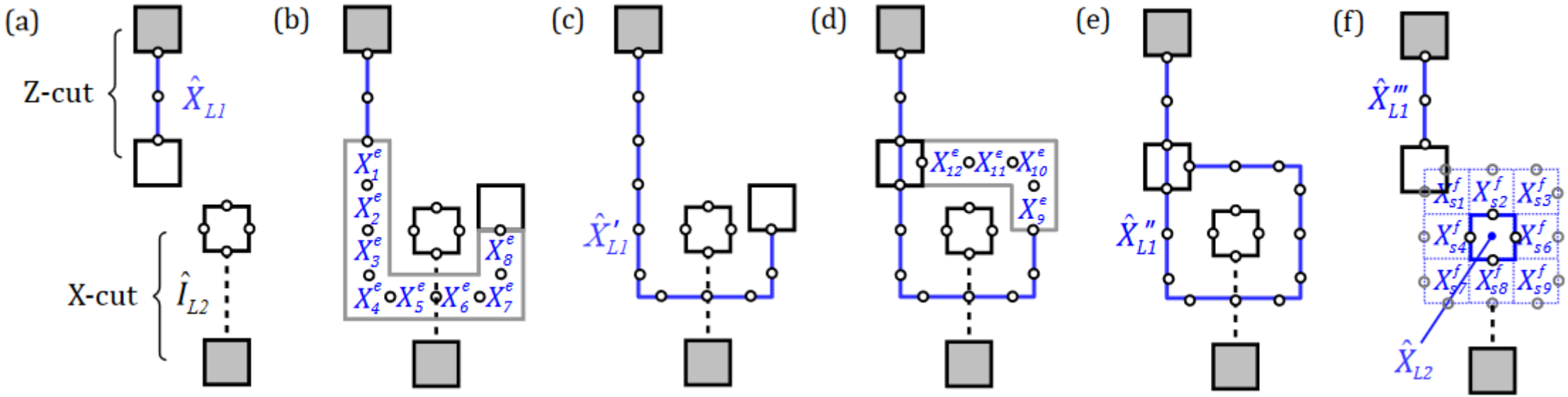
CNOT gate in 2d surface code

Gottesman-Knill

- Logical CNOT gate can be implemented in 2d surface code by certain braiding transformations, which involve moving qubits around on the surface.

$U(\sigma_x \otimes I \otimes \sigma_y \otimes \cdots) U^\dagger$

(For details, see arXiv:1208.0928 [quant-ph].)



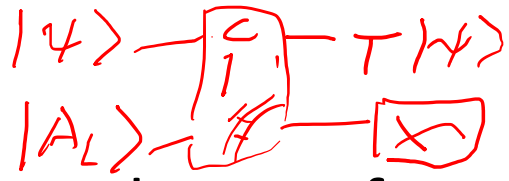
Universal fault–tolerant QC in 2d surface code

- X, Z, Hadamard and CNOT gates not enough for universal QC. **Can be simulated efficiently by a classical computer!** ("Clifford gates", **Gottesman-Knill theorem**)
- Universal computation requires also, e.g., the **T gate**:

$$T|0\rangle = |0\rangle, \qquad T|1\rangle = e^{i\pi/4}|1\rangle.$$

- Implementation of T gates in 2d surface code requires "**magic state distillation**". High fidelity copies of "magic states" of the form

$$|A\rangle_L = \frac{1}{\sqrt{2}}\left(|0\rangle_L + e^{\frac{i\pi}{4}}|1\rangle_L\right)$$

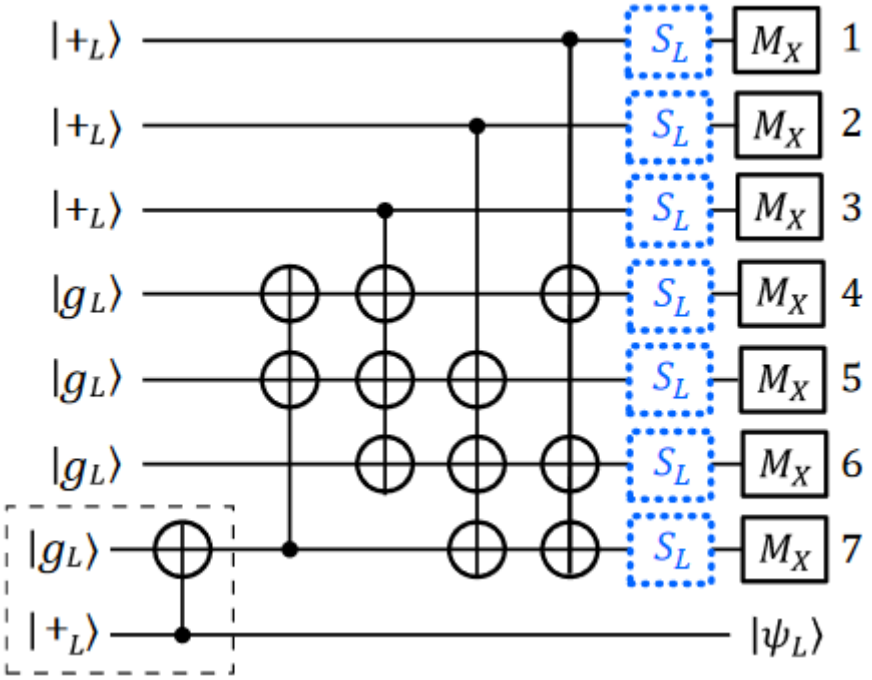


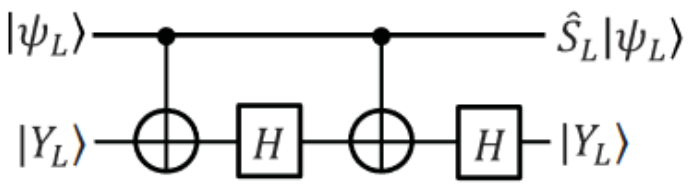
are prepared by a special distillation process from many less perfect few-qubit states, and then used to implement the logical T gate.

- It is, in fact, this distillation process, which may cause most overhead. (However, see arXiv:1905.06903 [quant-ph].)

Some details on the T gate implementation

- The implementation of T gate requires several steps:
  1. Production of $|Y\rangle_L = \frac{1}{\sqrt{2}}(|0\rangle_L + i|1\rangle_L)$ on single data qubits by native X-rotations.
  2. State distillation of high-fidelity versions of $|Y\rangle_L$ on surface code qubits. Each distillation step uses several lower fidelity copies of $|Y\rangle_L$. Error rate $p \mapsto 7p^3$ at each step.
  3. Implementation of logical S gate using high fidelity $|Y\rangle_L$ states.
  4. State distillation of $|A\rangle_L$ using logical S gates. Error rate $p \mapsto 35p^3$ at each step.
  5. Implementation of logical T gate using the $|A\rangle_L$ magic state and logical S gate.





[1907.09528] Topological and subsystem codes on low-degree graphs with flag qubits (arxiv.org)

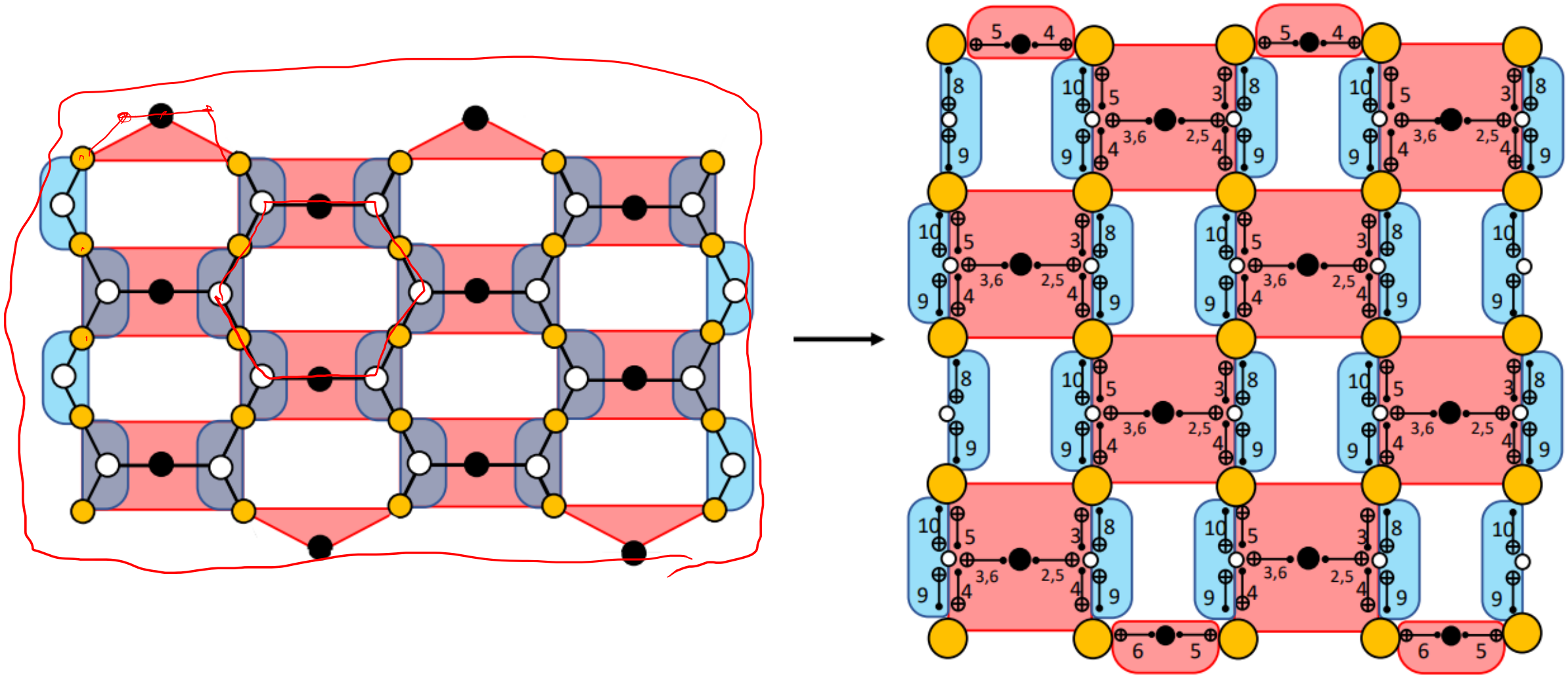


FIG. 2. The left of the figure corresponds to the actual layout of the $d = 5$ heavy hexagon code which encodes one logical qubit. The data qubits are represented by yellow vertices, white vertices are the flag qubits and dark vertices represent the ancilla to measure the X-type gauge generators (red areas) and the Z-type gauge generators (blue areas). In the bulk, products of the two Z-type gauge generators at each white face forms a Z-type stabilizer. The right of the figure provides a circuit illustration of the heavy hexagon code with the scheduling of the CNOT gates used the measure the X-type and Z-type gauge generators.

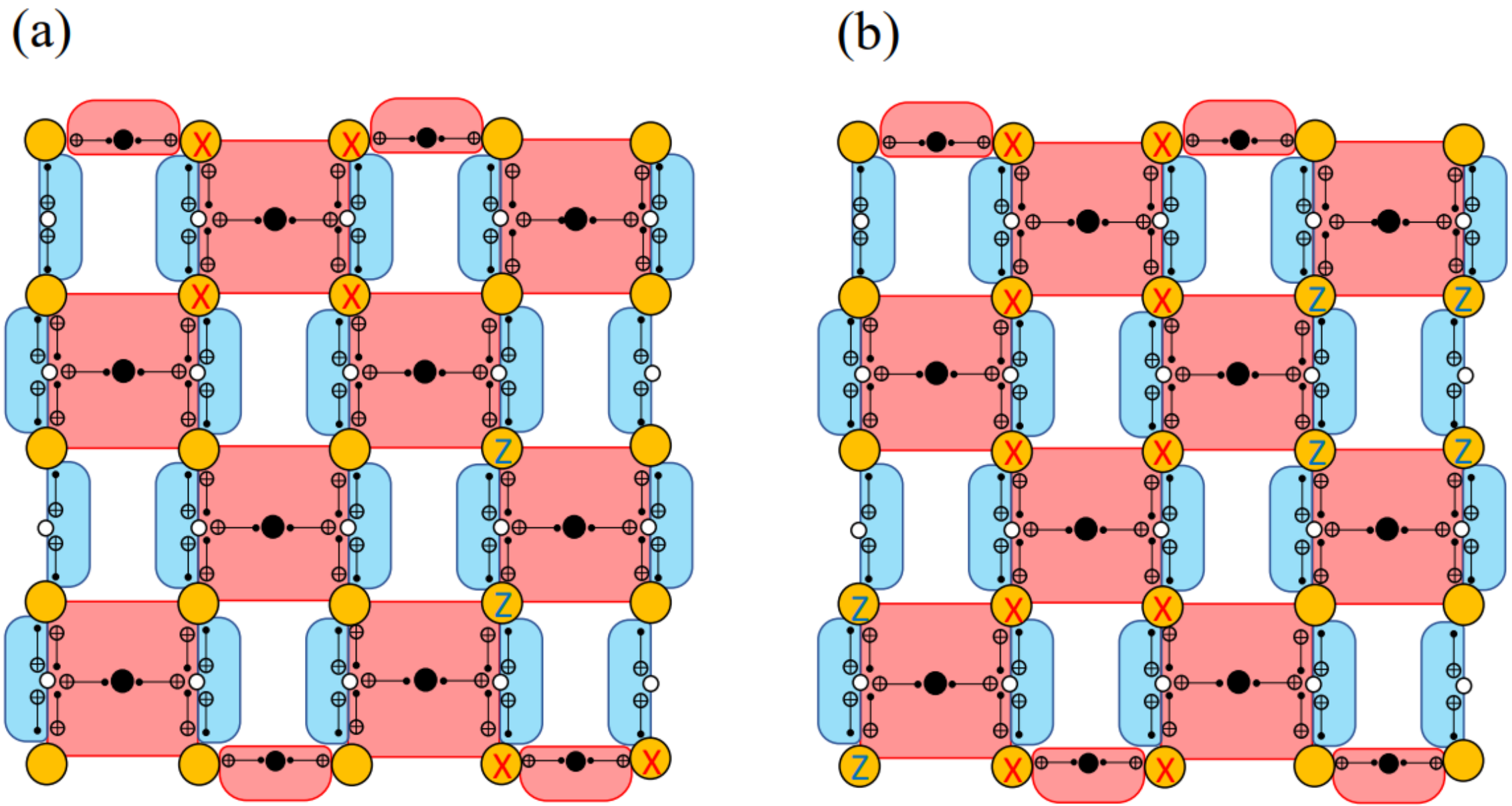


FIG. 3. (a) Gauge generators: weight-four X-type in the bulk, weight-two X-type on the upper and lower boundaries, and weight-two Z-type. (b) Stabilizer operators: a two-column vertical strip with X-type, weight-four Z-type in the bulk, and weight-two Z-type on the left and right boundaries.

The gauge group of the heavy hexagon code is

$$\mathcal{G} = \langle Z_{i,j}Z_{i+1,j},\ X_{i,j}X_{i,j+1}X_{i+1,j}X_{i+1,j+1},\ X_{1,2m-1}X_{1,2m},\ X_{d,2m}X_{d,2m+1}\rangle$$

The stabilizer group which specifies the logical subspace $\mathcal{H}_{\mathcal{L}}$ is the center of the gauge group or, explicitly,

$$\mathcal{S} = \langle Z_{i,j}Z_{i,j+1}Z_{i+1,j}Z_{i+1,j+1},\ Z_{2m,d}Z_{2m+1,d},\ Z_{2m-1,1}Z_{2m,1},\ \prod_i X_{i,j}X_{i,j+1}\rangle \quad (2)$$

In general, a distance d version of the code will have d data qubits along each row and each column of the hexagonal lattice so that the code parameters are given by $[\![d^2, 1, d]\!]$. In addition, a distance d implementation of the code requires a total of $\frac{d+1}{2}(d-1)$ syndrome measurement qubits and $d(d-1)$ flag qubits. Hence the total number of qubits in the implementation of the code is $\frac{5d^2-2d-1}{2}$.

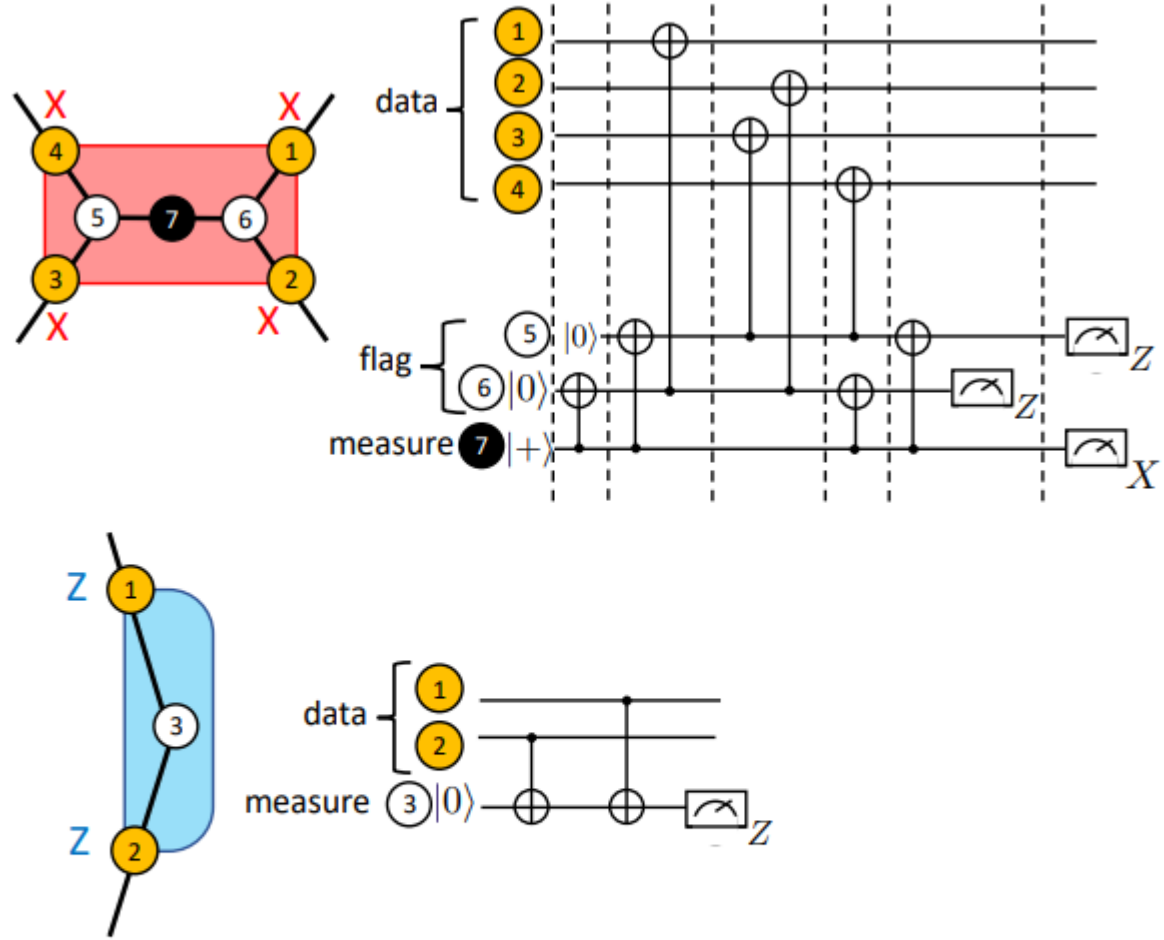
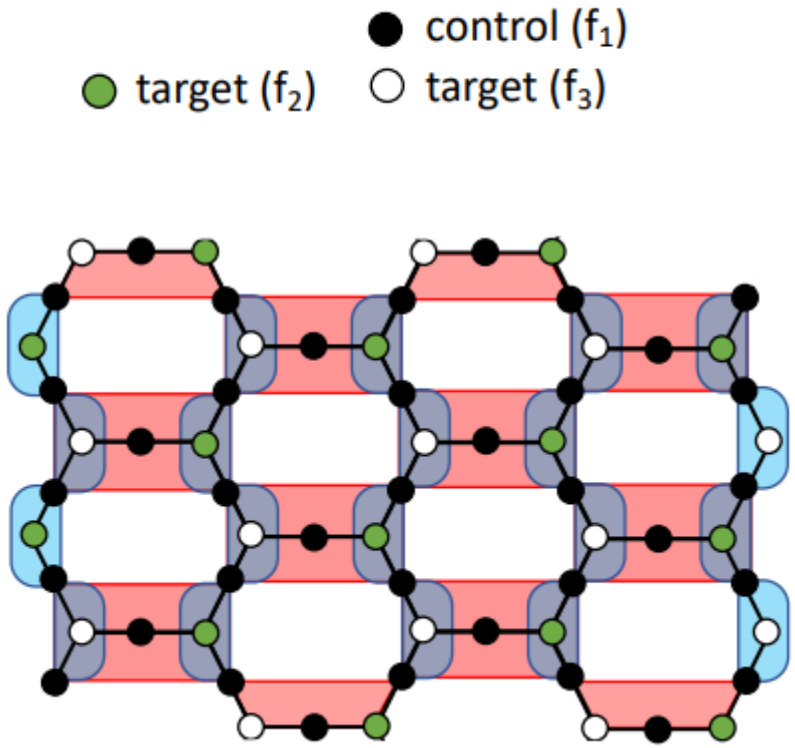
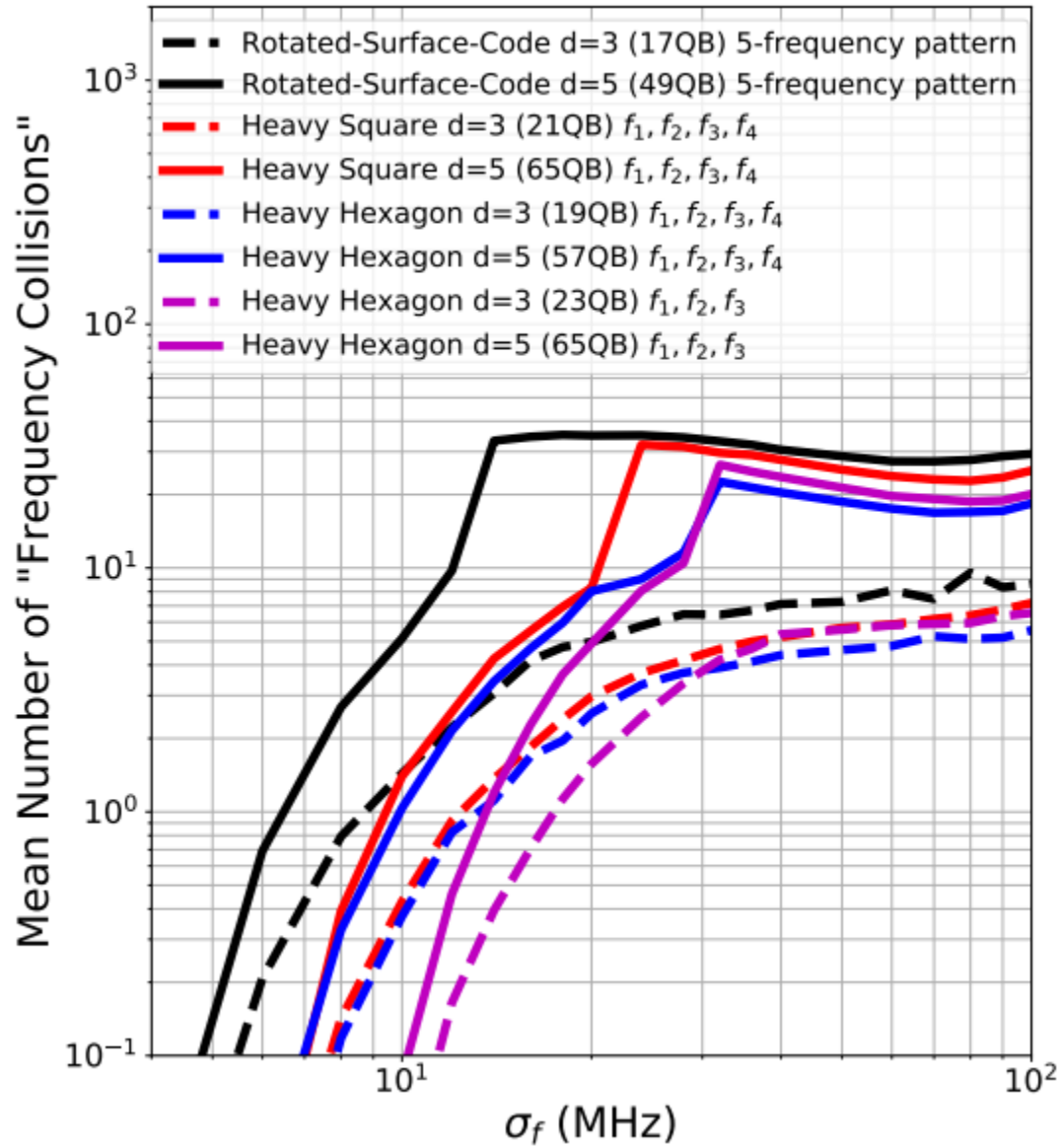


FIG. 4. Circuit to perform the X and Z-type parity measurements of the heavy hexagon code. Two flag qubits (white circles) are used to measure the weight-four X-type gauge generators.


control (f_1)
target (f_2)
target (f_3)


Rotated-Surface-Code d=3 (17QB) 5-frequency pattern
Rotated-Surface-Code d=5 (49QB) 5-frequency pattern
Heavy Square d=3 (21QB) f_1, f_2, f_3, f_4
Heavy Square d=5 (65QB) f_1, f_2, f_3, f_4
Heavy Hexagon d=3 (19QB) f_1, f_2, f_3, f_4
Heavy Hexagon d=5 (57QB) f_1, f_2, f_3, f_4
Heavy Hexagon d=3 (23QB) f_1, f_2, f_3
Heavy Hexagon d=5 (65QB) f_1, f_2, f_3
Mean Number of "Frequency Collisions"
10^3
10^2
10^1
10^0
10^{-1}
10^1
10^2
σ_f (MHz)

References

Quantum error correction intro: arXiv:1907.11157
Subsystem codes: arXiv:quant-ph/0508131, arXiv:quant-ph/0506023
Surface codes: arXiv:1208.0928
Flag qubits: arXiv:1705.02329
Heavy hexagon code: arXiv:1907.09528