Topological Subsystem Codes with High-Weight Stabilizers and Moderate Thresholds

Ben Criger [1 2] Barbara Terhal [1]

[1]JARA IQI, RWTH Aachen

[2]Qutech, TU Delft

August 18, 2016

Introduction

Multiple Two-Qubit Gate Implementations

PHYSICAL REVIEW A **75**, 032329 (2007)

Quantum-information processing with circuit quantum electrodynamics

Alexandre Blais,[1,2] Jay Gambetta,[1] A. Wallraff,[1,3] D. I. Schuster,[1] S. M. Girvin,[1] M. H. Devoret,[1] and R. J. Schoelkopf[1]
[1]*Departments of Applied Physics and Physics, Yale University, New Haven, Connecticut 06520, USA*
[2]*Département de Physique et Regroupement Québécois sur les Matériaux de Pointe, Université de Sherbrooke, Sherbrooke, Québec, Canada J1K 2R1*
[3]*Department of Physics, ETH Zurich, CH-8093 Zürich, Switzerland*
(Received 30 November 2006; published 22 March 2007)

We theoretically study single and two-qubit dynamics in the circuit QED architecture. We focus on the current experimental design [Wallraff *et al.*, Nature (London) **431**, 162 (2004); Schuster *et al.*, *ibid.* **445**, 515 (2007)] in which superconducting charge qubits are capacitively coupled to a single high-Q superconducting coplanar resonator. In this system, logical gates are realized by driving the resonator with microwave fields. Advantages of this architecture are that it allows for multiqubit gates between non-nearest qubits and for the realization of gates in parallel, opening the possibility of fault-tolerant quantum computation with superconduting circuits. In this paper, we focus on one- and two-qubit gates that do not require moving away from the charge-degeneracy "sweet spot." This is advantageous as it helps to increase the qubit dephasing time and does not require modification of the original circuit QED. However, these gates can, in some cases, be slower than those that do not use this constraint. Five types of two-qubit gates are discussed, these include gates based on virtual photons, real excitation of the resonator, and a gate based on the geometric phase. We also point out the importance of selection rules when working at the charge degeneracy point.

DOI: 10.1103/PhysRevA.75.032329 PACS number(s): 03.67.Lx, 73.23.Hk, 74.50.+r, 32.80.−t

Multiple Two-Qubit Gate Implementations

PHYSICAL REVIEW A **75**, 032329 (2007)

Quantum-information processing with circuit quantum electrodynamics

Alexandre Blais,[1,2] Jay Gambetta,[1] A. Wallraff,[1,3] D. I. Schuster,[1] S. M. Girvin,[1] M. H. Devoret,[1] and R. J. Schoelkopf[1]
[1]*Departments of Applied Physics and Physics, Yale University, New Haven, Connecticut 06520, USA*
[2]*Département de Physique et Regroupement Québécois sur les Matériaux de Pointe, Université de Sherbrooke, Sherbrooke, Québec, Canada J1K 2R1*
[3]*Department of Physics, ETH Zurich, CH-8093 Zürich, Switzerland*

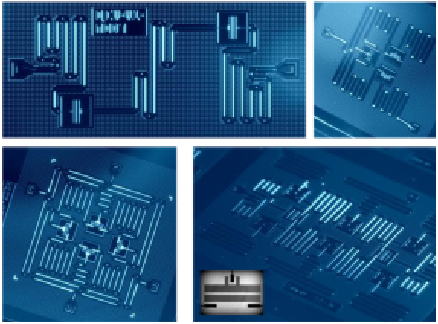
(Received 30 November 2006; published 22 March 2007)

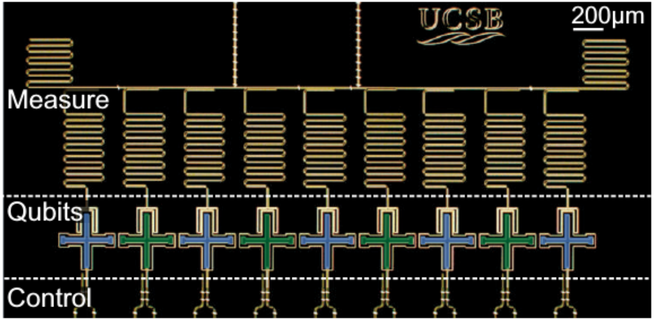
We theoretically study single and two-qubit dynamics in the circuit QED architecture. We focus on the current experimental design [Wallraff *et al.*, Nature (London) **431**, 162 (2004); Schuster *et al.*, *ibid.* **445**, 515 (2007)] in which superconducting charge qubits are [illegible] coplanar resonator. In this system, logical gates are [illegible] Advantages of this architecture are that it allows fo[illegible] realization of gates in parallel, opening the possibil[illegible] duting circuits. In this paper, we focus on one- and two-qubit gates that do not require moving away from the charge-degeneracy "sweet spot." This is advantageous as it helps to increase the qubit dephasing time and does not require modification of the original circuit QED. However, these gates can, in some cases, be slower than those that do not use this constraint. Five types of two-qubit gates are discussed, these include gates based on virtual photons, real excitation of the resonator, and a gate based on the geometric phase. We also point out the importance of selection rules when working at the charge degeneracy point.

Five types of two-qubit gates

DOI: 10.1103/PhysRevA.75.032329 PACS number(s): 03.67.Lx, 73.23.Hk, 74.50.+r, 32.80.−t

Fidelity Vs. Range

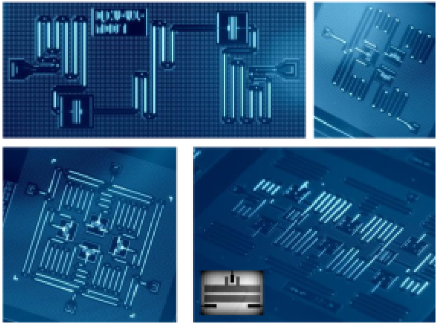


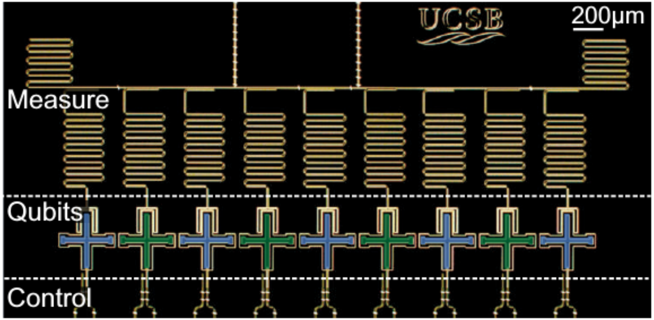


ArXiv: 1510.04375
Indirect Coupling
$F \sim 94\%$

ArXiv: 1411.7403
Direct Coupling
$F \sim 99\%$

Fidelity Vs. Range





ArXiv: 1510.04375
Indirect Coupling
$F \sim 94\%$

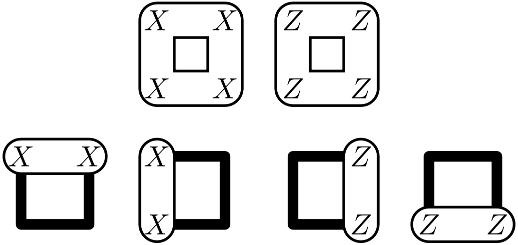
ArXiv: 1411.7403
Direct Coupling
$F \sim 99\%$

Can we use short- and long-range gates as part of the same layout?

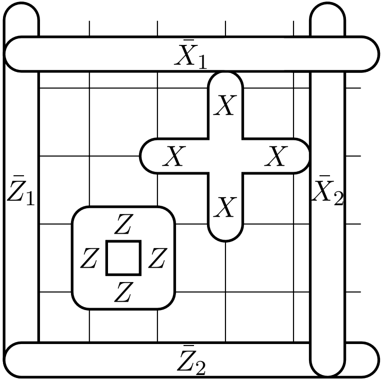
The $[\![4,\ 2,\ 2]\!]$-Toric Code

Introducing Short-Range Gates by Concatenation

ArXiv:1604.04062, soon to appear in QIC.

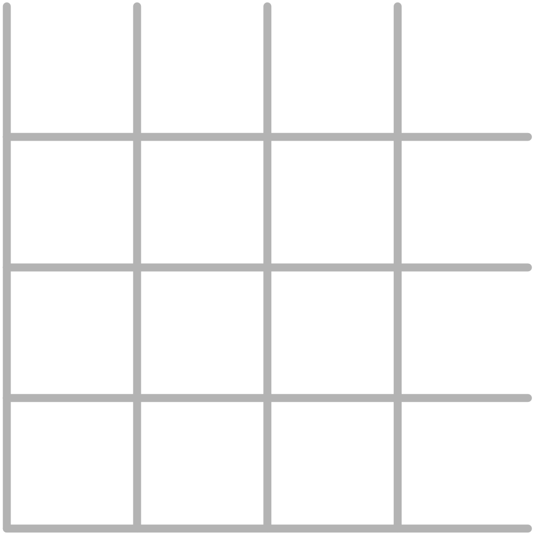


short-range: the $[\![4, 2, 2]\!]$ code



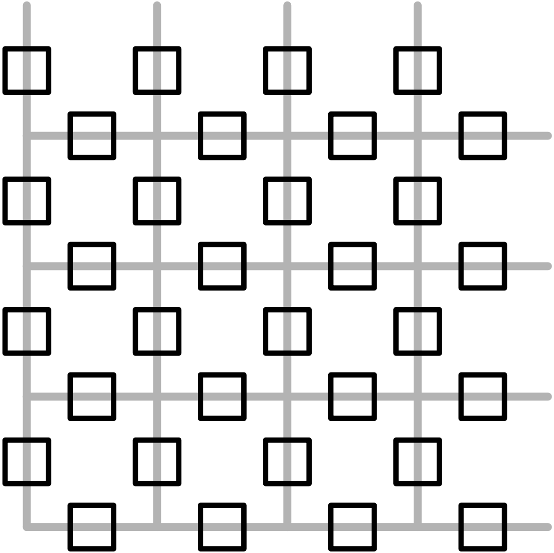
long-range: toric code

The $[\![4,\ 2,\ 2]\!]$-Toric Code



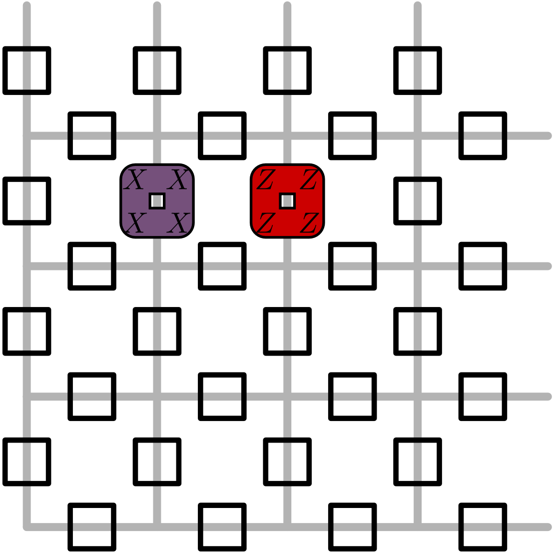
- begin with a square lattice

The $[\![4,\ 2,\ 2]\!]$-Toric Code



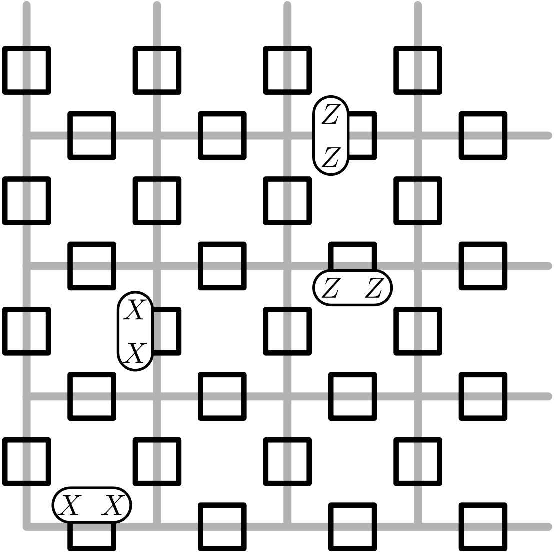
- begin with a square lattice
- replace physical qubits on edges with squares (physical qubits now on vertices, $n \to 4n$)

The $[\![4,2,2]\!]$-Toric Code



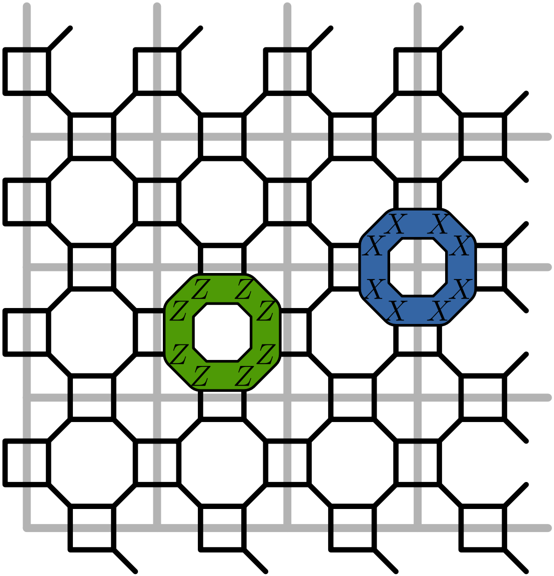
- begin with a square lattice
- replace physical qubits on edges with squares (physical qubits now on vertices, $n \to 4n$)
- local square stabilizers

The $[\![4, 2, 2]\!]$-Toric Code



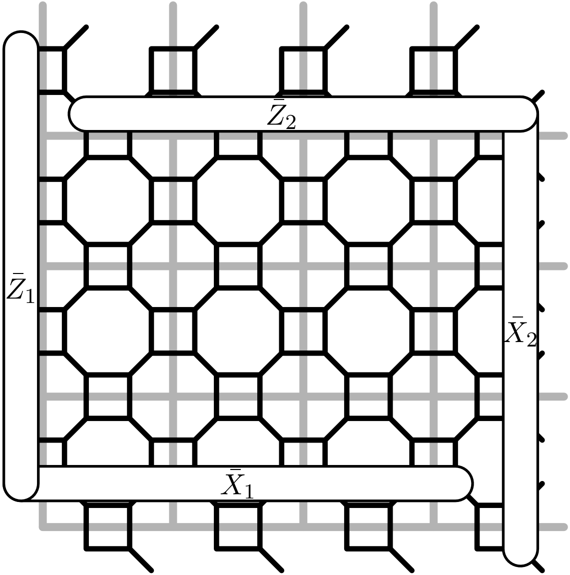
- begin with a square lattice
- replace physical qubits on edges with squares (physical qubits now on vertices, $n \to 4n$)
- local square stabilizers
- local logical operators

The $[\![4,\ 2,\ 2]\!]$-Toric Code



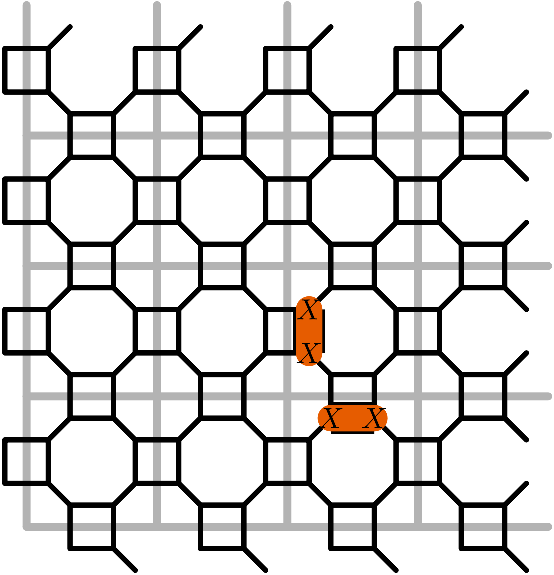
- begin with a square lattice
- replace physical qubits on edges with squares (physical qubits now on vertices, $n \to 4n$)
- local square stabilizers
- local logical operators
- toric code stabilizers as products of logicals (now weight-eight)

The $[\![4, 2, 2]\!]$-Toric Code



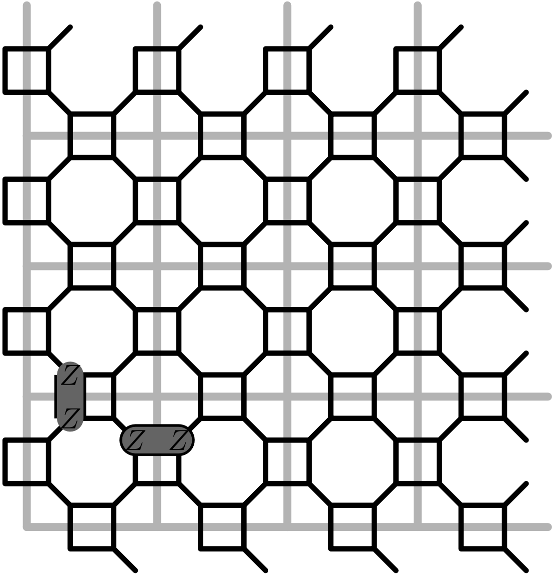
- begin with a square lattice
- replace physical qubits on edges with squares (physical qubits now on vertices, $n \to 4n$)
- local square stabilizers
- local logical operators
- toric code stabilizers as products of logicals (now weight-eight)
- toric code logicals as products of logicals ($d \to 2d$)

The $[\![4,2,2]\!]$-Toric Code



- begin with a square lattice
- replace physical qubits on edges with squares (physical qubits now on vertices, $n \to 4n$)
- local square stabilizers
- local logical operators
- toric code stabilizers as products of logicals (now weight-eight)
- toric code logicals as products of logicals ($d \to 2d$)
- remaining logicals become gauge operators

The $[\![4,\,2,\,2]\!]$-Toric Code

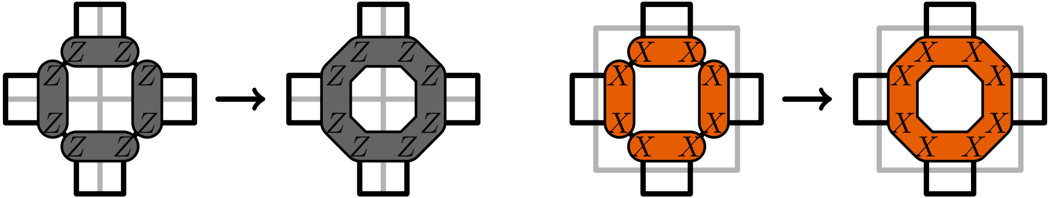


- begin with a square lattice
- replace physical qubits on edges with squares (physical qubits now on vertices, $n \to 4n$)
- local square stabilizers
- local logical operators
- toric code stabilizers as products of logicals (now weight-eight)
- toric code logicals as products of logicals ($d \to 2d$)
- remaining logicals become gauge operators
- looks a lot like a color code ...

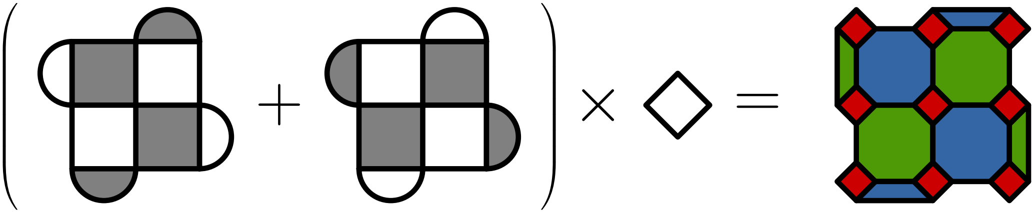
Distraction: Color Codes

Color Codes as Concatenated Codes

The gauge operators from the $[\![4, 2, 2]\!]$-toric code can be encoded into a toric code:

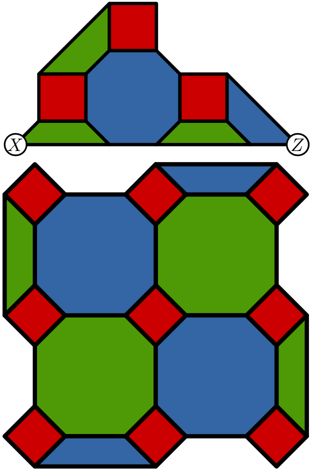


As long as every qubit is on a square, this is a color code:



Triangular Boundary Conditions/Logical Cliffords

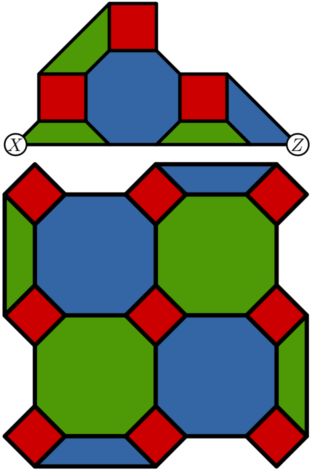
Removing checks from a triangular color code doesn't work:



- Extra weight-1 logical operators appear at corners.

Triangular Boundary Conditions/Logical Cliffords

Removing checks from a triangular color code doesn't work:

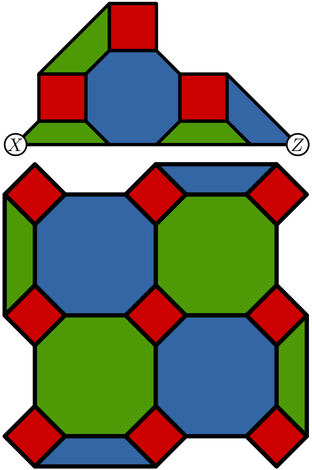


- Extra weight-1 logical operators appear at corners.

- Concatenated code is not self-dual $\rightarrow$ no transversal logical H

Triangular Boundary Conditions/Logical Cliffords

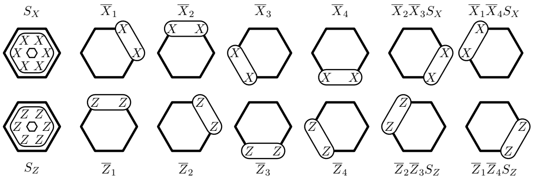
Removing checks from a triangular color code doesn't work:

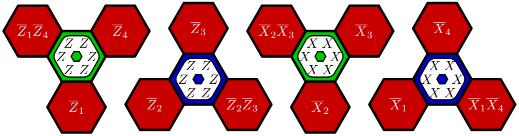


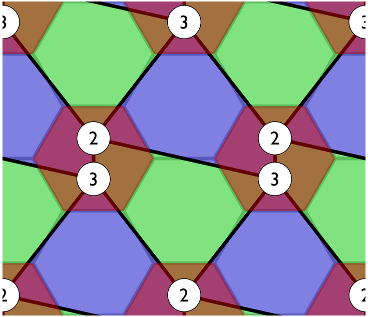
- Extra weight-1 logical operators appear at corners.
- Concatenated code is not self-dual $\rightarrow$ no transversal logical H
- Concatenated code has even n $\rightarrow$ no transversal logical P

The Hexagonal Color Code

We can also concatenate the $[\![6,\ 4,\ 2]\!]$ code with two toric codes to get a color code:

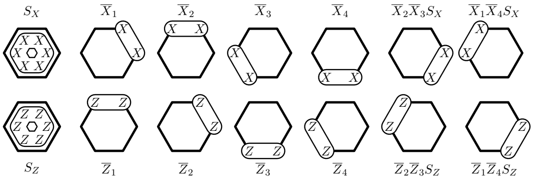


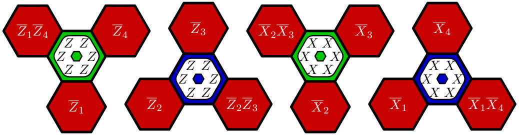


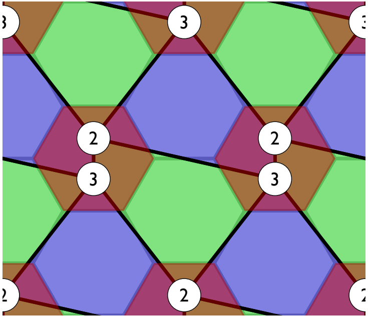


The Hexagonal Color Code

We can also concatenate the $[\![6,\,4,\,2]\!]$ code with two toric codes to get a color code:



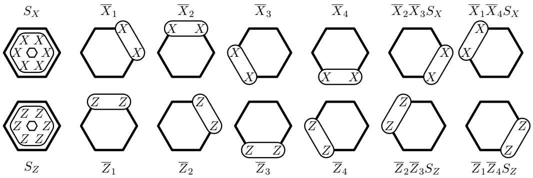


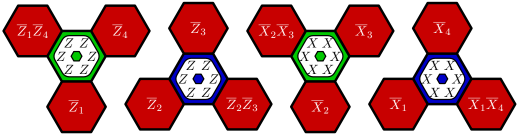


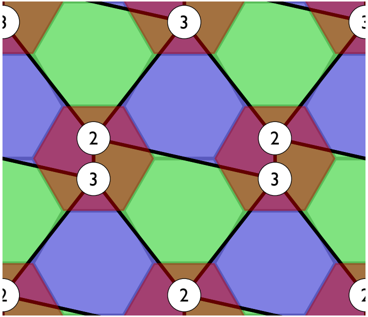
- Equivalence between toric and color codes was already known (ArXiv:1007.4601, 1503.02065)

The Hexagonal Color Code

We can also concatenate the $[\![6,\,4,\,2]\!]$ code with two toric codes to get a color code:



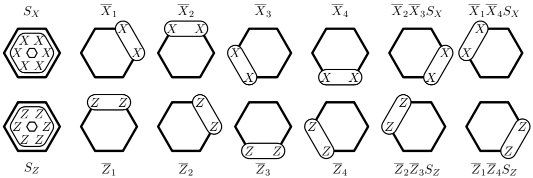


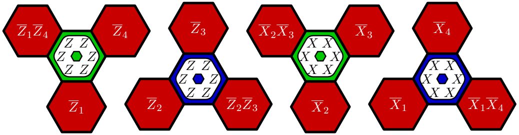


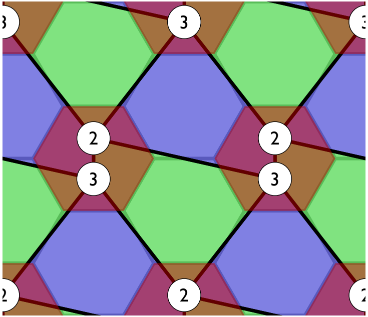
- Equivalence between toric and color codes was already known (ArXiv:1007.4601, 1503.02065)
- This construction is very simple

The Hexagonal Color Code

We can also concatenate the $[\![6,\,4,\,2]\!]$ code with two toric codes to get a color code:



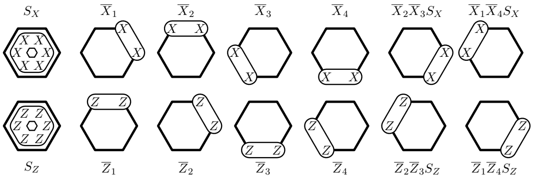


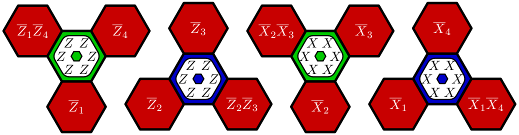


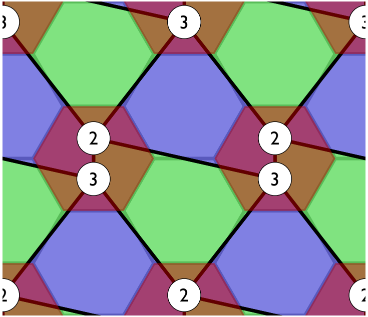
- Equivalence between toric and color codes was already known (ArXiv:1007.4601, 1503.02065)
- This construction is very simple
- Likely extends to other lattices/“bottom codes”

The Hexagonal Color Code

We can also concatenate the $[\![6,4,2]\!]$ code with two toric codes to get a color code:



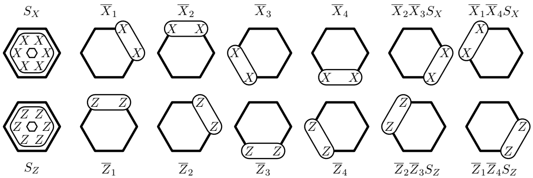


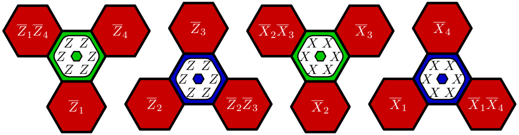


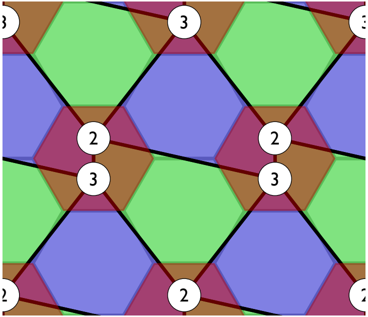
- Equivalence between toric and color codes was already known (ArXiv:1007.4601, 1503.02065)
- This construction is very simple
- Likely extends to other lattices/“bottom codes”
- Maybe extends to higher dimensions

The Hexagonal Color Code

We can also concatenate the $[\![6,\ 4,\ 2]\!]$ code with two toric codes to get a color code:



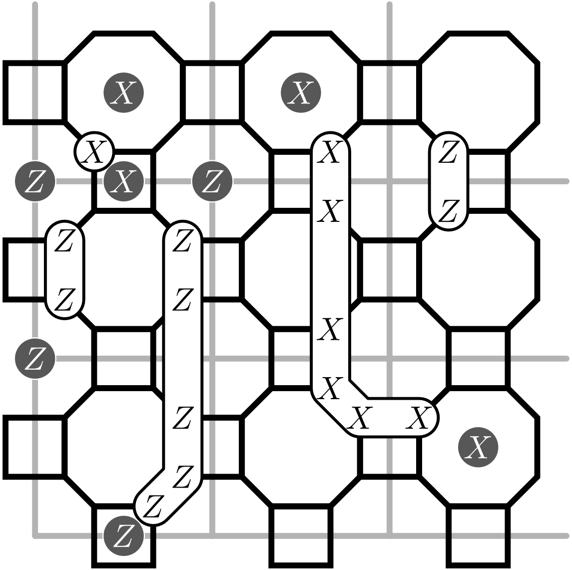




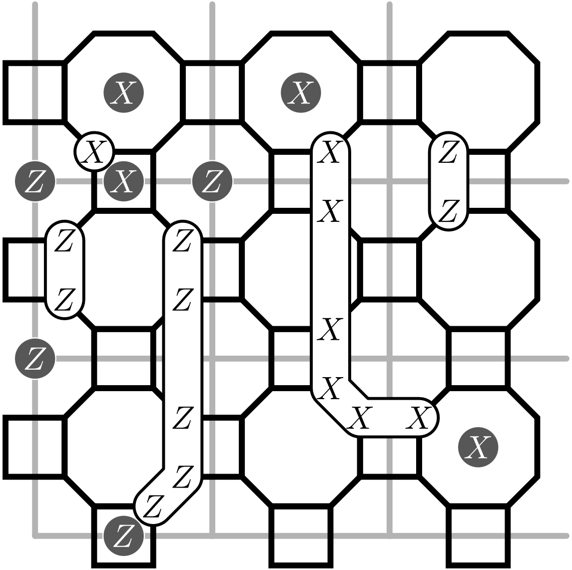
- Equivalence between toric and color codes was already known (ArXiv:1007.4601, 1503.02065)
- This construction is very simple
- Likely extends to other lattices/“bottom codes”
- Maybe extends to higher dimensions
- Small chance it can help with decoding

Back To Work: Decoding the $[\![4, 2, 2]\!]$-Toric Code

IID X/Z Errors

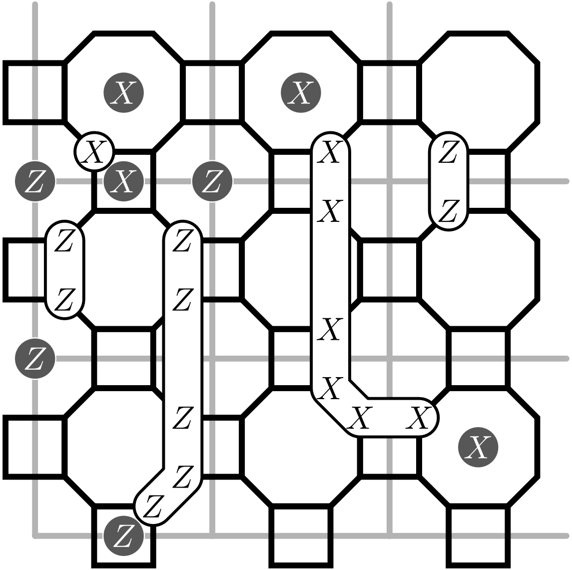


IID X/Z Errors



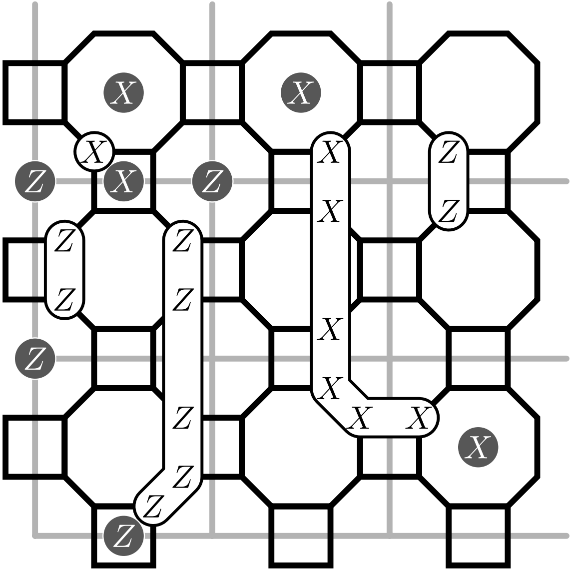
- $[\![4,\ 2,\ 2]\!]$-toric code is CSS

IID X/Z Errors



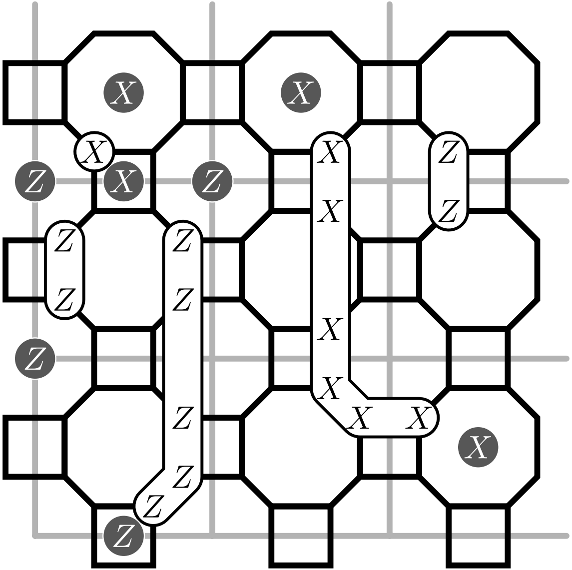
- $[\![4,\ 2,\ 2]\!]$-toric code is CSS
- Syndromes appear at endpoints of a chain, as in the toric code

IID X/Z Errors



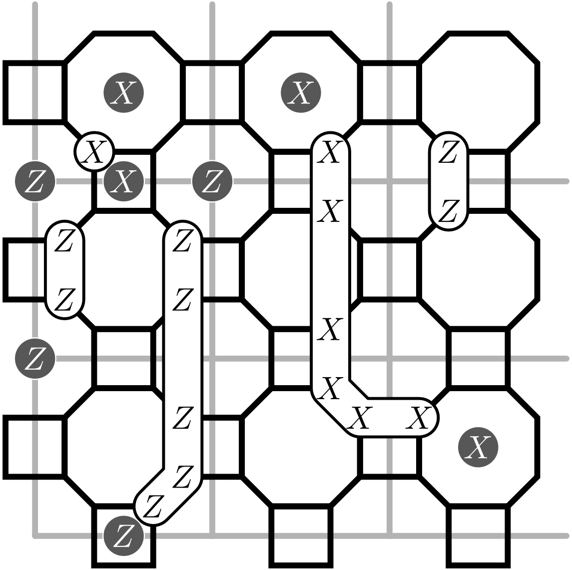
- $[\![4,\ 2,\ 2]\!]$-toric code is CSS
- Syndromes appear at endpoints of a chain, as in the toric code
- Weight of an error still length-like:

IID X/Z Errors



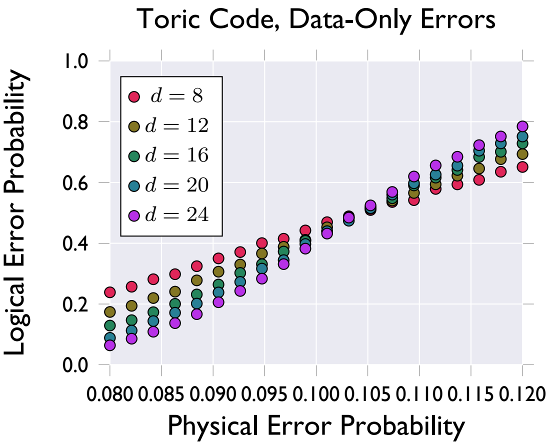
- $[\![4,\,2,\,2]\!]$-toric code is CSS
- Syndromes appear at endpoints of a chain, as in the toric code
- Weight of an error still length-like:
  - chains between octagons use a Manhattan metric

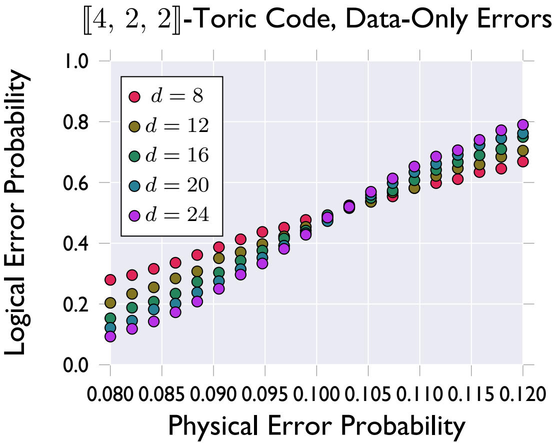
IID X/Z Errors



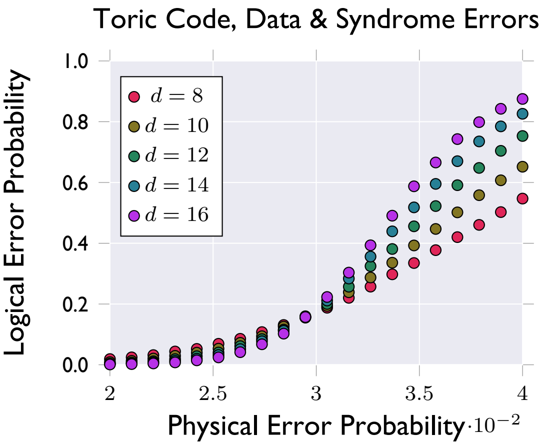
- $[\![4,\ 2,\ 2]\!]$-toric code is CSS
- Syndromes appear at endpoints of a chain, as in the toric code
- Weight of an error still length-like:
  - chains between octagons use a Manhattan metric
  - square-to-octagon (square-to-square) chains use the minimum octagon-to-octagon length plus one(two)

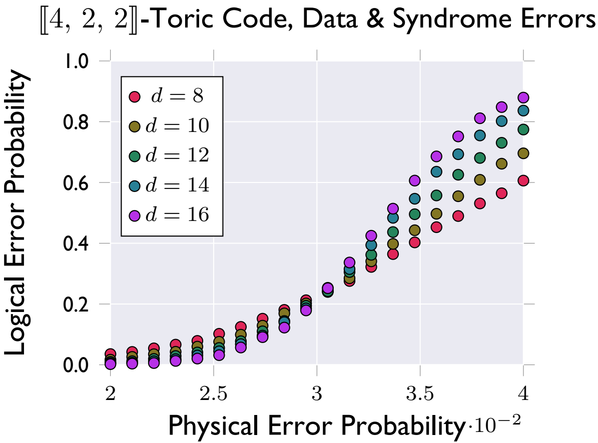
IID X/Z Errors





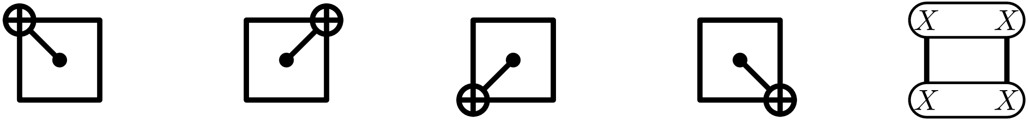
IID X/Z/Stabiliser Errors



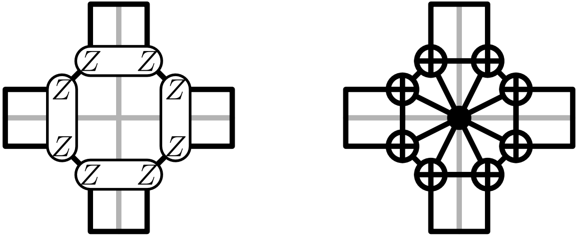


Aside: Gauge Generators Vs. Error Propagation

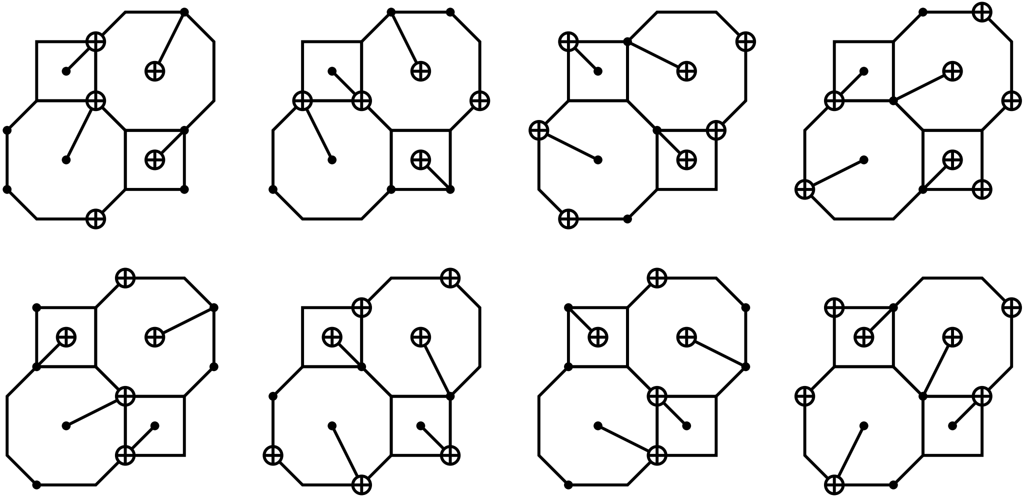
Errors which propagate through CNOT gates on squares can result in gauge operators:



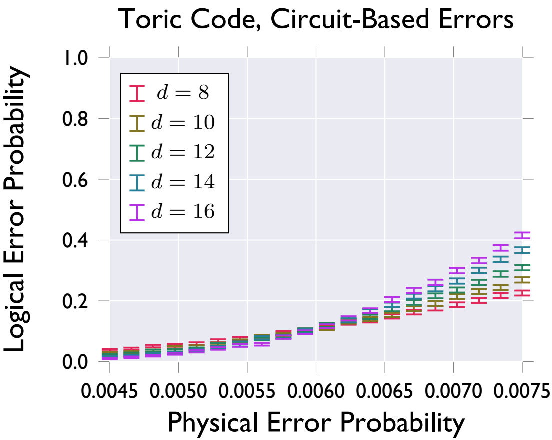
However, octagons whose CNOTs propagate X errors have Z gauge operators on the perimeter, and vice versa:

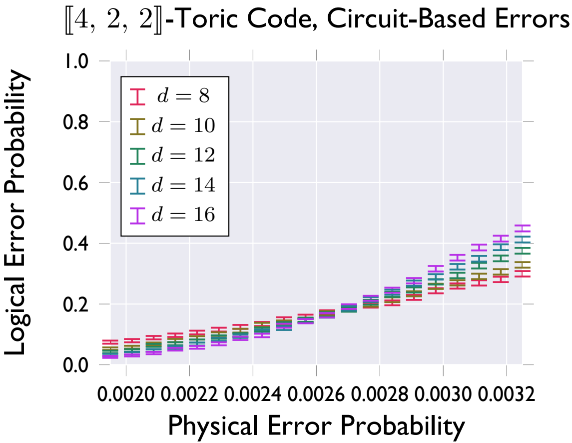


Circuit-Based Errors: Eight-Step Circuit



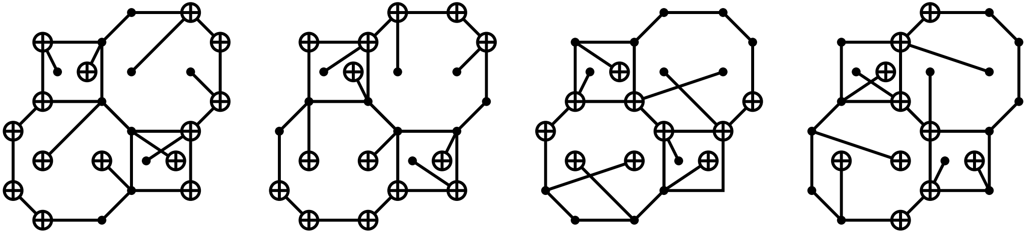
Circuit-Based Errors: Eight-Step Circuit





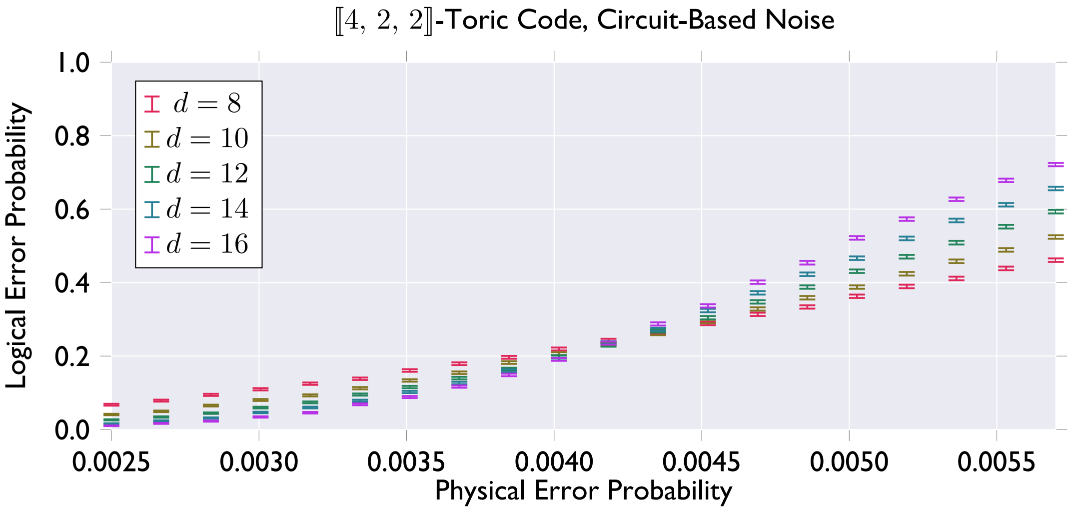
Aside: Bell Pair Ancillas

- There are data qubits sitting idle at every timestep in the eight-step circuit
- All ancilla qubits are being used at all times
- Solution: more ancilla qubits (prepare octagon ancillas in Bell states)

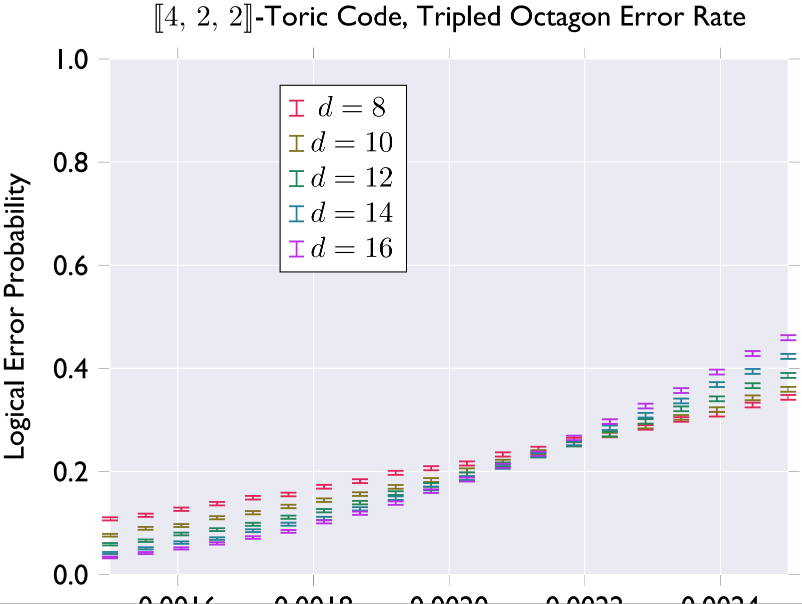


Note: This has the same number of ancilla qubits as the toric code.

Circuit-Based Errors: Four-Step Circuit



The Original Goal



Conclusions

What Have We Learned?

- We can use a level of concatenation to take advantage of high-fidelity short-range gate implementations.
- We can construct color codes by concatenation.
- Large stabilizers (weight > 4) don't destroy FT thresholds.
- Subsystem codes can have high-ish thresholds.
- Unverified high-weight (> 1) ancilla states can increase thresholds, especially where stabilizer weights differ.

What Should We Do?

- Adapt advanced decoding techniques ("diagonal edges") to decoding the $[\![4,\, 2,\, 2]\!]$-toric code.
- Look for more places to insert short-range operations (ancilla preparation?).
- Account for differences in gate time.
- Use high-weight ancillas to extract syndromes from the surface code.
- Prove that all 2D (nD? gauge?) color codes can be constructed by concatenation.
- Find out if color codes can be decoded using schemes for concatenated codes.

Thanks for your attention!