

Targeted Clifford Logical Gates for Hypergraph Product Codes

Adway Patra and Alexander Barg

Department of ECE and Institute for Systems Research, University of Maryland, College Park, MD 20742

Starting with an explicit framework for designing logical Clifford circuits for CSS codes, we construct logical gates for Hypergraph Product Codes. We first derive symplectic matrices for CNOT, CZ, Phase, and Hadamard operators, which together generate the Clifford group. This enables us to design explicit transformations that result in targeted logical gates for arbitrary codes in this family. As a concrete example, we give logical circuits for the $[[18, 2, 3]]$ toric code.

1 Introduction

In the class of stabilizer qubit codes, a large body of work has been devoted to the construction and analysis of various families of quantum codes that have the additional property of low stabilizer weight, known as quantum low-density parity-check, or qLDPC codes [\[Ec200\]](#). These codes have been drawing much attention in quantum coding research because of theoretical results that assert the possibility of fault-tolerant computing with constant overhead [\[8, 17, 22, 27\]](#). This restriction implies that the code stabilizers act nontrivially only on a constant number of physical qubits, which enables low-complexity stabilizer measurement circuits. Among the early constructions in this class are the toric codes [\[Ec200\]](#) [\[24\]](#) with stabilizers defined by the vertices or faces of a square grid with periodic boundaries, their close relatives known as surface codes [\[5\]](#), and other topological codes [\[2, 38\]](#). These families rely on geometrically simple constructions and have low-weight stabilizers, but they only encode a small number of logical qubits (two for the toric code), which results in a low information rate relative to the number of physical qubits. More general families of qLDPC codes afford higher rates while still relying on stabilizers of bounded support. Recent research on qLDPC codes has yielded spectacular advances, including code families of asymptotically nonvanishing rate and a nonvanishing proportion of correctable errors [\[15, 29, 34\]](#).

Fault-tolerant quantum computations rely on error-correction procedures together with circuit design that protects the computational procedure from noise propagation through faulty gates and measurements. Logical operations performed on the encoded data should be compatible with the overall noise protection of the computation. This gives rise to two related, equally important research problems, designing elementary logical operations in the form of explicit circuits, and constructing fault-tolerant implementations that can be realized in hardware. The problem of designing logical operators

Adway Patra: apatra@umd.edu

Alexander Barg: abarg@umd.edu

for quantum error-correcting codes has been at the forefront of quantum algorithms research since the inception of quantum computing [20, 41].

The task of supporting universal quantum computation involves implementing a class of operators in the logical Clifford group, which together with a single non-Clifford gate suffice to provide universality. Since this group is generated by a small number of simple gates, researchers have viewed it as a viable path toward designing practical quantum computations. In this paper we focus on the design of Clifford logical gates for a class of quantum stabilizer codes known as homological product codes $\mathcal{E}_{\text{2D}}^{\text{C}}$ [3], and more specifically, the hypergraph product (HGP) codes $\mathcal{E}_{\text{2D}}^{\text{C}}$ of [42]. These codes belong to the qLDPC family, combining simple description with good error-correcting properties. They encode a number of logical qubits that scales as the constant fraction of the code length n and correct the order of $\sqrt{n}$ errors in the encoding. HGP codes also have a number of favorable properties such as a simple algebraic description, low stabilizer weight, and planar hardware implementation, which places them among the leading candidates for implementing fault-tolerant computations in practice [43, 45].

Notable prior works on the constructions of logical gates for HGP codes include the works [6, 28, 37, 45]. The first of these, [28], provided simple descriptions for logical Pauli operators for HGP codes, and [37] found an explicit basis for these operators. Using it, the authors of [37] designed transversal global Clifford logicals¹ for a subclass of HGP codes called symmetric codes. Their results for such codes also included a circuit implementation for certain targeted CZ and CNOT gates as well as single-qubit Hadamard and Phase gates using state injection. The authors of [6] designed transversal gates for global Clifford operators for a certain class of CSS codes which includes symmetric HGP codes. Finally, [45] constructed targeted Clifford logicals by combining the global gates of [37] with two separate ancilla codes for teleporting some of the logical qubits out of the target code. This design approach leads to increased space complexity because it requires the order of $2n$ ancilla qubits for performing the logic. A very recent work [30] aims at designing non-Clifford targeted logical operators such as CCZ for a general code family of sheaf codes, which contains many recent product constructions of qLDPC codes. Other recent papers [7, 47] use cohomology invariants and cup products to construct constant-depth circuits for a class of Clifford and non-Clifford logical operations in several copies of homological qLDPC codes.

An earlier paper [39] on CSS code logic gave an algorithm for finding all symplectic matrices, and hence circuits, for any logical Clifford operator for general CSS codes. Their approach is built upon the well-known representation of logical Clifford operators by binary symplectic matrices [9, 21]. The algorithm of [39] finds a set of linear equations based on the constraints imposed by the target logical operator and then finds *all* symplectic matrices that perform the action of this operator. Thereafter, one can convert any of these symplectic matrices into circuits using elementary Clifford gates [13, 33]. However, the number of solutions to the linear system grows exponentially with the number of stabilizer generators, making it difficult to optimize the circuit parameters even for moderate-size codes.

In this work we take an alternative approach to finding these symplectic matrices that does not require searching through multiple possibilities, resulting in a general framework for finding explicit circuits for any logical Clifford operations. It is well known that the stabilizer generators of an $[[n, k]]$ CSS code form a part of a symplectic basis of $\mathbb{F}_2^{2n}$. The logical Clifford operators perform a linear transformation between symplectic bases. By specifying this mapping of basis vectors we derive an

¹A *global* operator acts nontrivially on all the logical qubits. As opposed to this, logicals with action limited to a subset of qubits are called *targeted*. Their structure is of interest because it provides added flexibility in algorithm design.

explicit form of the matrix of this transformation, and these matrices precisely implement the Clifford logical operators for the code (up to global phases).

Generally, this technique applies to any CSS code (or, in fact, any stabilizer code with an explicit choice of logical Pauli operators), but we rely on the structure of HGP codes to construct circuits for single-qubit/two-qubit logical operators that span the logical Clifford group. This structure translates into a special form of the symplectic matrices for the logical Cliffords, allowing us to simplify the circuit building algorithm, resulting in simpler explicit procedures for the targeted logicals compared to the state of the art. Our design works for any HGP code and for all logical qubits of the code, lifting the limitations inherent to the procedures of [6, 37, 45]. We illustrate the construction by explicitly designing logical Clifford gates for a prominent example of the HGP code family, namely the toric code. The circuits that we construct do not require any additional ancilla qubits or protocols such as state injection and are built purely out of basic single and two-qubit gates. At the same time, they are not fault-tolerant by design, requiring additional error-correction ancillas to support reliable computation. We leave this challenge for future work.

The structure of the paper is as follows. After covering notation and preliminaries in the next section, we derive symplectic matrices for the logical operators in Sec. 3. Our main results, presented in Sec. 4, are concerned with an explicit set of transformations from the symplectic matrices to the logical circuits for each of the CNOT, CZ, Phase, and Hadamard targeted (single- or two-qubit) logical operators. In Sec. 4.2.5 we present explicit Clifford circuits for the $[[18,2,3]]$ toric code. Some proofs and further examples are collected in Section A.

2 Preliminaries

We begin with the basics on quantum stabilizer and CSS codes, their logical operators, and a description of the HGP code family in the form used in the paper. In the paper for any natural number n , $[n]$ will denote the set $\{1, 2, \dots, n\}$ while for two natural numbers a, b , $a \leq b$, $[a : b] = \{a, a + 1, \dots, b - 1, b\}$. Given an $r \times n$ matrix M , we denote by $M(i, :)$ and $M(:, j)$ its i -th row and j -th column, respectively, and write $M(i_1 : i_2, j_1 : j_2)$ to refer to the submatrix of M indexed by the rows in $[i_1 : i_2]$ and columns in $[j_1 : j_2]$. Finally, we denote by $|x|$ the Hamming weight of the binary vector x .

2.1 Stabilizer and CSS Codes

Let $\mathcal{S}$ be an Abelian subgroup of the Pauli group $\mathcal{P}_n$ on n qubits. A stabilizer code $\mathcal{C}(\mathcal{S})$ is a subspace of the Hilbert space $(\mathbb{C}^2)^{\otimes n}$ that forms the common eigenspace of $\mathcal{S}$ with eigenvalue $+1$ [19]. Below we use the notation $\mathcal{S}$ to denote the stabilizer group or a set of its generators interchangeably. Let $N(\mathcal{S}) = \{M \in \mathcal{P}_n | SM = MS \text{ for all } S \in \mathcal{S}\}$ be the normalizer group of $\mathcal{S}$ in the Pauli group. Of course, $\mathcal{S} \subset N(\mathcal{S})$, so we can consider the quotient group $\mathcal{L} = N(\mathcal{S})/\mathcal{S}$. We denote by $L \in \mathcal{L}$ a representative operator of its coset, which can be chosen arbitrarily. Any element $L \in \mathcal{L}$, $L \neq I$ has the property that, for any $|\psi\rangle \in \mathcal{C}(\mathcal{S})$:

$$SL|\psi\rangle = LS|\psi\rangle = L|\psi\rangle, \quad \text{for all } S \in \mathcal{S}$$

and so $L|\psi\rangle$ is also a codeword of $\mathcal{C}(\mathcal{S})$. We observe that L maps codewords to codewords, i.e., it preserves the codespace. Its elements are Pauli operators, and they perform operations on the encoded (logical) qubits, while not taking the result outside of the codespace. For this reason, they are called

logical Pauli operators. Characterizing logical Paulis and similar logical operators outside the Pauli group is of primary importance to quantum computing as it allows operations on the encoded data that are protected from noise by the properties of the code.

CSS codes $\mathbb{E}_{\text{co}}^{\text{C}}$ [10, 40] form a subclass of stabilizer codes $\mathbb{E}_{\text{co}}$ whose stabilizer set is constructed using a pair of classical binary linear codes (C_X, C_Z) such that $C_X^\perp \subseteq C_Z$. Letting H_X be an $m_X \times n$ parity-check matrix of the classical code C_X , we form the set of X stabilizers of the quantum code as $\otimes_{j \in [n]} X^{(H_X)_{ij}}$ for each row $H_X(i, :)$, $i = 1, \dots, m_X$ of the code C_X , and we do the same for the Z stabilizers and the $m_Z \times n$ matrix H_Z . The X and Z stabilizers commute by design because of the subcode assumption, which is equivalent to $H_X H_Z^\top = 0$.

Equivalently, a CSS code $\mathcal{Q}$ can be described in homological terms. Consider the following 3-term chain complex

$$\mathbb{F}_2^{m_Z} \xrightarrow{H_Z^\top} \mathbb{F}_2^n \xrightarrow{H_X} \mathbb{F}_2^{m_X} \quad (1)$$

with boundary operators ∂_2 and ∂_1 given by the matrices $H_Z^\top$ and H_X , respectively. By the assumption, $C_Z^\perp \subseteq C_X$, which translates to $H_X H_Z^\top = 0$, assuring that the complex is well defined. The elements of $\ker(H_X)$ correspond to Z -type Paulis that commute with all X stabilizers (cycles), while the elements of $\text{Im}(H_Z^\top)$ correspond to Z -type Paulis given by products of the Z -type stabilizer generators (trivial boundaries). Hence, the elements of the (first) homology group $H_1(\mathcal{Q}, \mathbb{F}_2) \cong \ker(H_X) / \text{Im}(H_Z^\top)$ form nontrivial cycles (cycles without boundary), giving rise to Z -type logical Pauli operators on $\mathcal{Q}$. In other words, these are the Z -type Paulis that commute with all X -type stabilizers but are not themselves Z stabilizers.

Flipping the arrows, we also consider the co-complex

$$\mathbb{F}_2^{m_Z} \xleftarrow{H_Z} \mathbb{F}_2^n \xleftarrow{H_X^\top} \mathbb{F}_2^{m_X}.$$

where the matrices $H_X^\top$ and H_Z define the coboundary operators δ_0 and δ_1 . The elements of $\ker(H_Z)$, $\text{Im}(H_X^\top)$ and the first cohomology group $H^1(\mathcal{Q}, \mathbb{F}_2) \cong \ker(H_Z) / \text{Im}(H_X^\top)$, respectively, denote co-cycles, coboundaries and nontrivial or essential cocycles, which give rise to X -type Pauli logical operators. This rephrasing offers a convenient way to find the logical Pauli operators of a CSS code, as fixing a basis of H_1 and H^1 fixes a basis for the logical Pauli space.

2.2 Clifford Groups

Let $\mathbb{U}_n$ denote the group of unitary matrices acting on $(\mathbb{C}^2)^{\otimes n}$. The Clifford group $\text{Cliff}_n \subset \mathbb{U}_n$ (more precisely, the *physical Clifford group*) is the normalizer of the n -qubit Pauli group $\mathcal{P}_n$,

$$\text{Cliff}_n = \{U \in \mathbb{U}_n : U P U^\dagger \in \mathcal{P}_n, \text{ for all } P \in \mathcal{P}_n\}.$$

Recall that the Clifford group Cliff_n can be generated by a small set of operators, for instance, by $\{H, S, CNOT\}$ (the Hadamard, Phase, and $CNOT$ (or CZ) gates), which form a set of elementary gates that can serve as building blocks for Clifford circuits. Below we denote logical operators by adding a bar above the operator symbol, for instance, $\bar{P}$ refers to a logical Pauli operator. The logical Clifford group of an $[[n, k]]$ quantum code $\mathcal{Q}$ is defined as the set of those n -qubit unitary operators that normalize the stabilizer group, $\mathcal{S}$, and normalize the subgroup generated by the logical Paulis, $\bar{\mathcal{P}}_k$, of $\mathcal{Q}$,

$$\bar{\text{Cliff}}_k := \{U \in \mathbb{U}^n : U S U^\dagger \in \mathcal{S}, U \bar{P} U^\dagger \in \bar{\mathcal{P}}_k, \text{ for all } S \in \mathcal{S}, \bar{P} \in \bar{\mathcal{P}}_k\}.$$

For every $U \in \overline{\text{Cliff}}_k$ and every code state $|\psi\rangle$, we have $U|\psi\rangle \in \mathcal{Q}$ and $U\bar{P}|\psi\rangle = (U\bar{P}U^\dagger)U|\psi\rangle$ for every $\bar{P} \in \bar{\mathcal{P}}_k$.

As an outcome of this discussion, consider the operators $U \in \overline{\text{Cliff}}_k \cap \text{Cliff}_n$. It is this operator set that we study in this work because any such U performs a logical operation on the code state (manipulates the encoded qubits) while at the same time affording a quantum circuit that can be formed only of Clifford gates.

2.3 Hypergraph Product Codes

A hypergraph product code of [42], $\mathcal{Q}(C_a, C_b)$, is a CSS code constructed from two classical codes $C_a[n_a, k_a, d_a]$ and $C_b[n_b, k_b, d_b]$ with parity-check matrices $H_a \in \mathbb{F}_2^{m_a \times n_a}$ and $H_b \in \mathbb{F}_2^{m_b \times n_b}$, where $m_a = n_a - k_a, m_b = n_b - k_b$. The X and Z stabilizers of an HGP code are given by the rows of the matrices H_X and H_Z as shown below:

$$H_X = [H_a \otimes I_{n_b} | I_{m_a} \otimes H_b^\top], \quad H_Z = [I_{n_a} \otimes H_b | H_a^\top \otimes I_{m_b}]. \quad (2)$$

It is easy to check that $H_X \cdot H_Z^\top = 2(H_a \otimes H_b^\top) = 0 \pmod{2}$, showing that $\mathcal{Q}$ is a well-defined CSS code. The number of physical qubits (the length) of the code $\mathcal{Q}$ equals $n = n_a n_b + m_a m_b$. To define the dimension and the distance, recall that a transpose code $A^\top$ of a classical code A is the space orthogonal over $\mathbb{F}_2$ to the column space of its parity-check matrix H ; in other words, the columns of H serve as the parity relations for $A^\top$. Then the dimension and distance of the code $\mathcal{Q}$ equal $k = k_a k_b + k_a^\top k_b^\top$ and $d = \min\{d_a, d_b, d_a^\top, d_b^\top\}$. As shown in [42], with a proper choice of the constituent codes, the number of logical qubits $k = \Theta(n)$ and the distance scales as $\Theta(\sqrt{n})$. If the matrices H_a and H_b have small rows weight (give rise to classical LDPC codes), then the code $\mathcal{Q}$ is itself a qLDPC code.

It will be convenient to view the physical qubits of the HGP code arranged in two grids (sectors) of size $n_a \times n_b$ and $m_a \times m_b$ respectively, calling them the left and the right sector. In this convention, we can identify individual qubits by indices $(i, h, L), 1 \leq i \leq n_a, 1 \leq h \leq n_b$ and $(j, l, R), 1 \leq j \leq m_a, 1 \leq l \leq m_b$, which refer to the row and column indices and to one of the two sectors as appropriate. From the definition of the stabilizers in Eq. (2), we see that in this view, each stabilizer, whether X or Z , is supported on a single column or row of left sector and a single column or row of right sector, and within them it coincides with the support of a row or a column of H_a or H_b . Specifically, the factors of X stabilizer generators can be indexed by a pair (h, j) such that the generator $S_X(h, j)$ acts on a subset of qubits in column h of the left sector and row j of the right sector. The Z stabilizer generators can similarly be indexed by pairs (i, l) such that $S_Z(i, l)$ has support on row i of the left sector and column l of the right sector.

A convenient way of thinking of HGP codes in linear-algebraic terms arises from considering the tensor product of 1-complexes. This construction, called *homological product* [3], generalizes the above combinatorial approach as follows. We view the classical codes that form the base of the construction as 1-complexes $\mathbb{F}_2^{n_i} \xrightarrow{H_i} \mathbb{F}_2^{m_i}, i \in \{a, b\}$ and consider the tensor product spaces

$$\mathcal{C}_2 = \mathbb{F}_2^{n_a} \otimes \mathbb{F}_2^{m_a}, \mathcal{C}_1 = \mathbb{F}_2^{n_a} \otimes \mathbb{F}_2^{m_b} \oplus \mathbb{F}_2^{m_a} \otimes \mathbb{F}_2^{m_b}, \mathcal{C}_0 = \mathbb{F}_2^{m_a} \otimes \mathbb{F}_2^{n_b}.$$

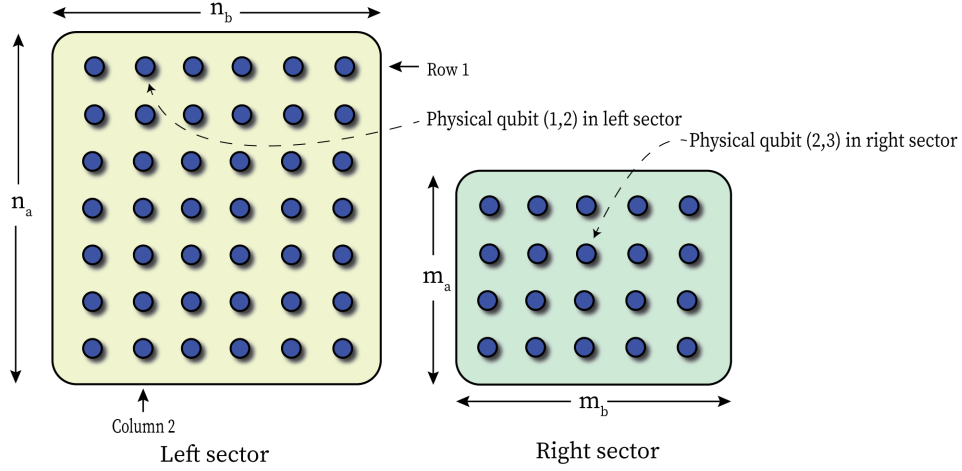


Figure 1: Two-grid view of physical qubits in HGP codes

The homological (hypergraph) product code $\mathcal{Q}$ defined by the matrices H_X, H_Z (2) can be described by the following diagram:

$$\begin{array}{ccccc}
 & & \mathbb{F}_2^{n_a} \otimes \mathbb{F}_2^{m_b} & & \\
 & \nearrow^{I_{n_a} \otimes H_b^\top} & & \nwarrow_{H_a \otimes I_{n_b}} & \\
 \mathbb{F}_2^{n_a} \otimes \mathbb{F}_2^{m_b} & & & & \mathbb{F}_2^{m_a} \otimes \mathbb{F}_2^{n_b} \\
 & \searrow_{H_a \otimes I_{m_b}} & & \nearrow_{I_{m_a} \otimes H_b^\top} & \\
 & & \mathbb{F}_2^{m_a} \otimes \mathbb{F}_2^{m_b} & &
 \end{array} \tag{3}$$

$$\mathcal{C}_2 \xrightarrow{\partial_2} \mathcal{C}_1 \xrightarrow{\partial_1} \mathcal{C}_0$$

where $\partial_2 = H_Z^\top$ and $\partial_1 = H_X$ in accordance with Eq. (1). The bottom part of this diagram describes the *total complex* of the 1-complexes $\mathcal{A} = \mathbb{F}_2^{n_a} \xrightarrow{H_a} \mathbb{F}_2^{m_a}$ and $\mathcal{B} = \mathbb{F}_2^{m_b} \xrightarrow{H_b^\top} \mathbb{F}_2^{n_b}$, written explicitly as

$$\mathbb{F}_2^{n_a} \otimes \mathbb{F}_2^{m_b} \xrightarrow{H_Z^\top} (\mathbb{F}_2^{n_a} \otimes \mathbb{F}_2^{n_b}) \oplus (\mathbb{F}_2^{m_a} \otimes \mathbb{F}_2^{m_b}) \xrightarrow{H_X} \mathbb{F}_2^{m_a} \otimes \mathbb{F}_2^{n_b}, \tag{4}$$

and it is this complex that gives rise to a hypergraph product code. Physical qubits of the code $\mathcal{Q}$ correspond to the basis of the space $\mathcal{C}_1$, and thus the length of the code $\mathcal{Q}$ again is given by $n = n_a n_b + m_a m_b$.

3 Constructing Clifford circuits for CSS codes

Suppose that $\mathcal{Q}$ is a CSS code with k logical qubits. The *logical Pauli operators* map code states to code states non-identically, while preserving the entire codespace. For the code $\mathcal{Q}$ there exist k logical X operators $\bar{X}_i, i = 1, \dots, k$ and k logical Z operators $\bar{Z}_i, i = 1, \dots, k$, all of which commute with the stabilizers. Furthermore, the logical $\bar{X}_i$ operator anticommutes with $\bar{Z}_i$ and commutes with all the other logical Paulis. As discussed above, a basis of logical Z operators $\{\bar{Z}_i\}_{i=1}^k$ can be found

by fixing a basis of the homology space H_1 [19, 44]. By construction, this choice is equivalent to choosing a representative for each coset in $\ker(H_X)/\text{Im}(H_Z^\top)$. Similarly, a choice of the basis for the cohomology space H^1 gives a set of logical X operators $\{\bar{X}_i\}_{i=1}^k$. Note that all the logical Paulis together generate a non-Abelian subgroup $\bar{\mathcal{P}}_k := \langle \{\bar{X}_i, \bar{Z}_i : i = 1, \dots, k\} \rangle$ of $\mathcal{P}^n$ of order $2k$, called the *logical Pauli group*.

It is clear that since these logical operators are products of physical X or Z operators, they can be implemented in parallel in a circuit. However, to support universal quantum computations we also need to be able to implement any logical (non-Pauli) Clifford operation on any subset of logical qubits (even this is not enough for universal computations, which additionally require a non-Clifford operator, but this is outside the scope of this paper). Below we present a procedure to construct circuits for logical Clifford operators for a general CSS code.

3.1 Symplectic representation and circuit design algorithm

Designing logical operators for the Clifford group can be phrased relying on concepts from symplectic geometry over $\mathbb{F}_2$. Given binary vectors $a, b \in \mathbb{F}_2^n$, let

$$D(a, b) := X^{a_1} Z^{b_1} \otimes \dots \otimes X^{a_n} Z^{b_n}.$$

The elements of the n -qubit Pauli group $\mathcal{P}_n$ are then given by the matrices $\pm D(a, b)$ and $\pm i D(a, b)$ for all $a, b \in \mathbb{F}_2^n$. The standard symplectic form is defined as

$$([a, b], [a', b'])_s = a'b^\top + b'a^\top = [a, b]\Omega[a', b']^\top$$

where $\Omega = \begin{bmatrix} 0 & I_n \\ I_n & 0 \end{bmatrix}$. Note that two n -qubit Paulis $D(a, b)$ and $D(a', b')$ commute if and only if their binary representations are symplectically orthogonal $([a, b], [a', b'])_s = 0$. Hence, the mapping $\phi : \mathcal{P}_n \rightarrow \mathbb{F}_2^{2n}$, defined by $\phi(i^\kappa D(a, b)) = [a, b]$, for $\kappa \in \{0, 1, 2, 3\}$, allows an efficient representation of any n qubit Paulis up to the global phase. Let $\tilde{\mathcal{P}}_n$ refer to the group $\mathcal{P}_n / \langle i I_n \rangle$, i.e, the Pauli group with the center quotiented out. With a minor abuse of notation, $\phi : \tilde{\mathcal{P}}_n \rightarrow \mathbb{F}_2^{2n}$ is an isomorphism. The usefulness of this representation for the construction of quantum codes was realized in [9]; see also [21, 25] for an overview.

Now recall that elements of Cliff_n are those unitaries that normalize the Pauli matrices and, hence, $\text{Cliff}_n \cong \text{Aut}(\mathcal{P}_n)$. Furthermore, the action of any element $g \in \text{Cliff}_n$ is determined by the action of g on the set $\{D(e_i, 0) : i \in [n]\} \cup \{D(0, e_i) : i \in [n]\}$, where e_i is the standard basis vector, simply because the collection of these sets forms a basis of $\tilde{\mathcal{P}}_n$. This establishes a direct relationship between Clifford operators and binary symplectic matrices.

Definition 3.1. A symplectic matrix is a $2n \times 2n$ binary matrix that preserves the symplectic form. Symplectic matrices form the binary symplectic group $Sp(2n, \mathbb{F}_2)$. Equivalently, $F = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \in Sp(2n, \mathbb{F}_2)$ if and only if $F\Omega F^\top = \Omega$, or in other words

$$AB^\top = BA^\top, \quad CD^\top = DC^\top, \quad AD^\top + BC^\top = I_n.$$

The relationship between symplectic matrices and Clifford operators is formalized in the following theorem.

Theorem 3.1 ([39]). *The automorphism of $\mathcal{P}_n$ induced by $g \in \text{Cliff}_n$ satisfies*

$$gE(a, b)g^\dagger = \pm E([a, b]F_g)$$

where $E(a, b) = i^{ab^\top} D(a, b)$ and $F_g = \begin{bmatrix} A_g & B_g \\ C_g & D_g \end{bmatrix} \in Sp(2n, \mathbb{F}_2)$.

Remark 3.1. Note that Pauli operators, which are themselves in Cliff_n , in this formulation are represented by the identity matrix.

3.2 From symplectic matrices to circuit design

Once the symplectic representation of a Clifford operator is found, we can convert it to a Clifford circuit via decomposing the matrix into elementary row operations. It was shown in [1, 13] that any symplectic matrix can be decomposed into elementary matrices corresponding to single- and two-qubit operations in the set $\{H, S, CNOT, CZ, SWAP\}$. We list these operations in Table 1 along with their effect on an arbitrary symplectic matrix by left and right multiplication. For $g \in \text{Cliff}_n$ with F_g being its corresponding symplectic matrix, let $\{V_{k,L} : k = 1, \dots, g_L\}$ and $\{V_{k,R} : k = 1, \dots, g_R\}$ be two sequences of symplectic matrices from this elementary set such that

$$\left(\prod_{k=1}^{g_L} V_{k,L} \right) F_g \left(\prod_{k=1}^{g_R} V_{k,R} \right) = I.$$

Then, utilizing the fact that these elementary symplectic matrices are invertible, we get the following decomposition

$$F_g = \left(\prod_{k=g_L}^1 V_{k,L}^{-1} \right) \left(\prod_{k=g_R}^1 V_{k,R}^{-1} \right). \quad (5)$$

Translating each of these elementary matrices to the corresponding elementary Clifford gate gives a circuit that performs the desired effect of the unitary U (up to possible Pauli corrections, refer to Sec. 3.4 for details). Alternatively, we could also find a sequence of operations that, when performed on the $I_{2n \times 2n}$ matrix, give us the symplectic matrix F , and then translate this sequence to Clifford gates in the reverse order. We illustrate this process in Example A.1 elsewhere in the paper.

3.3 Building an arbitrary logical Clifford circuit

We now focus on the case when the unitary $g \in \text{Cliff}_n$ is also a logical Clifford operator of an $[[n, k]]$ CSS code. By definition, such a g has to separately normalize the stabilizer space and the logical Pauli space, i.e.,

$$\begin{aligned} \langle \{gD(H_X(i, :), 0)g^\dagger\}_{i=1}^{m_X} \cup \{gD(0, H_Z(i, :))g^\dagger\}_{i=1}^{m_Z} \rangle &= \mathcal{S}, \\ \langle \{g\bar{X}_i g^\dagger\}_{i=1}^k \cup \{g\bar{Z}_i g^\dagger\}_{i=1}^k \rangle &= \bar{\mathcal{P}}_k. \end{aligned}$$

Translating these requirements to the symplectic domain using the map ϕ and Theorem 3.1, we obtain constraints on the corresponding symplectic matrix F_g . Solving the resulting system of linear equations leads to finding the matrices representing logical operators, which can also be converted to circuits [39]. However, as we show below, these matrices can also be found in a more explicit and efficient manner.

Clifford Element	Left Multiplication	Right Multiplication
H_i	Switch the i th row of $(A B)$ and $(C D)$	Switch the i th column of $(\frac{A}{C})$ and $(\frac{B}{D})$
S_i	Add the i th row of $(C D)$ to the i th row $(A B)$	Add the i th column of $(\frac{A}{C})$ to the i th column of $(\frac{B}{D})$
$CNOT_{i,j}$	Add the j th row of $(A B)$ to the i th row of $(A B)$ and adds i th row $(C D)$ to the j th row of $(C D)$	Add the i th column of $(\frac{A}{C})$ to the j th column of $(\frac{A}{C})$ and adds j th column of $(\frac{B}{D})$ to the i th column of $(\frac{B}{D})$
$CZ_{i,j}$	Add the j th row of $(C D)$ to the i th row of $(A B)$ and adds i th row $(C D)$ to the j th row of $(A B)$	Add the i th column of $(\frac{A}{C})$ to the j th column of $(\frac{B}{D})$ and adds j th column of $(\frac{A}{C})$ to the i th column of $(\frac{B}{D})$
$SWAP_{ij}$	Swaps rows i and j of $(A B)$ and rows i and j of $(C D)$	Swaps columns i and j of $(\frac{A}{C})$ and columns i and j of $(\frac{B}{D})$

Table 1: Elementary circuit operations generating the Clifford group and their action on symplectic matrices

For a logical operator $g \in \overline{\text{Cliff}}_k \cap \text{Cliff}_n$, let $\Gamma_g : \overline{\mathcal{P}}_k \rightarrow \overline{\mathcal{P}}_k$ be the corresponding automorphism of the logical Pauli group. Denote the operation enacted by g on the stabilizer group by $\Upsilon_g, \Upsilon_g : \mathcal{S} \rightarrow \mathcal{S}$. As mentioned before, the Z -type Pauli logical operators can be chosen by fixing a basis of the homology space $H_1 = \ker(H_X) / \text{Im}(H_Z^\top)$; let us denote them by $\{\overline{X}_i\}_{i=1}^k$. Similarly, a basis for the cohomology space $H^1 = \ker(H_Z) / \text{Im}(H_X^\top)$ gives the X -type Pauli logical operators; denote them $\{\overline{Z}_i\}_{i=1}^k$. More precisely:

$$\phi(\overline{X}_i) = [l_{i,X}, 0], \quad \phi(\overline{Z}_i) = [0, l_{i,Z}] \quad \text{for all } i = 1, \dots, k.$$

Let L be the $2k \times 2n$ block-diagonal matrix with rows 1 to k given by $\{\phi(\overline{X}_i)\}$ and rows $k+1$ to $2k$ given by $\{\phi(\overline{Z}_i)\}$. Since $\tilde{\mathcal{P}}_n \xrightarrow{\phi} \mathbb{F}_2^{2n}$ is an isomorphism, $\phi(\overline{\mathcal{P}}_k) = \text{span}(L)$, and the automorphism Γ_g can be described by a $2k \times 2k$ invertible binary matrix $M_g^{(L)}$ acting on L from the left. Similarly, if we define $H = \begin{bmatrix} H_X & 0 \\ 0 & H_Z \end{bmatrix}$, then the action of g on the stabilizer space, Υ_g , can be described by a $(n-k) \times (n-k)$ invertible binary matrix $M_g^{(S)}$, acting on H from the left.

Example 3.2. For a set $I \subseteq [k]$, the logical Clifford operation $g := \otimes_{i \in I} \overline{H}_i$ that fixes the stabilizers pointwise is described by the matrices

$$M_g^{(L)} = \sum_{i \in I} \left(e_i^\top e_{i+k} + e_{i+k}^\top e_i \right) + \sum_{i \in [k] \setminus I} \left(e_i^\top e_i + e_{i+k}^\top e_{i+k} \right), \quad M_g^{(S)} = I_{n-k}. \quad \square$$

While $M_g^{(S)}$ can be any matrix from $\text{GL}(n-k, \mathbb{F}_2)$, the matrix $M_g^{(L)}$ in addition to being invertible, also needs to respect the fact that, for all i , $\overline{X}_i$ and $\overline{Z}_i$ anticommute. We describe these conditions

below. Define

$$\begin{aligned} I_{i,X} &= \text{supp}(M_g^{(L)}(i, 1 : k)), & I_{i,Z} &= \text{supp}(M_g^{(L)}(k+i, 1 : k)) \\ J_{i,X} &= \text{supp}(M_g^{(L)}(i, k+1 : 2k)), & J_{i,Z} &= \text{supp}(M_g^{(L)}(k+i, k+1 : 2k)). \end{aligned}$$

Expressing this in a different but equivalent form,

$$\begin{aligned} \phi(\Gamma_g(\overline{X}_i)) &= \left[\sum_{m \in I_{i,X}} l_{m,X}, \sum_{n \in J_{i,X}} l_{n,Z} \right] \\ \phi(\Gamma_g(\overline{Z}_i)) &= \left[\sum_{m \in I_{i,Z}} l_{m,X}, \sum_{n \in J_{i,Z}} l_{n,Z} \right]. \end{aligned}$$

Below for two sets A, B we write $|A|_2 = |B|_2$ as a short notation for $|A| = |B| \pmod{2}$. Further for $P \in \tilde{\mathcal{P}}_n$, define $\phi_X(P) := \phi(P)(1 : k)$, $\phi_Z(P) := \phi(P)(k+1 : 2k)$.

Proposition 3.3. (1) For $i, j \in [k]$ and $P \in \{X, Z\}$,

$$|I_{i,P} \cap J_{j,P}|_2 = |I_{j,P} \cap J_{i,P}|_2.$$

(2) For $i, j \in [k]$,

$$|I_{i,X} \cap J_{j,Z}|_2 = \delta_{ij} + |I_{j,Z} \cap J_{i,X}|_2.$$

Proof. (1) For $i = j$ the claim is trivially satisfied. Assume that $i \neq j$ and let $P = X$. For any two logical X operators $\overline{X}_i, \overline{X}_j$, $(\phi(\overline{X}_i), \phi(\overline{X}_j))_s = 0$. This implies that

$$(\phi(\Gamma_g(\overline{X}_i)), \phi(\Gamma_g(\overline{X}_j)))_s = 0,$$

which, by definition of the symplectic inner product, implies

$$(\phi_X(\Gamma_g(\overline{X}_i)), \phi_Z(\Gamma_g(\overline{X}_j))) = (\phi_Z(\Gamma_g(\overline{X}_i)), \phi_X(\Gamma_g(\overline{X}_j))),$$

where $(\cdot, \cdot)$ is the standard inner product. Writing this relation explicitly,

$$\left(\sum_{m \in I_{i,X}} l_{m,X}, \sum_{n \in J_{j,X}} l_{n,Z} \right) = \left(\sum_{n \in J_{i,X}} l_{n,Z}, \sum_{m \in I_{j,X}} l_{m,X} \right),$$

which implies that $|I_{i,X} \cap J_{j,X}|_2 = |I_{j,X} \cap J_{i,X}|_2$. The case of $P = Z$ is fully analogous.

(2) The proof is similar to the previous one. Let $i, j \in [k]$ and let $\overline{X}_i, \overline{Z}_j$ be a pair of logical operators. Then $(\phi(\overline{X}_i), \phi(\overline{Z}_j))_s = \delta_{ij}$, implying that $(\phi(\Gamma_g(\overline{X}_i)), \phi(\Gamma_g(\overline{Z}_j)))_s = \delta_{ij}$, or

$$(\phi_X(\Gamma_g(\overline{X}_i)), \phi_Z(\Gamma_g(\overline{Z}_j))) = \delta_{ij} + (\phi_Z(\Gamma_g(\overline{X}_i)), \phi_X(\Gamma_g(\overline{Z}_j))).$$

Writing this explicitly,

$$\left(\sum_{m \in I_{i,X}} l_{m,X}, \sum_{n \in J_{j,Z}} l_{n,Z} \right) = \delta_{ij} + \left(\sum_{n \in J_{i,X}} l_{n,Z}, \sum_{m \in I_{j,X}} l_{m,X} \right),$$

which implies that $|I_{i,X} \cap J_{j,Z}|_2 = \delta_{ij} + |I_{j,Z} \cap J_{i,X}|_2$. □

We see that the symplectic representation of Paulis gives us a convenient way to analyze the actions of logical Clifford operators on $\overline{\mathcal{P}}_k$ and $\mathcal{S}$; more precisely, commutation relations of the Pauli operators translate into certain restrictions on the matrices $M_g^{(S)}$ and $M_g^{(L)}$. We further note that the generators of $\mathcal{S}$ and $\overline{\mathcal{P}}_k$ in their binary symplectic form, i.e., the rows of H and L , constitute a part of a symplectic basis of $\mathbb{F}_2^{2n}$.

Definition 3.2. A symplectic basis for $\mathbb{F}_2^{2n}$ is a set of pairs $\{(u_i, v_i)\}_{i=1}^n$ of length- $2n$ binary vectors such that $(u_i, v_j)_s = \delta_{ij}$, $(u_i, u_j)_s = (v_i, v_j)_s = 0$ for all $i, j \in [n]$.

Note that the rows of any matrix in $Sp(2n, \mathbb{F}_2)$ form a symplectic basis for $\mathbb{F}_2^{2n}$. There exists a symplectic Gram-Schmidt orthogonalization procedure that converts the standard basis of $\mathbb{F}_2^{2n}$ to a symplectic basis.

Now let

$$\begin{aligned} u_i &= \phi(\overline{X}_i) = L(i, :), i \in [k], & v_i &= \phi(\overline{Z}_i) = L(k+i, :), i \in [k] \\ u_{i+k} &= H(i, :), i \in [m_X], & u_{i+k+m_X} &= H(i+m_X, :), i \in [m_Z], \end{aligned} \quad (6)$$

where m_X and m_Z are the numbers of rows of H_X and H_Z , respectively. Clearly, $(u_i, v_j)_s = \delta_{ij}$ for $i \in [n], j \in [k]$ and $(u_i, u_j)_s = 0, i, j \in [n], (v_i, v_j)_s = 0, i, j \in [k]$. Hence, fixing a CSS code and fixing a basis of its logical Pauli space gives us a total of $n+k$ vectors that span a subspace of $\mathbb{F}_2^{2n}$. A symplectic matrix F_g corresponding to a specified Clifford logical operator, maps this set of vectors to another set while preserving the symplectic inner products. The new set of vectors also forms a part of a basis (another basis) of $\mathbb{F}_2^{2n}$. Using this, we can explicitly construct F_g in terms of these two sets of vectors. To build an intuition of this process, let us recall the definition of symplectic transvections.

Definition 3.3. Given a vector $x \in \mathbb{F}_2^{2n}$, a symplectic transvection is a map $T_x : \mathbb{F}_2^{2n} \rightarrow \mathbb{F}_2^{2n}$ defined by $T_x(y) = y + (y, x)_s x$. Equivalently, T_x is given by the symplectic matrix $F_x = I + \Omega x^\top x$.

It is well known that symplectic transvections generate $Sp(2n, \mathbb{F}_2)$ [23, Thm. 3.4]. Furthermore, it is known that physical Clifford gates have very simple transvection decompositions. Namely, for n physical qubits, the Hadamard, Phase, CNOT and CZ gates have the following form:

$$\begin{aligned} H_i &\equiv T_{[e_i, e_i]}, \\ S_i &\equiv T_{[0, e_i]}, \\ CNOT_{i \rightarrow j} &\equiv T_{[e_j, 0]} T_{[e_j, e_i]} T_{[0, e_i]}, \\ CZ_{i,j} &\equiv T_{[0, e_i]} T_{[0, e_i + e_j]} T_{[0, e_j]}, \end{aligned} \quad (7)$$

where e_i is a vector of the standard basis of $\mathbb{F}_2^n$. Since H, S , and $CNOT$ generate the physical Clifford group on n qubits (up to Paulis), the above (products of) transvections generate the group of symplectic matrices corresponding to the physical Clifford operators on n qubits. We shall later see that the logical Clifford operator space is also generated by logical operators whose symplectic matrix representations take the form similar to Eq. (7).

While a symplectic transvection T_x transforms any vector y not orthogonal to x , i.e., $(y, x)_s = 1$, to $y + x$, we shall require a slightly different variation of this fact. Let $x, w \in \mathbb{F}_2^{2n}$ be such that $(w, x)_s = 0$ and define

$$T_{w,x}(y) = y + (y, x)_s w.$$

The operator $T_{w,x}(\cdot)$ maps any vector y non-orthogonal to x to a vector $y + w$, and since $w \perp_s x$, it preserves the symplectic form. In other words, the matrix $F_{w,x} = I + \Omega x^\top w$ is symplectic. This

implies that, for a symplectic basis $\{(u_i, v_i)\}_{i=1}^n$, the operator $T_{u_i+u'_i, v_i}$ maps u_i to u'_i while leaving all other basis vectors unchanged. We obtain the following proposition.

Proposition 3.4. *Let $\{(u_i, v_i)\}_{i=1}^n$ and $\{(u'_i, v'_i)\}_{i=1}^n$ be two sets of vectors each constituting a symplectic basis for $\mathbb{F}_2^{2n}$. Then the following matrix*

$$F = I + \sum_{i=1}^n \Omega(v_i^\top(u_i + u'_i) + u_i^\top(v_i + v'_i)).$$

satisfies $u_i F = u'_i, v_i F = v'_i, i = 1, \dots, n$.

Proof. For $r = 1, \dots, n$,

$$u_r F = u_r + \sum_{i=1}^n (u_r \Omega v_i^\top(u_i + u'_i) + u_r \Omega u_i^\top(v_i + v'_i)) = u_r + (u_r + u'_r) = u'_r.$$

Similarly, $v_r F = v'_r$ for all $r = 1, \dots, n$. □

We are now ready to state our first theorem.

Theorem 3.5. *Let g be a Clifford logical operator for an $[[n, k]]$ CSS code that acts on $\overline{\mathcal{P}}_k$ by the matrix $M_g^{(L)}$. Let $L' = (M_g^{(L)} + I_{2k})L$. Assume that g fixes the stabilizers pointwise, i.e., $M_g^{(S)} = I_{n-k}$, and define the matrices*

$$\begin{aligned} A_g &= I + \sum_{i=1}^k l_{i,Z}^\top L'(i, 1:n), & B_g &= \sum_{i=1}^k l_{i,Z}^\top L'(i, n+1:2n) \\ C_g &= \sum_{i=1}^k l_{i,X}^\top L'(k+i, 1:n), & D_g &= I + \sum_{i=1}^k l_{i,X}^\top L'(k+i, n+1:2n). \end{aligned}$$

Then the matrix

$$F_g = \begin{bmatrix} A_g & B_g \\ C_g & D_g \end{bmatrix},$$

is a valid symplectic matrix for the logical operation g .

Proof. Let $\{(u_i, v_i)\}_{i=1}^k \cup \{u_i\}_{i=k+1}^n$ be the vectors defined in Eq. (6). To complete the basis, choose $v_i \in \mathbb{F}_2^{2n}, i \in [k+1:n]$ such that

$$(u_i, v_j)_s = \delta_{ij}, \quad i \in [n], j \in [k+1:n], \quad (v_i, v_j)_s = 0, \quad i \in [n], j \in [k+1:n].$$

Then $\{(u_i, v_i)\}_{i=1}^n$ form a symplectic basis for $\mathbb{F}_2^{2n}$. Now let $\hat{L} = M_g^{(L)}L$ and let

$$\begin{aligned} u'_i &= \hat{L}(i, :), & v'_i &= \hat{L}(k+i, :), & i &= 1, \dots, k, \\ u'_i &= u_i, & v'_i &= v_i, & i &= k+1, \dots, n. \end{aligned}$$

By Proposition 3.4, the matrix

$$\begin{aligned} F_g &= I + \sum_{i \in [k]} \Omega(v_i^\top[u_i + u'_i] + u_i^\top[v_i + v'_i]) \\ &= I + \sum_{i \in [k]} \left(\begin{bmatrix} l_{i,Z}^\top \\ 0 \end{bmatrix} L'(i, :) + \begin{bmatrix} 0 \\ l_{i,X}^\top \end{bmatrix} L'(k+i, :) \right) \\ &= I + \sum_{i \in [k]} \begin{bmatrix} l_{i,Z}^\top L'(i, 1:n) & l_{i,Z}^\top L'(i, n+1:2n) \\ l_{i,X}^\top L'(k+i, 1:n) & l_{i,X}^\top L'(k+i, n+1:2n) \end{bmatrix}. \end{aligned} \tag{8}$$

satisfies $u_i F_g = u'_i, v_i F_g = v'_i, i \in [k]$ and fixes the other vectors in the basis. $\square$

Specializing Theorem 3.5 to the logical $\overline{H}_i, \overline{P}_i, \overline{CNOT}_{i \rightarrow j}$ and $\overline{CZ}_{i,j}$, we obtain the following representations (note the similarities with Eq. (7)).

Corollary 3.6.

- (1) A single-qubit logical Phase gate acting on the i -th logical qubit is given by the following symplectic transvection:

$$\overline{S}_i \equiv T_{[0, l_{i,Z}]}.$$

- (2) A single-qubit logical Hadamard gate acting on the i -th logical qubit is given by the following symplectic transvection:

$$\overline{H}_i \equiv T_{[l_{i,X}, l_{i,Z}]}.$$

- (3) A two-qubit logical CNOT gate with the i -th and j -th logical qubits as control and target, respectively, is given by the following composition of symplectic transvections:

$$\overline{CNOT}_{i \rightarrow j} \equiv T_{[l_{j,X}, 0]} T_{[l_{j,X}, l_{i,Z}]} T_{[0, l_{i,Z}]}.$$

- (4) A two-qubit logical CZ gate between the i -th and j -th logical qubits is given by the following composition of symplectic transvections:

$$\overline{CZ}_{i,j} \equiv T_{[0, l_{i,Z}]} T_{[0, l_{i,Z} + l_{j,Z}]} T_{[0, l_{j,Z}]}.$$

Proof. (1) From Theorem 3.5, the symplectic matrix for $\overline{S}_i$ is given by

$$F_g = I + \begin{bmatrix} 0 & l_{i,Z}^\top l_{i,Z} \\ 0 & 0 \end{bmatrix} = I + \Omega[0, l_{i,Z}]^\top [0, l_{i,Z}] = T_{[0, l_{i,Z}]}.$$

- (2) Similarly, the symplectic matrix for $\overline{H}_i$ is given by

$$F_g = I + \begin{bmatrix} l_{i,Z}^\top l_{i,X} & l_{i,Z}^\top l_{i,Z} \\ l_{i,X}^\top l_{i,X} & l_{i,X}^\top l_{i,Z} \end{bmatrix} = I + \Omega[l_{i,X}, l_{i,Z}]^\top [l_{i,X}, l_{i,Z}] = T_{[l_{i,X}, l_{i,Z}]}.$$

- (3) The symplectic matrix for $\overline{CNOT}_{i \rightarrow j}$ is given by

$$\begin{aligned} F_g &= I + \begin{bmatrix} l_{i,Z}^\top l_{j,X} & 0 \\ 0 & l_{j,X}^\top l_{i,Z} \end{bmatrix} = I + \Omega[l_{j,X}, 0]^\top [l_{j,X}, 0] + \Omega[l_{j,X}, l_{i,Z}]^\top [l_{j,X}, l_{i,Z}] + \Omega[0, l_{i,Z}]^\top [0, l_{i,Z}] \\ &= T_{[l_{j,X}, 0]} T_{[l_{j,X}, l_{i,Z}]} T_{[0, l_{i,Z}]} \end{aligned}$$

- (4) The symplectic matrix for $\overline{CZ}_{i,j}$ is given by

$$\begin{aligned} F_g &= I + \begin{bmatrix} 0 & l_{i,Z}^\top l_{j,Z} + l_{j,Z}^\top l_{i,Z} \\ 0 & 0 \end{bmatrix} = I + \Omega[0, l_{i,Z}]^\top [0, l_{i,Z}] + \Omega[0, l_{j,Z}]^\top [0, l_{j,Z}] \\ &\quad + \Omega[0, l_{i,Z} + l_{j,Z}]^\top [0, l_{i,Z} + l_{j,Z}] \\ &= T_{[0, l_{i,Z}]} T_{[0, l_{i,Z} + l_{j,Z}]} T_{[0, l_{j,Z}]} \end{aligned} \quad \square$$

The above corollary, combined with the fact that any Clifford logical operator can be decomposed into elementary logical Hadamard, Phase and CNOT (or CZ) operators, directly implies the following theorem.

Theorem 3.7. *For an $[[n, k]]$ CSS code, any logical Clifford operator can be represented as a product of transvection operators from the following set:*

$$\{T_{[l_i, X, l_i, Z]}\}_{i=1}^k \cup \{T_{[0, l_i, Z]}\}_{i=1}^k \cup \{T_{[l_j, X, 0]}T_{[l_j, X, l_i, Z]}T_{[0, l_i, Z]}\}_{i, j \in [k], i \neq j}.$$

This theorem implies that any Clifford logical operator can be implemented by first decomposing the k -qubit logical operator into elementary operators ($H, S, CNOT/CZ$) using the Bruhat or transvection [13, 35, 36] decomposition and then performing the corresponding n -qubit transvections on the physical qubits. However, note that the decomposition might not be unique, and the process of circuit building using the elementary building blocks given in Corollary 3.6 may be less efficient in terms of the circuit parameters compared to a direct implementation using Theorem 3.5.

The matrix F_g as defined by Theorem 3.5, acts non-trivially only on the subspace of $\mathbb{F}_2^{2n}$ spanned by the $2k$ vectors $\{u_i, v_i\}_{i=1}^k$. The exact nature of this action is determined by the Clifford logical operation g , i.e., the operational description of the Clifford operator defines a nontrivial mapping between vectors $u_i \rightarrow u'_i$ and $v_i \rightarrow v'_i, i \in [k]$. As for the remaining part of the basis $\{u_i, v_i\}_{i=k+1}^n$, the action of F_g fixes them pointwise.

Recall that generally, a logical Clifford operator normalizes the stabilizer group and does not necessarily fix the stabilizers pointwise. Moreover, so far we have only fixed $n + k$ basis vectors using the stabilizer generators and Pauli logical generators of the CSS code, while neither specifying the other $n - k$ basis vectors nor the action of the Clifford logical operator on them. We shall do that now, which will allow us to construct all the possible symplectic matrices corresponding to a particular logical action on the logical Pauli space.

Definition 3.4 ([1]). Given a set of independent generators of the stabilizer group $\mathcal{S}$, a *destabilizer* for a generator $s \in \mathcal{S}$ is a Pauli operator in $\mathcal{P}_n$ that anticommutes with s and commutes with every other generator and logical Pauli operator. For a CSS code with $n - k$ stabilizer generators defined by H_X and H_Z , there exists a choice of $n - k$ destabilizers. The destabilizers jointly form the destabilizer group $\mathcal{D}$.

Proposition 3.8. *A choice of $n - k$ vectors $\{v_i\}_{i=k+1}^n$ to complete the symplectic basis corresponds to a choice of destabilizers.*

Proof. Recall that by Eq. (6), $u_{k+1}, \dots, u_n$ represent the stabilizers. For any $v_i, i = k + 1, \dots, n$, we have $(u_i, v_i)_s = 1$ and $(u_j, v_i)_s = 0, j \neq i$, i.e., $D(v_i(1 : n), v_i(n + 1 : 2n))$ anticommutes with exactly one stabilizer generator $D(u_i(1 : n), u_i(n + 1 : 2n))$ and commutes with the others. Furthermore, $(u_j, v_i)_s = 0, (v_j, v_i)_s = 0, j = 1, \dots, k$ implies that $D(v_i(1 : n), v_i(n + 1 : 2n))$ commutes with the logical Paulis. Hence $D(v_i(1 : n), v_i(n + 1 : 2n))$ is a valid destabilizer. $\square$

In the next lemma we find the number of choices for the set of destabilizers, each of which gives a different way to complete the basis described before. The proof follows the arguments in Theorem 7 of [39].

Lemma 3.9. *The set of vectors $\{v_i\}_{i=k+1}^n$ can be chosen in $2^{(n-k)(n-k+1)/2}$ different ways.*

In addition to defining an automorphism of the stabilizer group $\mathcal{S}$, a Clifford logical operator induces an automorphism $\Sigma_g : \mathcal{D} \rightarrow \mathcal{D}$ of the destabilizers. Let R be a $(n-k) \times 2n$ matrix with the i -th row equal to v_{k+i} and let $M_g^{(R)}$ be the invertible binary matrix corresponding to the automorphism Σ_g . The following proposition shows that Σ_g is completely described by the action, Υ_g , of g on $\mathcal{S}$.

Proposition 3.10. $(M_g^{(R)})^\top = (M_g^{(S)})^{-1}$.

Proof. We have that $H\Omega R^\top = I_{n-k}$, and since the actions of $M_g^{(S)}$ and $M_g^{(R)}$ on H and R respectively preserve the symplectic inner product, we have

$$M_g^{(S)} H \Omega (M_g^{(R)} R)^\top = I_{n-k},$$

which implies $M_g^{(S)} (M_g^{(R)})^\top = I_{n-k}$. $\square$

Hence, it suffices to define the action of g on the stabilizer space and the logical Pauli space. This gives the total number of symplectic matrices for a given g , as argued in the next lemma.

Lemma 3.11. *The total number of symplectic matrices for a given logical operator specified by Γ_g is at most $|GL(n-k, \mathbb{F}_2)| 2^{(n-k)(n-k+1)/2}$.*

Proof. By Lemma 3.9, there are $2^{(n-k)(n-k+1)/2}$ of ways to complete the symplectic basis. For each such choice, the number of ways of choosing Υ_g (and hence Σ_g) clearly equals $|GL(n-k, \mathbb{F}_2)|$. However, some of the resulting matrices may coincide, so the expression is an upper bound on the number of distinct matrices. $\square$

We are now ready to state a generalized version of Theorem 3.5. While Theorem 3.5 gives one very simple matrix representation for the operator g (the one that acts trivially on all the stabilizers), the next theorem yields all the possible matrices.

Theorem 3.12. *Let g be a Clifford logical operator for an $[[n, k]]$ CSS code with its respective actions on $\overline{\mathcal{P}}_k$ and $\mathcal{S}$ defined by the matrices $M_g^{(L)}$ and $M_g^{(S)}$ respectively. Let R be a valid choice for the destabilizer matrix: $H\Omega R^\top = I_{n-k}$, $L\Omega R^\top = 0$. Let*

$$\begin{aligned} L' &= (M_g^{(L)} + I_{2k})L, & H' &= (M_g^{(S)} + I_{n-k})H \\ R' &= (((M_g^{(S)})^{-1})^\top + I_{n-k})R. \end{aligned}$$

Then the following matrix

$$\begin{aligned} F_g = I + \sum_{i \in [k]} & \begin{bmatrix} l_{i,Z}^\top L'(i, 1 : n) & l_{i,Z}^\top L'(i, n+1 : 2n) \\ l_{i,X}^\top L'(k+i, 1 : n) & l_{i,X}^\top L'(k+i, n+1 : 2n) \end{bmatrix} \\ & + \sum_{i=1}^{n-k} \Omega [R(i, :)^\top H'(i, :) + H(i, :)^\top R'(i, :)] \end{aligned}$$

is a valid symplectic matrix for the logical operation g .

Proof. The proof is similar to Theorem 3.5. $\square$

Although this theorem may yield more efficient circuits compared to Theorem 3.5, explicit results appear difficult to come by because of the large number of symplectic matrices that implement the logical operator g .

3.4 Applying Pauli Correction

Although circuits constructed using the above framework are guaranteed to perform the desired Pauli transformations with respect to their support, they can potentially introduce unwanted phase changes to some Paulis. This may happen because of two reasons, first because symplectic representation ignores global phases and also because the elementary operations of Table 1 as listed there disregard potential Pauli corrections. Note that any global phases introduced in a logical Pauli operator by the circuit can be safely ignored because that would simply change the global phase of a codeword. However, phase changes in a stabilizer require special attention. In that case, the circuit does not perform a logical operation because it does not preserve the code space: namely, some stabilizers S are mapped to $-S$. Note that this is the only possible phase change because a Clifford automorphism cannot change the order of a Pauli in the group $\mathcal{P}_n$, i.e., mapping S to iS is not allowed by definition. A small adjustment to the circuits ensures that the resulting operation performs the same logic and preserves all the stabilizers. As we show next, this is always possible.

Proposition 3.13. *Let g be a Clifford logical operator of an $[[n, k]]$ CSS code, let C be a Clifford circuit on n qubits performing the desired operation on the logical Pauli space, and let U_C be the corresponding unitary matrix. For a set of stabilizer generators $\mathcal{S}$, let $\mathcal{S}_F \subseteq \mathcal{S}$ be the subset such that*

$$U_C S U_C^\dagger = (-1)^{\mathbb{1}_{\{S \in \mathcal{S}_F\}}} S \quad \text{for all } S \in \mathcal{S}.$$

Then there exists $P \in \mathcal{P}_n$, such that $P U_C$ performs the same logical operation g while preserving all the stabilizer generators of $\mathcal{S}$.

Proof. For every $S \in \mathcal{S}_F$, let d_S be a destabilizer for S such that the set $\{d_S : S \in \mathcal{S}_F\}$ is formed of commuting operators. Let $\tilde{P} = \prod_{S \in \mathcal{S}_F} d_S$, then

$$\begin{aligned} \tilde{P} U_C S U_C^\dagger \tilde{P}^\dagger &= -\tilde{P} S \tilde{P}^\dagger = -d_S S d_S^\dagger = S, \quad \text{for all } S \in \mathcal{S}_F, \\ \tilde{P} U_C S U_C^\dagger \tilde{P}^\dagger &= \tilde{P} S \tilde{P}^\dagger = S, \quad \text{for all } S \in \mathcal{S} \setminus \mathcal{S}_F. \end{aligned}$$

Furthermore, for any logical Pauli $\bar{P} \in \bar{\mathcal{P}}_k$, $\tilde{P} \bar{P} \tilde{P}^\dagger = \bar{P}$ by definition of destabilizers. $\square$

Hence after constructing the circuits using the symplectic matrices, one may need to perform an additional layer of Pauli operators to make sure the stabilizer phases do not change. This added layer of operations increases the circuit depth by at most 1 while not affecting the other parameters.

4 Logical Gates for HGP Codes

In this section, we use the above framework together with the matrix representations of Clifford logical operators derived above to construct circuit realizations of these operators for HGP codes. Among the approaches to implementing the Clifford group operators for CSS codes appearing in the literature are: state injection [4], code deformation [16, 28], and others. Our approach is more circuit-focused in the sense that we attempt to construct logical circuits using a small number of quantum gates on the physical qubits. Namely, our approach relies on an explicit form of the bases of the symplectic vector spaces spanned by the stabilizer generators and logical Pauli operators. This approach ties well with HGP codes for which these bases are recorded below.

We shall be interested in the following parameters for such physical Clifford circuits.

Definition 4.1. For a Clifford circuit C acting on a code state of an $[[n, k]]$ CSS code, the support of C , denoted $\chi(C)$, is the subset of physical qubits on which the circuit acts nontrivially. In other words, a circuit C enacting a unitary $U \in \mathcal{U}^n$, has support $\chi(C)$ if and only if U acts as identity on every qubit in $[n] \setminus \chi(C)$ and $\chi(C)$ is the smallest such set.

A Clifford circuit can be implemented by a sequential application of elementary Clifford gates to a quantum state. A *layer* in a Clifford circuit is a set of gates that can be applied simultaneously, or equivalently, gates whose supports do not overlap. This leads us to our second circuit parameter of interest.

Definition 4.2. For a Clifford circuit C operating on an n -qubit state, the depth of C , denoted $\delta(C)$, is the minimum number of layers such that gates in the same layer do not overlap on qubits.

The general goal of circuit design is to minimize both of these parameters, leading to faster implementation with lower complexity and reduced chance of error propagation while performing the desired logical operation on the codespace. We construct explicit circuits that perform single-qubit logical Hadamard, Phase, and two-qubit CNOT, CZ operators for any family of HGP codes. If the length and dimension of the component codes of a HGP code $\mathcal{Q}(C_1, C_2)$ are taken to be about $\sqrt{n}$ (see also Sec.4.2 below), then the support and depth of the constructed circuits scale as $\Theta(\sqrt{n})$. Combined with Theorem 3.7, this implies that any Clifford logical operation on HGP codes that acts on a constant number of logical qubits, will have a Clifford circuit with χ and δ of order $\Theta(\sqrt{n})$.

4.1 Logical Paulis

Understanding the circuitry of logical operators on a quantum code begins with implementing the logical Paulis on the code space, more specifically, being able to perform single qubit logical X and Z Paulis. The logical X and Z operators of an HGP code $\mathcal{Q}(C_a, C_b)$ can be chosen to have support on a line of qubits in either of the two grids (see Fig. 1). The following lemma describes this special and very useful structure.

Lemma 4.1 ([28], Lemma 1). *Let $\mathcal{Q}(C_a, C_b)$ be an HGP code with parity-check matrices H_a and H_b (2). The following is true for the logical Pauli operators of $\mathcal{Q}$:*

1. *The logical X operators are spanned by*

$$((\mathbb{F}_2^{n_a} / \text{Im}(H_a^\top)) \otimes \ker(H_b) | 0^{m_a m_b}) \cup (0^{n_a n_b} | \ker(H_a^\top) \otimes (\mathbb{F}_2^{m_b} / \text{Im}(H_b)))$$

2. *The logical Z operators are spanned by*

$$(\ker(H_a) \otimes (\mathbb{F}_2^{n_b} / \text{Im}(H_b^\top)) | 0^{m_a m_b}) \cup (0^{n_a n_b} | (\mathbb{F}_2^{m_a} / \text{Im}(H_a)) \otimes \ker(H_b^\top))$$

To interpret this lemma, we refer to the description of HGP codes in Sec. 2.3. To connect the claims of the lemma to Fig. 1, choose a standard basis in the quotient space $\mathbb{F}_2^{n_u} / H$, where H is one of $H_a, H_b, H_a^\top, H_b^\top$, and n_u matches this choice (i.e., is one of n_a, n_b, m_a, m_b). Note further that $\text{Im}(H_a^\top) \cong C_a^\perp$ and $\ker(H_a^\top) \cong \mathbb{F}_2^{n_a} / C_a^\perp \cong C_a^*$, and similar claims hold for H_b . Then this lemma implies that any element of a basis of logical X 's can be supported on a single row of the left sector or a single column of the right sector. There are exactly k_a logical X 's supported on a row (any row) of the left sector, and the corresponding binary vectors form a basis of C_a . The same claims are true with

respect to a basis of logical Z 's, with the roles of rows and columns interchanged, and C_b swapped in for the code C_a .

A further specialization of Lemma 4.1 was given in [37]. We begin with citing their definition.

Definition 4.3. An $m \times n$ binary matrix A is said to be *strongly lower triangular* (SLT) if

1. Any column j has an entry (a pivot) p_j such that $A_{ij} = 0$ for all $i > p_j$;
2. $p_{j_1} \neq p_{j_2}$ for $1 \leq j_1 < j_2 \leq n$;
3. Reordering the columns if necessary, if $A_{p_j,j} = 1$, then $A_{p_j,t} = 0$ for $t > j$.

We call a basis SLT if, once written in matrix form, this matrix can be brought to the SLT form upon performing row and column permutations.

Theorem 4.2 ([37]). *Let $\mathcal{Q}(C_a, C_b)$ be an HGP code with constituent parity-check matrices H_a and H_b . Suppose that $\{a_i\}, \{\alpha_j\}, \{b_h\}, \{\beta_l\}$ are SLT bases for $\ker(H_a), \ker(H_a^\top), \ker(H_b), \ker(H_b^\top)$ respectively, where the pivot of a_i is i and similar for the other bases. Then*

1. *The logical X operators are spanned by*

$$\{e_i \otimes b_h | 0^{m_a m_b}\} \cup \{0^{n_a n_b} | \alpha_j \otimes e_l\}$$

for all combinations of $e_i \in (\mathbb{F}_2^{n_a} / \text{Im}(H_a^\top)), e_l \in (\mathbb{F}_2^{m_b} / \text{Im}(H_b))$;

2. *The logical Z operators are spanned by*

$$\{a_i \otimes e_h | 0^{m_a m_b}\} \cup \{0^{n_a n_b} | e_j \otimes \beta_l\}$$

for all combinations of $e_h \in (\mathbb{F}_2^{m_b} / \text{Im}(H_b^\top)), e_j \in (\mathbb{F}_2^{n_a} / \text{Im}(H_a))$.

With this choice of the representatives of the bases, $|\overline{X}_i \cap \overline{Z}_j| = \delta_{ij}$, i.e., every $\overline{X}_i$ overlaps with $\overline{Z}_i$ on a single qubit and is disjoint from all the other $\overline{Z}_j$'s.

4.2 Logical Clifford operations

In this section we give explicit symplectic representations and circuit-building procedures for logical single-qubit and two-qubit Clifford operators for general HGP codes. The first of these steps is performed by specializing Theorem 3.5 and its corollary, Cor. 3.6, to HGP codes. After that, we construct circuit representations of these matrices following the procedure described after Theorem 3.1 using the operations from Table 1.

The circuits we construct perform the following logical tasks: a single-qubit logical Phase, Hadamard, two-qubit logical CNOT, and CZ. Without loss of generality, we will assume that the targeted logical operation is performed in the left sector for the Hadamard and Phase logical gates, noting that gates for the right sector can be derived similarly. We also assume that the targeted operator fixes the stabilizers pointwise, i.e., Υ_g is trivial. While this assumption, inherited from Theorem 3.5, yields a concise descriptions of the symplectic matrices leading to explicit physical circuits, we would like to reiterate that more efficient circuits might be possible by using Theorem 3.12 (this was mentioned earlier at the end of Section 3.3).

Notational conventions: The following notation will be used throughout this section. An HGP code constructed from classical codes defined by the matrices $H_a \in \mathbb{F}_2^{m_a \times n_a}, H_b \in \mathbb{F}_2^{m_b \times n_b}$ will

be denoted by $\mathcal{Q}(C_a, C_b)$. We denote by $\{a_i\}, \{\alpha_j\}, \{b_h\}, \{\beta_l\}$ SLT bases for $\ker(H_a), \ker(H_a^\top), \ker(H_b), \ker(H_b^\top)$, respectively, and denote by $P(\cdot)$ the set of pivots for the basis argument (see Def. 4.3). Throughout, $n = n_a n_n + m_a m_b$ is the number of physical qubits (the length of the HGP code, cf. (2)), and in our statements about asymptotic scaling of the circuit parameters, we implicitly refer to a sequence of HGP codes with increasing n . We also assume throughout that the constituent codes C_a, C_b in the HGP construction are chosen such that their dimension and distance scale as a constant proportion of their length, e.g., $k_a = \Theta(n_a), d_a = \Theta(n_a)$, and n_a, n_b scale in proportion to $\sqrt{n}$. As a result, the weights of the vectors in each of the 4 bases in Theorem 4.2 scale as $\Theta(\sqrt{n})$. These assumptions will be factored into the statements in this section without further mention.

The logical qubits in the two sectors can be naturally labeled as follows: $\{\bar{q}_{i,h,L} : i \in P(\{a_i\}), h \in P(\{b_h\})\}$ in the left sector and $\{\bar{q}_{j,l,R} : j \in P(\{\alpha_j\}), l \in P(\{\beta_l\})\}$ in the right sector. There are a total of $k = k_a k_b + k_a^\top k_b^\top$ logical qubits. We denote by $\overline{\text{Cliff}}_k$ the logical Clifford group for this code and write the logical Paulis as

$$\begin{aligned} \{\bar{X}_{i,h,L} : i \in P(\{a_i\}), h \in P(\{b_h\})\}, & \quad \{\bar{X}_{j,l,R} : j \in P(\{\alpha_j\}), l \in P(\{\beta_l\})\} \\ \{\bar{Z}_{i,h,L} : i \in P(\{a_i\}), h \in P(\{b_h\})\}, & \quad \{\bar{Z}_{j,l,R} : j \in P(\{\alpha_j\}), l \in P(\{\beta_l\})\}. \end{aligned}$$

The sets of physical qubits supporting these operators are given in Theorem 4.2.

4.2.1 Targeted Logical Phase Gate

We limit ourselves to the logicals acting on left-sector qubits since the construction for the right ones is fully analogous. Let $g = \bar{S}_{i,h,L}$ be the single-qubit logical Phase gate on the logical qubit $\bar{q}_{i,h,L}$. The automorphism Γ_g corresponding to the operational description of $\bar{S}_{i,h,L}$ is described as follows:

$$\begin{cases} \Gamma_g(\bar{X}_{i,h,L}) = \bar{X}_{i,h,L} \bar{Z}_{i,h,L} \\ \Gamma_g(\bar{Z}_{i,h,L}) = \bar{Z}_{i,h,L} \\ \Gamma_g(\bar{P}_{i_1,h_1,A}) = \bar{P}_{i_1,h_1,A} \quad \text{for all } (i_1, h_1) \neq (i, h), \bar{P} \in \{\bar{X}, \bar{Z}\}, A \in \{L, R\} \end{cases} \quad (9)$$

Assume further that g fixes the stabilizers pointwise. In the following proposition, we construct a symplectic matrix representation for g . To simplify the notation, let $\tilde{n} = n_a n_b, \tilde{m} = m_a m_b$.

Proposition 4.3. *For $\mathcal{Q}(C_a, C_b)$, a single-qubit logical targeted Phase gate that acts on the (i, h, L) -th logical qubit has the following symplectic representation:*

$$F_{i,h,L}^S = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & \tilde{B} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}$$

where

$$\tilde{B} = (a_i \otimes e_h)^\top (a_i \otimes e_h).$$

Proof. By Corollary 3.6, the logical phase gate on the (i, h, L) -th logical qubit is given by $T_{[0, v_{i,h,L}]}$, where $v_{i,h,L} = [a_i \otimes e_h, 0]$. Hence, $F_{i,h,L}^S = I + \begin{bmatrix} 0 & v_{i,h,L}^\top v_{i,h,L} \\ 0 & 0 \end{bmatrix}$. $\square$

Theorem 4.4. *Let $\mathcal{Q}(C_a, C_b)$ be an HGP code. There exists a Clifford circuit $\mathcal{C}$, with $\chi(\mathcal{C}) = \Theta(\sqrt{n})$ and $\delta(\mathcal{C}) = \Theta(\sqrt{n})$ that implements $\bar{S}_{i,h,L}$.*

Proof. Let $v = (a_i \otimes e_h)$. We shall construct the matrix $F_{i,h,L}^S$ by performing the following row operations on the matrix $I_{2n \times 2n}$:

1. Choose $x \in \text{supp}(v)$. For every $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $(y + n)$ to row $(x + n)$.
2. Add row $(x + n)$ to row x .
3. For every $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $(y + n)$ to row $(x + n)$.

It is easy to see that the row operations produce the matrix F from $I_{2n \times 2n}$; we include a detailed proof in Appendix A.2.1. The next step is to translate these row operations to physical gates using the operations in Table 1. Let I denote the physical qubit indices corresponding to the support of the $\bar{Z}_{i,h,L}$ logical Pauli. Steps (1) and (3) of row operations translate to applying CNOT gates and step (2) translates to applying a Phase gate. Since the circuit realizes a Clifford unitary having F as its symplectic representation, the circuit is guaranteed to fix the support of the stabilizer generators and perform the desired transformations on the logical Pauli space. However, the circuit can still introduce undesired phase changes to some stabilizers, which may change the codespace. It turns out that adding a single X gate to qubit $x \in I$ suffices to prevent this and preserve the stabilizer space with respective phases: we include a complete proof in Appendix A.2.2.

For reference, we give the complete circuit building procedure in Algorithm 1.

Algorithm 1 Logical phase gate circuit design algorithm

- 1: Select any qubit $x \in I$.
 - 2: **CNOT layer**
For $y \in I \setminus \{x\}$, apply CNOT from y to x .
 - 3: Apply Phase on qubit x .
 - 4: **CNOT layer**
For $y \in I \setminus \{x\}$, apply CNOT from y to x .
 - 5: Apply X on qubit x
-

Finally we check the parameters of the circuit $\mathcal{C}$. It only acts on the set of qubits indexed by I , whose size equals the Hamming weight of $|a_i\rangle$, hence $\chi = |a_i|$. Since there are two layers of $|a_i| - 1$ CNOT gates each having the same qubit as their target, and the Phase gates and X gate also operate on this qubit, the depth $\delta = 2|a_i|$. Since $\{a_i\}$ is an SLT basis of $\ker(H_a)$ then by Theorem 4.2, $|a_i| = \Theta(\sqrt{n})$. Hence we conclude that $\chi = \Theta(\sqrt{n}), \delta = \Theta(\sqrt{n})$. $\square$

Note that the sequence of row operations to construct the desired symplectic matrix from the identity matrix is not unique. One can potentially find other sequences of row operations which, when translated to physical gates, could result in circuits with different parameters. As an example, we include an alternative proof of Theorem 4.4 in Appendix A.3. While the circuit constructed there has a larger gate count, it has a simpler design since that all qubits in the support are acted upon in a symmetric way.

4.2.2 Targeted Logical Hadamard Gate

Now let g be the single-qubit logical Hadamard $\bar{H}_{i,h,L}$ acting on logical qubit $\bar{q}_{i,h,L}$, and let $F_{i,h,L}^H$ be its symplectic matrix representation as given in Theorem 3.1. The mapping Γ_g exchanges $\bar{X}$ and $\bar{Z}$, i.e., $\Gamma_g(\bar{X}_{i,h,L}) = \bar{Z}_{i,h,L}$, $\Gamma_g(\bar{Z}_{i,h,L}) = \bar{X}_{i,h,L}$ and acts identically on the logical Paulis outside the (i, h) location. The matrix form of this gate is given in the next proposition.

Proposition 4.5. *For $\mathcal{Q}(C_a, C_b)$, a single-qubit logical targeted Hadamard gate that acts on the (i, h, L) -th logical qubit has the following symplectic representation:*

$$F_{i,h,L}^H = \begin{bmatrix} \tilde{A} & 0_{\tilde{n} \times \tilde{m}} & \tilde{B} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ \tilde{C} & 0_{\tilde{n} \times \tilde{m}} & \tilde{D} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}$$

where

$$\begin{aligned} \tilde{A} &= I_{\tilde{n} \times \tilde{n}} + (a_i \otimes e_h)^\top (e_i \otimes b_h), \\ \tilde{B} &= (a_i \otimes e_h)^\top (a_i \otimes e_h), \\ \tilde{C} &= (e_i \otimes b_h)^\top (e_i \otimes b_h), \\ \tilde{D} &= I_{\tilde{n} \times \tilde{n}} + (e_i \otimes b_h)^\top (a_i \otimes e_h). \end{aligned}$$

Proof. By Corollary 3.6, the logical Hadamard gate on the (i, h, L) -th logical qubit is given by $T_{[u_{i,h,L}, v_{i,h,L}]}$ where $u_{i,h,L} = [e_i \otimes b_h, 0]$, $v_{i,h,L} = [a_i \otimes e_h, 0]$. Hence,

$$F_{i,h,L}^H = I + \begin{bmatrix} v_{i,h,L}^\top u_{i,h,L} & v_{i,h,L}^\top v_{i,h,L} \\ u_{i,h,L}^\top u_{i,h,L} & u_{i,h,L}^\top v_{i,h,L} \end{bmatrix}.$$

□

Theorem 4.6. *Let $\mathcal{Q}(C_a, C_b)$ be an HGP code. There exists a Clifford circuit C , with $\chi(C) = \Theta(\sqrt{n})$ and $\delta(C) = \Theta(\sqrt{n})$ that implements $\bar{H}_{i,h,L}$.*

Proof. Let us rewrite the matrix $F_{i,h,L}^H$ in a more convenient form. Denote $u = (e_i \otimes b_h)$, $v = (a_i \otimes e_h)$, then

$$F_{i,h,L}^H = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u & 0_{\tilde{n} \times \tilde{m}} & v^\top v & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ u^\top u & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} + u^\top v & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}.$$

Let $I = \text{supp}(u)$ and $J = \text{supp}(v)$ be the supports of the logical Paulis $\bar{X}_{i,h,L}$ and $\bar{Z}_{i,h,L}$ and note that, due to the remark after Theorem 4.2, $I \cap J = (i, h, L)$. Let $\rho = n_a(i-1) + h$.

Next, we give a sequence of row operations that transforms the $2n \times 2n$ identity matrix to $F_{i,h,L}^H$.

1. For each $x \in \text{supp}(v) \setminus \{\rho\}$, add row $(x+n)$ to row ρ and add row $(\rho+n)$ to row x .
2. For each $y \in \text{supp}(u)$, exchange rows y and $(y+n)$.

3. For each $y \in \text{supp}(u) \setminus \{\rho\}$, add row $(y + n)$ to row ρ and add row $\rho + n$ to row y .
4. For each $y \in \text{supp}(u)$, exchange rows y and $(y + n)$.
5. For each $y \in \text{supp}(u) \setminus \{\rho\}$, add row y to row ρ and add row $(\rho + n)$ to row $(y + n)$.
6. For each $x \in \text{supp}(v) \setminus \{\rho\}$, add row ρ to row x and add row $(x + n)$ to row $(\rho + n)$.
7. Exchange rows ρ and $(\rho + n)$.

It is not difficult to verify that these operations result in the correct symplectic matrix F . We give a detailed proof in Appendix A.4.1.

Next, we construct the circuit $\mathbf{C}$ by translating each of the steps above to corresponding physical gates, reversing the order of the gates compared to the above steps in accordance with (5). Steps (1) and (3) correspond to CZ gates, steps (2), (4) and (7) correspond to Hadamard gates and steps (5), (6) correspond to CNOT gates. Although this circuit preserves the stabilizers up to their support, it does not preserve the phases of all stabilizers. However, we show in Appendix A.4.2 that adding a single Y gate negates any undesired phase change. The complete circuit design algorithm is described in Algorithm 2 and a conceptual visualization diagram is included in Figure 2.

Algorithm 2 Logical Hadamard gate circuit design algorithm

- 1: Hadamard on qubit ρ
 - 2: **CNOT layer**
For $x \in J \setminus \{\rho\}$, apply *CNOT* from x to ρ
 - 3: **CNOT layer**
For $y \in I \setminus \{\rho\}$, apply *CNOT* from ρ to y
 - 4: **Hadamard layer**
For $y \in I$, apply Hadamard to qubit y
 - 5: **CZ layer**
For $y \in I \setminus \{\rho\}$, apply *CZ* between y and ρ
 - 6: **Hadamard layer**
For $y \in I$, apply Hadamard to qubit y
 - 7: **CZ layer**
For $x \in J \setminus \{\rho\}$, apply *CZ* between x and ρ
 - 8: Y gate on qubit ρ
-

Finally we look at the parameters of the circuit. The circuit acts only on the set of qubits indexed by $I \cup J$ and so $\chi = |a_i| + |b_h| - 1$. Furthermore, qubit ρ is operated on the most in the circuit and appears in $2(|I| + |J|) + 2$ gates, which is the depth δ of the circuit. Since $\{a_i\}$ is an SLT basis of $\ker(H_a)$ and $\{b_h\}$ is an SLT basis of $\ker(H_b)$, by Theorem 4.2, $|a_i| = |b_h| = \Theta(\sqrt{n})$, and we obtain for both parameters χ, δ the order of $\sqrt{n}$ estimate. $\square$

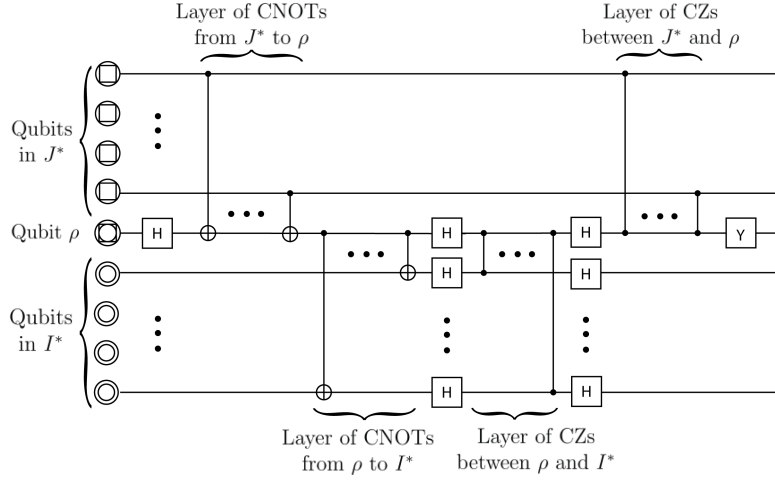


Figure 2: A conceptual visualization of the logical Hadamard circuit. Here $J^* = J \setminus \{\rho\}$, $I^* = I \setminus \{\rho\}$.

4.2.3 Targeted Logical CNOT Gate

In the next two sections we present constructions of two-qubit logical gates. We shall consider two cases according as the two logical qubits are in the same or in different sectors.

Let $\bar{q}_{i_1, h_1, A_1}$ and $\bar{q}_{i_2, h_2, A_2}$ be two distinct logical qubits, with $A_1, A_2 \in \{L, R\}$ and let $g = \overline{CNOT}_{(i_1, h_1, A_1) \rightarrow (i_2, h_2, A_2)}$ denote the logical CNOT operation with (i_1, h_1, A_1) acting as the control and (i_2, h_2, A_2) acting as the target. Again assume that g fixes the stabilizers pointwise. The mapping Γ_g of the logical Paulis is described as follows:

$$\begin{aligned} \Gamma_g(\bar{X}_{i_1, h_1, A_1}) &= \bar{X}_{i_1, h_1, A_1} \bar{X}_{i_2, h_2, A_2}, \\ \Gamma_g(\bar{Z}_{i_2, h_2, A_2}) &= \bar{Z}_{i_1, h_1, A_1} \bar{Z}_{i_2, h_2, A_2}, \\ \Gamma_g(\bar{X}_{i, h, A}) &= \bar{X}_{i, h, A} \quad \text{for all } (i, h, A) \neq (i_1, h_1, A_1), \\ \Gamma_g(\bar{Z}_{i, h, A}) &= \bar{Z}_{i, h, A} \quad \text{for all } (i, h, A) \neq (i_2, h_2, A_2), A \in \{L, R\} \end{aligned} \quad (10)$$

Proposition 4.7. *For $\mathcal{Q}(C_a, C_b)$, the two-qubit targeted logical CNOT with (i_1, h_1, L) as the control and (i_2, h_2, L) as the target has the following symplectic representation:*

$$F_{(i_1, h_1, L) \rightarrow (i_2, h_2, L)}^{CNOT} = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \tilde{D}^\top & 0_{\tilde{n} \times \tilde{m}} & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} + \tilde{D} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}$$

where

$$\tilde{D}^\top = (a_{i_1} \otimes e_{h_1})^\top (e_{i_2} \otimes b_{h_2}).$$

Similarly, the targeted logical CNOT with (i, h, L) as the control and (j, l, R) as the target has the following symplectic representation:

$$F_{(i, h, L) \rightarrow (j, l, R)}^{CNOT} = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & \tilde{D}^\top & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & \tilde{D} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}$$

where

$$\tilde{D}^\top = (a_i \otimes e_h)^\top (\alpha_j \otimes e_l).$$

Proof. By Corollary 3.6, the logical CNOT gate with the (i_1, h_1, L) -th logical qubit as the control and (i_2, h_2, L) -th logical qubit as the target is given by

$$T_{[u_{i_2, h_2, L}, 0]} T_{[u_{i_2, h_2, L}, v_{i_1, h_1, L}]} T_{[0, v_{i_1, h_1, L}]}$$

where $u_{i_2, h_2, L} = [e_{i_2} \otimes b_{h_2}, 0]$, $v_{i_1, h_1, L} = [a_{i_1} \otimes e_{h_1}, 0]$. Hence,

$$F_{(i_1, h_1, L) \rightarrow (i_2, h_2, L)}^{CNOT} = I + \begin{bmatrix} v_{i_1, h_1, L}^\top u_{i_2, h_2, L} & 0 \\ 0 & u_{i_2, h_2, L}^\top v_{i_1, h_1, L} \end{bmatrix}.$$

The case where the two logical qubits are in different sectors can be proven similarly. $\square$

Theorem 4.8. Let $\mathcal{Q}(C_a, C_b)$ be an HGP code.

(a) There exists a Clifford circuit $\mathcal{C}$ that implements $\overline{CNOT}_{(i_1, h_1, L) \rightarrow (i_2, h_2, L)}$ for logical qubits $\bar{q}_{i_1, h_1, L}$ and $\bar{q}_{i_2, h_2, L}$ in the same sector.

(b) There exists a Clifford circuit $\mathcal{C}$ that implements $\overline{CNOT}_{(i, h, L) \rightarrow (j, l, R)}$ for logical qubits $\bar{q}_{i, h, L}$ and $\bar{q}_{j, l, R}$ in different sectors.

In both cases the support $\chi(\mathcal{C})$ and depth $\delta(\mathcal{C})$ are of order $\Theta(\sqrt{n})$

Proof. We shall show that for both cases one can find elementary row operations from Table 1 which, when applied to the identity matrix, produce the desired symplectic matrices of Proposition 4.7.

(a) Consider the case where the two qubits are in different sectors and let $v = (a_i \otimes e_h)$, $u = (\alpha_j \otimes e_l)$. The target matrix is:

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & v^\top u & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & u^\top v & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}$$

Perform the following row operations on $I_{2n \times 2n}$:

1. Select a row index $x \in \text{supp}(v)$. For all $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $y + n$ to row $x + n$.
2. For all $\bar{y} \in \text{supp}(u)$, add row $\bar{y} + \tilde{n}$ to row x and add row $x + n$ to row $\bar{y} + \tilde{n} + n$.
3. For all $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $y + n$ to row $x + n$.

(b) Now consider the case when both logical qubits are in the same sector and let $v = (a_{i_1} \otimes e_{h_1})$, $u = (e_{i_2} \otimes b_{h_2})$. Due to the fact that $\{a_i\}, \{\alpha_j\}, \{b_h\}, \{\beta_l\}$ are SLT bases, we have that $\text{supp}(u) \cap \text{supp}(v) = \emptyset$ whenever $i_1 \neq i_2$ or $h_1 \neq h_2$. The target matrix is:

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u & 0_{\tilde{n} \times \tilde{m}} & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} + u^\top v \end{bmatrix}$$

Perform the following row operations on $I_{2n \times 2n}$:

1. Select a row index $x \in \text{supp}(v)$. For all $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $y + n$ to row $x + n$.
2. For all $\bar{y} \in \text{supp}(u)$, add row $\bar{y}$ to row x and add row $x + n$ to row $\bar{y} + n$.
3. For all $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $y + n$ to row $x + n$.

Note that the only change from the previous case is in the second step. It can be easily verified that in both cases performing operations (1)-(3) on $I_{2n \times 2n}$ produces the desired matrix F . We collect the details of the calculations in Appendix A.5.

The next task is to translate the row operations to physical gates, which is easily done using Table 1. Each of the steps above corresponds to performing a series of physical CNOT gates on specific qubits. Combining the two cases, we give a generic algorithm, Algorithm 3, for constructing the circuit that performs a CNOT between any pair of logical qubits of an HGP code. Let I_c be the set of physical qubits that are in the support of the Z logical Pauli of the control qubit and let I_t be set of qubits in the support of the X logical Pauli of the target qubit. Note that due to the SLT property, I_c and I_t always have empty overlap irrespective of their relative location (the same sector or not).

Algorithm 3 Logical CNOT gate circuit design algorithm

- 1: Select any qubit $x \in I_c$
 - 2: **CNOT layer**
For $y \in I_c \setminus \{x\}$, apply *CNOT* from y to x
 - 3: **CNOT layer**
For $y \in I_t$, apply *CNOT* from x to y
 - 4: **CNOT layer**
For $y \in I_c \setminus \{x\}$, apply *CNOT* from y to x
-

Let us check that the constructed circuits preserve the stabilizers with their phases. In this case this is straightforward because HGP codes belong to the CSS family, and hence the generators of the stabilizer group are composed of either X operators only or Z operators only. A circuit composed purely of CNOT gates cannot introduce any phase change to Paulis composed purely of either X or Z operators. Hence, the stabilizer generators of the code are preserved with their phases.

Clearly, the circuit acts only on the set of qubits indexed by $I_c \cup I_t$ and so $\chi = |a_{i_1}| + |b_{h_2}|$. The circuit depth is at most $\delta = |b_{h_2}| + 2|a_{i_1}| - 2$. Since $\{a_i\}$ is an SLT basis of $\ker(H_a)$ and $\{b_h\}$ is an SLT basis of $\ker(H_b)$, by Theorem 4.2, $|a_{i_1}| = |b_{h_2}| = \Theta(\sqrt{n})$, and we conclude that both χ, δ for the circuit are of order $\Theta(\sqrt{n})$. $\square$

4.2.4 Targeted Logical CZ Gate

Although the logical Hadamard, Phase, and CNOT gates are enough to generate the logical Clifford group, we additionally give a construction of the CZ gate as a possible alternative to CNOT. Again we have to consider two cases depending on whether the logical qubits involved are in the same sector or not.

Let $\bar{q}_{i_1, h_1, A_1}$ and $\bar{q}_{i_2, h_2, A_2}$ be two distinct logical qubits with $A_1, A_2 \in \{L, R\}$ and let $g = \overline{CZ}_{(i_1, h_1, A_1) \leftrightarrow (i_2, h_2, A_2)}$ denote the logical CZ between these qubits, with the arrow indicating that it is symmetric with respect to the two logical qubits involved. For this operator, the logical mapping of Paulis Γ_g is described as follows:

$$\begin{aligned}\Gamma_g(\bar{X}_{i_1, h_1, A_1}) &= \bar{X}_{i_1, h_1, A_1} \bar{Z}_{i_2, h_2, A_2}, \\ \Gamma_g(\bar{X}_{i_2, h_2, A_2}) &= \bar{Z}_{i_1, h_1, A_1} \bar{X}_{i_2, h_2, A_2}, \\ \Gamma_g(\bar{X}_{i, h, A}) &= \bar{X}_{i, h, A} \quad \text{for all } (i, h, A) \neq (i_1, h_1, A_1), (i_2, h_2, A_2), \\ \Gamma_g(\bar{Z}_{i, h, A}) &= \bar{Z}_{i, h, A} \quad \text{for all } (i, h, A), A \in \{L, R\}.\end{aligned}$$

Proposition 4.9. *Let $\mathcal{Q}(C_a, C_b)$ be an HGP code. The two-qubit targeted logical CZ between (i_1, h_1, L) and (i_2, h_2, L) has the following symplectic representation:*

$$F_{(i_1, h_1, L) \leftrightarrow (i_2, h_2, L)}^{CZ} = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & \tilde{B} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}$$

where

$$\tilde{B} = (a_{i_1} \otimes e_{h_1})^\top (a_{i_2} \otimes e_{h_2}) + (a_{i_2} \otimes e_{h_2})^\top (a_{i_1} \otimes e_{h_1}).$$

Similarly, the targeted logical CZ between qubits (i, h, L) and (j, l, R) has the following symplectic representation:

$$F_{(i, h, L) \leftrightarrow (j, l, R)}^{CZ} = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & 0_{\tilde{n} \times \tilde{n}} & \tilde{B} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & \tilde{B}^\top & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}$$

where

$$\tilde{B} = (a_i \otimes e_h)^\top (e_j \otimes \beta_l).$$

Proof. We prove the claim for the case of different sectors. By Corollary 3.6, a logical CZ gate between logical qubits (i, h, L) and (j, l, R) is given by

$$T_{[0, v_{i, h, L}]} T_{[0, v_{i, h, L} + v_{j, l, R}]} T_{[0, v_{j, l, R}]},$$

where $v_{i, h, L} = [a_i \otimes e_h, 0]$, $v_{j, l, R} = [0, e_j \otimes \beta_l]$. Hence,

$$F_{(i, h, L) \leftrightarrow (j, l, R)}^{CZ} = I + \begin{bmatrix} 0 & v_{i, h, L}^\top v_{j, l, R} + v_{j, l, R}^\top v_{i, h, L} \\ 0 & 0 \end{bmatrix}. \quad \square$$

Theorem 4.10. *Let $\mathcal{Q}(C_a, C_b)$ be an HGP code.*

(a) *There exists a Clifford circuit $\mathcal{C}$ that implements $\overline{CZ}_{(i_1, h_1, L) \leftrightarrow (i_2, h_2, L)}$ for logical qubits $\bar{q}_{i_1, h_1, L}$ and $\bar{q}_{i_2, h_2, L}$ in the same sector.*

(b) *There exists a Clifford circuit $\mathcal{C}$ that implements $\overline{CZ}_{(i, h, L) \leftrightarrow (j, l, R)}$ for logical qubits $\bar{q}_{i, h, L}$ and $\bar{q}_{j, l, R}$ in different sectors.*

In both cases the support $\chi(\mathcal{C})$ and depth $\delta(\mathcal{C})$ are of order $\Theta(\sqrt{n})$.

Proof. We will again show that both for cases (a) and (b) we can find a combination of elementary row operations from Table 1 which when performed on the identity matrix produce the desired symplectic matrices of Proposition 4.9.

(a) First let us consider the case where the two qubits are in different sectors and let $v = (a_i \otimes e_h), u = (e_j \otimes \beta_l)$. The target matrix is:

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & 0_{\tilde{n} \times \tilde{n}} & v^\top u \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & u^\top v & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}$$

Perform the following row operations on a $2n \times 2n$ identity matrix in order:

1. Select a row index $x \in \text{supp}(v)$. For all $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $y + n$ to row $x + n$.
2. Select a row index $\bar{x} \in \text{supp}(u)$. For all $\bar{y} \in \text{supp}(u) \setminus \{\bar{x}\}$, add row $\bar{x} + \tilde{n}$ to row $\bar{y} + \tilde{n}$ and add row $\bar{y} + \tilde{n} + n$ to row $\bar{x} + n$.
3. Add row $\bar{x} + \tilde{n} + n$ to row x and add row $x + n$ to row $\bar{x} + \tilde{n}$.
4. For all $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $y + n$ to row $x + n$.
5. For all $\bar{y} \in \text{supp}(u) \setminus \{\bar{x}\}$, add row $\bar{x} + \tilde{n}$ to row $\bar{y} + \tilde{n}$ and add row $\bar{y} + \tilde{n} + n$ to row $\bar{x} + n$.

(b) Now consider the case when both logical qubits are in the same sector and let $v = (a_{i_1} \otimes e_{h_1}), u = (a_{i_2} \otimes e_{h_2})$ and the target matrix is:

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & v^\top u + u^\top v & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}.$$

In this case, the two vectors u and v may have a non-trivial overlap. However, we note that due to the fact that $\{a_i\}, \{b_h\}$ are SLT bases for $\ker(H_a), \ker(H_b)$ respectively, we have $v_{\tilde{n}(i_1-1)+h_1} = 1, u_{\tilde{n}(i_1-1)+h_1} = 0$ and $u_{\tilde{n}(i_2-1)+h_2} = 1, v_{\tilde{n}(i_2-1)+h_2} = 0$. Note that this property was trivially satisfied by the vectors in the previous case because the logical qubits were in different sectors and so the supports of the two logical Z Pauli operators were disjoint. With this property in mind, consider the following row operations on $I_{2n \times 2n}$ in order:

1. For all $y \in \text{supp}(v) \setminus \{\tilde{n}(i_1-1) + h_1\}$, add row $\tilde{n}(i_1-1) + h_1$ to row y and add row $y + n$ to row $\tilde{n}(i_1-1) + h_1 + n$.
2. For all $\bar{y} \in \text{supp}(u) \setminus \{\tilde{n}(i_2-1) + h_2\}$, add row $\tilde{n}(i_2-1) + h_2$ to row $\bar{y}$ and add row $\bar{y} + n$ to row $\tilde{n}(i_2-1) + h_2$.
3. Add row $\tilde{n}(i_1-1) + h_1 + n$ to row $\tilde{n}(i_2-1) + h_2$ and add row $\tilde{n}(i_2-1) + h_2 + n$ to row $\tilde{n}(i_1-1) + h_1$.

4. For all $y \in \text{supp}(v) \setminus \{\tilde{n}(i_1 - 1) + h_1\}$, add row $\tilde{n}(i_1 - 1) + h_1$ to row y and add row $y + n$ to row $\tilde{n}(i_1 - 1) + h_1 + n$.
5. For all $\bar{y} \in \text{supp}(u) \setminus \{\tilde{n}(i_2 - 1) + h_2\}$, add row $\tilde{n}(i_2 - 1) + h_2$ to row $\bar{y}$ and add row $\bar{y} + n$ to row $\tilde{n}(i_2 - 1) + h_2$.

The proof of correctness of the two sequences of row operations is not difficult and is given in Appendix A.6. We translate the row operations to physical gates using Table 1. The row operations in both cases correspond to CNOT and CZ gates. The two cases have been combined by carefully choosing the qubits x and $\bar{x}$ which is always possible as we have argued. Hence, the generic algorithm for constructing the circuit for performing CZ between arbitrary logical qubits for any HP codes is given in Algorithm 4. Let I_1 and I_2 be the sets of physical qubits that are in the support of the Z logical Paulis corresponding to the two logical qubits.

Algorithm 4 Logical CZ gate circuit design algorithm

- 1: Select a qubit $x \in I_1 \setminus I_2$ and select a qubit $\bar{x} \in I_2 \setminus I_1$
 - 2: **CNOT layer**
For $y \in I_1 \setminus \{x\}$, apply *CNOT* from y to x
 - 3: **CNOT layer**
For $\bar{y} \in I_2 \setminus \{\bar{x}\}$, apply *CNOT* from $\bar{y}$ to $\bar{x}$
 - 4: *CZ* between qubits x and $\bar{x}$
 - 5: **CNOT layer**
For $y \in I_1 \setminus \{x\}$, apply *CNOT* from y to x
 - 6: **CNOT layer**
For $\bar{y} \in I_2 \setminus \{\bar{x}\}$, apply *CNOT* from $\bar{y}$ to $\bar{x}$
-

We now show that the circuit given in Algorithm 4 preserves the stabilizers with phases. We shall do this by tracking the changes of any X - or Z -type stabilizer as it progresses along the circuit. Consider the three stages of the circuit: the first stage composed purely of CNOT gates, the second stage of only a single CZ gate, and the third stage composed again of CNOT gates. Any Z -type stabilizer, when acted upon by the successive stages remains a Z type stabilizer with no possible phase change at any point. On the other hand an X type stabilizer can be converted to a mix of X and Z type due to the CZ gate between qubits x and $\bar{x}$ in the second stage. Even if this happens, i.e, the second stage introduces a Z operator on qubit x and/or $\bar{x}$, we know that this change will be reversed by the third stage of CNOT gates by introduction of another Z operator on the corresponding qubit(s) due to the fact that the circuit is guaranteed to fix all the X stabilizers up to their support. These operations do not introduce any negative signs as $XZ = Y, YZ = X, Z^2 = I$. Hence all stabilizers are preserved with their phase.

The circuit acts only on the set of qubits indexed by $I_1 \cup I_2$, hence $\chi = |a_i| + |\beta_l|$, while the circuit depth is $\delta = 2(|a_i| + |\beta_l|) + 1$. The parameter τ is easily calculated to be $\max\{|a_i|, |\beta_l|\}$. Since $\{a_i\}$ is an SLT basis of $\ker(H_a)$ and $\{\beta_l\}$ is an SLT basis of $\ker(H_b^\top)$ then by Theorem 4.2, $|a_i| = |\beta_l| = \Theta(\sqrt{n})$, and we have that χ, δ, τ for the circuit are all of order $\Theta(\sqrt{n})$. Choosing the two logical qubits to be in the same sector only requires minor modification to the algorithm above while the circuit parameters remain of the same order. $\square$

4.2.5 Examples for toric codes

Toric codes form a particular case of the HGP family obtained if the constituent classical codes are repetition codes; see [42, Sec.3] for the details of this specification. To exemplify the general construction presented above, we explicitly construct circuits for logical Cliffords for the $\mathcal{T}_3[[18, 2, 3]]$ toric code. Note that

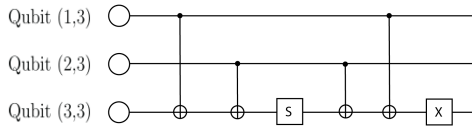
$$H_a = H_b = H_a^\top = H_b^\top = H = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}; \quad \ker(H) = (111),$$

and $P\left(\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}\right) = \{3\}$. We use the two-grid notation introduced in Section 2.3 to denote the physical qubits by (i, h, L) for the left sector and (j, l, R) for the right sector with $i, h, j, l = 1, 2, 3$, ignoring the third index whenever there is no possibility of confusion. The code encodes two logical qubits, which we call the left and the right logical qubit below.

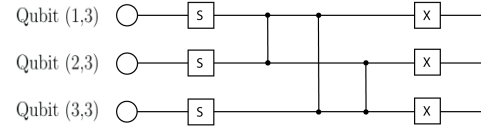
Logical Phase gate on left logical qubit: By Proposition 4.3, the symplectic matrix that performs logical Phase on the left logical qubit is given by:

$$F_{\bar{S}_L} = \left[\begin{array}{cc|cc} I_{18} & v^\top v & 0 \\ 0 & 0 & I_{18} \end{array} \right]$$

with $v = [001001001]$. Using Algorithm 1, we obtain a circuit that implements the logical Phase $\bar{S}_L$, shown in Fig. 3a. An alternative circuit, using Algorithm 5, is given in Fig. 3b.



(a) Circuit implementation using Algorithm 1



(b) Circuit implementation using Algorithm 5

Figure 3: Logical Phase gate $\bar{S}_L$ for the code $\mathcal{T}_3$

Logical Hadamard gate on left logical qubit: By Proposition 4.5, the symplectic matrix that performs the logical Hadamard operation on the left logical qubit is given by:

$$F_{\bar{H}_L} = \left[\begin{array}{cc|cc} I_9 + v^\top u & 0 & v^\top v & 0 \\ 0 & I_9 & 0 & 0 \\ u^\top u & 0 & I_9 + u^\top v & 0 \\ 0 & 0 & 0 & I_9 \end{array} \right]$$

with $u = (000000111)$ and $v = (001001001)$. Using Algorithm 2, we obtain the circuit in Fig. 4 that implements the logical Hadamard operation:

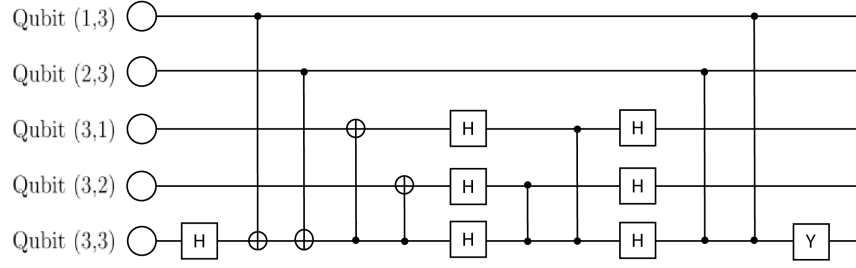


Figure 4: Logical Hadamard gate $\overline{H}_L$ for the code $\mathcal{T}_3$

Logical left-to-right qubit CNOT gate: By Proposition 4.7, the symplectic matrix that performs logical CNOT with left and right logical qubits as control and target, respectively, has the form

$$F_{\overline{H}_L} = \left[\begin{array}{cc|cc} I_9 & v^\top v & 0 & 0 \\ 0 & I_9 & 0 & 0 \\ \hline 0 & 0 & I_9 & 0 \\ 0 & 0 & v^\top v & I_9 \end{array} \right]$$

with $v = (001001001)$. Using Algorithm 3, we obtain the circuit in Fig. 5 for the logical CNOT:

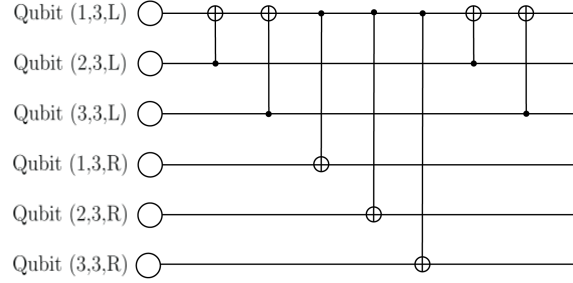


Figure 5: Logical CNOT gate $\overline{CNOT}_{L \rightarrow R}$ for the code $\mathcal{T}_3$

Logical CZ gate between the left and right logical qubit: By Proposition 4.9, the symplectic matrix that performs logical CZ between the left and right logical qubits has the form Using Proposition 4.9, we have the symplectic matrix performing logical CZ between the left and right logical qubit:

$$F_{\overline{H}_L} = \left[\begin{array}{cc|cc} I_9 & 0 & 0 & v^\top u \\ 0 & I_9 & u^\top v & 0 \\ \hline 0 & 0 & I_9 & 0 \\ 0 & 0 & 0 & I_9 \end{array} \right]$$

with $u = (000000111)$ and $v = (001001001)$. Using Algorithm 4, we see that the circuit in Fig. 6 implements the logical operation.

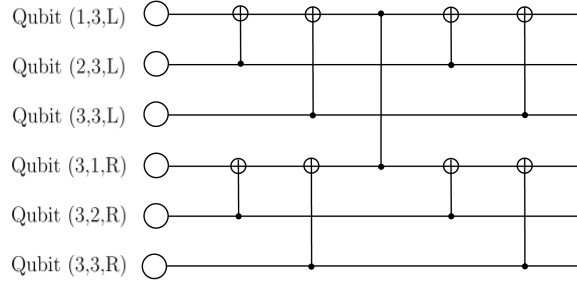


Figure 6: Logical CZ gate $\overline{CZ}_{L,R}$ for the code $\mathcal{T}_3$

Concluding this section, we also mention surface codes [18], which are obtained from toric codes by simply discarding the periodicity of the lattice, and represent a practically popular variant of toric codes. By construction, surface codes encode a single logical qubit. For this reason, two-qubit target logicals do not apply, and the single-qubit logical Phase and Hadamard gates can be constructed similar to their construction for toric codes presented above.

5 Concluding remarks

The circuit design procedures of this paper do not account for the fault tolerance property. To make them more attuned to practical implementations, we can rely on a variety of techniques known in the literature. A general idea of designing fault-tolerant circuits is to resort to the framework of pieceable fault tolerance [26, 46] which consists of inserting intermediate stabilizer measurements in between physical layers of the circuit to determine/detect possible errors occurring during physical gate operations. This technique was used for HGP codes in [31, 37] to design fault-tolerant gates. Intermediate measurements between the layers of a Clifford circuit were utilized in [14, 32] to advance the framework of Space-Time codes, which involves tracking the modification of stabilizers under the action of the physical gates on the state. Note that, since our circuits use only physical Clifford gates, throughout the layers of the circuit, the code remains a stabilizer code, and hence intermediate Pauli measurements are sufficient for detecting errors. However, it appears difficult to keep track of the weights of these intermediate stabilizers and logical Pauli operators, which can cause problems for intermediate error correction.

Whenever the HGP codes have distance 3 or less, one can replace each of the two-qubit gates of the circuit with fault-tolerant gadgets using flag qubits [11, 12]. These gadgets require a constant overhead, and so the overhead for fault tolerance is a constant factor of the number of two-qubit gates in the circuit. This includes the $[[18, 2, 3]]$ toric code which we used to exemplify the general circuit design procedure of this paper.

Thinking of generalizations of our results, recent constructions of qLDPC codes [15, 29, 34] reach into the range of parameters superior to HGP codes, and designing logical gates for them is therefore of significant interest. Since our approach of Sec. 3 applies to any CSS code, it also applies to these code families. To complete the description of the logical gates, we only need to find an explicit basis for the logical Pauli operators. Many of the qLDPC code families are based on different kinds of code

products, so if we manage to find a basis of logical Paulis for them and use it to construct symplectic matrices, this can lead to feasible circuit construction procedures. In particular, since homological product codes [3] are almost equivalent to HGP codes, for them this path looks within reach.

Finally, our general result, Theorem 3.12, relies on a simplifying assumption that the circuit acts trivially on all the stabilizers. This assumption was adopted to simplify the presentation, but it is not essential for the implementation of logical Clifford operators. Lifting this assumption in Theorem 3.12 might lead to lower-depth and/or transversal circuits for HGP codes or other related code families, improving the parameters of our designs.

Acknowledgment

This research was supported in part by US NSF grant CCF-2330909.

References

- [1] S. Aaronson and D. Gottesman. Improved simulation of stabilizer circuits. *Phys. Rev. A*, 70:052328, Nov 2004. doi:[10.1103/PhysRevA.70.052328](https://doi.org/10.1103/PhysRevA.70.052328).
- [2] H. Bombin and M. A. Martin-Delgado. Homological error correction: classical and quantum codes. *J. Math. Phys.*, 48(5):052105, 35, 2007. doi:[10.1063/1.2731356](https://doi.org/10.1063/1.2731356).
- [3] S. Bravyi and M. B. Hastings. Homological product codes. In *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '14, page 273–282, New York, NY, USA, 2014. Association for Computing Machinery. doi:[10.1145/2591796.2591870](https://doi.org/10.1145/2591796.2591870).
- [4] S. Bravyi and A. Kitaev. Universal quantum computation with ideal Clifford gates and noisy ancillas. *Physical Review A*, 71(2):022316, 2005. doi:[10.1103/PhysRevA.71.022316](https://doi.org/10.1103/PhysRevA.71.022316).
- [5] S. B. Bravyi and A. Y. Kitaev. Quantum codes on a lattice with boundary. *arXiv preprint quant-ph/9811052*, 1998. arXiv:[quant-ph/9811052](https://arxiv.org/abs/quant-ph/9811052), doi:[10.48550/arXiv.quant-ph/9811052](https://doi.org/10.48550/arXiv.quant-ph/9811052).
- [6] N. P. Breuckmann and S. Burton. Fold-transversal Clifford gates for quantum codes. *Quantum*, 8:1372, 2024. arXiv:[2202.06647](https://arxiv.org/abs/2202.06647), doi:[10.22331/q-2024-06-13-1372](https://doi.org/10.22331/q-2024-06-13-1372).
- [7] N. P. Breuckmann, M. Davydova, J. N. Eberhardt, and N. Tantivasadakarn. Cups and gates I: Cohomology invariants and logical quantum operations, 2024. arXiv:[2410.16250](https://arxiv.org/abs/2410.16250), doi:[10.48550/arXiv.2410.16250](https://doi.org/10.48550/arXiv.2410.16250).
- [8] N. P. Breuckmann and J. N. Eberhardt. Quantum low-density parity-check codes. *PRX Quantum*, 2(4):040101, 2021. doi:[10.1103/PRXQuantum.2.040101](https://doi.org/10.1103/PRXQuantum.2.040101).
- [9] A. R. Calderbank, E. M. Rains, P. W. Shor, and N. J. A. Sloane. Quantum error correction and orthogonal geometry. *Physical Review Letters*, 78(3):405–408, 1997. doi:[10.1103/PhysRevLett.78.405](https://doi.org/10.1103/PhysRevLett.78.405).
- [10] A. R. Calderbank and P. W. Shor. Good quantum error-correcting codes exist. *Phys. Rev. A*, 54:1098–1105, Aug 1996. doi:[10.1103/PhysRevA.54.1098](https://doi.org/10.1103/PhysRevA.54.1098).
- [11] R. Chao and B. W. Reichardt. Fault-tolerant quantum computation with few qubits. *npj Quantum Information*, 4(1):42, 2018. doi:[10.1038/s41534-018-0085-z](https://doi.org/10.1038/s41534-018-0085-z).

- [12] Z. Chen and N. Rengaswamy. Tailoring fault-tolerance to quantum algorithms, 2024. [arXiv:2404.11953](#), [doi:10.48550/arXiv.2404.11953](#).
- [13] J. Dehaene and B. De Moor. The Clifford group, stabilizer states, and linear and quadratic operations over $\text{gf}(2)$. *Physical Review A*, 68, 04 2003. [doi:10.1103/PhysRevA.68.042318](#).
- [14] N. Delfosse and A. Paetznick. Spacetime codes of Clifford circuits. 2023. [arXiv:2304.05943](#), [doi:10.48550/arXiv.2304.05943](#).
- [15] I. Dinur, M.-H. Hsieh, T.-C. Lin, and T. Vidick. Good quantum ldpc codes with linear time decoders. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023*, page 905–918, New York, NY, USA, 2023. Association for Computing Machinery. [doi:10.1145/3564246.3585101](#).
- [16] A. Dua, A. Kubica, L. Jiang, S. T. Flammia, and M. J. Gullans. Clifford-deformed surface codes. *PRX Quantum*, 5(1):010347, 2024. [doi:10.1103/PRXQuantum.5.010347](#).
- [17] O. Fawzi, A. Grospellier, and A. Leverrier. Constant overhead quantum fault tolerance with quantum expander codes. *Commun. ACM*, 64(1):106–114, Dec. 2020. [doi:10.1145/3434163](#).
- [18] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland. Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A*, 86:032324, Sep 2012. [doi:10.1103/PhysRevA.86.032324](#).
- [19] D. Gottesman. *Stabilizer Codes and Quantum Error Correction*. PhD thesis, California Institute of Technology, 1997. [arXiv:quant-ph/9705052](#), [doi:10.48550/arXiv.quant-ph/9705052](#).
- [20] D. Gottesman. Theory of fault-tolerant quantum computation. *Physical Review A*, 57(1):127, 1998. [doi:10.1103/PhysRevA.57.127](#).
- [21] D. Gottesman. An introduction to quantum error correction and fault-tolerant quantum computation. In *Quantum Information Science and its Contributions to Mathematics*, volume 68 of *Proceedings of Symposia in Applied Mathematics*, pages 13–58. American Mathematical Society, 2010. [doi:10.1090/psapm/068/2762145](#).
- [22] D. Gottesman. Fault-tolerant quantum computation with constant overhead. 2014. Published in *Quantum Information & Computation*, Vol. 14, No. 15–16 (2014), pp. 1338–1372. [arXiv:1310.2984](#), [doi:10.48550/arXiv.1310.2984](#).
- [23] L. C. Grove. *Classical Groups and Geometric Algebra*, volume 39 of *Graduate Studies in Mathematics*. American Mathematical Society, 2002. [doi:10.1090/gsm/039](#).
- [24] A. Y. Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52(6):1191–1249, 1997. [doi:10.1070/RM1997v052n06ABEH002155](#).
- [25] A. Y. Kitaev, A. Shen, and M. N. Vyalyi. *Classical and Quantum Computation*, volume 47 of *Graduate Studies in Mathematics*. American Mathematical Society, 2002. [doi:10.1090/gsm/047](#).
- [26] E. Knill, R. Laflamme, and W. H. Zurek. Resilient quantum computation. *Science*, 279(5349):342–345, 1998. [doi:10.1126/science.279.5349.342](#).
- [27] A. A. Kovalev and L. P. Pryadko. Fault tolerance of quantum low-density parity check codes with sublinear distance scaling. *Physical Review A*, 87(2):020304, 2013. [doi:10.1103/PhysRevA.87.020304](#).

- [28] A. Krishna and D. Poulin. Fault-tolerant gates on hypergraph product codes. *Phys. Rev. X*, 11:011023, Feb 2021. [doi:10.1103/PhysRevX.11.011023](https://doi.org/10.1103/PhysRevX.11.011023).
- [29] A. Leverrier and G. Zémor. Quantum Tanner codes. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 872–883, 2022. expanded version in arXiv:2202.13641. [doi:10.1109/FOCS54457.2022.00117](https://doi.org/10.1109/FOCS54457.2022.00117).
- [30] T.-C. Lin. Transversal non-Clifford gates for quantum LDPC codes on sheaves, 2024. [arXiv:2410.14631](https://arxiv.org/abs/2410.14631), [doi:10.48550/arXiv.2410.14631](https://doi.org/10.48550/arXiv.2410.14631).
- [31] A. J. Malcolm, A. N. Glaudell, P. Fuentes, D. Chandra, A. Schotte, C. DeLisle, R. Haenel, A. Ebrahimi, J. Roffe, A. O. Quintavalle, et al. Computing efficiently in QLDPC codes. 2025. [arXiv:2502.07150](https://arxiv.org/abs/2502.07150), [doi:10.48550/arXiv.2502.07150](https://doi.org/10.48550/arXiv.2502.07150).
- [32] S. Martiel and A. Javadi-Abhari. Low-overhead error detection with spacetime codes. 4 2025. [arXiv:2504.15725](https://arxiv.org/abs/2504.15725), [doi:10.48550/arXiv.2504.15725](https://doi.org/10.48550/arXiv.2504.15725).
- [33] D. Maslov and M. Roetteler. Shorter stabilizer circuits via Bruhat decomposition and quantum circuit transformations. *IEEE Transactions on Information Theory*, 64(7):4729–4738, 2018. [doi:10.1109/TIT.2018.2825602](https://doi.org/10.1109/TIT.2018.2825602).
- [34] P. Panteleev and G. Kalachev. Asymptotically good quantum and locally testable classical ldpc codes. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2022*, page 375–388, New York, NY, USA, 2022. Association for Computing Machinery. [doi:10.1145/3519935.3520017](https://doi.org/10.1145/3519935.3520017).
- [35] T. Pllaha, N. Rengaswamy, O. Tirkkonen, and R. Calderbank. Un-Weyl-ing the Clifford hierarchy. *Quantum*, 4:370, December 2020. [doi:10.22331/q-2020-12-11-370](https://doi.org/10.22331/q-2020-12-11-370).
- [36] T. Pllaha, K. Volanto, and O. Tirkkonen. Decomposition of Clifford gates. In *2021 IEEE Global Communications Conference (GLOBECOM)*, pages 01–06, 2021. [doi:10.1109/GLOBECOM46510.2021.9685501](https://doi.org/10.1109/GLOBECOM46510.2021.9685501).
- [37] A. O. Quintavalle, P. Webster, and M. Vasmer. Partitioning qubits in hypergraph product codes to implement logical gates. *Quantum*, 7:1153, Oct. 2023. [doi:10.22331/q-2023-10-24-1153](https://doi.org/10.22331/q-2023-10-24-1153).
- [38] R. Raussendorf and J. Harrington. Fault-tolerant quantum computation with high threshold in two dimensions. *Physical Review Letters*, 98(19):190504, 2007. [doi:10.1103/PhysRevLett.98.190504](https://doi.org/10.1103/PhysRevLett.98.190504).
- [39] N. Rengaswamy, R. Calderbank, S. Kadhe, and H. D. Pfister. Logical Clifford synthesis for stabilizer codes. *IEEE Transactions on Quantum Engineering*, 1:1–17, 2020. [doi:10.1109/TQE.2020.3023419](https://doi.org/10.1109/TQE.2020.3023419).
- [40] A. Steane. Error correcting codes in quantum theory. *Phys. Rev. Letters*, 77:793–797, 1996. [doi:10.1103/PhysRevLett.77.793](https://doi.org/10.1103/PhysRevLett.77.793).
- [41] A. M. Steane. Efficient fault-tolerant quantum computing. *Nature*, 399(6732):124–126, 1999. [doi:10.1038/20127](https://doi.org/10.1038/20127).
- [42] J.-P. Tillich and G. Zémor. Quantum ldpc codes with positive rate and minimum distance proportional to the square root of the blocklength. *IEEE Transactions on Information Theory*, 60(2):1193–1202, 2014. [doi:10.1109/TIT.2013.2292061](https://doi.org/10.1109/TIT.2013.2292061).

- [43] M. A. Tremblay, N. Delfosse, and M. E. Beverland. Constant-overhead quantum error correction with thin planar connectivity. *Phys. Rev. Lett.*, 129:050504, Jul 2022. [doi:10.1103/PhysRevLett.129.050504](https://doi.org/10.1103/PhysRevLett.129.050504).
- [44] M. Wilde. Logical operators of quantum codes. *Physical Review A*, 79, 03 2009. [doi:10.1103/PhysRevA.79.062322](https://doi.org/10.1103/PhysRevA.79.062322).
- [45] Q. Xu, H. Zhou, G. Zheng, D. Bluvstein, J. P. B. Ataiades, M. D. Lukin, and L. Jiang. Fast and parallelizable logical computation with homological product codes. *Phys. Rev. X*, 15:021065, May 2025. [doi:10.1103/PhysRevX.15.021065](https://doi.org/10.1103/PhysRevX.15.021065).
- [46] T. J. Yoder, R. Takagi, and I. L. Chuang. Universal fault-tolerant gates on concatenated stabilizer codes. *Phys. Rev. X*, 6:031039, Sep 2016. [doi:10.1103/PhysRevX.6.031039](https://doi.org/10.1103/PhysRevX.6.031039).
- [47] G. Zhu, S. Sikander, E. Portnoy, A. W. Cross, and B. J. Brown. Non-Clifford and parallelizable fault-tolerant logical gates on constant and almost-constant rate homological quantum LDPC codes via higher symmetries, 2024. [arXiv:2310.16982](https://arxiv.org/abs/2310.16982), [doi:10.48550/arXiv.2310.16982](https://doi.org/10.48550/arXiv.2310.16982).

A Examples, Proofs, Extensions

A.1 Example for circuit design

Example A.1. (Refer to Sec. 3.2) Let g be a Clifford operator on 3 qubits and let

$$F_g = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

We can reduce this matrix to the identity by the following row operations:

1. Exchange rows 2 and 5.
2. Add row 5 to row 2.
3. Exchange rows 2 and 5.
4. Add row 6 to row 1 and add row 4 to row 3.
5. Add row 2 to row 1 and add row 4 to row 5.

that is $A_4 A_3 A_1 A_2 A_1 F_g = I$, where

$$A_4 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, A_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Hence, $F_g = A_1^{-1} A_2^{-1} A_1^{-1} A_3^{-1} A_4^{-1}$, and translating each of the matrices to the corresponding gate, we obtain the circuit for g shown in Figure 7.

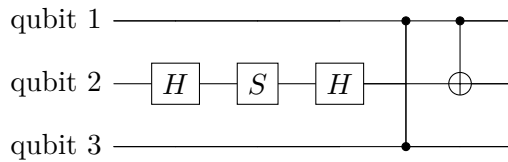


Figure 7: Circuit for Example A.1

A.2 Omitted steps in the proof of Theorem 4.4

A.2.1 Correctness proof of row operations in Theorem 4.5.

Recall that we want to show that for any vector $v \in \mathbb{F}_2^{\tilde{n}}$, performing the following sequence of row operations:

1. Choose $x \in \text{supp}(v)$. For every $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $(y + n)$ to row $(x + n)$.
2. Add row $(x + n)$ to row x .
3. For every $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $(y + n)$ to row $(x + n)$.

on $I_{2n \times 2n}$ gives rise to the symplectic matrix:

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & v^\top v & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}.$$

Since $\text{supp}(v) \subset [\tilde{n}]$, i.e., we are operating only on the qubits of the left sector, for notational simplicity it is convenient to restrict F to these qubits only. The resulting submatrix is:

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & v^\top v \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix}.$$

Let $\bar{v} = v + e_x$ and consider the following two $2\tilde{n} \times 2\tilde{n}$ matrices:

$$F_1 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} \end{bmatrix}, \quad F_2 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_x^\top e_x \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix},$$

corresponding respectively to the row operations of steps (1) and (2) above. Note that the row operations in step (3) are the same as those of step (1). We have

$$\begin{aligned} F_1 F_2 F_1 &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_x^\top e_x \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} \end{bmatrix} \\ &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & e_x^\top e_x + \bar{v}^\top e_x \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} \end{bmatrix} \\ &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & v^\top e_x \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} \end{bmatrix} \\ &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & v^\top v \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix} = F \end{aligned}$$

and this completes the proof.

A.2.2 Correctness proof of Algorithm 1

For $i = 2, 3, 4, 5$, let U_i be the Clifford unitary applied to perform steps $1 \leq j \leq i$ of Algorithm 1. Recalling the notation of Section 2.3, we see that I is contained in column h of the left sector. The X stabilizer generator $S_X(h', j)$ is supported on column h' of the left sector and row j of the right sector, while the Z stabilizer generator $S_Z(i', l)$ is supported on row i' of the left sector and column l of the right sector. First consider an X stabilizer $S_X(h', j)$ and let X be its support.

Case I: If $h' \neq h$, then $I \cap X = \emptyset$ and hence C preserves $S_X(h', j)$.

Case II: If $h' = h$, then I has even overlap with X because $S_X(h, j)$ commutes with $\bar{Z}_{i,h,L}$. The layer of CNOT gates applies an X to qubit x for every $y \in I \cap X$. We have two subcases:

IIa : If $x \notin I \cap X$, an even number of X s are applied to qubit x and so

$$U_2 S_X(h, j) U_2^\dagger = S_X(h, j).$$

Therefore the Phase gate on qubit x does not change the stabilizer, and it is also unaffected by the following layer of CNOTs by the same logic. The X type stabilizer is also unchanged by the X gate on qubit x . Hence

$$U_5 S_X(h, j) U_5^\dagger = S_X(h, j).$$

IIb: If, on the other hand, $x \in I \cap X$, an odd number of X s are applied to qubit r and so

$$U_2 S_X(h, j) U_2^\dagger = \bigotimes_{r \in I \cap X \setminus \{x\}} X_r \bigotimes_{r \in X \setminus I} X_r.$$

Therefore the Phase gate on qubit x does not change the stabilizer further, and since the following layer of CNOTs again applies an odd number X s to qubit x , we have

$$U_4 S_X(h, j) U_4^\dagger = S_X(h, j).$$

Finally the stabilizer is unchanged by the last X operator on qubit x .

Hence any X stabilizer does not gain a phase due to the series of CZ and phase gates.

Next consider the Z stabilizer $S_Z(i', l)$ and let its support be Z . Note that Z can have at most one overlap with I on the qubit (i', h) .

- Case I: If $x \notin I \cap Z$, then the stabilizer $S_Z(i', l)$ is unaffected by the layer of CNOTs, the phase gate on qubit x and the following layer of CNOTs and the X gate on qubit x .
- Case II: If $x \in I \cap Z$, then we have

$$U_2 S_Z(i', l) U_2^\dagger = \bigotimes_{r \in I \cup Z} Z_r$$

and so

$$U_3 S_Z(i', l) U_3^\dagger = - \bigotimes_{r \in I \cup Z} Z_r$$

due to the action of the single Phase gate on qubit x . Due to the next layer of CNOTs, we have

$$U_4 S_Z(i', l) U_4^\dagger = - \bigotimes_{r \in Z} Z_r$$

and finally due to the X gate on qubit x

$$U_5 S_Z(i', l) U_5^\dagger = S_Z(i', l)$$

Hence the circuit in Algorithm 1 preserves all the stabilizers with phases.

A.3 An alternative proof for Theorem 4.4

Let $v = (a_i \otimes e_h)$. We shall construct the matrix

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & v^\top v & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix} \quad (11)$$

by performing the following row operations on the matrix $I_{2n \times 2n}$:

1. For every $x, y \in \text{supp}(v)$, $x < y$, add row $(x + n)$ to row y and add row $(y + n)$ to row x .
2. For every $x \in \text{supp}(v)$, add row $(x + n)$ to row x .

To see that these operations on $I_{2n \times 2n}$ produce the matrix F , let $M = v^\top v$ and note that it can be written as

$$M = \sum_{r \in \text{supp}(v)} \sum_{s \in \text{supp}(v) \setminus \{r\}} e_r^\top e_s + \sum_{r \in \text{supp}(v)} e_r^\top e_r.$$

It is easy to see that the row operations in Step (1) above produce the first part of the sum while the row operations in Step (2) produce the second part of the sum.

The next step is to translate these row operations to physical gates using the operations in Table 1. Let I denote the physical qubit indices corresponding to the support of the $\bar{Z}_{i,h,L}$ logical Pauli. Step (1) of row operations translates to applying CZ gates between every pair of qubits in I and step (2) translates to applying a Phase gate to each qubit in I . Note that in the circuit realization, these steps have to be applied in the opposite order, i.e., the Phase gates of step (2) will precede the CZ gates of step (1) in the circuit (cf. (5)). Since the circuit realizes a Clifford unitary having F as its symplectic representation, the circuit is guaranteed to fix the support of the stabilizer generators and perform the desired transformations on the logical Pauli space. However, the circuit can still introduce undesired phase changes to some stabilizers, which may change the codespace. We show next that adding a layer of X gates to the qubits in I suffices to preserve the stabilizer space.

Summarizing, the circuit building procedure can be written as in Algorithm 5.

For $i = 1, 2, 3$, let U_i be the Clifford unitary applied to perform steps $1 \leq j \leq i$ of the Algorithm 5. Recalling the notation of Section 2.3, we see that I is contained in column h of the left sector. The X stabilizer generator $S_X(h', j)$ is supported on column h' of the left sector and row j of the right sector, while the Z stabilizer generator $S_Z(i', l)$ is supported on row i' of the left sector and column l of the right sector. First consider an X stabilizer $S_X(h', j)$ and let X be its support.

Algorithm 5 Logical Phase gate circuit design algorithm

1: Phase layer

For $s \in I$, apply Phase on qubit s

2: CZ layer

For $r, s \in I, s < r$, apply CZ between s and r

3: X layer

For $r, s \in I, s < r$, apply X on qubit s

Case I: If $h' \neq h$, then $I \cap X = \emptyset$ and hence C preserves $S_X(h', j)$.

Case II: If $h' = h$, then I has even overlap with X because $S_X(h, j)$ commutes with $\bar{Z}_{i,h,L}$. The layer of Phase gates in step (1) of 5 maps every X on qubits in $I \cap X$ to Y and so

$$U_1 S_X(h, j) U_1^\dagger = \bigotimes_{r \in I \cap X} Y_r \bigotimes_{r \in X \setminus I} X_r.$$

Since there is a CZ gate between every pair of $I \cap X$ due to step (2) of Algorithm 5, every qubit in $I \cap X$ is operated on by an odd number of CZ gates, which maps Y_r to X_r for $r \in I \cap X$, and so

$$U_2 S_X(h, j) U_2^\dagger = S_X(h, j).$$

Hence any X stabilizer does not gain a phase due to the series of CZ and phase gates.

Next consider the Z stabilizer $S_Z(i', l)$ and let its support be Z . Note that Z can have at most one overlap with I on the qubit (i', h) . Since a single Phase gate changes Z to $-Z$, if $I \cap Z$ is non-empty, i.e., contains the qubit (i', h) , then

$$U_1 S_Z(i', l) U_1^\dagger = \bigotimes_{r \in I \cap Z} (-Z_r) \bigotimes_{r \in Z \setminus I} Z_r = -S_Z(i', l),$$

and we have

$$U_2 S_Z(i', l) U_2^\dagger = (-1)^{|I \cap Z|} S_Z(i', l)$$

because CZ gates do not change Z type Paulis. This sign is reversed by the layer of X gates on qubits in I , implying

$$U_3 S_Z(i', l) U_3^\dagger = S_Z(i', l).$$

Hence the circuit in Algorithm 5 preserves all the stabilizers with phases.

Finally we check the parameters of the circuit C . It only acts on the set of qubits indexed by I , whose size equals the Hamming weight of $|a_i|$, hence $\chi = |a_i|$. For the depth, note that although there are $\binom{|a_i|}{2} = \frac{|a_i|(|a_i|-1)}{2}$ CZ gates, we can apply $\frac{|a_i|}{2}$ CZ gates (or $\frac{|a_i|-1}{2}$ CZ gates if $|a_i|$ is odd) with non-overlapping supports in parallel in a single layer. Since the Phase gates and X gates add 2 layers, the depth $\delta \leq |a_i| + 2$. Since $\{a_i\}$ is an SLT basis of $\ker(H_a)$ then by Theorem 4.2, $|a_i| = \Theta(\sqrt{n})$. Hence we conclude that $\chi = \Theta(\sqrt{n}), \delta = \Theta(\sqrt{n})$.

A.4 Omitted steps in the proof of Theorem 4.6

A.4.1 Correctness proof of row operations in Theorem 4.6

Recall that we want to show that for two vectors $u, v \in \mathbb{F}_2^{\tilde{n}}$, $\text{supp}(u) \cap \text{supp}(v) = \{\rho\}$, performing the following sequence of row operations:

1. For each $x \in \text{supp}(v) \setminus \{\rho\}$, add row $x + n$ to row ρ and add row $\rho + n$ to row x .
2. For each $y \in \text{supp}(u)$, exchange row y and row $y + n$.
3. For each $y \in \text{supp}(u) \setminus \{\rho\}$, add row $y + n$ to row ρ and add row $\rho + n$ to row y .
4. For each $y \in \text{supp}(u)$, exchange row y with row $y + n$.
5. For each $y \in \text{supp}(u) \setminus \{\rho\}$, add row y to row ρ and add row $\rho + n$ to row $y + n$.
6. For each $x \in \text{supp}(v) \setminus \{\rho\}$, add row ρ to row x and add row $x + n$ to row $\rho + n$.
7. Exchange row ρ and row $\rho + n$.

on $I_{2n \times 2n}$ gives rise to the symplectic matrix

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u & 0_{\tilde{n} \times \tilde{m}} & v^\top v & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ u^\top u & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} + u^\top v & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}.$$

Since $\text{supp}(v), \text{supp}(u) \subset [\tilde{n}]$, i.e., we are operating only on the qubits of the left sector, for notational simplicity it would be convenient to restrict our symplectic matrix to these qubits only. The restricted matrix is:

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u & v^\top v \\ u^\top u & I_{\tilde{n} \times \tilde{n}} + u^\top v \end{bmatrix}.$$

Let $\bar{u} = u + e_\rho$, $\bar{v} = v + e_\rho$ and note that $\bar{u}e_\rho^\top = \bar{v}e_\rho^\top = \bar{u}\bar{v}^\top = 0$. Consider the following matrices in $Sp(2\tilde{n}, \mathbb{F}_2)$:

$$F_1 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_\rho^\top \bar{v} + \bar{v}^\top e_\rho \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix}, \quad F_2 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{n}} \\ \bar{u}^\top e_\rho + e_\rho^\top \bar{u} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix}, \quad F_3 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + e_\rho^\top \bar{u} & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + \bar{u}^\top e_\rho \end{bmatrix}$$

$$F_4 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_\rho & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_\rho^\top \bar{v} \end{bmatrix}, \quad F_5 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + e_\rho^\top e_\rho & e_\rho^\top e_\rho \\ e_\rho^\top e_\rho & I_{\tilde{n} \times \tilde{n}} + e_\rho^\top e_\rho \end{bmatrix}$$

corresponding respectively to the row operations of step (1), steps (2-4), step (5), step (6) and step (7). We claim that $F = F_5 F_4 F_3 F_2 F_1$. We have:

$$\begin{aligned}
F_4 F_3 &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_\rho & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_\rho^\top \bar{v} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + e_\rho^\top \bar{u} & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + \bar{u}^\top e_\rho \end{bmatrix} \\
&= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_\rho + e_\rho^\top \bar{u} + \bar{v}^\top e_\rho e_\rho^\top \bar{u} & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_\rho^\top \bar{v} + \bar{u}^\top e_\rho + e_\rho^\top \bar{v} \bar{u}^\top e_\rho \end{bmatrix} \\
&= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top u + e_\rho^\top \bar{u} & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_\rho^\top \bar{v} + \bar{u}^\top e_\rho \end{bmatrix} \\
&= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u + e_\rho^\top e_\rho & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + u^\top e_\rho + e_\rho^\top v \end{bmatrix}
\end{aligned}$$

and so

$$\begin{aligned}
F_5 F_4 F_3 &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + e_\rho^\top e_\rho & e_\rho^\top e_\rho \\ e_\rho^\top e_\rho & I_{\tilde{n} \times \tilde{n}} + e_\rho^\top e_\rho \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u + e_\rho^\top e_\rho & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + u^\top e_\rho + e_\rho^\top v \end{bmatrix} \\
&= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u + e_\rho^\top e_\rho + e_\rho^\top u & e_\rho^\top v \\ e_\rho^\top u & I_{\tilde{n} \times \tilde{n}} + u^\top e_\rho \end{bmatrix} \\
&= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u + e_\rho^\top \bar{u} & e_\rho^\top v \\ e_\rho^\top u & I_{\tilde{n} \times \tilde{n}} + u^\top e_\rho \end{bmatrix}.
\end{aligned}$$

Next

$$\begin{aligned}
F_2 F_1 &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{n}} \\ \bar{u}^\top e_\rho + e_\rho^\top \bar{u} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_\rho^\top \bar{v} + \bar{v}^\top e_\rho \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix} \\
&= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_\rho^\top \bar{v} + \bar{v}^\top e_\rho \\ \bar{u}^\top e_\rho + e_\rho^\top \bar{u} & I_{\tilde{n} \times \tilde{n}} + \bar{u}^\top e_\rho e_\rho^\top \bar{v} + e_\rho^\top \bar{u} e_\rho^\top \bar{v} + \bar{u}^\top e_\rho \bar{v}^\top e_\rho + e_\rho^\top \bar{u} \bar{v}^\top e_\rho \end{bmatrix} \\
&= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_\rho^\top \bar{v} + \bar{v}^\top e_\rho \\ \bar{u}^\top e_\rho + e_\rho^\top \bar{u} & I_{\tilde{n} \times \tilde{n}} + \bar{u}^\top \bar{v} \end{bmatrix}.
\end{aligned}$$

Finally (note the order of the factors, corresponding to (5)),

$$\begin{aligned}
F_5 F_4 F_3 F_2 F_1 &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u + e_\rho^\top \bar{u} & e_\rho^\top v \\ e_\rho^\top u & I_{\tilde{n} \times \tilde{n}} + u^\top e_\rho \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_\rho^\top \bar{v} + \bar{v}^\top e_\rho \\ \bar{u}^\top e_\rho + e_\rho^\top \bar{u} & I_{\tilde{n} \times \tilde{n}} + \bar{u}^\top \bar{v} \end{bmatrix} \\
&= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u + e_\rho^\top \bar{u} + e_\rho^\top \bar{u} & e_\rho^\top \bar{v} + \bar{v}^\top e_\rho + v^\top \bar{v} + e_\rho^\top v \\ e_\rho^\top u + \bar{u}^\top e_\rho + e_\rho^\top \bar{u} + u^\top \bar{u} & e_\rho^\top \bar{v} + I_{\tilde{n} \times \tilde{n}} + u^\top e_\rho + \bar{u}^\top \bar{v} \end{bmatrix} \\
&= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + v^\top u & v^\top v \\ u^\top u & I_{\tilde{n} \times \tilde{n}} + u^\top v \end{bmatrix} = F
\end{aligned}$$

A.4.2 Correctness proof of Algorithm 2

As per the notation of Sec. 2.3, first note that I is a subset of qubits in row i while J is a subset of qubits in column h of the left sector and $I \cap J = \{\rho\}$. For $i = 1, 2, \dots, 7$, let U_i be the Clifford

unitary applied to perform steps $1 \leq j \leq i$ of Algorithm 2. Recall that an X (Z) stabilizer generator is supported on a column (row) of the left sector and a row (column) of the right sector. Let us consider the following cases for an X stabilizer $S_X(h', j)$ with X denoting its support:

Case I: Suppose that $h' \neq h$ and note that $I \cap X$ is either empty or contains the physical qubit (i, h') . In the former case, $S_X(h', j)$ is unaffected by the circuit. If, on the other hand, qubit (i, h') is in $I \cap X$, then the X Pauli on that qubit is acted upon by a single CNOT gate (as the target qubit) and a single CZ gate, none of which introduce any global sign changes.

Case II: Now consider an X stabilizer generator $S_X(h, j)$. Due to the fact that the X type stabilizer commutes with the logical Pauli Z operator, we know that $|J \cap X|$ will be even. Here two possible subcases can happen.

II.a: First suppose that $\rho \notin X$. Due to the even overlap with set J , an even number of CNOT gates have their control qubit in $J \cap X$ in Step (3) and an even number of CZ gates have one of their qubits in $J \cap X$ in Step (7) of Algorithm 2. Hence overall phase does not change.

II.b: Now let $\rho \in X$. We track the modification of the stabilizer via the conjugation relation $U_i S_X(h, j) U_i^\dagger$. First, due to the H gate on qubit ρ , we have

$$U_1 S_X(h, j) U_1^\dagger = \bigotimes_{r \in X \setminus \{\rho\}} X_r \otimes Z_\rho.$$

After this, the Z Pauli on qubit ρ propagates to all qubits in $J \setminus \{\rho\}$ and the X Paulis on qubits in $J \cap X \setminus \{\rho\}$ propagate to qubit ρ . Since $|J \cap X|$ is even, we have an overall phase change

$$U_2 S_X(h, j) U_2^\dagger = - \bigotimes_{r \in X \setminus J} X_r \bigotimes_{r \in J \setminus X} Z_r \bigotimes_{r \in X \cap J} Y_r.$$

In the next stage of CNOT gates, the Y Pauli on qubit ρ propagates an X to every qubit in $I \setminus \{\rho\}$.

$$U_3 S_X(h, j) U_3^\dagger = - \bigotimes_{r \in X \cup I \setminus J} X_r \bigotimes_{r \in J \setminus X} Z_r \bigotimes_{r \in X \cap J} Y_r.$$

Next, due to the Hadamard gates, the Y Pauli on qubit ρ gets a phase change and the X Paulis on qubits in $I \setminus \{\rho\}$ are transformed to Z Paulis. The layer of CZ gates cancels the Z Paulis on qubits in $I \setminus \{\rho\}$. Finally the Y Pauli on qubit ρ acquires another phase change due to the second Hadamard gate. Combining all these changes, we have

$$U_6 S_X(h, j) U_6^\dagger = - \bigotimes_{r \in X \setminus J} X_r \bigotimes_{r \in J \setminus X} Z_r \bigotimes_{r \in X \cap J} Y_r.$$

After this, the layer of CZ gates introduces a Z Pauli on each qubit in $J \setminus \{\rho\}$ due to the Y operator on qubit ρ . This does not introduce any phase change:

$$U_7 S_X(h, j) U_7^\dagger = - \bigotimes_{r \in X} X_r = -S_X(h, j).$$

Finally the Y gate on qubit ρ maps $-S_X(h, j)$ back to $S_X(h, j)$.

A similar analysis on the Z stabilizer generators reveals that

$$U_7 S_Z(i', l) U_7^\dagger = -S_Z(i', l)$$

if and only if $i = i'$ and the support of $S_Z(i, l)$ contains qubit ρ . The Y gate on qubit ρ maps $-S_Z(i, l)$ back to $S_Z(i, l)$. In conclusion, the circuit C, built using Algorithm 2, preserves the stabilizers together with their global phases.

A.5 Correctness proof of row operations in Theorem 4.8

For $v \in \mathbb{F}_2^{\tilde{n}}, u \in \mathbb{F}_2^{\tilde{m}}$, we wish to show that performing the following sequence of row operations:

1. Select a row index $x \in \text{supp}(v)$. For all $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $y + n$ to row $x + n$.
2. For all $\bar{y} \in \text{supp}(u)$, add row $\bar{y} + \tilde{n}$ to row x and add row $x + n$ to row $\bar{y} + \tilde{n} + n$.
3. For all $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $y + n$ to row $x + n$.

on $I_{2n \times 2n}$ gives rise to the symplectic matrix

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & v^\top u & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & u^\top v & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}.$$

Fix an x in $\text{supp}(v)$ and let e_x be the length $\tilde{n}$ unit vector, let $\bar{v} = v + e_x$. Consider the following symplectic matrices corresponding to the row operations in step (1) and (2) respectively (note that step (3) is the same as step (1))

$$F_1 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{m}} & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}, F_2 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_x^\top u & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & u^\top e_x & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}.$$

We have

$$\begin{aligned} F_2 F_1 &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_x^\top u & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & u^\top e_x & I_{\tilde{m} \times \tilde{m}} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{m}} & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix} \\ &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & e_x^\top u & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & u^\top v & I_{\tilde{m} \times \tilde{m}} \end{bmatrix} \end{aligned}$$

and finally

$$\begin{aligned}
F_1 F_2 F_1 &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{m}} & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix} \\
&\quad \times \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & e_x^\top u & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & u^\top v & I_{\tilde{m} \times \tilde{m}} \end{bmatrix} \\
&= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & v^\top u & 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & u^\top v & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}
\end{aligned}$$

and this completes the proof. The proof for the case when both logical qubits are in the same sector is analogous and hence is not repeated.

A.6 Correctness proof of row operations in Theorem 4.10

We prove the correctness for the case when both qubits are in the same sector, as the proof of the case of different sectors is completely analogous. Let $v, u \in \mathbb{F}_2^{\tilde{n}}$ be vectors such that there exists $x \in \text{supp}(v) \setminus \text{supp}(u)$ and $y \in \text{supp}(u) \setminus \text{supp}(v)$. The target matrix is:

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & v^\top u + u^\top v & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} \\ 0_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} & I_{\tilde{n} \times \tilde{n}} & 0_{\tilde{n} \times \tilde{m}} \\ 0_{\tilde{m} \times \tilde{n}} & 0_{\tilde{m} \times \tilde{m}} & 0_{\tilde{m} \times \tilde{n}} & I_{\tilde{m} \times \tilde{m}} \end{bmatrix}.$$

We wish to prove that the following row operations performed on $I_{2n \otimes 2n}$ in order produce the matrix F :

1. For all $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $y + n$ to row $x + n$.
2. For all $\bar{y} \in \text{supp}(u) \setminus \{y\}$, add row y to row $\bar{y}$ and add row $\bar{y} + n$ to row y .
3. Add row $x + n$ to row y and add row $y + n$ to row x .
4. For all $y \in \text{supp}(v) \setminus \{x\}$, add row x to row y and add row $y + n$ to row $x + n$.
5. For all $\bar{y} \in \text{supp}(u) \setminus \{y\}$, add row y to row $\bar{y}$ and add row $\bar{y} + n$ to row y .

Since the operations are completely contained within rows $\{1, \dots, \tilde{n}\}$ and $\{n + 1, \dots, n + \tilde{n}\}$, for notational simplicity we consider the restricted matrix

$$F = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & v^\top u + u^\top v \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix}.$$

Let $\bar{v} = v + e_x$, $\bar{u} = u + e_y$ and note that $e_y v^\top = e_x u^\top = 0$. Consider the following matrices in $Sp(2\tilde{n}, \mathbb{F}_2)$ corresponding to the row operations in step (1), (2) and (3) respectively, noting that steps (4) and (5) are the same as steps (1) and (2),

$$F_1 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} \end{bmatrix}, \quad F_2 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{u}^\top e_y & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_y^\top \bar{u} \end{bmatrix},$$

$$F_3 = \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_x^\top e_y + e_y^\top e_x \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix}.$$

We have

$$\begin{aligned} F_3 F_2 F_1 &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_x^\top e_y + e_y^\top e_x \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{u}^\top e_y & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_y^\top \bar{u} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} \end{bmatrix} \\ &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_x^\top e_y + e_y^\top e_x \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x + \bar{u}^\top e_y & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} + e_y^\top \bar{u} \end{bmatrix} \\ &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x + \bar{u}^\top e_y & e_x^\top e_y + e_y^\top e_x + e_x^\top \bar{u} + e_y^\top \bar{v} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} + e_y^\top \bar{u} \end{bmatrix} \\ &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x + \bar{u}^\top e_y & e_x^\top u + e_y^\top v \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} + e_y^\top \bar{u} \end{bmatrix}. \end{aligned}$$

And finally

$$\begin{aligned} F_1 F_2 F_3 F_2 F_1 &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} \end{bmatrix} \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{u}^\top e_y & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_y^\top \bar{u} \end{bmatrix} \\ &\quad \times \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x + \bar{u}^\top e_y & e_x^\top u + e_y^\top v \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} + e_y^\top \bar{u} \end{bmatrix} \\ &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x + \bar{u}^\top e_y & 0_{\tilde{n} \times \tilde{n}} \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} + e_y^\top \bar{u} \end{bmatrix} \\ &\quad \times \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} + \bar{v}^\top e_x + \bar{u}^\top e_y & e_x^\top u + e_y^\top v \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} + e_x^\top \bar{v} + e_y^\top \bar{u} \end{bmatrix} \\ &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & e_x^\top u + e_y^\top v + \bar{v}^\top u + \bar{u}^\top v \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix} \\ &= \begin{bmatrix} I_{\tilde{n} \times \tilde{n}} & v^\top u + u^\top v \\ 0_{\tilde{n} \times \tilde{n}} & I_{\tilde{n} \times \tilde{n}} \end{bmatrix} \end{aligned}$$

and this completes the proof.