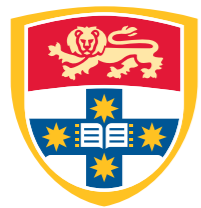


Sparse Quantum Codes from Quantum Circuits

Steve Flammia



THE UNIVERSITY OF
SYDNEY

eQus

ARC CENTRE OF EXCELLENCE FOR
ENGINEERED QUANTUM SYSTEMS

QEC 2014
15 December 2014
ETH Zürich

Joint work with
D Bacon, A W Harrow, and J Shi
arxiv:1411.3334

Quantum Error Correction

- ✱ Quantum error correction allows us to deal with the inevitable presence of noise in a quantum computation
- ✱ Most quantum codes are **stabilizer codes**.
Ex: $n=4$ code that detects any single-qubit Pauli error

4 qubits: 

Stabilizers:

$$S_X = \begin{matrix} X & X \\ X & X \end{matrix}$$

$$S_Z = \begin{matrix} Z & Z \\ Z & Z \end{matrix}$$

Logical qubits:

$$L_X^1 = \begin{matrix} X & X \\ I & I \end{matrix}$$

$$L_Z^1 = \begin{matrix} Z & I \\ Z & I \end{matrix}$$

$$L_X^2 = \begin{matrix} X & I \\ X & I \end{matrix}$$

$$L_Z^2 = \begin{matrix} Z & Z \\ I & I \end{matrix}$$

Subsystem Codes

- Use excess logical qubits as “gauge”, and correct errors only up to transformations on this gauge space.
- Subsystem codes can be **sparser**, implying simpler syndrome measurements, higher thresholds (sometimes)

4 qubits: 

$$S_X = G_X^1 G_X^2$$

$$S_Z = G_Z^1 G_Z^2$$

Gauge:

$$G_X^1 = \begin{matrix} X & I \\ X & I \end{matrix} \quad G_X^2 = \begin{matrix} I & X \\ I & X \end{matrix}$$

$$G_Z^1 = \begin{matrix} Z & Z \\ I & I \end{matrix} \quad G_Z^2 = \begin{matrix} I & I \\ Z & Z \end{matrix}$$

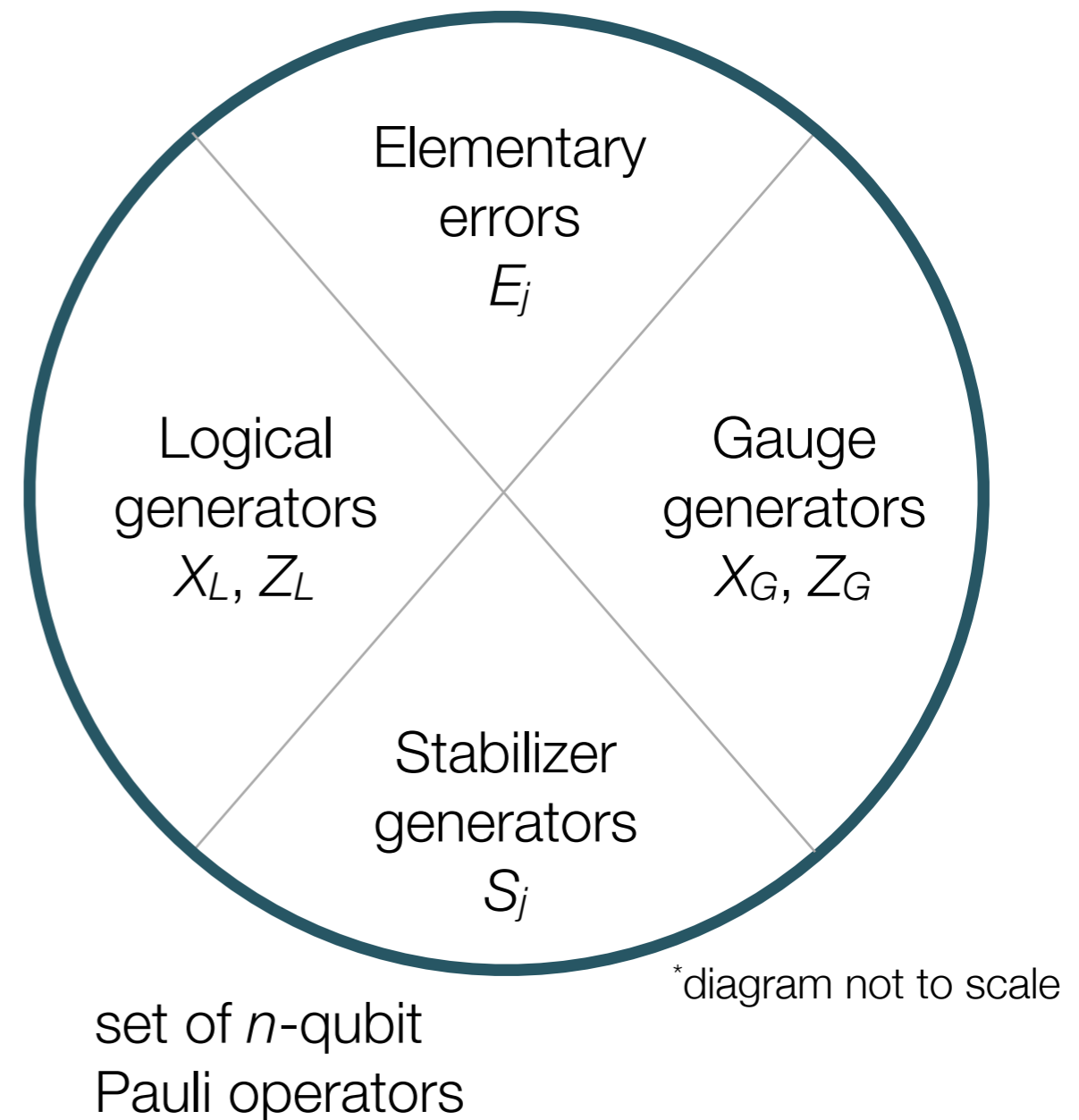
Logical:

$$L_X = \begin{matrix} X & X \\ I & I \end{matrix}$$

$$L_Z = \begin{matrix} Z & I \\ Z & I \end{matrix}$$

The Structure of Subsystem Codes

- ☼ Code is defined by a set of **gauge generators**
- ☼ Center $Z(G)$ of the gauge group is the stabilizer group (for a choice of signs)
- ☼ Logical operators commute with G and permute the code space (normalizer $N(G)$)
- ☼ **Bare** logical operators act trivially on the gauge qubits; **dressed** ones act nontrivially

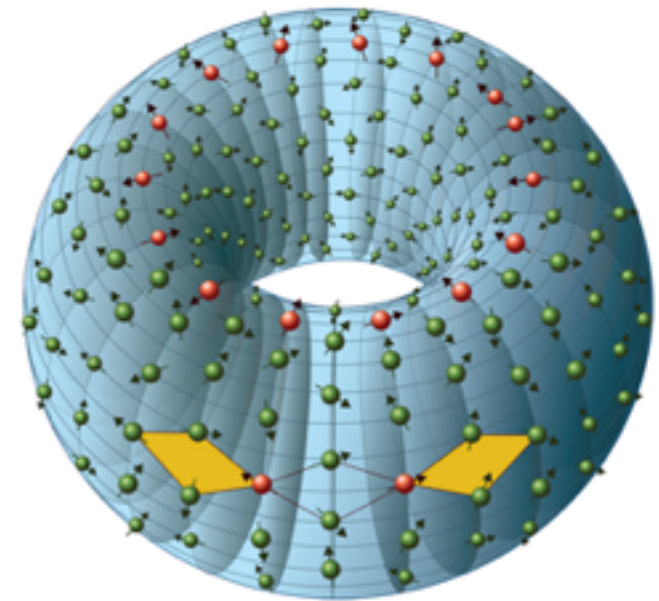


Sparse Codes

- ✱ A code family is called $[n,k,d]$ if it encodes k logical qubits into n physical qubits and can detect any Pauli error of weight $< d$.
- ✱ A code with a given set of gauge generators is called **s-sparse** if:
 - ✱ every gauge generator has weight $\leq s$
 - ✱ every physical qubit is acted on nontrivially by $\leq s$ gauge generators.
 - ✱ ex: row- and column-sparse parity-check matrix
- ✱ A code is called just **sparse** if $s = O(1)$, independent of n

The Importance of Sparse Codes

- ✱ Only at most s qubits need to be measured at a time, instead of $O(n)$
- ✱ \Rightarrow higher thresholds, parallelized architectures, simpler decoding algorithms, FTQC with low overhead
- ✱ Ex: topological codes; LDPC codes
 - ✱ toric code, color codes, hypergraph product codes, ...
- ✱ A **major challenge** is to find sparse quantum codes that perform well, e.g. with $k, d = O(n)$ and **fast decoders**



Main Result

Theorem 1. *Given any $[n_0, k_0, d_0]$ quantum stabilizer code with stabilizer generators of weight $w_1, \dots, w_{n_0-k_0}$, there is an associated $[n, k, d]$ quantum subsystem code whose gauge generators have weight $O(1)$ and where $k = k_0$, $d = d_0$, and $n = O(n_0 + \sum_i w_i)$. This mapping is constructive given the stabilizer generators of the base code.*

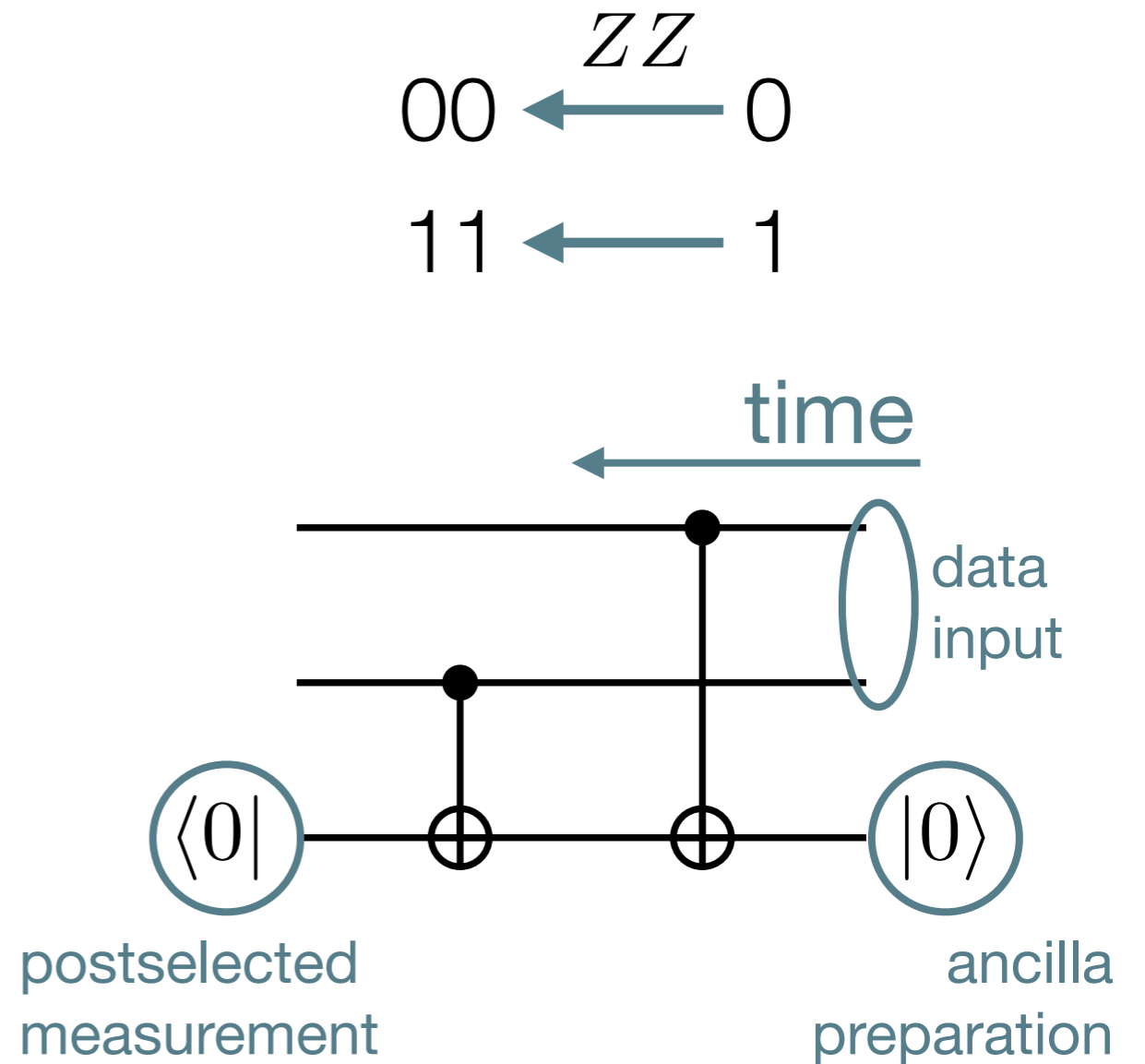
A systematic way to convert *any* stabilizer code into a **sparse** subsystem code with the **same** k and d parameters

The **price** is an increase in the number of physical qubits equal to the sum of the original generator weights

The proof is **hard**, but the construction itself is quite **simple**

From Codes to Circuits to Codes Again...

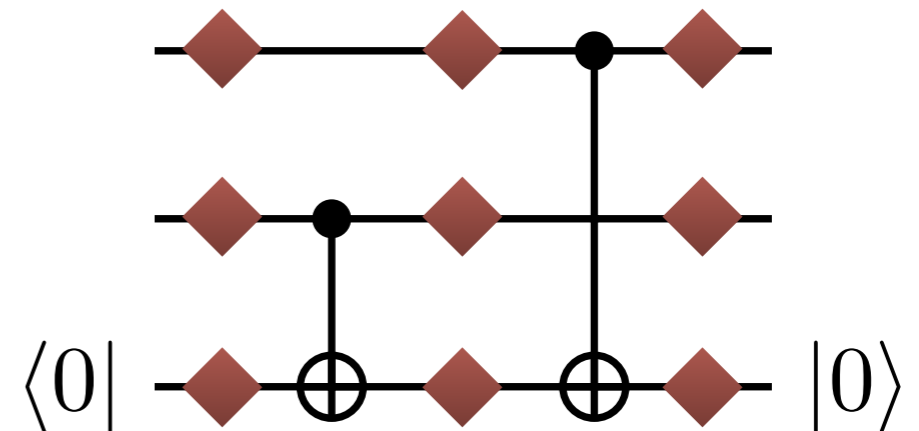
- ✱ Begin with a stabilizer code of your choice
- ✱ Write a quantum circuit for **measuring the stabilizers** of this code.



From Codes to Circuits to Codes Again...

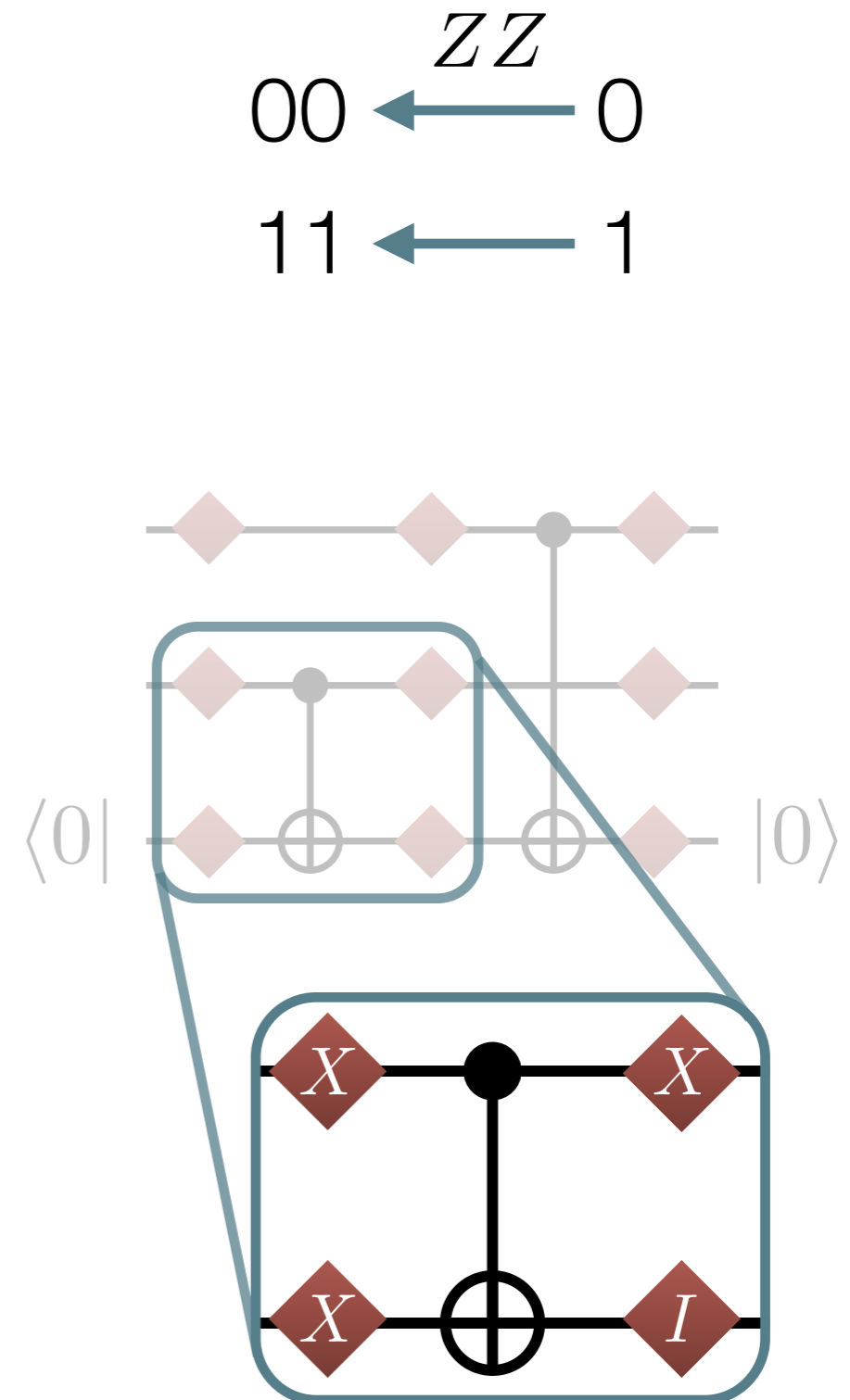
- ✻ Begin with a stabilizer code of your choice
- ✻ Write a quantum circuit for **measuring the stabilizers** of this code.
- ✻ Turn the circuit elements into **input/output qubits**

$$\begin{array}{l} 00 \xleftarrow{ZZ} 0 \\ 11 \xleftarrow{\quad} 1 \end{array}$$



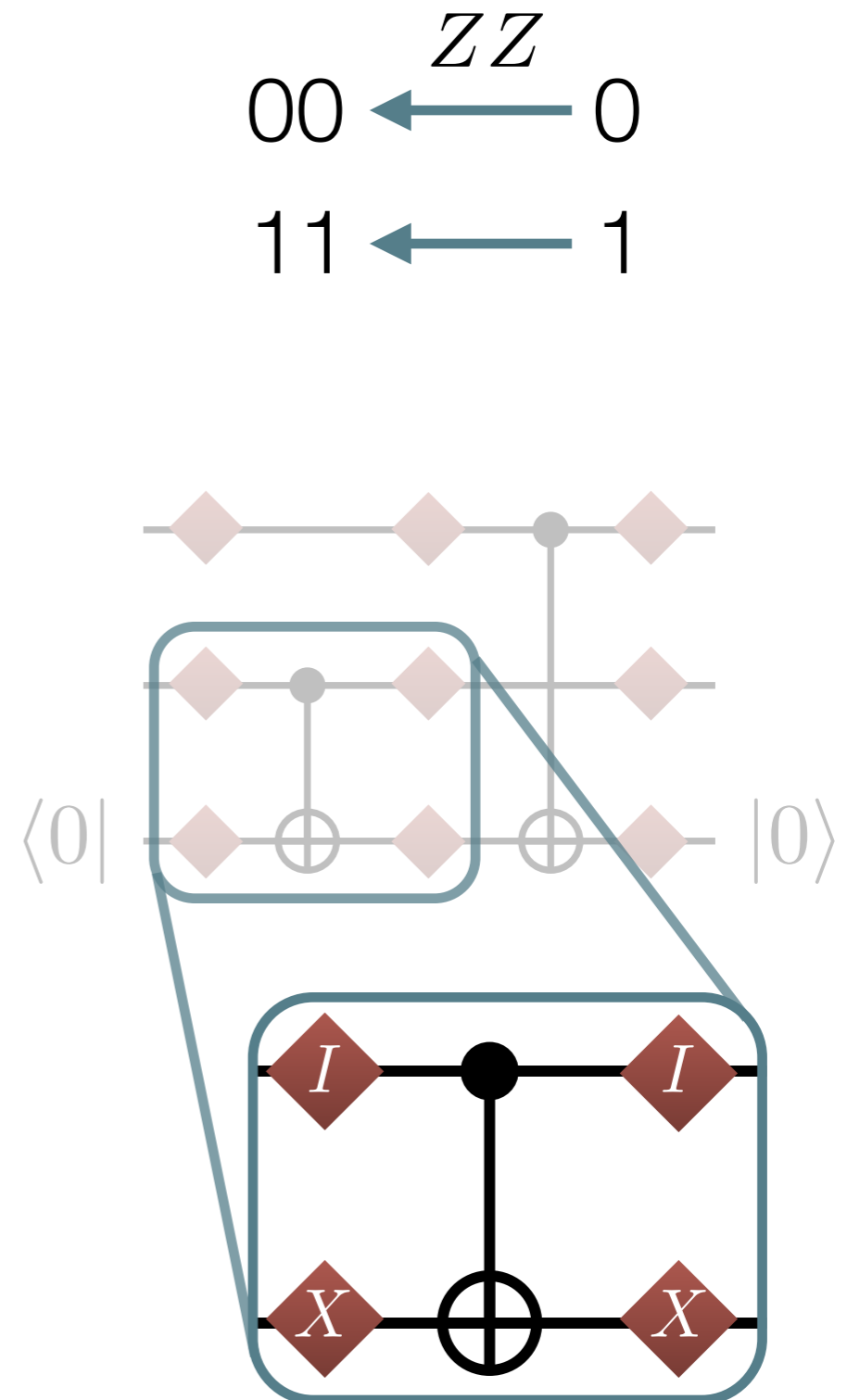
From Codes to Circuits to Codes Again...

- ✱ Begin with a stabilizer code of your choice
- ✱ Write a quantum circuit for **measuring the stabilizers** of this code.
- ✱ Turn the circuit elements into **input/output qubits**
- ✱ Add gauge generators via **Pauli circuit identities.**



From Codes to Circuits to Codes Again...

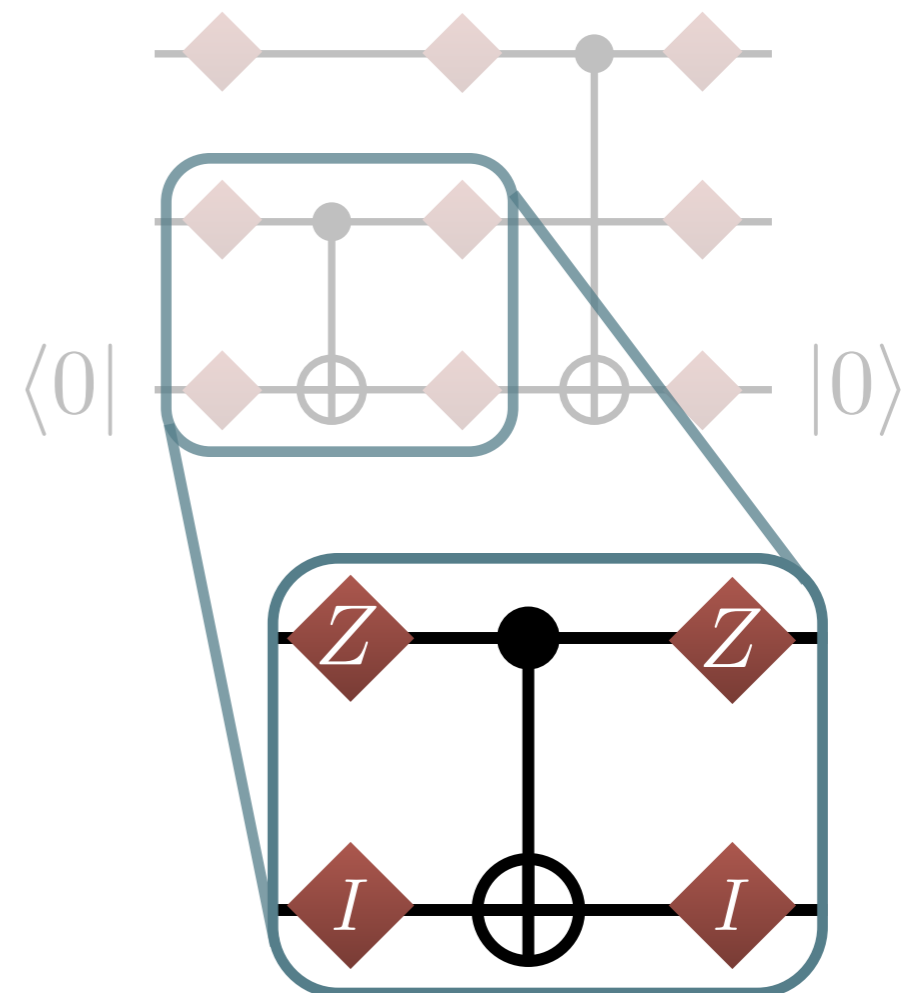
- ✱ Begin with a stabilizer code of your choice
- ✱ Write a quantum circuit for **measuring the stabilizers** of this code.
- ✱ Turn the circuit elements into **input/output qubits**
- ✱ Add gauge generators via **Pauli circuit identities.**



From Codes to Circuits to Codes Again...

- ✱ Begin with a stabilizer code of your choice
- ✱ Write a quantum circuit for **measuring the stabilizers** of this code.
- ✱ Turn the circuit elements into **input/output qubits**
- ✱ Add gauge generators via **Pauli circuit identities.**

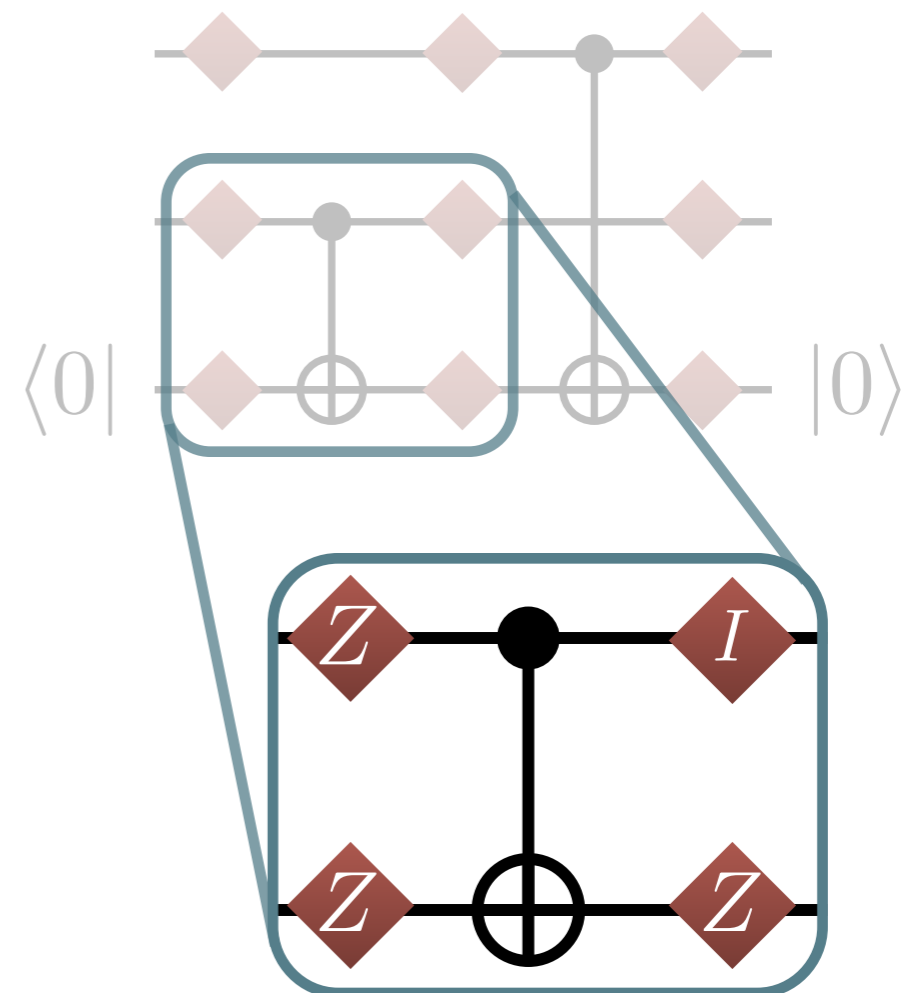
$$\begin{array}{l} 00 \xleftarrow{ZZ} 0 \\ 11 \xleftarrow{\quad} 1 \end{array}$$



From Codes to Circuits to Codes Again...

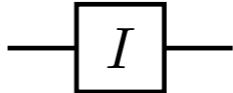

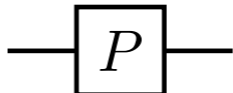
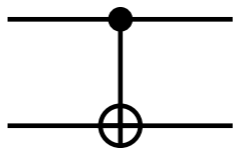
- ✱ Begin with a stabilizer code of your choice
- ✱ Write a quantum circuit for **measuring the stabilizers** of this code.
- ✱ Turn the circuit elements into **input/output qubits**
- ✱ Add gauge generators via **Pauli circuit identities.**

$$\begin{array}{l} 00 \xleftarrow{ZZ} 0 \\ 11 \xleftarrow{\quad} 1 \end{array}$$



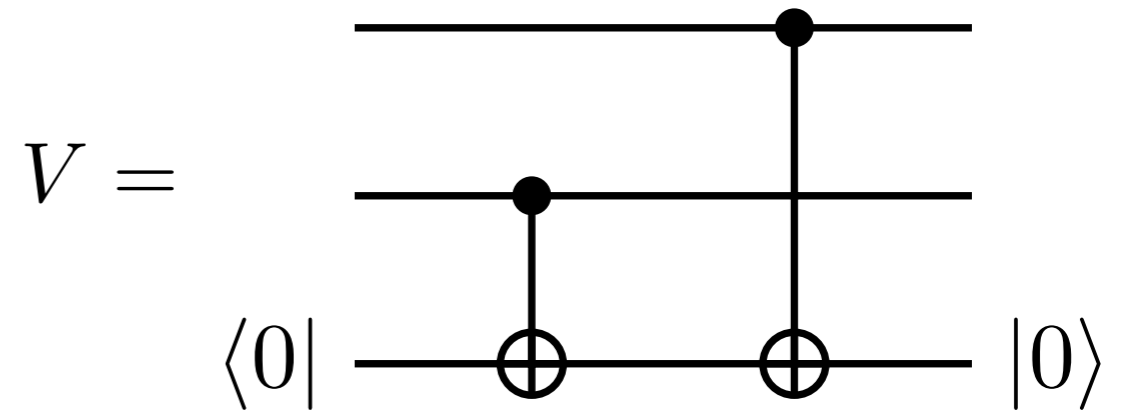
From Codes to Circuits to Codes Again...

- Begin with a stabilizer code of your choice
- Write a quantum circuit for **measuring the stabilizers** of this code.
- Turn the circuit elements into **input/output qubits**
- Add gauge generators via **Pauli circuit identities**
- This defines the code

Circuit element	Gauge generators
	XX, ZZ
	ZX, XZ
	YX, ZZ
	$XX \quad II \quad ZZ \quad ZI$ XI, XX, II, ZZ
$\langle 0 $ —	Z
— $ 0\rangle$	Z

Properties of this Construction

- ☼ Circuits as linear operators preserving the code space



$$V = |00\rangle\langle 00| + |11\rangle\langle 11|$$

$$\mathcal{C} = \text{span}(\{|00\rangle, |11\rangle\})$$

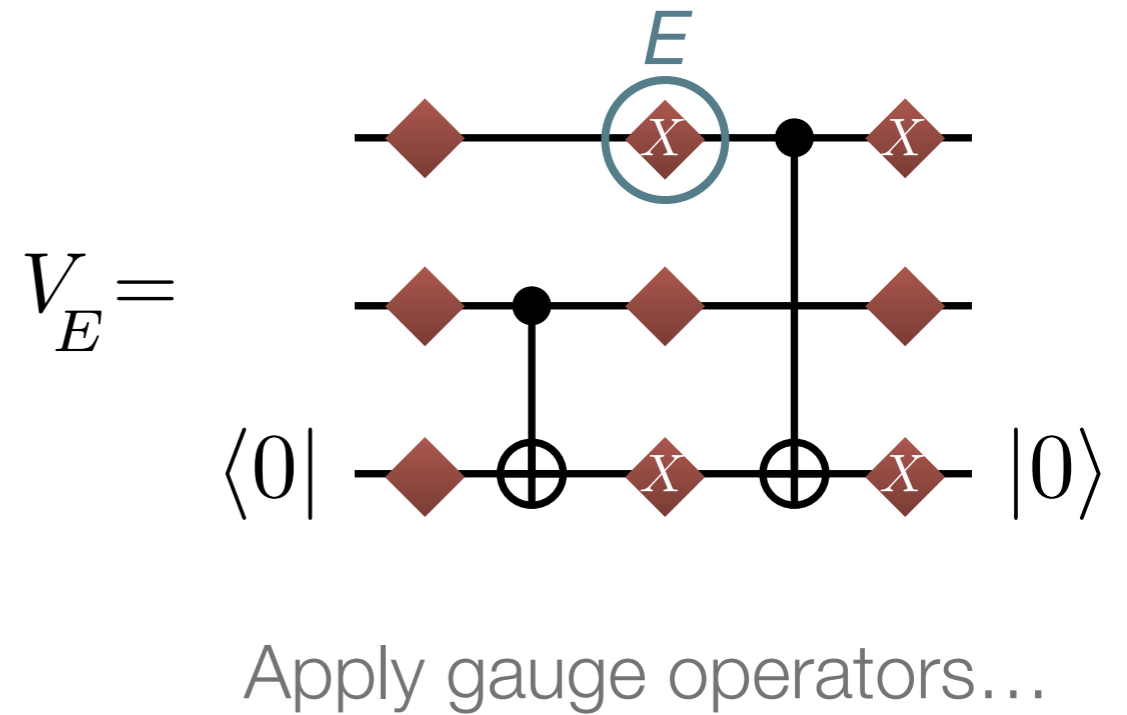
V is a good circuit

General condition:

$$V \text{ is good iff } V^\dagger V = \Pi_{\mathcal{C}}$$

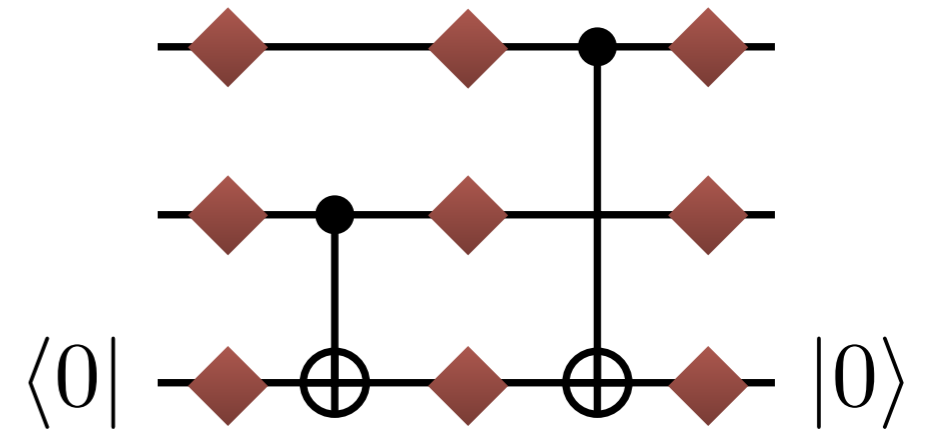
Properties of this Construction

- ⊛ Circuits as linear operators preserving the code space
- ⊛ Gauge equivalence of errors: $V_E = \pm V_{GE}$



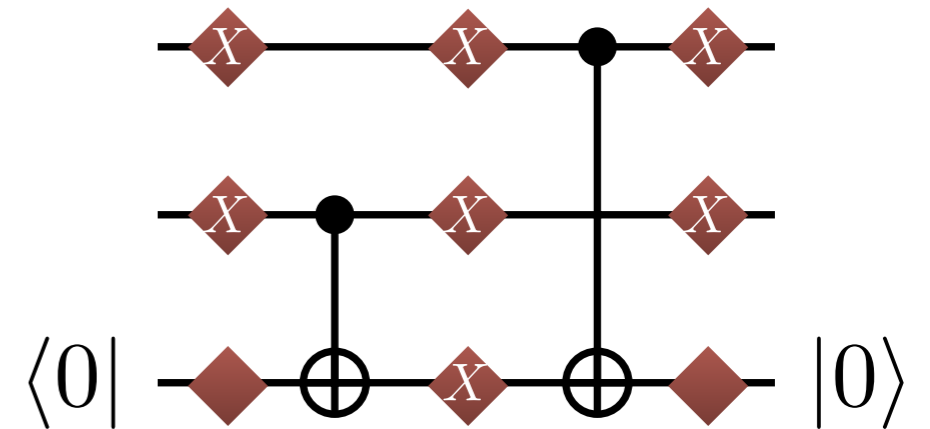
Properties of this Construction

- ✱ Circuits as linear operators preserving the code space
- ✱ Gauge equivalence of errors: $V_E = \pm V_{GE}$
- ✱ **Squeezegee** lemma: using gauge operations, we can localize errors to the initial data qubits



Stabilizer and Logical Operators

- * **Spackling**: like squeegee, but you leave a residue
- * Spackling of logical operators gives the new logical operators
- * Spackling of stabilizers on the inputs and ancillas are the new stabilizers
- * Everything else is gauge or detectable error
- * ...what about distance?



$$L_X = \begin{matrix} X & X & X \\ X & X & X \\ I & X & I \end{matrix} \quad L_Z = \begin{matrix} Z & Z & Z \\ I & I & I \\ I & I & I \end{matrix}$$

$$S = \begin{matrix} Z & Z & Z \\ Z & Z & Z \\ I & I & I \end{matrix} \quad S_a = \begin{matrix} Z & Z & I \\ Z & I & I \\ Z & Z & Z \end{matrix}$$

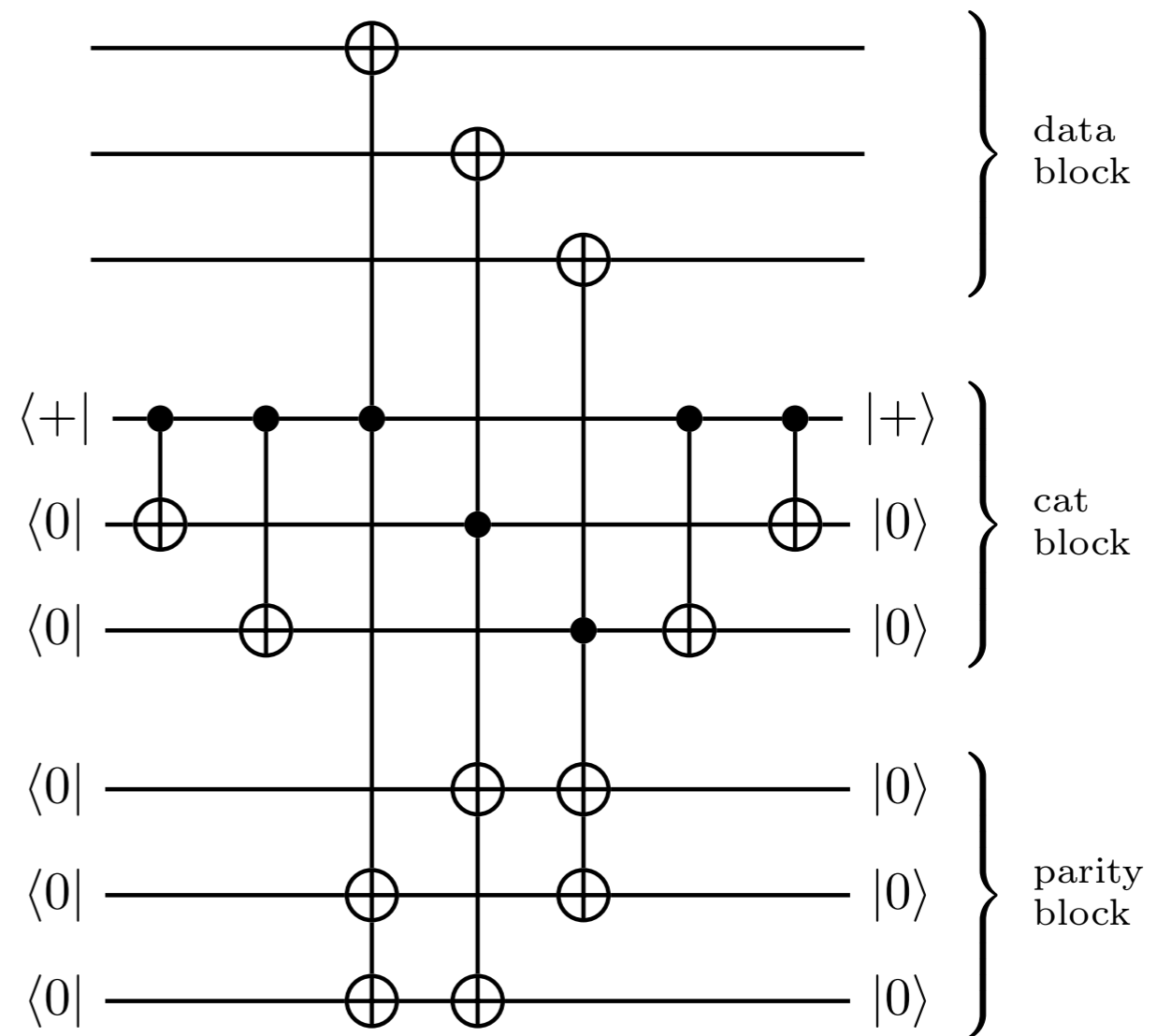
*even/odd effect means that circuits wires must have odd length

Code Distance and Fault Tolerance

- ✿ For most syndrome-measurement circuits, the new code is uninteresting
- ✿ If we use a **fault-tolerant circuit** then we preserve the code distance
- ✿ **Fault tolerance**: for every error pattern E , either $V_E = 0$ or there exists E' on inputs s.t. $V_{E'} = V_E$ and $|E'| \leq |E|$
- ✿ Strange constraints:
 - ✿ Circuit must be **Clifford** (so no majority vote)
 - ✿ No **classical feedback** or **post-processing** allowed
 - ✿ However, we only need to **detect** errors

Fault-Tolerant Gadgets

- ✱ Use modified Shor/
DiVincenzo cat states
- ✱ Build a cat, and postselect
...not fault tolerant
- ✱ Redeem this idea by
coupling to **expanders**
- ✱ constant-degree
expanders exist with
sufficient edge expansion
to make this fault tolerant



Wake Up!

Theorem 1. *Given any $[n_0, k_0, d_0]$ quantum stabilizer code with stabilizer generators of weight $w_1, \dots, w_{n_0-k_0}$, there is an associated $[n, k, d]$ quantum subsystem code whose gauge generators have weight $O(1)$ and where $k = k_0$, $d = d_0$, and $n = O(n_0 + \sum_i w_i)$. This mapping is constructive given the stabilizer generators of the base code.*

- ✱ Created sparse subsystem codes with the same k and d parameters as the base code
- ✱ Used fault-tolerant circuits in a new way, via expanders
- ✱ Extra ancillas are required according to the circuit size

Almost “Good” Sparse Subsystem Codes

- ✱ Start with an $[n_0, 1, d_0]$ random stabilizer code (so that $d_0 = O(n_0)$ with high probability)
- ✱ Concatenate this m times to get an $[n_0^m, 1, d_0^m]$ code
- ✱ Sum over the stabilizer weights gives $n = n_0^{O(m)}$
- ✱ Apply Theorem 1 with $m = O(\sqrt{\log n})$

Sparse subsystem codes exist with
 $d = O(n^{1-\varepsilon})$ and $\varepsilon = O(1/\sqrt{\log n})$.

Best previous distance for sparse codes was
 $d = O(\sqrt{n \log n})$ by Freedman, Meyer, Luo 2002

*Thank you
Sergei Bravyi!

Local Subsystem Codes Without Strings

- ✿ Take the circuit construction from the previous result
- ✿ Using SWAP gates and wires, spread the circuit over the vertices of a cubic lattice in D dimensions
- ✿ Let $n=L^D$ be the total number of qubits

Local subsystem codes exist with
 $d = O(L^{D-1-\varepsilon})$ and $\varepsilon = O(1/\sqrt{\log n})$.

Compared to Known Bounds

- ✱ Local subsystem codes in D dimensions
 $d \leq O(L^{D-1})$
- ✱ Our code: $d = \Omega(L^{D-1-\varepsilon})$
- ✱ Best known local stabilizer codes: $d = O(L^{D/2})$
- ✱ Local commuting projector codes
 $kd^{2/(D-1)} \leq O(n)$
- ✱ Our codes: $kd^{2/(D-1)} = \Omega(n)$
(use the hypergraph product codes and Thm 1)

Local Subsystem Codes Without Strings

- ✱ Specialize to $D=3$
- ✱ Sparse subsystem code on a lattice with $[L^3, O(1), L^{2-\varepsilon}]$
- ✱ **No strings**, either for **bare** or **dressed** logical operators
 - ✱ cf. Bombin's gauge color codes
- ✱ ...on the other hand it's a *subsystem* code
- ✱ How does this compare to other candidate self-correcting quantum memories?

$$*\varepsilon = O(1/\sqrt{\log n})$$

Comparing Candidate Self-Correcting Memories

Code	Self-correcting?	Comments
3D Bacon-Shor (Bacon 2005)	no	No threshold, so no self-correction (Pastawski <i>et al.</i> 2009)
Welded Code (Michnicki 2014)	no	See Brown <i>et al.</i> 2014 review article for discussion
Cubic Code (Haah 2011)	marginal	poly(L) memory lifetime for $L < e^{\beta/3}$ (Bravyi & Haah 2013)
Embedded Fractal Product Codes (Brell 2014)	maybe	very large ground-state degeneracy?
Gauge Color Codes (Bombin 2013)	???	Does have a threshold, also has string-like dressed operators
This talk (BFHS 2014)	???	No strings, concatenated codes have a threshold

Not depicted: Codes with long-range couplings (e.g. several works by the Loss group) or Hamma *et al.* 2009
See the talk by Olivier Landon-Cardinal on Friday for more discussion of these types of codes.

Challenges with Gauge Hamiltonians

- ✱ Gauge Hamiltonians are sometimes gapped:
(Kitaev 2005; Brell *et al.* 2011; Bravyi *et al.* 2013)
- ✱ ...but sometimes not:
(Bacon 2005; Dorier, Becca, & Mila 2005)
- ✱ The simplest example of our code (a wire) reduces to Kitaev's quantum wire, which is gapped as long as the couplings aren't equal in magnitude
- ✱ Our codes are a **vast generalization** of Kitaev's wire to arbitrary circuits!
- ✱ This undoubtedly has a rich phase diagram... might there be a gapped self-correcting phase, or something more?

Conclusion & Open Questions

- ✿ Showed a generic way to turn stabilizer codes into sparse subsystem codes
- ✿ New connection between **quantum error correction & fault-tolerant quantum circuits**
- ✿ What are the limits for sparse *stabilizer* codes?
- ✿ Self-correcting memory from the gauge Hamiltonian?
- ✿ Efficient, fault-tolerant decoding for these codes?
- ✿ Improve the rate? (Bravyi & Hastings 2013)
- ✿ Extend these results to allow for subsystem codes?
- ✿ See [arxiv:1411.3334](https://arxiv.org/abs/1411.3334) for more details!