Universal fault-tolerant quantum computation with Bacon-Shor codes

Theodore J. Yoder
Department of Physics, Massachusetts Institute of Technology

We present a fault-tolerant universal gate set consisting of Hadamard and controlled-controlled-Z (CCZ) gates on Bacon-Shor subsystem codes. Transversal non-Clifford gates on these codes are intriguing in that higher levels of the Clifford hierarchy become accessible as the code becomes more asymmetric. For instance, in an appropriate gauge, Bacon-Shor codes on an \( m \times m^k \) lattice have transversal \( k \)-qubit-controlled \( Z \). Through a variety of tricks, including intermediate error-correction and non-Pauli recovery, we reduce the overhead required for fault-tolerant CCZ. We calculate pseudo-thresholds for our universal gate set on the smallest \( 3 \times 3 \) Bacon-Shor code and also compare our gates with magic-states within the framework of a proposed ion trap architecture.

Shor’s 9-qubit code was the first quantum code discovered [1] and is still popular due to its conceptual simplicity. Later, it was realized that viewing Shor’s code as a subsystem code (the so-called Bacon-Shor code) leads to even easier protocols for error-correction [2, 3] with just local interactions on an \( m \times n \) lattice of qubits. An important concept in quantum coding theory is a threshold, the value of the physical error rate below which encoding quantum data begins to help reduce errors. Surprisingly, the simple Bacon-Shor code even boasts some of the highest known thresholds for concatenated codes [3].

Behind such thresholds, however, actually lie several assumptions on how a universal gate set is constructed [4]. Since the quoted thresholds are calculated by simulating the encoded CNOT gadget, but are for universal computation, the assumption is made that for non-Clifford gates, a magic-state [5] should be distilled and injected at an arbitrarily high level of concatenation. The resulting scheme is simply not resource realistic for fault-tolerant experiments operating in the low-distance limit.

It is more realistic to study a universal gate set for low-distance codes directly, and it is this program that we adhere to here. This is possible in principle, as already evidenced dating back to Shor [6] and similar constructions elsewhere [7] that verify magic-states directly at low distance. Optimizing low-distance constructions of non-Clifford gates is the next step. Prior work in this direction is promising. For instance, gauge-fixing can convert between the 7-qubit and 15-qubit codes [8, 9] to take advantage of their complementary universal transversal gate sets. Alternatively, concatenation of complementary codes also yields a universal set of gates [10].

Our starting point is a different strategy, wherein instead of combining two different codes, non-transversal gates are constructed directly and made fault-tolerant via stabilizer measurements intermediate in the circuit. Since the intermediate error-correction cycles effectively break the circuit into fault-tolerant pieces, this is dubbed “pieceable” fault-tolerance [11]. One advantage of this approach is relatively broad applicability, including to the Bacon-Shor code family. Code-specific simplifications of the general-purpose designs in [11] exist and these can offer substantial improvements in how much intermediate error-correction is required. The Bacon-Shor code’s simple structure also lends itself well to these strategies.

Here we develop an appealing scheme for universal fault-tolerant computing on Bacon-Shor codes using fault-tolerant Hadamard (\( H \)) and controlled-controlled-\( Z \) (CCZ) gates. In its simplest form, the scheme hinges on the observation that extending a symmetric Bacon-Shor code into an asymmetric one enlarges the class of transversal non-Clifford gates on the code. Indeed, make the code asymmetric enough, i.e. \( m \times m^k \), and a gate (namely the \( k \)-qubit-controlled \( Z \)) from the \( k \)-th level of the Clifford hierarchy [12] becomes transversal. If the code does not meet the asymmetry requirement, then adding intermediate error-correction can still make the construction fault-tolerant. Since many uses of intermediate error-correction are undesirable, we can reduce the number by clever circuit design. Additional simplifications are achieved using non-Pauli recovery operations. Non-Pauli recovery lies outside the standard formalism for fault-tolerant computing with stabilizer codes but is permitted by the Knill-Laflamme conditions [13]. Such overhead-reducing innovations may be crucial for experimentally realizing small instances of fault-tolerance.

The smallest instance of our constructions is a fault-tolerant CCZ on the \( 3 \times 3 \) Bacon-Shor code that uses no intermediate error-correction. Because \( H \) is transversal on symmetric Bacon-Shor codes, the \( 3 \times 3 \) code has a particularly simple fault-tolerant universal gate set that moreover requires no postselection.

For the \( 3 \times 3 \) code, we compute exREC pseudo-thresholds [14] for a universal gate set under circuit depolarizing noise and find \( \sim 8 \times 10^{-5} \) for the largest gate CCZ. We also discuss overhead, and compare with magic-state implementations of CCZ in the ion trap MUSICQ [15] architecture. In this context, we estimate a roughly 4 times faster implementation of fault-tolerant Toffoli, a key part of the quantum circuits used for Shor’s factoring [16]. We conclude that ion traps, with their ability to easily implement non-local gates, are in a promising position to take full advantage of fault-tolerant optimizations.

The Bacon-Shor codes are subsystem codes, as opposed to subspace codes. As such, there are three relevant sets of Paulis, the stabilizers, gauge operators, and logical operators. Let \( \{ m \} = \{ 0, 1, \ldots, m - 1 \} \) and \( \{ m \} = \{ 1, 2, \ldots, m - 1 \} \) denote subsets of integers.
out the code qubits in an \( m \times n \) lattice (Fig. 1), so qubits are indicated by coordinates \((i, j) \in [m] \times [n]\) and single-qubit Paulis by subscripts \(X_{i,j}, Y_{i,j}, Z_{i,j}\). As in [3], denote columns and rows of Paulis by, respectively,

\[
Z_{s,j} = Z_{0,j} Z_{1,j} \cdots Z_{m-1,j}, \quad (1)
\]

\[
X_{i,s} = X_{i,0} X_{i,1} \cdots X_{i,n-1}. \quad (2)
\]

The stabilizer group \( S \) is generated from

\[
\tilde{Z}_j = Z_{s,j-1} Z_{s,j}, \quad \forall j \in (n), \quad (3)
\]

\[
\tilde{X}_i = X_{i-1,s} X_{i,s}, \quad \forall i \in (m). \quad (4)
\]

That is, the stabilizers are an even number of columns of \( Z \)'s or an even number of rows of \( X \)'s.

Gauge operators are generated by

\[
\tilde{Z}_{i,j} = Z_{i,j-1} Z_{i,j}, \quad \forall i \in [m], j \in (n), \quad (5)
\]

\[
\tilde{X}_{h,k} = X_{h-1,k} X_{h,k}, \quad \forall h \in (m), k \in [n], \quad (6)
\]

while (lowest-weight) bare logical operators are any single column of \( Z \)'s and any single row of \( X \)'s. For instance, \( \tilde{Z} = Z_{s,0} \) and \( \tilde{X} = X_{0,s} \). The code distance of an \( m \times n \) Bacon-Shor code is \( \min(m,n) \), but asymmetric codes also offer greater protection against one type of error [17]. A (destructive) transversal measurement of all qubits in the \( Z \)- (\( X \)-) basis suffices to measure \( \tilde{X} \) and \( \tilde{Z} \).

The gauge offers degrees of freedom not available in subspace stabilizer codes. Indeed, notice that from measurements of the gauge operators we can infer a measurement of the stabilizers, since

\[
\tilde{Z}_j = \prod_{i \in [m]} Z_{i,j}, \quad \tilde{X}_i = \prod_{j \in [n]} X_{i,j}. \quad (7)
\]

Measuring local gauge operators can be easier than measuring non-local, high-weight stabilizers directly.

Although the gauge offers an advantage during error-correction, during our logical gates we will want to fix a gauge. On paper, fixing a gauge amounts to adding a maximal, commuting subset of the gauge operators to the stabilizer. Common gauges are the \( Z \)-gauge (formed by placing all \( Z_{i,j} \) into \( S \)) and the \( X \)-gauge (placing all \( \tilde{X}_{h,k} \) into \( S \)). However, the rotated surface code [18] is also just a gauge choice of the Bacon-Shor code.

In practice, we can fix a gauge by measuring the gauge operators. Steane error-correction [19] achieves a high threshold while also being extremely simple for Bacon-Shor codes. Indeed, the logical states \( |0_X \rangle \) and \( |+_Z \rangle \), where subscripts denote \( X \)- or \( Z \)-gauge, are simply tensor products of CAT states,

\[
|0_X \rangle = (|+ \rangle^\otimes m + |- \rangle^\otimes m)^\otimes n \quad \text{and} \quad |+_Z \rangle = (|0 \rangle^\otimes n + |1 \rangle^\otimes n)^\otimes m.
\]

Preparing these fault-tolerantly is easy for \( m = n = 3 \), because 3-qubit CAT states need not be postselectively verified. Gauge operators \( \tilde{X}_{h,k} \) (and therefore stabilizers \( X \) as well) are measured using \( |0_X \rangle \), while \( \tilde{Z}_{i,j} \) (and so \( \tilde{Z} \) as well) are measured using \( |+_Z \rangle \). Measuring the \( X \)-gauge followed by the \( Z \)-gauge leaves the code in the \( Z \)-gauge, and reversing the order of measurement leaves it in the \( X \)-gauge.

There is a subtlety in the process of gauge fixing. When the syndrome measurement is ordered such that the gauge changes (from \( X \)- to \( Z \)- or vice versa) we obtain more information about the errors than if we had ordered the measurements such that the gauge does not change. In the former case, call it type-1 correction, if the code began in the \( Z \)-gauge (\( X \)-gauge), we learn the values of all \( \tilde{Z}_{i,j} (\tilde{X}_{h,k}) \) and the values of all \( \tilde{X}_i (\tilde{Z}_j) \). However, in the latter case, type-2 correction, we learn only the values of all \( \tilde{X}_i \) and all \( \tilde{Z}_j \). Transversal logical gates can use either type of correction to achieve fault-tolerance, but the fault-tolerance of our non-transversal CCZ gates can be dependent on the added information gathered from type-1 error-correction. Nevertheless, if the gauge change of type-1 is undesired it can always be changed back by a subsequent type-1 correction.

Extending a Bacon-Shor code can be done fault-tolerantly. To transfer the encoded quantum state from an \( m \times n \) code to an \( m' \times n \) code, \( m' > m \), prepare an \( (m' - m) \times n \) Bacon-Shor codeblock in \( |+_Z \rangle \). Join the ancilla block with the initial block by measuring the \( Z \)-gauge operators across the boundary. This could be done by performing \( m' \times n \) Steane error-correction for instance. Adding more rows instead is the Hadamard conjugate of this process. One can also remove columns by measuring the individual qubits in the \( X \)-basis and rows by measuring them in the \( Z \)-basis.

It is well known that Toffoli and Hadamard are a universal gate set for quantum computation [20]. Replacing Toffoli with controlled-controlled-\( Z \) (CCZ) also makes a universal gate set. We now discuss how to implement Hadamard and CCZ on a Bacon-Shor code.

On symmetric \((i.e. m \times m)\) Bacon-Shor codes, logical \( \overline{\text{H}} \) is a transversal gate up to a qubit permutation [3]. On asymmetric Bacon-Shor codes (i.e. \( m \times n \) with \( n > m \)), \( \overline{\text{H}} \) can be done via teleportation [21]. Preparing \( |\overline{+}_Z \rangle \), coupling to the target codeblock with CCZ (which we show later is transversal), and measuring \( \tilde{X} \) on the target codeblock suffices to teleport the original encoded state to the ancilla codeblock with \( \overline{\text{H}} \) applied. This protocol also implies CCZ is universal on its own (given \( X \)- and \( Z \)-basis state preparation and measure-
We discuss this corollary more in Appendix A.

To obtain computational universality, we implement logical CCZ, a three-qubit gate, with full-distance (i.e., under circuit depolarizing noise, a distance $d$ code recovers from $[(d-1)/2]$ faulty circuit components). Assume that all code blocks (labeled $A, B, C$) begin in the $Z$-gauge. Logical CCZ, denoted $\overline{\text{CCZ}}$, can be created from physical CCZ gates in round-robin fashion [11]:

$$\overline{\text{CCZ}} = \prod_{u, v, w \in \text{supp}(\bar{Z})} \text{CCZ}(u, v, w).$$

All gates following $\prod$-symbols in this paper mutually commute, so ordering is unnecessary.

Ostensibly then, $m \times n$ Bacon-Shor codes would use $m^3$ physical CCZ gates to implement $\overline{\text{CCZ}}$ in a depth $m^2$ circuit, because $Z$ has support $\text{supp}(\bar{Z})$ of size at least $m$. To make Eq. (8) fault-tolerant, it is sufficient to measure $Z$ after each timestep of CCZ gates. This suffices because $X$ errors can be detected and corrected before they propagate $Z$ errors through subsequent CCZ gates. In the terminology of [11], we say the circuit is fault-tolerant in $m$ pieces, a number of pieces equal to the circuit depth. However, two kinds of simplifications can generally reduce the number of pieces.

The first of these simplifications exploits the code stabilizer to reduce the depth of the circuit. By definition, for a stabilizer $s \in S$ and a state $|\bar{\psi}\rangle$ in the codespace, $s|\bar{\psi}\rangle = |\bar{\psi}\rangle$. Thus, we might say $s$ is an implementation of logical identity, $\bar{I}$. However, it is only one such implementation. For instance, controlled-$s$ and controlled-controlled-$s$ are also implementations of $\bar{I}$ for any control qubit(s) (note, the controls could be taken in any basis, but we will only use controls in the $Z$-basis here). In $Z$-gauge Bacon-Shor codes, $\bar{Z}_{i,j} \in S$ and thus,

$$\bar{I} = \text{CCZ}(u_A, v_B, (i, j-1)_C)\text{CCZ}(u_A, v_B, (i, j)_C),$$

for all $u \in [m] \times [n], v \in [m] \times [n], i \in [m], j \in [n]$. Similar expressions hold under permutation of $A, B, C$.

By multiplying $\overline{\text{CCZ}}$ from Eq. 8 by implementations of $\bar{I}$ from Eq. 9, we create lower depth implementations of $\overline{\text{CCZ}}$. This can be visualized as moving the control nodes of the CCZ gates across rows of a $Z$-gauge Bacon-Shor code block, thereby spreading CCZ gates across all code qubits. For $m \times m$ Bacon-Shor codes, it is trivial to reduce the depth of the circuit to $m$. For $m \times m^2$ Bacon-Shor codes, $\overline{\text{CCZ}}$ becomes depth-1, i.e. transversal. Fig. 2 shows the $3 \times 9$ case. Generally, $m \times n$ Bacon-Shor codes can implement $\overline{\text{CCZ}}$ with a depth $\lceil m^2/n \rceil$ circuit, which translates directly to a fault-tolerant $\overline{\text{CCZ}}$ in $\lceil m^2/n \rceil$ pieces, as discussed above. From this argument we see a space-time tradeoff emerge: an $m \times n$ Bacon-Shor code supports a depth $h$ circuit for $\overline{\text{CCZ}}$ if $hn \geq m^2$. The appropriate generalization for a $\overline{\text{C}^k\text{Z}}$ gate (a $Z$ with $k$ controls) is $hn \geq m^k$ for any integer $k \geq 1$.

Explicitly, these circuits for $\overline{\text{C}^k\text{Z}}$ on $m \times n$ codes can be arranged in terms of subcircuits indexed by a $k$-digit $m$-ary number $p$. The subcircuit $p = p_k \cdots p_2 p_1 \cdots p_0$ is

$$C_p = \prod_{i=0}^{m-1} \overline{\text{C}^k\text{Z}}((i, j), (i \oplus p_{k-1}, j), \ldots, (i \oplus p_0, j))$$

with addition $\oplus$ modulo $m$ and for some choice of column $j$ depending on $p$. The standard choice would be $j = p \pmod{n}$. Subcircuits with the same value of $j$ must be done in subsequent timesteps, contributing to the circuit depth $h = \lfloor m^k/n \rfloor$. The product of all such subcircuits implements the logical gate: $\overline{\text{C}^k\text{Z}} = \prod_{p=0}^{n-1} C_p$.

The advantage of this organization is that, if the three interacting codeblocks are layered upon one another in the plane, the physical $\overline{\text{C}^k\text{Z}}$ gates interact qubits within a column, i.e. distant from one another by at most $m-1$ lattice spacings (rather than the worst case $n-1 \geq m-1$).

That $\overline{\text{C}^k\text{Z}}$ with $k > 1$ on 2D Bacon-Shor codes must use long-range gates (or, equivalently, SWAP circuits of non-constant depth) is necessitated by arguments similar to those of Bravyi-König [22] for topological subspace codes. In Appendix B, we present this argument and note that for all $k$ our constructions use optimal gate range.

As a practical matter, either substantially extending the code or using a larger depth circuit and correcting $X$ errors after every timestep may be unappealing. As our second simplification, we can reduce the number of intermediate error corrections by using an idea from [11] called 2-transversality, wherein each qubit interacts with at most two qubits from each other codeblock. We leave a more thorough description of this simplification to Appendix C, but note here that it can reduce the number of pieces used to implement $\overline{\text{C}^k\text{Z}}$ to $\lceil [m/2]^k/n \rceil$. In particular, a $3 \times 4$ Bacon-Shor code (just 12 code qubits) can implement $\overline{\text{CCZ}}$ without intermediate error-correction.

However, we can do even better, implementing $\overline{\text{CCZ}}$ on smaller Bacon-Shor codes without the need for intermediate error-correction. In doing so, we no longer assume that error-correction consists of projection to the stabilizer space with strictly Pauli recovery. This highlights the inequivalence of the Knill-Laflamme conditions for the existence of a general recovery map and the (necessarily stronger) conditions for the existence of Pauli...
CCZ never uses postselection. Magic-state CCZ to be 4-times faster than using a CCZ with magic-states for Steane’s 7-qubit code, following the design in [15, 21], uses 56-126 qubits depending on error-correction scheme, with 66 striking a balance between error rate, qubit count, and gate time. Our Bacon-Shor CCZ can use between 30-81 qubits, with 54 (9 × 3 data plus 9 × 3 ancillas) striking a good balance. Notice that the Bacon-Shor scheme leaves a comfortable 46 qubits per ELU for entanglement generation. Physical qubits are reusable in our estimations and we assume all qubits are in the same ELU. If not, teleporting codeblocks to the same “interacting” ELU adds a small overhead to the numbers in Table I. More circuit details are in Appendix C.

To illustrate the advantages of such a small CCZ construction, we provide a comparison with a magic state method for ion traps. In the MUSICQ architecture [15], qubits are grouped into elementary logical units, or ELUs, with $N_q$ qubits per ELU (where $N_q \leq 100$ is considered daunting but possible) all of which may interact via two-qubit gates. Qubits within different ELUs interact by teleportation through shared entanglement generated by photon interference. Since it takes roughly two orders of magnitude longer to generate entanglement between ELUs than it takes to interact qubits within an ELU, we set aside $C_q \approx N_q/2$ qubits per ELU for interaction with the other ELUs. Due to excellent state lifetimes in ion traps, it is reasonable to assume that entanglement is generated and stored just before a computation.

Thus, the MUSICQ architecture excels at implementing non-local gates. This comes at the cost of limited, though not nonexistent, parallel operations. To compare our Bacon-Shor CCZ with magic-states for Steane’s 7-qubit code, we follow [15] and assume up to twelve multi-qubit operations (CNOTs and CCZs) can be performed in parallel within an ELU. We idealize single-qubit gate and state preparation time as 1µs, 2-qubit and 3-qubit gate time as 10µs, and measurement time as 30µs [15].

Our comparison is shown in Table I. Roughly we expect a Bacon-Shor CCZ to be 4-times faster than using a magic state while also using fewer qubits. Also note that in all scenarios, the magic-state approach uses postselection to both prepare the magic-state and prepare CAT or Steane states for error-correction. In contrast, the Bacon-Shor CCZ never uses postselection. Magic-state CCZ for the 7-qubit code, following the design in [15, 21], uses 56-126 qubits depending on error-correction scheme, with 66 striking a balance between expected error rate, qubit count, and gate time. Our Bacon-Shor CCZ can use between 30-81 qubits, with 54 (9 × 3 data plus 9 × 3 ancillas) striking a good balance. Notice that the Bacon-Shor scheme leaves a comfortable 46 qubits per ELU for entanglement generation. Physical qubits are reusable in our estimations and we assume all qubits are in the same ELU. If not, teleporting codeblocks to the same “interacting” ELU adds a small overhead to the numbers in Table I. More circuit details are in Appendix C.

We have also calculated pseudothresholds for our universal gate set on the 3 × 3 Bacon-Shor code using exact counting. The results are shown in Table II. Calculation details and logical error rate plots are in Appendix E.

What is the most practical route toward scalable, universal, fault-tolerant quantum computation? The question is fraught with many dependencies and subtleties, such as hardware capabilities, the gate set implemented, the noise model and rate, etc. If we drop the scalability requirement and set our sights instead on near term fault-tolerance at low distance, we can start optimizing.

It is in this spirit that we have developed our low-overhead universal computing scheme for the 3×3 Bacon-Shor. And the results are relatively promising — a high pseudothreshold and a size comfortably fitting into quantum computing architectures of the near future.

However, can we put scalability back without too much cost? Indeed, with the Bacon-Shor codes, this seems challenging. It is well-known that Bacon-Shor codes fail to have an asymptotic threshold as a topological family. Concatenation offers a threshold in theory, but in practice it is hard to implement and pays a price in overhead.

An intriguing alternative to improve scalability is to exploit the limited ability of 3D Bacon-Shor codes to be self-correcting [2]. $Z$-gauge Bacon-Shor codes in 3D are codespace equivalent to asymmetric codes in 2D. However, what were 1-dimensional rows of qubits in 2D codes become 2-dimensional lattices in 3D. The Ising ZZ inter-

| Gate  | $p$          | $p/10^{2-q}$ |
|-------|--------------|--------------|
| I & H | $4.1 \times 10^{-4}$ | $1.9 \times 10^{-4}$ |
| CNOT  | $1.4 \times 10^{-4}$ | $5.3 \times 10^{-4}$ |
| CCZ   | $8.2 \times 10^{-5}$ | $6.1 \times 10^{-4}$ |

TABLE II. A table of exREC [7] pseudothreshold [14] lower bounds for gates on the 3 × 3 Bacon-Shor code. The thresholds of all gates are calculated by exact counting. Shown are two versions of the circuit depolarizing noise model, one in which components all err with the same probability $p$ and the other in which a qubit component err with probability $p/10^{2-q}$. The thresholds shown are the error rates for CNOT below which the encoded gate is better than physical.
teraction can be applied between all lattice neighbors. Thus, the Z-gauge 3D Bacon-Shor code inherits the thermal (though not Hamiltonian-perturbative [23]) stability of the 2D Ising model to protect $X$. Moreover, since $X$ errors are suppressed in this manner, the need for intermediate correction in our circuits is reduced. Though not completely solving the scalability problem ($Z$ is still vulnerable) it could offer a substantial simplification of our construction for architectures that can support it.

The author gratefully acknowledges Ken Brown, Steve Flammia, and Ryuji Takagi for discussions and their comments on the manuscript, as well as Cody Jones for ideas relating Bacon-Shor codes and the surface code. This project was supported financially by the Department of Defense (DoD) through the National Defense Science and Engineering Graduate (NDSEG) Fellowship program.

[1] Peter W Shor, “Scheme for reducing decoherence in quantum computer memory,” Physical Review A 52, R2493 (1995).
[2] Dave Bacon, “Operator quantum error-correcting subsystems for self-correcting quantum memories,” Physical Review A 73, 012340 (2006).
[3] Panos Aliferis and Andrew W Cross, “Subsystem fault tolerance with the Bacon-Shor code,” Physical Review Letters 98, 220502 (2007).
[4] Panos Aliferis, “Level reduction and the quantum threshold theorem,” arXiv preprint quant-ph/0703230 (2007).
[5] Sergey Bravyi and Alexei Kitaev, “Universal quantum computation with ideal clifford gates and noisy ancillas,” Physical Review A 71, 022316 (2005).
[6] Peter W Shor, “Fault-tolerant quantum computation,” in Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on (IEEE, 1996) pp. 56–65.
[7] Panos Aliferis, Daniel Gottesman, and John Preskill, “Quantum accuracy threshold for concatenated distance-3 codes,” Quantum Information & Computation 6, 97–165 (2006).
[8] Adam Paetznick and Ben W Reichardt, “Universal fault-tolerant quantum computation with only transversal gates and error correction,” Physical Review Letters 111, 090505 (2013).
[9] Jonas T Anderson, Guillaume Duclos-Cianci, and David Poulin, “Fault-tolerant conversion between the Steane and Reed-Muller quantum codes,” Physical Review Letters 113, 080501 (2014).
[10] Tomasz Jochym-OConnor and Raymond Laflamme, “Using concatenated quantum codes for universal fault-tolerant quantum gates,” Physical Review Letters 112, 010505 (2014).
[11] Theodore J Yoder, Ryuji Takagi, and Isaac L Chuang, “Universal fault-tolerant gates on concatenated stabilizer codes,” Physical Review X 6, 031039 (2016).
[12] Daniel Gottesman and Isaac L Chuang, “Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations,” Nature 402, 390–393 (1999).
[13] Emanuel Knill and Raymond Laflamme, “Theory of quantum error-correcting codes,” Physical Review A 55, 900 (1997).
[14] Krysta M Svore, Andrew W Cross, Isaac L Chuang, and Alfred V Aho, “A flow-map model for analyzing pseudothresholds in fault-tolerant quantum computing,” Quantum Information & Computation 6, 193–212 (2006).
[15] C Monroe, R Raussendorf, A Ruthven, KR Brown, P Maunz, L-M Duan, and J Kim, “Large-scale modular quantum-computer architecture with atomic memory and photonic interconnects,” Physical Review A 89, 022317 (2014).
[16] Peter W Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,” SIAM review 41, 303–332 (1999).
[17] Peter Brooks and John Preskill, “Fault-tolerant quantum computation with asymmetric Bacon-Shor codes,” Physical Review A 87, 032310 (2013).
[18] Yu Tomita and Krysta M Svore, “Low-distance surface codes under realistic quantum noise,” Physical Review A 90, 062320 (2014).
[19] Andrew M Steane, “Active stabilization, quantum computation, and quantum state synthesis,” Phys. Rev. Lett. 78, 2252 (1997).
[20] Dorit Aharonov, “A simple proof that Toffoli and Hadamard are quantum universal,” arXiv preprint quant-ph/0301040 (2003).
[21] Xinlan Zhou, Debbie W Leung, and Isaac L Chuang, “Methodology for quantum logic gate construction,” Physical Review A 62, 052316 (2000).
[22] Sergey Bravyi and Robert König, “Classification of topologically protected gates for local stabilizer codes,” Physical Review Letters 110, 170503 (2013).
[23] Fernando Pastawski, Alastair Kay, Norbert Schuch, and J Ignacio Cirac, “Limitations of passive protection of quantum information,” Quantum Information & Computation 10, 580–618 (2010).
[24] Bryan Eastin and Emanuel Knill, “Restrictions on transversal encoded quantum gate sets,” Physical Review Letters 102, 110502 (2009).
[25] Fernando Pastawski and Beni Yoshida, “Fault-tolerant logical gates in quantum error-correcting codes,” Physical Review A 91, 012305 (2015).
[26] Michael A Nielsen and Isaac L Chuang, “Quantum information and quantum computation,” Cambridge: Cambridge University Press 2, 23 (2000).
[27] Debbie W Leung, Michael A Nielsen, Isaac L Chuang, and Yoshihisa Yamamoto, “Approximate quantum error correction can lead to better codes,” Physical Review A 56, 2567 (1997).
[28] Cédric Bény and Ogyvan Oreshkov, “General conditions for approximate quantum error correction and near-optimal recovery channels,” Physical Review Letters 104, 120501 (2010).
[29] Hayato Goto, “Minimizing resource overheads for fault-tolerant preparation of encoded states of the Steane code,” Scientific reports 6 (2016), 10.1038/srep19578.
[30] Theodore J Yoder and Isaac H Kim, “The surface code with a twist,” Quantum 1, 2 (2017).
[31] D Gottesman, The Heisenberg representation of quantum computers, Tech. Rep. (Los Alamos National Lab., 1998).
Appendix A: Computational universality without single-qubit gates

Most universal gate sets for quantum computation include some single-qubit gates. Indeed, this can be an efficient choice for implementing some algorithms because fault-tolerant versions of those gates are likely small, and, moreover, much more is known about compiling with single-qubit gates. However, single-qubit gates are not a necessary part of a computationally universal gate set, while multi-qubit gates are.

**Theorem 1.** Assuming the availability of $|0⟩$, $|+⟩$ and measurement in the $X$- and $Z$-bases, CCZ is quantum computationally universal.

**Proof.** We start from the computationally universal set $(H, CCZ)$ [20], then implement $H$ using the circuit described in the main text (also see Fig. 3). This involves two gates not explicitly in our gate set. First, CZ can be implemented using CCZ controlled on a $|1⟩$ ancilla. We can prepare $|1⟩$ nondeterministically by measuring $|+⟩$ in the $Z$-basis, succeeding with probability $1/2$. Second, $X$ might have to be applied via classical control. However, we can track this Pauli through the subsequent circuit. At most, it results in needing an additional CZ gate for every CCZ, a polynomial blowup in circuit size. We note that both complications are more easily remedied, in particular without any nondeterminism, by adding $X$ to the gate set, which is transversal for any stabilizer code. \(\square\)

Thus, any stabilizer code that can implement CCZ has fault-tolerant computational universality. Codes with transversal CCZ do not violate the Eastin-Knill no-go [24] because the injection of $H$ is not completely unitary. Interestingly, if transversal CZ is available in a CSS (Calderbank-Shor-Steane) stabilizer code, the injection of $H$ in Fig. 3 uses no more resources than Steane error-correction would, because $Z$ errors are correctable using information from the transversal $X$-measurement. This is perhaps an even easier route to universality with the 15-qubit Reed-Muller code (and the larger quantum Reed-Muller family) than gauge-fixing à la [8, 9].

Finally, we note Toffoli is quantum universal on its own given the same access to preparation and measurement. Since classical reversible computation is universal with just Toffoli, this reveals the power of quantum computation as “simply” the ability to prepare and measure in the conjugate basis.

Appendix B: Bravyi-König for Bacon-Shor codes

In this section, we view the 2D Bacon-Shor codes as a topological family to better understand the limitations of logical gates. Bravyi-König [22] made general arguments restricting the ability of logical operators for stabilizer codes with local generators in $D$ spatial dimensions. Subsequently, Pastawski-Yoshida [25] made similar arguments for subsystem codes that possess a threshold. Unfortunately, neither of these results directly apply to the Bacon-Shor CCZ gates that we have developed here. Our gates work only in the fixed $Z$-gauge, and therefore the $X$-stabilizers fail to satisfy the spatial locality constraint of Bravyi-König. The Bacon-Shor code family also notoriously fails to have a threshold, and so fails to satisfy the assumptions of Pastawski-Yoshida.

Nevertheless, following the simpler argument of Bravyi-König for the restriction of logical gates on the 2D surface code, we can develop the following theorem for 2D, $Z$-gauge Bacon-Shor codes.

**Theorem 2.** Consider a constant-depth circuit $U$ that is a logical operator on a constant number of copies a 2D, $Z$-gauge Bacon-Shor code with distance $d$. Then, $U$ is a Clifford operation as long as the gates in $U$ have ranges in the $x$- and $y$-dimensions $R_x$ and $R_y$, respectively, satisfying $(R_x + 1)(R_y + 1) < O(d)$.

We note that our CCZ constructions on $m \times m^2$ codes saturate the bound, because $R_y = m - 1 = d - 1$ and $R_x = 0$. Indeed, even our $C^kZ$ constructions on $m \times m^k$ codes saturate the bound for all $k$.

**Proof.** Consider a stack of a constant number of 2D, $Z$-gauge Bacon-Shor codes depicted in Fig. 4. Denote pairs of horizontal and vertical regions of qubits by $\xi_j$ and $\zeta_j$ with $j = 1, 2$. These regions can be chosen to have constant width and be separated by $O(d)$ qubits. Consider two logical Pauli operators $P$ and $Q$ acting on all code-blocks. Without loss of generality we may take $P$ to lie within $\omega_1 := \xi_1 \cup \zeta_1$ and $Q$ to lie within $\omega_2 := \xi_2 \cup \zeta_2$.

Write the group commutator

$$K = P(UQU^\dagger)P(UQU^\dagger).$$  \hspace{1cm}  \text{(B1)}$$

It is clear that $K$ is a logical operator since $U$ is. Assume for now (to be shown later) that $K$ is a trivial logical operator, i.e.

$$K \Pi = c \Pi,$$  \hspace{1cm}  \text{(B2)}$$

where $\Pi$ is the projector onto the codespace and $c$ is a constant. Writing $PK \Pi = c \Pi$ and squaring it, we get

$$(PK)^2 \Pi = c^2 P^2 \Pi,$$  \hspace{1cm}  \text{(B3)}$$

where commutation of all logical operators with $\Pi$ was used. Since $P^2 = Q^2 = 1$ as they are Pauli operators, we get $c = \pm 1$. Thus, Eq. (B2) becomes

$$P(UQU^\dagger) \Pi = \pm(UQU^\dagger)P \Pi,$$  \hspace{1cm}  \text{(B4)}$$

FIG. 3. Implementing $H$ by one-bit teleportation [21].
representing commutation or anticommutation of $P$ and $UQU^\dagger$ with respect to the codespace. This holds for all $P$ and $Q$, implying $UQU^\dagger$ is a logical Pauli operator for all $Q$, and thus $U$ is a logical Clifford operator by definition.

It remains to show that Eq. (B2) holds, and it is here we use the bound on the range of gates in $U$. Let $\mathcal{B}(A, r)$ denote the set of qubits within distance $r$ of a set of qubits $A$. If $U$ has depth $h$ and consists of gates with range $R_x$ and $R_y$ in the $x$- and $y$-dimensions, then $V := UQU^\dagger \in \mathcal{B}(\xi_2, h R_y) \cap \mathcal{B}(\zeta_2, h R_x) := \beta_2$. Gates of $U$ outside the “lightcone” of $\alpha_2$ act trivially on $Q$ and cancel, leading to $V$ supported in a limited area. Now $V$ is a depth $(2h+1)$ circuit of bounded range gates, and so $K = PV PV^\dagger$ is similarly supported only in the region $\mathcal{B}(\xi_1, (2h+1)R_y) \cap \mathcal{B}(\zeta_1, (2h+1)R_x) := \beta_1$. Yet, at the same time, because $V \in \beta_2$, we have $K \in \beta_2$. Thus, $K \in \beta_1 \cap \beta_2$, a region which has size upper bounded by

$$|\beta_1 \cap \beta_2| < O(h^2 R_x R_y + h R_x + h R_y). \quad (B5)$$

If $h^2 R_x R_y + h R_x + h R_y < O(d)$, then $K$ can only be the trivial logical operator and so Eq. (B2) is proved. In the case that $U$ has constant depth $h = O(1)$, then $(R_x + 1)(R_y + 1) < O(d)$ is sufficient for $K$ to be trivial.  

We note briefly that 3D Bacon-Shor codes [2] in the Z-gauge can perform CCZ without ranged gates, as long as the three codeblocks are allowed to be oriented differently. Symmetric 3D Bacon-Shor codes $m \times m \times m$ are constructed from $m \times m \times m$ planes of the 2D Ising model with nearest neighbor ZZ interactions ($XX$ interactions exist between adjacent planes). Let $\hat{n}_j$ be the vector perpendicular to all these $m$ planes for codeblock $j$. Taking $\hat{n}_1 = \hat{x}$, $\hat{n}_2 = \hat{y}$, and $\hat{n}_3 = \hat{z}$ is sufficient for the layout to support CCZ without ranged gates. The generalization to local $CZ$ gates in $k + 1$-dimensional Bacon-Shor codes is straightforward (including $k = 1$).

Appendix C: The conditions for Pauli recovery and fault-tolerant circuits that violate them

In this section, our goal is to decide when circuits of CCZ gates between Bacon-Shor codeblocks are fault-tolerant, and in particular, if they are able to be made fault-tolerant by using only the “standard” stabilizer code recovery: a projective measurement of all stabilizers followed by a classically controlled recovery chosen from the Pauli group $\mathcal{P}$. Actually, we define two special-case recovery procedures – stabilizer projective recovery (SPR) and stabilizer projective Pauli recovery (SPPR) – the latter (a subset of the former) corresponding to the stabilizer code “standard” $\mathcal{C}$. We then elucidate when SPR and SPPR exist for a given error channel on a stabilizer code. Next, we define CCZ-form circuits and prove a simpler characterization of the existence of SPR for them. Then, we bound the asymmetry of Bacon-Shor codes required such that CCZ-form circuits implementing CCZ are fault-tolerant using only SPR. Finally, we discuss the fault-tolerance (using non-Pauli recovery) of CCZ designs on Bacon-Shor codes. Interestingly, while our recovery is not SPR, it still borrows most of its circuitry from that class of recovery.

We begin by formally defining SPR and SPPR.

**Definition 3.** For a stabilizer code with generators $\{s_k\}_{k \in \mathcal{F}}$ define the set of orthogonal projectors indexed by $\alpha \in \{0, 1\}^r$ as

$$P_{\alpha} = \prod_{k=0}^{r-1} \frac{1 + (-1)^{s_k} \gamma_k}{2}. \quad (C1)$$

Then a stabilizer projective recovery (SPR) $\mathcal{R}$ is defined

$$\mathcal{R}(\sigma) = \sum_{\alpha \in \{0,1\}^r} \mathcal{R}_\alpha (P_{\alpha} \sigma P_{\alpha}). \quad (C2)$$

for arbitrary quantum channels $\mathcal{R}_\alpha$. If $\mathcal{R}_\alpha(\rho) = U_\alpha \rho U_\alpha^\dagger$ with $U_\alpha \in \mathcal{P}$ for all $\rho$ and all $\alpha$, then the recovery is stabilizer projective with Pauli recovery (SPPR).

That is, both SPR and SPPR assume that a complete set of stabilizers is measured, followed by a classically controlled channel. In SPPR this channel is simply a Pauli operator. Stabilizer codes and logical gates on stabilizer codes traditionally use SPPR, as this is sufficient for fault-tolerance of transversal operations.

However, SPR and SPPR are inherently weaker than the entire class of recovery allowed by the Knill-Laflamme conditions.

**Theorem 4 (Knill-Laflamme [13]).** Consider a quantum code with projector $P$ and quantum operation $\mathcal{E}$ with Kraus operators $\{E_j\}_j$. There exists a recovery operation $\mathcal{R}$ correcting $\mathcal{E}$ (i.e. $\mathcal{R}(\mathcal{E}(\rho)) \propto \rho$ for all $\rho$ in the codespace) if and only if

$$PE_j^\dagger E_j P = \gamma_{ij} P, \quad (C3)$$

for a Hermitian matrix $\gamma$. 

![FIG. 4. A stack of a constant number (e.g. 3) of 2D, Z-gauge Bacon-Shor codes can be visualized in 2D.](image)
For the proof of this famous theorem, we refer to \([13, 26]\). We note here a subtlety, however. In general, a trace-preserving error channel \(\mathcal{E}\) is never exactly correctable, because it can conceivably involve catastrophic yet very low probability events. Indeed, it is more reasonable that Eq. \((C3)\) is merely approximately satisfied. To make this concrete, introduce an error parameter \(\epsilon\) (e.g. this could be the depolarizing error rate), and demand that

\[
PE_i^\dagger (\epsilon) E_j (\epsilon) P = \gamma_{ij} (\epsilon) P + O (\epsilon^d). 
\]  

(C4)

We say the code has effective distance \(d\) (with respect to the error channel \(\mathcal{E}\)) if this approximate condition holds. In \([27]\) it is argued that Eq. \((C4)\) is sufficient for the existence of a recovery operation \(R\) such that the fidelity \(F\) is bounded like

\[
F := \min_{\psi \in \mathcal{C}} |\langle \psi \mid \mathcal{R} \circ \mathcal{E} (|\psi \rangle \langle \psi|) \rangle| \geq 1 - O (\epsilon^d). 
\]  

(C5)

The minimization is over all states in the codespace \(\mathcal{C}\).

As an example, single-qubit depolarizing noise over all \(n\) qubits in a code has Kraus operators

\[
\{ \epsilon^{\vec{v}} \sigma_1^{v_1} \otimes \sigma_2^{v_2} \cdots \otimes \sigma_n^{v_n} : \vec{v} \in \{ 0, 1, 2, 3 \}^n \},
\]  

(C6)

where subscripts indicate the affected qubit and superscripts the Pauli operator. The Hamming weight \(|\vec{v}|\) is defined as the number of non-zero elements of \(\vec{v}\). In this case, effective distance \(d\) implies error-correction of all errors of weight \(< d\), as per the definition of code distance. Later, we consider the effective distance of circuit noise channels, those in which each Kraus operator is a collection of circuit faults that have propagated to the end of the circuit.

Generalizing similar ideas to entanglement fidelity has yielded both necessary and sufficient conditions for approximate correction \([28]\). While these ideas are likely applicable here, for simplicity we stick with the notion of approximate correction in Eq. \((C4)\).

Our immediate goal, however, is to develop conditions analogous to Knill-Laflamme for performing SPR.

**Theorem 5.** Given a stabilizer code with projector \(P\) and a quantum operation \(\mathcal{E}\) with Kraus operators \(\{ E_j \}\), there exists a SPR \(R\) correcting \(\mathcal{E}\) if and only if there exist hermitian matrices \(\gamma^{(\alpha)}\) such that for all \(\alpha, i, j\)

\[
PE_i^\dagger P_\alpha E_j P = \gamma_{ij}^{(\alpha)} P. 
\]  

(C7)

**Proof.** Our arguments closely follow \([26]\). Indeed, we might consider \(P_\alpha E_j\) to be the operator elements of a new error channel and apply Theorem 4. Yet, this does not necessarily guarantee the recovery is SPR as we have defined it, and so we run through the complete proof for definitiveness. Note that because \(\sum_\alpha P_\alpha = I\), summing Eq. \((C7)\) over \(\alpha\) implies Eq. \((C3)\); SPR is indeed a subset of Knill-Laflamme recovery.

We prove the reverse direction first. Since each \(\gamma^{(\alpha)}\) is hermitian, each can be diagonalized \(d^{(\alpha)} = u^{(\alpha)} \gamma^{(\alpha)} u^{(\alpha)*}\). Define unitary equivalent Kraus operators

\[
F_j^{(\alpha)} = \sum_j u_j^{(\alpha)} E_j. 
\]  

That is, \(\mathcal{E}^{(\alpha)}\) with Kraus operators \(\{ F_j^{(\alpha)} \}\) satisfies \(\mathcal{E}^{(\alpha)} (\rho) = \mathcal{E} (\rho)\) for all \(\rho\) and \(\alpha\). Notice furthermore that for all \(\alpha\)

\[
P F_j^{(\alpha)} P^\dagger P^{(\alpha)} = d_{kl}^{(\alpha)} \rho, 
\]  

by using the definition of \(F_j^{(\alpha)}\) and Eq. \((C7)\).

Use the polar decomposition \(A = U \sqrt{A^\dagger A}\) on \(P_\alpha F_j^{(\alpha)} P\). This guarantees the existence of unitaries \(U_j^{(\alpha)}\) such that

\[
P_\alpha F_j^{(\alpha)} P = \sqrt{d_{kl}^{(\alpha)}} U_j^{(\alpha)} P. 
\]  

(C9)

For any fixed \(\alpha\), the projectors \(P_j^{(\alpha)} = U_j^{(\alpha)} P U_j^{(\alpha)*}\) are orthogonal,

\[
P_l^{(\alpha)} P_k^{(\alpha)} = \frac{U_l^{(\alpha)} P F_l^{(\alpha)*} P F_k^{(\alpha)} P}{\sqrt{d_{ll}^{(\alpha)} d_{kk}^{(\alpha)}}} = \frac{d_{kk}^{(\alpha)} F_l^{(\alpha)} P U_k^{(\alpha)} \dagger}{\sqrt{d_{ll}^{(\alpha)} d_{kk}^{(\alpha)}}}, 
\]  

(C10)

which is zero when \(l \neq k\). If \(\sum \rho_j^{(\alpha)} < I\) we can add another projector to complete the set and define

\[
R_\alpha (\rho) = \sum_k U_k^{(\alpha)*} \rho_j^{(\alpha)} P_k^{(\alpha)} U_k^{(\alpha)} \rho. 
\]  

(C12)

We can now show \(\mathcal{R}(\mathcal{E}(\rho) \propto \rho\) for any \(\rho = P \rho P\) in the codespace.

\[
\mathcal{R}(\mathcal{E}(\rho)) = \sum_\alpha \mathcal{R}_\alpha (P_\alpha \mathcal{E}(\rho) P_\alpha) 
\]  

(C13)

\[
= \sum_\alpha \mathcal{R}_\alpha (P_\alpha \mathcal{E}_\alpha (\rho) P_\alpha) 
\]  

\[
= \sum_\alpha \sum_{kl} P U_l^{(\alpha)*} P F_l^{(\alpha)} P \rho P F_l^{(\alpha)*} P U_k^{(\alpha)} P \rho \]  

\[
= \sum_\alpha \sum_{kl} d_{kk}^{(\alpha)} \rho \propto \rho. 
\]  

For the forward direction, we notice that \(\mathcal{R}(\mathcal{E}(P \rho P))\) defines a channel for all \(\rho\) (not just \(\rho\) in the codespace). By the assumption that \(\mathcal{R}\) corrects \(\mathcal{E}\), we have

\[
\mathcal{R}(\mathcal{E}(P \rho P)) = c P \rho P. 
\]  

(C14)

Linearity guarantees \(c\) does not depend on \(\rho\). Now, Eq. \((C14)\) holds for all \(\rho\) and therefore the Kraus operators of the channel on the left and the channel on the right must be unitary related. This means there are constants \(c_{ik}^{(\alpha)}\) such that

\[
R_l^{(\alpha)} P_\alpha E_k P = c_{ik}^{(\alpha)} P. 
\]  

(C15)
if \( \{ R_i^{(\alpha)} \} \) are the Kraus operators of \( R_\alpha \). Thus, using the completeness of these Kraus operators,

\[
\left( \sum_l c_{ij}^{(\alpha)*} c_{ik}^{(\alpha)} \right) P = PE_j^\dagger P_\alpha \left( \sum_l R_i^{(\alpha)*} R_i^{(\alpha)} \right) P_\alpha E_k P
\]

\[(C16)\]

\[
= PE_j^\dagger P_\alpha E_k P.
\]

\[(C17)\]

Because the parenthesized term on the left is a hermitian matrix, this is what we set out to show. □

Theorem 5 says that SPR works if projection to the codespace \( P_\alpha \) does not destroy the orthogonality of the error operators \( E_j \). Analogous to the notion of effective distance defined in Eq. (C4), we have a notion of effective distance \( d \) using SPR when

\[
P E_j^\dagger (\epsilon) P_\alpha E_j (\epsilon) P = \gamma_{ij}^{(\alpha)} (\epsilon) P + O(\epsilon^2). \tag{C18}\]

An SPR \( R \) exists such that the fidelity is bounded as \( \mathcal{F} \geq 1 - O(\epsilon^2) \) when Eq. (C18) holds.

We also develop conditions for performing SPPR, the stabilizer code standard.

**Theorem 6.** Given a stabilizer code with projector \( P \) and a quantum operation \( \mathcal{E} \) with Kraus operators \( \{ E_j \} \), there exists an SPPR \( \mathcal{R} \) correcting \( \mathcal{E} \) if and only if there exist constants \( c_{\alpha k} \) and unitaries \( U_\alpha \in \mathcal{P} \) such that for all \( j, \alpha \),

\[
P_\alpha E_j P = c_{\alpha j} U_\alpha P. \tag{C19}\]

**Proof.** We prove the reverse direction first. Using the set of Kraus operators \( \{ E_j \} \) and for \( \rho = P \rho P \) in the codespace,

\[
\mathcal{R}(\mathcal{E}(\rho)) = \sum_\alpha \sum_j U_\alpha^\dagger P_\alpha E_j \rho E_j^\dagger P_\alpha U_\alpha,
\]

\[(C20)\]

\[
= \sum_\alpha \sum_j U_\alpha^\dagger (P_\alpha E_j \rho P_\alpha E_j^\dagger) U_\alpha,
\]

\[(C21)\]

\[
= \sum_\alpha \sum_k \lvert c_{\alpha j} \rvert^2 U_\alpha^\dagger U_\alpha \rho U_\alpha^\dagger U_\alpha \tag{C22}\]

\[
\propto \rho. \tag{C23}\]

Now for the forward direction. We assume

\[
\mathcal{R}(\mathcal{E}(P \rho P)) = \sum_\alpha \sum_j U_\alpha^\dagger P_\alpha E_j \rho P E_j^\dagger P_\alpha U_\alpha = c P \rho P,
\]

\[(C24)\]

for constant \( c \) with \( U_\alpha \in \mathcal{P} \) for all \( \alpha \). Because this holds for all \( \rho \), the channel with one operator element \( \sqrt{c} P \) must be unitarily equivalent to the one with elements \( \{ U_\alpha^\dagger P_\alpha E_j P_\alpha \} \). This implies the existence of constants \( c_{\alpha j} \) such that

\[
U_\alpha^\dagger P_\alpha E_j P = c_{\alpha j} P. \tag{C25}\]

Multiplying both sides by \( U_\alpha \), we get Eq. (C19). It is worth noting that using any other unitarily equivalent set of operator elements for \( \mathcal{E} \) will just result in linear combinations of Eq. (C19) over the index \( j \) (and not \( \alpha \)), and so does not change the conclusions. Thus, to confirm SPPR correctability of a channel we only have to verify Eq. (C19) for one set of operator elements.

Theorem 6 says that a Pauli recovery operation can be used whenever projecting the operator elements of the error channel to the orthogonal codespace gives a Pauli error depending only on the projection result.

We now consider Theorems 5 and 6 in the context of CCZ-form circuits.

**Definition 7.** A CCZ-form circuit is composed entirely of CCZ gates, and moreover, the qubits can be partitioned into three sets (say \( A_i \) for \( i = 1, 2, 3 \)) such that any one CCZ gate acts on at most one qubit from each set. Without loss of generality, assume no CCZ gate is repeated (otherwise, they could be canceled). A CCZ gate is defined analogously for \( i = 1, 2, \ldots, k + 1 \).

To characterize when SPR is appropriate for CCZ-form circuits, we need to discuss how errors propagate through them. In general, circuits define “lightcones”, which contain all qubits that may be correlated. Lightcones also bound the region that errors may propagate. To be concrete, let \( C = \{ S_1, S_2, \ldots, S_k \} \) be a circuit broken into timesteps \( S_j \), each timestep a set of gates with disjoint support. For a set of qubits \( Q \) at time \( t \) the forward lightcone of \( Q \) is denoted \( L_t(Q) \). We can define this recursively,

\[
l_v(Q) = \{ i : \exists g \in S_v, j \in Q \text{ s.t. } \{ i, j \} \subseteq \text{supp}(g) \}, \tag{C26}\]

\[
L_t(Q) = L_{t+1}(L_{t+1}(Q)), \tag{C27}\]

with \( L_h(Q) = Q \). The lightcone of a gate \( g \in S_i \) is defined as the lightcone of that gate’s output qubits, \( L(g) = L_t(\text{supp}(g)) \).

However, CCZ-form circuits already restrict the propagation of errors more severely than naïve application of lightcones would suggest. Indeed, it is not hard to verify the following claim.

**Claim 8.** In a CCZ-form circuit with depth \( h \), the failure of a CCZ gate \( g \in S_i \) places

1. at most one \( X \) error per \( A_i \)
2. at most \( Z \) errors on all qubits in the modified lightcone

\[
\tilde{\mathcal{L}}(g) := \bigcup_{v=t+1}^h l_v(\text{supp}(g)). \tag{C28}\]

This follows from the fact that a CCZ gate commutes with Pauli \( Z \), but upon an incoming \( X \) error on one node propagates CZ between its other two nodes.

We now specialize to CCZ-form circuits on three 2D Bacon-Shor codeblocks. The codeblocks define the qubit...
partitioning \{A_i\}. To argue for fault-tolerance, we assume circuit depolarizing noise, i.e. a gate fails (called a fault) with probability \(p\) by applying any Pauli error on its support following the ideal application of the gate. An error-channel \(E\) can be defined as acting on the output qubits with Kraus operators \(\{E_j\}\) representing the products of depolarizing-noise generated Pauli errors propagated to the end of the circuit. A Kraus operator representing \(t\) faults has order \(O(p^{t/2})\). The error parameter \(\epsilon = \sqrt{p}\) is used to define effective distance via Eq. (C18) above. In words, effective distance \(d\) implies that up to \(d-1\) faults are detectable, and any \((d-1/2)\) faults are correctable.

**Lemma 9.** Let \(m, n\) be integers with \(n \geq m\) and \(m \geq 3\). Depolarizing noise in a CCZ-form circuit on three \(m \times n\) Z-gauge Bacon-Shor codeblocks is correctable with effective distance \(m\) using SPR if and only if all gates have modified lightcones (see Claim 8) that intersect any codeblock in at most two rows.

**Proof.** For the forward direction, we assume by way of contradiction that some gate has a lightcone intersecting codeblock \(A_1\) on at least three qubits. We appeal to Theorem 5 and find two \(E_1\) and a codespace projector \(P_\alpha\) that fail Eq. (C18), implying that the circuit is not effective distance \(m\), thereby obtaining a contradiction.

Label rows in \(A_i\) by \(r_1, r_2, \ldots, r_m\) and without loss of generality assume the first three intersect the lightcone of gate \(g\). Let \(E_1\) represent the set of faults containing \(g\) failing with \(XXX\) on its support and single-qubit \(Z\) errors on each of \(r_4, r_5, \ldots, r_{[m/2]+1}\). This is a total of \([m/2]-1\) faults. Let \(E_2\) represent the set of faults again containing \(XXX\) after \(g\), but also single-qubit \(Z\) errors on \(r_{[m/2]+2}, \ldots, r_m\). This is a total of \([m/2]\) faults. Note \(E_1E_2\) is the result of \(m-1 < m\) faults, and thus has order \(O(p^{(m-1)/2})\).

Choose \(P_\alpha\) corresponding to violation (i.e. projection onto the \(-1\)-eigenspace) of the \(Z\)-type stabilizers that indicate \(XXX\) on supp \((g)\) and violation of the \(X\)-type stabilizer spanning rows \(r_{[m/2]+1}\) and \(r_{[m/2]+2}\). All other stabilizers of any codeblock are not violated.

With this setup, we see \(PE_1^\dagger P_\alpha \propto PE_2^\dagger P_\alpha = PE_1^\dagger\), where

\[
E_1^\dagger \propto \prod_{q \in \text{supp}(g)} X_q \prod_{j=1}^{[m/2]+1} Z_{r_j} \tag{C29}
\]

in which \(X_q\) is an \(X\) on qubit \(q\) and \(Z_{r_j}\) indicates \(Z\) on any qubit in row \(r_j\) (they are all equivalent with respect to \(P_\alpha\) or \(P\)). By way of explanation, while propagating \(XXX\) on the support of \(g\) introduces \(Z\) errors, we can collapse these \(Z\) errors to Paulis using the projector \(P_\alpha\). Our choice of \(P_\alpha\) ensures \(Z_{r_1}Z_{r_2}Z_{r_3}\) is the sole result of this collapse.

Likewise, \(P_\alpha E_2 P \propto P_\alpha E_2^\dagger P = E_2^\dagger P\) with

\[
E_2^\dagger \propto \prod_{q \in \text{supp}(g)} X_q \prod_{j=\lceil m/2 \rceil+2}^m Z_{r_j} \tag{C30}
\]

Now notice \(PE_1^\dagger P_\alpha E_2 P \propto PE_2^\dagger E_1^\dagger P\). However, this is not proportional to \(P\) because \(E_2^\dagger\) and \(E_1^\dagger\) differ by a logical operator \(\bar{Z} = \prod_{j=1}^n Z_{r_j}\). So Eq. (C18) fails to hold.

For the reverse direction, it is enough to satisfy Eq. (C18) to notice that a logical error cannot be written onto the data with any combination of fewer than \(m\) faults. We do this using Claim 8. First, \(\bar{X}\) cannot be created because the minimum weight of \(\bar{X}\) is \(n \geq m\) and each CCZ gate failure introduces at most one \(X\) error per block. It remains to argue that \(Z\) errors cannot cause \(\bar{Z}\).

We do this by arguing that it always takes \(t\) faults for every \(t\) \(Z\) errors placed in a specific codeblock, say \(A_1\). Recall our goal is to cause \(\bar{Z}\) exactly with less than \(m\) faults. So every \(X\) error we introduce (and we must introduce at least one, otherwise it clearly takes \(m\) \(Z\) errors to cause \(\bar{Z}_1\)) must also be removed by an additional fault later in the circuit. While a single faulty CCZ gate, failing with some correlated \(X\) errors on its support, may introduce \(\leq 2\) \(Z\) errors to \(A_1\) (because of its restricted modified lightcone, see Claim 8) those \(X\) errors must then be erased via a later fault. Two faults lead to at most two \(Z\) errors. There is one other case to worry about though, when three faults can lead to three \(Z\) errors. If \(E_1\) and \(E_2\) are the \(X\) errors introduced by two different CCZ gates \(g_1\) and \(g_2\) failing, the ability to erase \(E_1E_2\) by failure of a single later CCZ gate \(g^{'\dagger}\) implies that the union of the modified lightcones \(\bar{L}(g_1) \cup \bar{L}(g_2)\) intersects \(A_1\) on at most \(3\) qubits, not \(4\). This is so because \(\bar{L}(g^{'\dagger}) \cap A_i\) (which is not empty) is a subset of both \(\bar{L}(g_1) \cap A_i\) and \(\bar{L}(g_2) \cap A_i\).

We also want some guarantee on the size of circuits for logical operators on Bacon-Shor codes. This might be thought of more a detailed version of Appendix B.

**Lemma 10.** A \(C^k\) Z-form circuit implementing \(C^k\bar{Z}\) on \(m \times n\) 2D Bacon-Shor codes in the \(Z\)-gauge must use \(m^{k+1}\) gates. Moreover, selecting a row from each codeblock, there is exactly one gate joining qubits from all those rows.

**Proof.** The proof proceeds inductively. Begin with \(CZ\). Assume each row is involved in \(\leq m\) \(CZ\) gates, otherwise they could be canceled via \(Z\)-gauge operators. If fewer than \(m^2\) gates were used in a \(CZ\)-form circuit \(C_1\), then there are rows \(r_{A_1}\) and \(r_{A_2}\) from codeblocks \(A_1\) and \(A_2\) that are not connected by a \(CZ\) gate. Thus, \(\overline{X}\) supported on \(r_{A_1}\) is conjugated by \(C_1\) we get \(\overline{X}\) times at most \(m-1\) Pauli \(Z\)s on codeblock \(A_2\). This is not enough to construct \(\overline{Z}\) on codeblock \(A_2\), so the circuit \(C_1\) cannot implement \(CZ\).

For the induction, assume \(C^{k-1}\bar{Z}\) uses \(m^k\) \(C^{k-1}\bar{Z}\) gates. If \(C^k\bar{Z}\) could be implemented with fewer than \(m^{k+1}\) gates by circuit \(C_k\), a gate would be missing that couples some set of rows \(\{r_{A_1}, r_{A_2}, \ldots, r_{A_{k+1}}\}\), one from each codeblock. Then, when \(\overline{X}\) supported on \(r_{A_1}\) is conjugated by \(C_k\), we get \(\overline{X}\) times a \(C^{k-1}\bar{Z}\)-form circuit with at most
\[ m^k - 1 \text{ gates, which is not enough for } C^{k-1}Z \text{ by the inductive assumption. So, } C_k \text{ cannot implement } C^kZ. \]

With Lemmas 9 and 10 in hand, we can argue that CCZ-form circuits implementing CCZ on Bacon-Shor codes cannot be fault-tolerant using SPR if the code is too symmetric.

**Theorem 11.** A CCZ-form circuit for CCZ on an \( m \times n \) 2D Bacon-Shor codes that is fault-tolerant to circuit depolarizing noise with SPR exists if \( n \geq \lceil m/2 \rceil^2 \) and only if \( n \geq \lceil m^2/4 \rceil \).

**Proof.** The "if" claim is constructive and uses the 2-transversality idea of [11]. For each codeblock \( A_i \), partition the rows into \( \lceil m/2 \rceil \) sets \( S_j^{(i)} \) of size at most two (i.e. \( \lceil m/2 \rceil \) pairs and one unpaired row if \( m \) is odd). There are \( \lceil m/2 \rceil^3 \) ways to choose a tuple \( S_j^{(1)}, S_j^{(2)}, S_j^{(3)} \), and with each we will associate a CCZ-form circuit \( C_{jkl} \) with \( \leq 8 \) CCZ gates that connects all rows in the sets in all ways (i.e. is "round-robin" as in [11]). For each column number \( c \in [n] \), choose \( \lceil m/2 \rceil \) of \( C_{jkl} \) that are disjoint. Since there are \( n \geq \lceil m/2 \rceil^2 \) columns, we can do all the \( C_{jkl} \) in parallel. Importantly, the construction of \( C_{jkl} \) guarantees no gate has modified lightcone intersecting any codeblock in more than two rows. So, Lemma 9 guarantees fault-tolerance with SPR. In fact, it is not hard to show SPPR is sufficient as well. Once \( t \) \( X \) errors are located (from the \( Z \)-gauge syndrome data) the possible locations of at most \( 2t \) \( Z \) errors per block are also located.

For the "only if" claim, we notice that any one qubit involved in \( > 4 \) CCZ gates is pigeonholed to have a modified lightcone including at least three rows of a codeblock. Moreover, each row is involved in \( m^2/2 \) gates, which means there exists a qubit involved in at least \( m^2/n \) gates. Thus, appealing to Lemma 9, the bound \( \lceil m^2/n \rceil \leq 4 \) is necessary for SPR to exist. Note that for even \( m \), this bound is equivalent to \( n \geq \lceil m^2/2 \rceil^2 \). \[ \square \]

It remains to describe the fault-tolerance of our \( 3 \times 3 \) code using non-SPR recovery. Recall the circuit Eq. (11). In Fig. 5 we show the timesteps for column \( j = 0 \).

Since CCZ gates are isolated to single columns and other columns are related by qubit permutations within the column we need only consider fault-tolerance of the construction for column \( j = 0 \).

Note first that Lemma 9 is violated. Any gate in timestep \( t = 0 \) has a modified lightcone spanning all rows of codeblocks A and C. These are the only gates violating the Lemma however, so we need only check that all errors (in particular the \( X \) errors) introduced at \( t = 0 \) are correctable. The circuit is already wired in such a way that single \( X \) errors cannot propagate \( Z \)s to more than three rows of any block. In the case of correlated \( X \) errors, the decoder (after measuring just the \( Z \)-gauge) knows a CCZ failed. If there is one \( X \) error per block, it knows exactly which failed and can apply suitable \( X \) and \( CZ \) correction. If two \( X \) errors are present, the decoder knows one of two CCZ gates failed. However, we have built the circuit so that the pair of \( X \) errors has a suitably restricted modified lightcone (\( X \) errors on blocks A and C) or such that \( CZ \)s can be unambiguously applied to correct part of the error propagation (for instance, it is sufficient to correct the \( CZ \) errors resulting from \( t = 2 \) if the possible faulty \( CZ \)s are known to be in either \( t = 0 \) or \( t = 1 \)). Remaining \( Z \) errors are now located to at most two rows of each codeblock, and measurement of the \( X \)-stabilizers is sufficient to correct them. See also Appendix E for an explicit description of the decoder.

**Appendix D: Circuits, volume, and logical gate times in MUSICQ**

In this section, we present circuits for the magic state injection protocols considered in Table I. We also discuss the circuit volume as a metric for comparing logical gates. Finally, we use projected physical gate times from the ion trap MUSICQ architecture [15] to estimate the spacetime volume (in units of \( \mu s \times \text{qubits} \)) and total time of logical CCZ gates.

The magic state used to implement CCZ is

\[ |\text{CCZ}\rangle = \frac{1}{2\sqrt{2}} \sum_{i,j,k \in \{0,1\}} (-1)^{ijk} |ijk\rangle = |CCZ\rangle^{+}_{\otimes 3}. \]

This is the +1-eigenstate of the three stabilizers

\[ S_1 = X_1 C_{Z_{23}}, \quad S_2 = X_2 C_{Z_{13}}, \quad S_3 = X_3 C_{Z_{12}}. \]

Notice \(|+\rangle + |+\rangle |0\rangle\) is already stabilized by \( S_1 \) and \( S_2 \), so we need only measure \( S_3 \) to create \(|\text{CCZ}\rangle\).

This is what is done logically in Fig. 6. This circuit prepares \(|\text{CCZ}\rangle\) fault-tolerantly for any distance three code
FIG. 6. Preparing $|\text{CCZ}\rangle$. The bracketed CZ indicates that logical transversal CZ is to be transversally controlled on the CAT state. CAT states are as large as a codeblock. The measurements are transversal, but the only meaningful result is the parity. Between measurements we perform full error-correction $\mathcal{M}$ on the codeblocks. If both parity measurements are the same, we accept, and if they are both 1, we need to apply $Z$ to the last codeblock.

FIG. 7. Preparing a CAT state that will have at most a single-qubit error if we postselect on the measurement returning 0.

with transversal CZ. We also need a way to prepare CAT states tolerant to one fault. This is done by Fig. 7. Using the $|\text{CCZ}\rangle$ state to implement CCZ on three codeblocks is done using Fig. 8.

FIG. 8. A $|\text{CCZ}\rangle$ state is used to implement CCZ fault-tolerantly. The logical measurements control logical implementations of the stabilizers $S_j$ from Eq. (D3). The final step $\mathcal{M}$ is complete stabilizer error-correction on all codeblocks.

However, relying on these magic-state constructions for implementing logical gates on low-distance codes is not necessarily a good idea. We now establish some metrics for comparing these circuits. Evaluating the magic-state constructions and our $3 \times 3$ Bacon-Shor CCZ, we find the latter decidedly advantaged; see Table I.

While qubit count and circuit depth are two simple circuit metrics, they are more easily manipulated. For example, it is well-known that a single CAT state can be repeatedly prepared, coupled to the data, and measured to extract all stabilizer syndromes. However, this takes a long time and thresholds suffer. A better metric should combine time and space. The circuit volume $CV$ does exactly this. If $n$ qubits are used and qubit $j$ is active for $s_j$ timesteps (i.e. is involved in $s_j$ state initializations, gates, and measurements) then $CV = \sum_{j=1}^{n} s_j$.

We can fine-tune this metric if we know how long each circuit component takes on the physical hardware, e.g. MUSICQ. Then, if a qubit $j$ is active for $t_j$ time, we define the spacetime volume $ST = \sum_{j=1}^{n} t_j$.

Both $CV$ and $ST$ are now relatively easily calculated for the magic-state circuits pictured here and our small CCZ on the $3 \times 3$ Bacon-Shor code in the MUSICQ architecture. MUSICQ single-qubit gates take time $1 \mu s$ and two and three-qubit gates take $10 \mu s$. State-preparation takes $1 \mu s$ and measurement takes $30 \mu s$ [15]. These are not state-of-the-art numbers but rather idealized times. We rounded the numbers in Table I, not because the counting is inexact but because small changes to the circuits and how they are parallelized will change the minor digits. For instance, we assume ancilla code blocks are available immediately when needed, rather than having to wait to reinitialize them (this is in contrast to the MUSICQ time calculation next). In any case, a rough counting suffices to distinguish our construction from the magic-state constructions.

In the interest of learning something about how fast fault-tolerant quantum computers might actually be, our last comparison regards the total time of logical gates, assuming fixed qubit counts. The number of qubits influences how parallelized circuits, such as Steane error-correction Fig. 9, can be. It is a somewhat arbitrary choice, but we attempt consistency by taking just one ancilla codeblock for each codeblock that needs error-correction (implying, for instance, the parts of Fig. 9 are done in series with no parallelization). It also happens that the constructions then fit within a 100 qubit elementary logical unit (ELU) [15]. For our Bacon-Shor construction for example, this means $3 \times 9$ data qubits and $3 \times 9$ ancillas, a total of 54. For magic-states on the 9-qubit code we need $2 \times 3 \times 9$ ancillas to hold the $|\text{CCZ}\rangle$ state and do the error-correction in Fig. 6, and we need $3 \times 9$ more for the data, a total of 81 qubits. For magic-states applied to the 7-qubit code, we use Goto’s method [29] of preparing Steane states with one verification ancilla, so that $3 \times 7 + 3 \times (7 + 1)$ ancillas suffice and $3 \times 7$ data qubits, for a total of 66 qubits.

For the magic-states we count both the time for creating $|\text{CCZ}\rangle$ and the time for injecting it. Only twelve multi-qubit gates can be done in parallel in (our as-
Once the syndrome is extracted, we need to decide on a recovery to perform, a process called decoding. This we perform by table lookup. All LEC, TEC, and Id (except for the CCZ TEC, which we describe separately) follow the same basic scheme. For each pattern of stabilizer measurements, determine the lowest weight Pauli error that is consistent. In particular, we make no attempt to optimize recovery over any structure of the circuit, an approach that may offer marginal, but not at all substantial improvement. Since we want to end in the \(+1\)-eigenspace of the new gauge (e.g. \(Z\)), we should also apply gauge operators (e.g. \(X_{i,j}\)) based on the final set of gauge measurements (e.g. of \(Z_{i,j}\)) to ensure this.

The decoder for the CCZ TEC is more complicated. We first measure \(Z\)-gauge operators. Assuming at most one fault, we learn the locations of all \(X\) errors, at most one per codeblock. The locations of \(X\) errors define a set \(C\) of CCZ gates, such that the failure of any one is capable of causing the \(X\) errors. Find the temporally last CCZ from the set \(C\) and assume it failed. Apply a recovery of \(X\) and \(Z\) gates to correct that failure. Also note that if one of the earlier CCZ gates failed instead, there could still be \(Z\) and \(CZ\) errors remaining on the data. Record all rows of all codeblocks that could still be affected by these errors. Next, \(X\)-gauge operators are measured. The \(X\)-stabilizer information, along with the recorded possible locations of \(Z\)-errors, allows us to correct any remaining \(Z\)-errors (assuming one fault). If there are two or more faults, this procedure will necessarily fail on some cases. For us, if the set \(C\) is empty, we default to the usual TEC used for logical transversal Clifford gates. While this decoding has a lengthy description in words, it can still be precomputed as a simple decoding table, and therefore the required classical computational overhead is just as little as for Clifford TEC.

Having described the exREC circuits and decoding, we now describe our simulation. Our error-model is standard circuit depolarizing noise – a \(q\)-qubit gate \(g\) (including identity \(I\)) is assigned a probability \(p_g\) of failing, and when it does each of the \((4^q - 1)\) non-identity, \(q\)-qubit Pauli errors has a probability \(p_{\text{Pauli}}\) of occurring. Initialization and measurement in Pauli bases are slightly different in that they are unaffected by one type of Pauli error (e.g. initialization of a \([0]\) state is indifferent to a subsequent \(Z\) error). Thus, initialization and measurement fail with probabilities \(p_i\) and \(p_m\), respectively, by suffering from the bad Pauli error. All components succeed or fail independently. The ability to use a separate failure rate for each component is a nice benefit of exact counting. In principle, we need not even use isotropic, or even constant-in-time, depolarizing noise, but do so for simplicity. We also note that, for simplicity, we always assume recovery operations (even non-Pauli) are perfect.

Our ultimate goal is to calculate the probability of failure of an exREC, \(P_{\text{fail}} = Pr[\text{fail}]\). However, doing so exactly would mean considering all combinations of faults, propagating them through the circuits, and checking the exREC correctness condition for each. Instead, as is typ-
cause terms in any given Pauli sum differ only in place-
tocation. In our case, only 
sums in two, regrouping terms based on their commu-
in the sum. Stabilizer measurements break the Pauli 
terms, storing the complex coefficients of each term 
we track all errors exactly, representing them as a sum of 
into several types of Pauli errors because our recovery 
components, are tracked through our 
circuit. For exRECs consisting of Clifford gates (i.e. all 
Pauli after the failed component, are tracked through our 
14

\[ P^{(2)}_{\text{fail}} = \Pr[\text{fail}, \leq 2 \text{ faults}] \quad \text{(E1)} \]

\[ P^{(2)}_{\text{succ}} = \Pr[\neg \text{fail}, \leq 2 \text{ faults}], \quad \text{(E2)} \]

which are functions of the depolarizing rates \( p_a \) for \( a \in \{\text{CCZ, CNOT, } H, I, i, m\} \). As a check, we make sure 
\( P^{(2)}_{\text{fail}} + P^{(2)}_{\text{succ}} = 1 - O(p^{3}_a) \). These two-fault counts provide 
upper and lower bounds on \( P_{\text{fail}} \) like 

\[ P^{(2)}_{\text{fail}} \leq P_{\text{fail}} \leq 1 - P^{(2)}_{\text{succ}}. \quad \text{(E3)} \]

The upper and lower bounds calculated by our counting 
can be seen plotted in Figs. 10 and 11. Solving \( P_{\text{fail}} = p_g \) defines the pseudothreshold for a gate \( g \). Solving instead 
\( 1 - P^{(2)}_{\text{succ}} = p_g \) and \( P^{(2)}_{\text{fail}} = p_g \) gives lower and upper bounds on the pseudothreshold.

Finally, we need to describe how errors, which begin as 
Pauli after the failed component, are tracked through our 
circuits. For exRECs consisting of Clifford gates (i.e. all 
the CCZ exREC) this tracking is a simple application 
of the Gottesman-Knill theorem \([26, 31]\). For the 
CCZ exREC, Pauli errors can become non-Pauli as they 
progress through the circuit. For CCZ, counting only up to two 
faults starts to become inaccurate.

\[ \log P_{\text{succ}}(E) = \log(1 - p_{\text{fail}}), \quad \log P_{\text{fail}}(E) = \log(p_{\text{fail}}). \]

where \( a_\alpha \in \mathbb{C} \) and \( \sigma_\alpha \in \mathcal{P} \) is in the Pauli group. Moreover, since \( E^\dagger E = I \) and thus \( \text{Tr}[E^\dagger E] = 2^n \), we know 
\( \sum |a_\alpha|^2 = 1 \). For two Paulis \( \sigma, \gamma \), we define \( [\sigma, \gamma] = 0 \) if 
they commute and \( [\sigma, \gamma] = 1 \) if they anticommute.

Let us now assume that at the output of the circuit we 
are expecting the pure stabilizer state 

\[ \rho = \prod_{g \in G(S)} \frac{1}{2} (I + g) = \frac{1}{2^n} \sum_{q \in S} q, \quad \text{(E5)} \]

where \( S \) is an Abelian subgroup of \( \mathcal{P} \) and \( G(S) \) is a generating set for \( S \). We actually have \( E \rho E^\dagger \) though. We 
can ask, what is the probability of finding one of the orthonormal states 

\[ \rho_{\vec{m}} = \prod_{g \in G(S)} \frac{1}{2} (I + (-1)^{\vec{m}_s} g) \quad \text{(E6)} \]

at the output? This probability is 

\[ \Pr[\vec{m}] = \text{Tr} [\rho_{\vec{m}} E \rho E^\dagger] \quad \text{(E7)} \]

\[ = \sum_{\alpha} a_\alpha \text{Tr} [\rho_{\vec{m}} \sigma_\alpha \rho E^\dagger] \quad \text{(E8)} \]

\[ = \sum_{\alpha} a_\alpha \text{Tr} [\sigma_\alpha \rho_{\vec{m}(\alpha)} \rho E^\dagger], \quad \text{(E9)} \]
where $\vec{m}(\alpha) g = \vec{m}_g \oplus [\sigma_\alpha, g]$. Now, $\rho \vec{m}(\alpha) \rho = 0$ unless $\vec{m}(\alpha) = \vec{0}$, in which case $\rho \vec{m}(\alpha) = \rho$ and, because of purity, $\rho^2 = \rho$. Thus,

$$\Pr[\vec{m}] = \sum_{\alpha \text{ s.t. } \vec{m}(\alpha) = \vec{0}} a_\alpha \text{Tr}[\sigma_\alpha \rho E^\dagger]$$

(E10)

$$= \sum_{\alpha \text{ s.t. } \vec{m}(\alpha) = \vec{0}} \sum_{\beta} a_\alpha a^*_\beta \text{Tr}[\sigma_\alpha \rho \sigma_\beta].$$

(E11)

Now, if $\sigma_\beta \sigma_\alpha \in S$ (or, equivalently, $\vec{m}(\beta) = \vec{m}(\alpha) = \vec{0}$), then the $(\alpha, \beta)$ trace term in the sum equals 1. However, if $\sigma_\beta \sigma_\alpha \notin S$, then the trace vanishes. We are left to conclude

$$\Pr[\vec{m}] = \left| \sum_{\alpha \text{ s.t. } \vec{m}(\alpha) = \vec{0}} a_\alpha \right|^2.$$  

(E12)