Fold-Transversal Clifford Gates for Quantum Codes

Nikolas P. Breuckmann[1] and Simon Burton[2]

[1]Department of Computer Science, University College London, WC1E 6BT London, United Kingdom
[2]Institute of Physics, Jagiellonian University, Łojasiewicza 11, 30-348 Kraków, Poland

We generalize the concept of folding from surface codes to CSS codes by considering certain dualities within them. In particular, this gives a general method to implement logical operations in suitable LDPC quantum codes using transversal gates and qubit permutations only.

To demonstrate our approach, we specifically consider a $[[30, 8, 3]]$ hyperbolic quantum code called Bring's code. Further, we show that by restricting the logical subspace of Bring's code to four qubits, we can obtain the *full* Clifford group on that subspace.

1 Introduction

We show how symmetries of CSS quantum codes can be utilized to implement encoded Clifford gates, using only transversal gates and qubit permutations. Our scheme is not only trivially fault-tolerant, but it also incurs no time-overhead and does not require any additional ancilla qubits. It can be understood as a generalization of the concept of folding surface codes due to Moussa [1], see Figure 1.

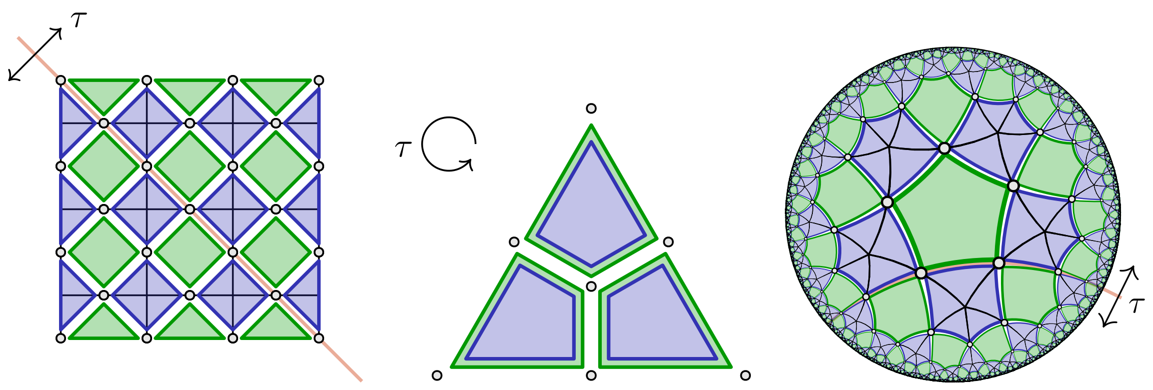
Folding of topological quantum codes is part of the folklore relating surface codes and color codes: folded surface codes give color codes, or alternatively, unfolded color codes give surface codes [2].

We generalize the concept of folding to arbitrary CSS codes by introducing *fold-transversal gates.* While our approach works for any sufficiently symmetric CSS code, we believe it is particularly suited to implement logical gates in low-density parity-check (LDPC) quantum codes. These codes have recently attracted a lot of attention, see [3] for a recent review. This is partially due to the fact that constant overhead fault-tolerant quantum computation can be realized with these codes [4, 5]. Recent developments have shown that it is possible to construct high-performance LDPC quantum codes from classical codes [6, 7, 8]. In particular, [8] gave a construction that was conjectured to have optimal asymptotic parameter scaling, which was subsequently proven in [9].

While the construction of LDPC quantum codes has made big advances, it is still unclear how to best manipulate the encoded information. The constant overhead in [4] was achieved by implementing gates via ancillary states. However, in order to keep the qubit overhead constant in their proof, the implementation of gates had to be serialized, making the scheme less practical.

For specific families of LDPC quantum codes, a few techniques to fault-tolerantly implement logical gates have been introduced. For hyperbolic surface codes, Breuckmann et al. considered code deformations in order to perform CNOT-gates [10]. Further, Lavasani–Barkeshli discussed the implementation of other Clifford gates on hyperbolic surface codes by using ancillary states [11]. Krishna–Poulin [12] considered generalizations of code deformation techniques of the surface code to hypergraph product codes in order to implement Clifford gates. On the other hand, Burton–Browne [13] showed that it is not possible to obtain transversal logical gates outside of the Clifford group using certain hypergraph

Nikolas P. Breuckmann: n.breuckmann@ucl.ac.uk, http://nikobreu.website
Simon Burton: simon@arrowtheory.com, http://arrowtheory.com



(i) Moussa's original construction on a surface code. The ZX-duality τ is reflecting along a diagonal.

(ii) The Steane code is self-dual, and so the identity permutation serves as a ZX-duality.

(iii) On a $\{5, 5\}$-tiled hyperbolic surface, a ZX-duality is shown reflecting along a geodesic.

Figure 1: How ZX-dualities define folds of quantum codes. Qubits are shown as grey circles. The green/blue faces correspond to the Z- and X-checks. The ZX-duality τ maps between the two types of checks. The codes (i) and (iii) are built from the homology of a tiling (cellulation) shown in black.

product codes.

More recently, Cohen et al. generalized lattice surgery between two logical operators of an arbitrary LDPC quantum code [14]. The surgery is facilitated by an ancillary surface code that grows with the code distance. Hence, for codes with large distance, it is necessary to serialize the quantum circuit to avoid a large overhead, as in [4].

In comparison, the logical gates constructed in our work do not incur any qubit overhead, as they are realized within the quantum code directly, avoiding the need for ancilla qubits. We also avoid any time overhead, as logical gates are implemented by a transversal circuit of single- and two-qubit gates, which can be applied in parallel. This also means that our implementation is compatible with any decoding scheme, as it can be executed between rounds of error correction.

We explicitly construct fault transversal gates for a certain $[[30, 8, 3]]$ hyperbolic surface code that we call Bring's code. We show that we can generate a subgroup of the Clifford group $\mathcal{C}_8$ using transversal circuits. By adding a single logical entangling gate, which can be obtained using other techniques, we generate the full Clifford group $\mathcal{C}_8$. Alternatively, when we restrict the logical subspace to four qubits, we can generate all of $\mathcal{C}_4$, using transversal circuits only.

In this work we do not consider any constraints on the locality of qubit interactions. In particular, we make use of qubit permutations, which in principle, can map any qubit to any other qubit. Thus, in order to take full advantage of our construction, a potential hardware architecture ideally supports non-local interactions between qubits. Fortunately, there have been several advances in experimental setups that can realize such unconstrained connectivity. Linear optical quantum computation, for example, is naturally flexible in terms of connectivity [15, 16, 17]. The Moussa folding construction is used by the fusion-based photonic scheme proposed in [18]. In a similar vain, modular architectures, in which modules are interlinked by a photonic interface, allow for non-local connectivity [19, 20, 21]. Other approaches to quantum computation, such as mobile qubits [22] or qubits coupled to a common cavity mode [23, 24], even allow for fast any-to-any connectivity between qubits. In particular, [25] proposes an architecture based on Rydberg atoms, or alternatively, ion-traps, in which it is possible to have fast, long-range interactions between hundreds of physical qubits.

Bring's code is small enough that this scheme could be implemented on NISQ hardware. Together with the fold-transversal gates it provides a simple set-up in which fault-tolerant Clifford gates could be demonstrated.

In Section 2 we introduce the theory of *fold-transversal* gates. When discussing stabilizer codes, we largely employ the language of homology which is introduced in Section 2.1. In Section 2.2 we give the definition of a ZX-*duality* of a homological code, and show how the symmetries of the code interact with the ZX-dualities. We then apply this to a class of highly symmetric two-dimensional hyperbolic surface codes in Section 2.3. These codes all have ZX-dualities, which are described in Theorem 5. Armed with this theory of ZX-dualities, in Section 2.4 we characterize when these dualities give rise to *fold-transversal* Clifford gates on the logical subspace of the code. In Section 3 we discuss the example of *Bring's code* and work out in detail the fold-transversal gates and their action on the logical subspace.

2 Folding along a ZX-duality

2.1 CSS quantum codes

A *stabilizer code* is defined by a subgroup $\mathcal{S}$ of the Pauli group $\mathcal{P}_n$ which is abelian and does not contain $-I$. We call the elements of some distinguished set of generators the *(stabilizer) checks* of the stabilizer code. A stabilizer code is a *Calderbank-Shor-Steane (CSS) code* if there exists a generating set of $\mathcal{S}$ such that each generator acts as either Pauli-X or Pauli-Z on all qubits in its support. Hence, we can define a CSS code C in terms of two binary matrices $H_X \in \mathbb{F}_2^{r_X \times n}$ and $H_Z \in \mathbb{F}_2^{r_Z \times n}$ where each row corresponds to the support indication vector of a Pauli-X or Pauli-Z generator, respectively. Note we do not require H_X or H_Z to have full rank. As X-type and Z-type Pauli operators commute if and only if the overlap of their supports contains an even number of qubits we must have

$$H_X H_Z^\top = 0. \tag{1}$$

In the language of homology, a CSS code C is equivalent to a chain complex over the field $\mathbb{F}_2 = \mathbb{Z}/2$:

$$C = \{\mathbb{F}_2^{r_Z} \xrightarrow{H_Z^\top} \mathbb{F}_2^n \xrightarrow{H_X} \mathbb{F}_2^{r_X}\}.$$

We can think of $H_Z^\top$ and H_X as *boundary operators* ∂_2 and ∂_1, respectively:

$$C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0$$

where $C_2 = \mathbb{F}_2^{r_Z}$, $C_1 = \mathbb{F}_2^n$ and $C_0 = \mathbb{F}_2^{r_X}$. The elements of C_i are called *i-chains*. Note that the space of i-chains comes with a natural basis due to the construction of C_i as a free vector space on a basis. The fact that "the boundary of a boundary is zero", i.e. $\partial_1 \partial_2 = 0$, is guaranteed by Equation (1). The group generated by the Z-type stabilizer operators $\mathcal{S}_Z$ corresponds to boundaries $B_1 = \operatorname{im} \partial_2$ and the Z-type Pauli-operators normalizing the X-type stabilizer operators $N(\mathcal{S}_X)_Z$ corresponds to cycles $Z_1 = \ker \partial_1$. Similarly, $\mathcal{S}_X$ and $N(\mathcal{S}_Z)_X$ correspond to coboundaries $B^1 = \operatorname{im} \delta_0$ and cocycles $Z^1 = \ker \delta_1$, respectively, where $\delta_0 = \partial_1^\top = H_X^\top$ and $\delta_1 = \partial_2^\top = H_Z$. The Z-type logical operators of the code are found as cycles modulo boundaries, which is the (first) *homology* of the code, $\mathcal{H}_1 := Z_1/B_2$. Dually, the X-type logical operators are cocycles modulo coboundaries. This is the (first) *cohomology* of the code, $\mathcal{H}^1 := Z^1/B^1$. A (co)cycle that is not trivial modulo a (co)boundary is called an *essential (co)cycle*.

2.2 ZX-Dualities

Given a CSS Code $C = (H_Z, H_X)$, the *dual* CSS code is

$$C^\top = \{\mathbb{F}_2^{r_X} \xrightarrow{H_X^\top} \mathbb{F}_2^n \xrightarrow{H_Z} \mathbb{F}_2^{r_Z}\}.$$

A *self-dual* CSS code C has $C = C^\top$.

An *isomorphism* of CSS codes $\sigma : C \to C'$ is a triple of permutation matrices $(\sigma_Z, \sigma_n, \sigma_X)$ where

$$\sigma_Z : \mathbb{F}_2^{r_Z} \to \mathbb{F}_2^{r'_Z},$$
$$\sigma_n : \mathbb{F}_2^{n} \to \mathbb{F}_2^{n'},$$
$$\sigma_X : \mathbb{F}_2^{r_X} \to \mathbb{F}_2^{r'_X},$$

such that the following diagram commutes:

$$\begin{tikzcd}
\mathbb{F}_2^{r_Z} \arrow[r, "H_Z^\top"] \arrow[d, "\sigma_Z"] & \mathbb{F}_2^n \arrow[r, "H_X"] \arrow[d, "\sigma_n"] & \mathbb{F}_2^{r_X} \arrow[d, "\sigma_X"] \\
\mathbb{F}_2^{r'_Z} \arrow[r, "H_Z'^\top"] & \mathbb{F}_2^{n'} \arrow[r, "H'_X"] & \mathbb{F}_2^{r'_X}
\end{tikzcd}$$

The *automorphism group* of a CSS code C is the group of self-isomorphisms $\mathrm{Aut}(C) = \{\sigma : C \to C\}$.

We are also interested in weaker notions of isomorphism between codes that allow for swapping the X- and Z-checks. Given a CSS code $C = (H_Z, H_X)$ we define the *underlying classical* code $C/\sim_{ZX}$ to be given by the parity check matrix H as:

$$C/\sim_{ZX} := \Big\{ \mathbb{F}_2^n \xrightarrow{H := \begin{pmatrix} H_X \\ H_Z \end{pmatrix}} \mathbb{F}_2^r \Big\}.$$

where $r := r_X + r_Z$. This classical code remembers all the checks of the quantum code, but it has forgotten whether each check is an X- or Z-type check.

An *automorphism* of a classical code with parity check matrix $H : \mathbb{F}_2^n \to \mathbb{F}_2^r$ is a pair of permutation matrices (σ_n, σ_r) where

$$\sigma_n : \mathbb{F}_2^n \to \mathbb{F}_2^n,$$
$$\sigma_r : \mathbb{F}_2^r \to \mathbb{F}_2^r,$$

such that the following diagram commutes:

$$\begin{tikzcd}
\mathbb{F}_2^n \arrow[r, "H"] \arrow[d, "\sigma_n"] & \mathbb{F}_2^r \arrow[d, "\sigma_r"] \\
\mathbb{F}_2^n \arrow[r, "H"] & \mathbb{F}_2^r
\end{tikzcd}$$

Given a CSS code $C = (H_Z, H_X)$ we write the automorphisms of the underlying classical code $C/\sim_{ZX}$ as

$$\mathcal{G} := \mathrm{Aut}(C/\sim_{ZX}).$$

The subgroup of $\mathcal{G}$ that fixes the X- and Z-sectors corresponds precisely to $\mathrm{Aut}(C)$ defined above. In more detail, these elements of $\mathcal{G}$ are pairs (σ_n, σ_r) where σ_r has block form

$$\sigma_r = \begin{pmatrix} \sigma_X & 0 \\ 0 & \sigma_Z \end{pmatrix}$$

and then $(\sigma_Z, \sigma_n, \sigma_X)$ is the corresponding automorphism of C.

The elements τ of $\mathcal{G}$ that swap the X- and Z-sectors are called *ZX-dualities.* These are pairs $\tau = (\tau_n, \tau_r)$ such that τ_r has the block form:

$$\tau_r = \begin{pmatrix} 0 & \tau_Z \\ \tau_X & 0 \end{pmatrix}.$$

In this case, $\tau = (\tau_Z, \tau_n, \tau_X)$ is an isomorphism $\tau : C \to C^\top$, and we call such a code *self-ZX-dual.* The set of all ZX-dualities of C we denote by $\mathcal{D}_{ZX}$.

Lemma 1. *Given a self-ZX-dual CSS code C, the set of ZX-dualities for C is a coset of* $\mathrm{Aut}(C)$ *in $\mathcal{G}$:*

$$\mathcal{D}_{ZX} = \tau \mathrm{Aut}(C),$$

where τ is any choice of a ZX-duality for C.

Proof. It is not hard to see that from a given ZX-duality $\tau : C \to C^\top$ and an isomorphism $\sigma : C \to C$ we get another ZX-duality from the composition $\tau\sigma$. Conversely, given another ZX-duality $\tau' = (\tau'_n, \tau'_r)$ we find that

$$\tau_r^{-1}\tau'_r = \begin{pmatrix} 0 & \tau_Z \\ \tau_X & 0 \end{pmatrix}^{-1} \begin{pmatrix} 0 & \tau'_Z \\ \tau'_X & 0 \end{pmatrix}$$
$$= \begin{pmatrix} \tau_X^{-1}\tau'_X & 0 \\ 0 & \tau_Z^{-1}\tau'_Z \end{pmatrix} \in \mathrm{Aut}(C).$$

Therefore we have shown the identity $\mathcal{D}_{ZX} = \tau\mathrm{Aut}(C)$. □

Corollary 2. *For a self-ZX-dual code C, the number of ZX-dualities equals the size of the automorphism group of C:*

$$|\mathcal{D}_{ZX}| = |\mathrm{Aut}(C)|.$$

Given CSS codes $C = (H_Z, H_X)$ and $C' = (H'_Z, H'_X)$ we can form the *direct sum* CSS code:

$$C \oplus C' := (H_Z \oplus H'_Z, H_X \oplus H'_X).$$

Lemma 3. *Given a CSS code C with ZX-duality $\tau : C \to C^\top$ then $C \oplus C$ has a ZX-duality without any fixed points.*

Proof. We define the ZX-duality given by $\tau \oplus \tau$ followed by the permutation that swaps the two summands:

$$C \oplus C \xrightarrow{\tau\oplus\tau} C^\top \oplus C^\top \xrightarrow{\text{swap}} C^\top \oplus C^\top = (C \oplus C)^\top .$$

□

In what follows we will often abuse notation and write $\tau(i)$ for the action of τ_n on the i-th basis vector of $\mathbb{F}_2^n$.

2.3 Hyperbolic surface codes

In this section we review the construction of hyperbolic triangle groups ([26] Section 2.4), and how they give rise to two dimensional hyperbolic surface codes [27, 28]. We then find the ZX-dualities in this context.

The Poincaré disc model of the hyperbolic plane is written as $\mathbb{D} = \{z \in \mathbb{C} : |z| < 1\}$. The group of orientation preserving automorphisms of $\mathbb{D}$ is given by the Möbius transforms,

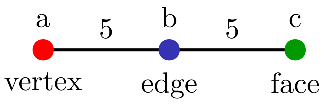
$$\begin{aligned}\mathrm{Aut}(\mathbb{D}^+) = \Big\{ z \mapsto \frac{az+b}{cz+d} \;\Big|\; & a,b,c,d \in \mathbb{R}, \\ & ad-bc = 1 \Big\} \\ \cong \mathrm{PSL}(2,\mathbb{R}).&\end{aligned}$$

By also including the reflection given by complex conjugation $z \mapsto \bar{z}$, we generate the full automorphism group $\mathrm{Aut}(\mathbb{D})$.

The Coxeter reflection group for the $\{5,5\}$-tiling of $\mathbb{D}$ has presentation

$$R_{5,5} := \langle a, b, c \mid a^2, b^2, c^2, (ab)^5, (ac)^2, (bc)^5 \rangle.$$

The generators a, b, c here correspond to reflections that *destabilize* a vertex, edge, or face respectively. We can summarize this presentation using the Coxeter diagram:



See Figure 2ii. The rotations ab, ac, bc correspond to the rotation around a face, edge or vertex respectively. The subgroup generated by these rotations we write as

$$R^+_{5,5} = \langle ab, ac, bc \rangle.$$

This subgroup has index 2 in $R_{5,5}$. Next we choose a normal subgroup Γ of $R^+_{5,5}$ with finite index, that acts freely on $\mathbb{D}$. The surface we are interested in is the compact Riemann surface

$$S = \mathbb{D}/\Gamma.$$

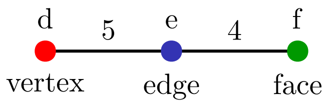
This surface is tiled by triangles, which we can identify with the elements of the group $\mathcal{G}^+ := R^+_{5,5}/\Gamma$ once we choose a fiducial tile to act upon. Up to conjugation, the stabilizer subgroup of a face, edge or vertex is generated by the rotations $ab, ac, bc \in \mathcal{G}^+$, respectively. Therefore, the chain complex of our quantum code is obtained from the free vector spaces generated by cosets of these:

$$\begin{aligned} C = \Big\{ & \mathbb{F}_2^{r_Z} := \mathbb{F}_2[\mathcal{G}^+/\langle ab\rangle] \\ & \xrightarrow{H_Z^\top} \mathbb{F}_2^n := \mathbb{F}_2[\mathcal{G}^+/\langle ac\rangle] \\ & \xrightarrow{H_X} \mathbb{F}_2^{r_X} := \mathbb{F}_2[\mathcal{G}^+/\langle bc\rangle] \Big\}. \end{aligned} \tag{2}$$

Proceeding with the theory developed in Section 2.2 we now seek the automorphism group $\mathcal{G}$ of the underlying classical code of C. We can do this using another Coxeter reflection group $R_{5,4}$ for the $\{5,4\}$-tiling of $\mathbb{D}$. This has presentation

$$R_{5,4} := \langle d, e, f \mid d^2, e^2, f^2, (de)^5, (df)^2, (ef)^4 \rangle.$$

The generators d, e, f here correspond to reflections that destabilize a vertex, edge, or face respectively. The rotations de, df, ef correspond to the rotation around a face, edge or vertex respectively. We can summarize this presentation using the Coxeter diagram:



See Figure 2i.

Lemma 4. *$R_{5,5}$ has an outer automorphism that swaps the generators a, c while fixing b. Moreover, $R_{5,5}$ is a subgroup of $R_{5,4}$ in which the outer automorphism is conjugation by f.*

Proof. The outer automorphism is clear from the presentation of $R_{5,5}$. There is a group monomorphism $R_{5,5} \rightarrowtail R_{5,4}$ given by sending the generators

$$a \mapsto e, \; b \mapsto d, \; c \mapsto f^{-1}ef,$$

and conjugation of these generators by f has the required effect. □

Theorem 5. *For the hyperbolic code C defined in Equation (2) we have the following:*

$$\mathrm{Aut}(C) = R_{5,5}/\Gamma,$$
$$\mathrm{Aut}(C/\sim_{ZX}) = R_{5,4}/\Gamma.$$

Furthermore, C is self-ZX-dual as exhibited by the ZX-duality $\tau = f \in R_{5,4}$.

Proof. (Sketch.) By Lemma 4 we can think of $R_{5,5}$ as a subgroup of $R_{5,4}$, where the latter has an additional generator f. The reflection f divides each of the fundamental triangles of $R_{5,5}$ with internal angles $\pi/2$, $\pi/5$, $\pi/5$ into two triangles with internal angles $\pi/2$, $\pi/4$, $\pi/5$, cf. Figures 2i and 2ii.

By studying the cosets which give rise to C and $C/\sim_{ZX}$ in the universal cover, and noting that Γ is normal and acts fixed-point free, our arguments regarding the cosets are preserved under the covering map induced by Γ onto the finite surface. □

Using this result and Lemma 1 we obtain all the ZX-dualities of our hyperbolic code C as

$$\mathcal{D}_{ZX} = fR_{5,5}/\Gamma.$$

A similar result holds for other symmetric Coxeter reflection groups $R_{l,l}$ for $l \geq 3$. There is always a subgroup inclusion $R_{l,l} \rightarrowtail R_{l,4}$ coming from the outer automorphism of $R_{l,l}$. In the case of $R_{3,3}$ this group is S_4 the (finite) symmetry group of the tetrahedron, and $R_{3,4}$ is the symmetry group of the cube. The inclusion $R_{3,3} \rightarrowtail R_{3,4}$ corresponds to the inclusion of a tetrahedron inside a cube. When $l = 4$, we get the group $R_{4,4}$ which is the (infinite) symmetry group of the square two-dimensional lattice. The inclusion $R_{4,4} \rightarrowtail R_{4,4}$ exhibits an inclusion of this lattice into a $1/\sqrt{2}$ scaled and rotated copy of the same lattice.

2.4 Fold-Transversal Clifford Gates

Some of the ZX-dualities in $\mathcal{D}_{ZX}$ can be used to implement logical gates using circuits of low, constant depth.

Given a ZX-duality τ, a unitary operator on the n qubits is called *fold-transversal* when it is a tensor product of single qubit unitaries, and two-qubit unitaries with support on the orbits $\{(i, \tau(i))\}_{i=1,\ldots,n}$ of τ.

A unitary operator on the n qubits of a CSS code C is called an *encoded logical gate* when it commutes with the codespace projector.

2.4.1 Hadamard-type fold-transversal gates

The simplest such gate is of what we call *Hadamard-type*, meaning it exchanges logical X-operators with logical Z-operators. Given a ZX-duality $\tau \in \mathcal{D}_{ZX}$ we define this as the following fold-transversal operator on the n physical bits:

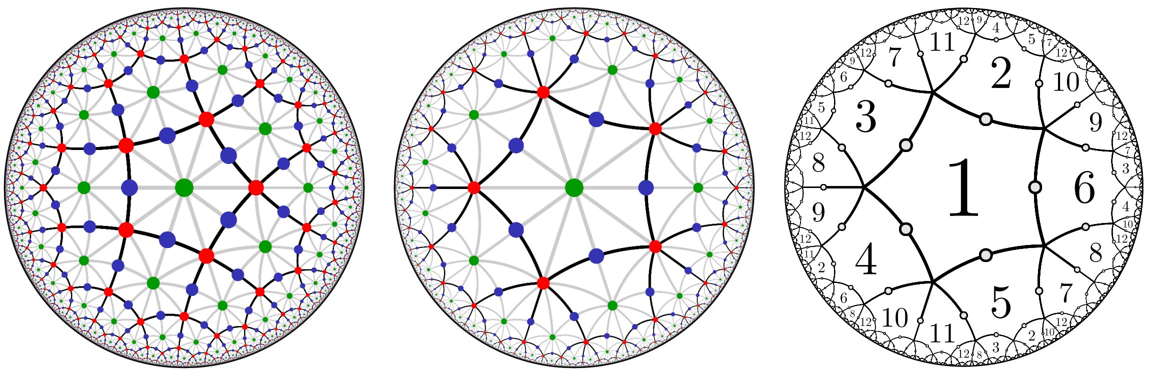
$$H_\tau = \bigotimes_{\substack{i=1,..,n \\ i<\tau(i)}} \mathrm{SWAP}_{i,\tau(i)} \bigotimes_{i=1}^{n} H_i$$

Theorem 6. *Given a ZX-duality τ for a CSS code C then H_τ is an encoded logical gate.*

Proof. The operator H_τ acts on elements of the Pauli group $\mathcal{P}_n$ via conjugation. Using the identities $H_i X_i H_i^\dagger = Z_i$ and $H_i Z_i H_i^\dagger = X_i$, as well as the fact that τ maps between X- and Z-checks, we see that H_τ leaves the stabilizer group $\mathcal{S}$ invariant. Thus, the normalizer of the stabilizer group $N(\mathcal{S})$ must also be invariant. In other words, H_τ leaves the code space invariant and maps logical operators onto logical operators. □

Note that H_τ is generally not simply a logical Hadamard, exchanging pairs of logical Pauli-operators $\bar{X}_i$ with $\bar{Z}_i$, as τ does not have to be compatible with the choice of logical basis. More severely, in general, we do not have $H_\tau^2 = I$. This is only the case if $\tau^2 = \mathrm{id}$.

When we construct H_τ for the surface code, using the ZX-duality τ shown in Figure 1i, we obtain the Hadamard-gate described by Moussa [1].



(i) The $\{5,4\}$-tiling of the Poincaré hyperbolic disc model. Vertices, edges and faces are marked with red, blue and green dots respectively. This tiling has symmetry group the Coxeter reflection group $R_{5,4}$.

(ii) The $\{5,5\}$-tiling of the Poincaré hyperbolic disc model. Vertices, edges and faces are marked with red, blue and green dots respectively. This tiling has symmetry group the Coxeter reflection group $R_{5,5}$.

(iii) Bring's curve as a quotient of the $\{5,5\}$-tiled hyperbolic disc. Faces are identified according to the number scheme. Each edge is associated with a qubit, shown with a grey circle, and faces/vertices define Z/X-check operators.

Figure 2: Hyperbolic codes are defined using the homology of compact hyperbolic surfaces. We construct these tiled surfaces as quotients of the $\{5,5\}$-tiling (ii), for example Bring's code (iii). We find ZX-dualities of the hyperbolic codes by forgetting the distinction between vertices and faces in (ii), which yields the $\{5,4\}$-tiling (i).

2.4.2 Phase-type fold-transversal gates

Given a ZX-duality $\tau \in \mathcal{D}_{ZX}$ we define the following fold-transversal operator on the n physical qubits:

$$S_\tau = \bigotimes_{\substack{i=1,\dots,n \\ i=\tau(i)}} S_i^{(\dagger)} \bigotimes_{\substack{i=1,\dots,n \\ i<\tau(i)}} \mathrm{CZ}_{i,\tau(i)}$$

for some choice of S_i and $S_i^\dagger$ on the fixed points $i=\tau(i)$. We call this a *phase-type* gate.

Theorem 7. *Let $\tau \in \mathcal{D}_{ZX}$ be a ZX-duality for a CSS code C, which is self-inverse ($\tau^2 = \mathrm{id}$) and whose set of stabilized qubits is even, i.e. $|\{i=1,\dots,n \mid \tau(i)=i\}| \equiv 0 \mod 2$. Further, suppose that each X-check $X^{\otimes s}$ with support $s \subset \{1,\dots,n\}$ has (a) even overlap with the set of invariant qubits $\{i=1,\dots,n|\tau(i)=i\}$ and (b) contains an even number of two element orbits $\{i,\tau(i)\}$, where $i \neq \tau(i)$. Then there exists a phase-type gate S_τ that is an encoded logical gate for C.*

Proof. Due to condition (a) there exists a partition of the set of invariant qubits of τ into A and B such that each X-check is half supported on A and half supported on B. We define the following unitary operator on the n physical bits:

$$S_\tau = \bigotimes_{i\in A} S_i \bigotimes_{j\in B} S_j^\dagger \bigotimes_{\substack{i=1,\dots,n \\ i<\tau(i)}} \mathrm{CZ}_{i,\tau(i)}$$

Let us verify that S_τ leaves the stabilizer group $\mathcal{S}$ invariant. We do this by showing that S_τ sends check operators to other elements of $\mathcal{S}$. By virtue of S_τ being Clifford, this then extends to a permutation of $\mathcal{S}$. As S_τ is diagonal in the computational basis, it commutes with all Z-checks. Consider an X-check $X^{\otimes s}$ with support $s \subset \{1,\dots,n\}$. First, let us assume that s does not contain qubits invariant under τ, i.e. $s \cap (A\cup B) = \emptyset$. Note that for single-qubit Pauli-X operators we have that

$$\mathrm{CZ}_{i,j}\, X_i\, \mathrm{CZ}_{i,j}^\dagger = X_i Z_j.$$

Further, due to condition (b) we have that $|s \cap \tau(s)|$ is even, so that we can commute the Pauli-X and Pauli-Z operators. Hence, the operator $X^{\otimes s}$ is mapped by S_τ onto the operator $X^{\otimes s} Z^{\otimes \tau(s)}$. By virtue of τ mapping X-checks onto Z-checks, we know that $Z^{\otimes \tau(s)} \in \mathcal{S}$. Now, let us assume that $s \cap (A \cup B) \neq \emptyset$.

For Pauli-X operators we have that $S_i X_i S_i^\dagger = iX_iZ_i$ and $S_i^\dagger X_i S_i = -iX_iZ_i$. The check $X^{\otimes s}$ is mapped onto

$$i^{|s\cap A|}(-i)^{|s\cap B|}X^{\otimes s}Z^{\otimes \tau(s)}.$$

Each phase gate S_i ($S_i^\dagger$) introduces an unwanted factor of i ($-i$). However, by assumption on τ, we are guaranteed that $|s\cap A| = |s\cap B|$, so that these factors cancel out. □

As for the Hadamard-type gates, we have shown that the phase-type gate S_τ leaves the code space invariant and maps logical operators onto logical operators. We stress again that the action of S_τ will generally be different from applying a logical $\bar{S}_i$ on each logical qubit. However, we do have $S_\tau^2 = I$ by construction.

When we construct S_τ for the surface code, using the ZX-duality τ shown in Figure 1i, we obtain the S-gate described by Moussa [1]. Note there is another ZX-duality that rotates the surface code by 90 degrees. However, this rotation fixes a single physical qubit and so we cannot construct a phase-type encoded logical gate from this ZX-duality.

The next result shows we can get a phase-type encoded logical gate by *stacking* two copies of a self-ZX-dual code.

Corollary 8. *Let $\tau \in \mathcal{D}_{ZX}$ be a ZX-duality for a CSS code C which is self-inverse, then $C \oplus C$ has phase-type encoded logical gate:*

$$S_\tau = \bigotimes_{i=1,\ldots,n} \mathrm{CZ}_{i,n+\tau(i)}\,.$$

Proof. We use the ZX-duality defined in Lemma 3. This ZX-duality has no fixed points and is also self-inverse so the result follows from the previous theorem. □

2.5 The Symplectic Group

Instead of dealing with unitary Clifford operators directly, we consider a well-known relation to the *symplectic group* $\mathrm{Sp}_{2n}(\mathbb{F}_2)$.

By definition, we have that the Pauli group $\mathcal{P}_n$ is a normal subgroup of the Clifford group $\mathcal{C}_n$. The quotient group $\mathcal{C}_n/\mathcal{P}_n$ is isomorphic to the symplectic group $\mathrm{Sp}_{2n}(\mathbb{F}_2)$ [29]. In other words, the following short sequence is exact:

$$\begin{tikzcd} \mathcal{P}_n \arrow[r, tail, "\iota"] & \mathcal{C}_n \arrow[r, two heads, "\pi"] & \mathrm{Sp}_{2n}(\mathbb{F}_2) \end{tikzcd}$$

where ι is the canonical embedding map and π the quotient map. This implies that any element of the Clifford group $U \in \mathcal{C}_n$ can be specified by an element of the symplectic group $\pi(U) \in \mathrm{Sp}_{2n}(\mathbb{F}_2)$ and a representative of the Pauli group $P \in \mathcal{P}_n$. This fact is well-known and used e.g. for sampling random elements of the Clifford group [30]. Note that we can trivially implement logical Pauli operators $\bar{P} \in \mathcal{P}_k$ transversally.

From now on, we will use the notation $\tilde{U} := \pi(U)$ for any operator $U \in \mathcal{C}_n$.

3 Bring's code

We demonstrate our construction on a hyperbolic surface code, called Bring's code, which encodes 8 logical qubits into 30 physical qubits. Hyperbolic codes are good candidates for constructing interesting fold-transversal gates, as they are finite-rate and have large symmetry groups [27]. One reason for choosing Bring's code in particular is that its symmetries can be visualized as a 3D polyhedron (see Figure 3). Also, it is small enough to analyze the group generated by the fold-transversal gates with a computer algebra system, such as GAP [31]. Larger hyperbolic codes already encode too many logical qubits to do so.

We briefly discuss some further examples, such as hypergraph product codes, balanced product codes and block codes, in Appendix B.

3.1 Construction

Using the notation from Section 2.3 for Bring's code we define Γ equal to the normal closure in $R^+_{5,5}$ of the group generated by $(abcb)^3$. This single generator is enough to define the quotient of the hyperbolic disc $\mathbb{D}$, see Figure 2iii. Somewhat surprisingly, this quotient has an immersion into three dimensional

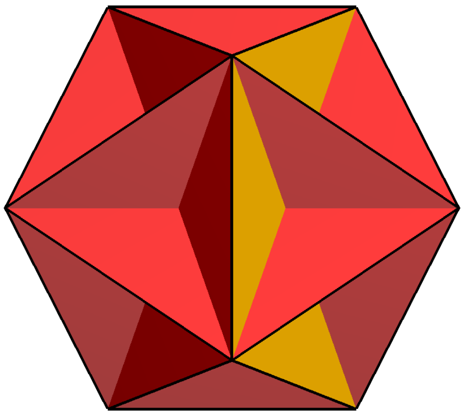


Figure 3: The great dodecahedron representation of Bring's code. Faces are pentagons which intersect. A single face is highlighted yellow. Each edge is associated with a qubit and vertices/faces define X/Z-check operators. This figure was made with Stella4D [32].

space known as the *great dodecahedron*. This is a Kepler-Poinsot polyhedron derived from the dodecahedron, see Figure 3. It has pentagonal faces, of which five meet at any vertex. By identifying the edges of the great dodecahedron with qubits and associating vertices/faces with X-/Z-checks, we obtain a quantum code with parameters $[[30, 8, 3]]$.

From a topological perspective, the great dodecahedron is a tiling (cellulation) of a genus-4 surface, $S = \mathbb{D}/\Gamma$. This tiling consists of 12 vertices, 30 edges and 12 faces. We refer to the derived code as *Bring's code*, as the underlying surface is known as *Bring's curve*, named after Erland Samuel Bring, who studied the algebraic equation that defines the surface as a complex curve. Bring's code was first defined in [27] and further analyzed in [10] and [33].

From Theorem 5, the group of symmetries of Bring's code is $\mathrm{Aut}(C) = R_{5,5}/\Gamma$ which in this case is found to be the permutation group S_5. In terms of the presentation of $R_{5,5}$ we have,

$$\begin{aligned}\mathrm{Aut}(C) &= R_{5,5}/\Gamma \\ &= \langle a, b, c \mid a^2, b^2, c^2, \\ &\qquad (ab)^5, (ac)^2, (bc)^5, (abcb)^3 \rangle \\ &= \mathrm{S}_5 \,.\end{aligned}$$

We also make the definitions $r := ab$ and $s := bc$ which are rotations around a face and vertex, respectively. Also from Theorem 5, we have the larger symmetry group which includes symmetries that exchange vertices and faces,

$$\begin{aligned}\mathrm{Aut}(C/\sim_{ZX}) &= R_{5,4}/\Gamma \\ &= \mathrm{S}_5 \times \mathrm{C}_2 \,.\end{aligned}$$

3.2 Logical bases

In the homological representation of CSS codes, the logical Z-operators correspond to a basis of the homology group $\mathcal{H}_1$. Essential cycles on Bring's surface can be associated to the 20 faces of the icosahedron, which forms the convex hull of the great dodecahedron, see Figure 3. First we note that a chain consisting of three edges of a triangle is a closed loop and hence it is clearly a cycle, i.e. an element of $Z_1 = \ker \partial_1$. Direct computation shows that it is also not a boundary, i.e. an element of $B_1 = \mathrm{im}\, \partial_2$. In fact, the set of all 20 chains associated to the triangles span the first homology group $\mathcal{H}_1$. However, as $k = \dim \mathcal{H}_1 = 8$, this spanning set can not be independent. Picking a suitable subset of these triangles gives rise to a basis.

A particularly nice basis is shown in Figure 4. The basis elements correspond to eight faces of the icosahedron. These faces are related by the action of the subgroup $\mathrm{C}_2 \times \mathrm{C}_2 \times \mathrm{C}_2 \subset \mathrm{S}_5$.

This choice of basis has a geometric representation that can be seen in Figure 5. We can place the icosahedron inside a cube, such that eight faces of the icosahedron can be associated with the eight vertices of the cube. The group $\mathrm{C}_2 \times \mathrm{C}_2 \times \mathrm{C}_2$ is a subgroup of the symmetry group of the cube which has a regular

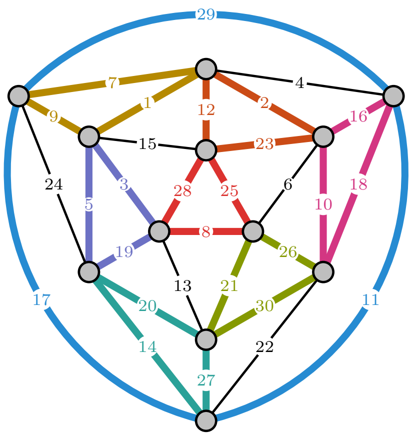


Figure 4: The basis $\mathfrak{B}_{\mathcal{H}_1}$ of the first homology group $\mathcal{H}_1$. Each element is a cycle of length three shown with thick lines of one color. Equivalently, this is a choice of logical Z-operators $\bar{Z}_1, \ldots, \bar{Z}_8$. Each triangular face of this planar embedding (of which there are 20, including the outer face) corresponds to a face of the convex hull of the great dodecahedron, which is the icosahedron.

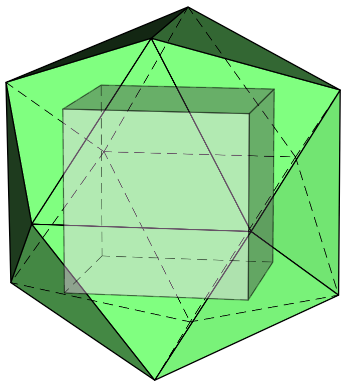


Figure 5: The convex hull of the great dodecahedron is the icosahedron. We can place a cube inside the icosahedron such that the eight vertices of the cube lie on the mid-points of eight faces of the icosahedron. The symmetries of the cube are also symmetries of the icosahedron. The orientation of the cube inside the icosahedron corresponds to a choice of basis in $\mathcal{H}_1$ where each triangle touched by a vertex of the cube is a basis vector, cf. Figure 4.

action on its vertices. It is generated by reflections corresponding to the three planes intersecting the cube in the middle. This placement of the icosahedron inside the cube is not unique and different placements correspond to different choices of a basis.

The logical X-operators correspond to a basis of the cohomology group $\mathcal{H}^1$. Similarly as for the homology basis, we can associate the essential cocycles with triangles of the icosahedron: for a given triangle there are three edges (not belonging to the triangle) which are adjacent to exactly one of the triangle's vertices and are not neighboring any of the triangle's edges. The chain formed by these edges is an essential cocycle. By associating eight triangles with the vertices of a cube we obtain a basis of $\mathcal{H}^1$. Unfortunately, choosing the bases of $\mathcal{H}_1$ and $\mathcal{H}^1$ in this fashion can not give rise to a symplectic basis of $\mathcal{H}_1 \oplus \mathcal{H}^1$.

In order to represent the logical gates that we are going to construct in terms of symplectic matrices, we need to fix ordered bases of $\mathcal{H}_1$ and $\mathcal{H}^1$. We fix the basis $\mathfrak{B}_{\mathcal{H}_1}$ of $\mathcal{H}_1$ as shown in Figure 4 with the order $\{1,7,9\}$, $\{2,12,23\}$, $\{3,5,19\}$, $\{10,16,18\}$, $\{21,26,30\}$, $\{8,25,28\}$, $\{11,17,29\}$, $\{14,20,27\}$. The ordered basis elements of $\mathfrak{B}_{\mathcal{H}^1}$, given in terms of supports on the edges (cf. Figure 4), are $\{1,6,11\}$, $\{2,8,22\}$, $\{4,5,25\}$, $\{10,13,17\}$, $\{24,28,30\}$, $\{12,19,26\}$, $\{9,16,27\}$, $\{15,20,29\}$.

The bases $\mathfrak{B}_{\mathcal{H}_1}$ and $\mathfrak{B}_{\mathcal{H}^1}$ give rise to a basis $\mathfrak{B}_{\mathcal{H}_1 \oplus \mathcal{H}^1}$ of the symplectic vector space $\mathcal{H}_1 \oplus \mathcal{H}^1$ in the obvious way. The symplectic product between the elements of $\mathfrak{B}_{\mathcal{H}_1 \oplus \mathcal{H}^1}$ is given by the matrix Φ with $\Phi_{i,j} = \langle b^i, b_j \rangle$ with $b^i \in \mathfrak{B}_{\mathcal{H}^1}$ and $b_j \in \mathfrak{B}_{\mathcal{H}_1}$.

$$\Phi = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

As Φ is not diagonal, $\mathfrak{B}_{\mathcal{H}_1 \oplus \mathcal{H}^1}$ is not a symplectic basis.

We can fix this by defining alternative bases as follows: Let $b'_i = b_i$ for $i \in \{1, \ldots, 5\}$, $b'_6 = b_2 + b_3 + b_5 + b_6$, $b'_7 = b_1 + b_4 + b_7$ and $b'_8 = b_1 + b_4 + b_7 + b_8$, defining a new basis $\mathfrak{B}'_{\mathcal{H}_1}$ of the first homology group. Further, let $b'^i = b^i$ for $i \in \{1, \ldots, 5\}$, $b'^6 = b^2 + b^3 + b^5 + b^6$, $b'^7 = b^8$ and $b'^8 = b^1 + b^4 + b^7$, defining a new basis $\mathfrak{B}'_{\mathcal{H}^1}$ of the first cohomology group. The resulting basis $\mathfrak{B}'_{\mathcal{H}_1 \oplus \mathcal{H}^1}$ of the combined symplectic space $\mathcal{H}_1 \oplus \mathcal{H}^1$ is a symplectic basis.

3.3 Permutation gates

Before we discuss the fold-transversal gate constrictions, we note that some non-trivial gates can be implemented simply by permuting qubits [34].

By definition, the automorphisms of Bring's code directly correspond to the symmetries of Bring's surface. The subgroup of symmetries of Bring's surface, which maps vertices to vertices and faces to faces, is S_5 and acts transitively on the 30 edges. This action is also faithful, so that we can write the generators a, b and c of S_5 in terms of cycles of S_{30}, i.e. cycles permuting the edge labels:

$$\begin{aligned}
a &= (2,3)(4,5)(6,8)(7,9)(10,13)(11,14)(12,15)(16,19)(18,20)(21,26)(22,27)(23,28)(24,29) \\
b &= (1,2)(3,6)(4,7)(5,10)(9,16)(11,17)(13,21)(14,22)(15,23)(18,24)(19,26)(20,30)(25,28) \\
c &= (2,4)(3,5)(6,11)(7,12)(8,14)(9,15)(10,18)(13,20)(17,25)(21,27)(22,26)(23,29)(24,28)
\end{aligned}$$

This action of S_5 on the edges extends to a linear representation ρ on the space of 1-chains.

Since clearly Z_1 and B_1 are invariant subspaces under any permutation matrix in the image of ρ, there exists a representation $\sigma_{\mathcal{H}_1} : \mathrm{S}_5 \to \mathrm{GL}(H_1)$ on the first homology group. Similarly, Z^1 and B^1 are invariant subspaces as well, so that we obtain a representation $\sigma_{\mathcal{H}^1} : \mathrm{S}_5 \to \mathrm{GL}(H^1)$ on the first cohomology group. See Appendix A.1 for the matrix representations.

The group of logical gates induced by permutations of the edges is generated by

$$\tilde{\sigma}_x = \left(\begin{array}{c|c} \tilde{\sigma}_{\mathcal{H}_1}(x) & 0 \\ \hline 0 & \tilde{\sigma}_{\mathcal{H}^1}(x) \end{array}\right)$$

where $x \in \{a, b, c\}$.

As the representation is faithful, we have that $\langle \tilde{\sigma}_a, \tilde{\sigma}_b, \tilde{\sigma}_c \rangle \cong \mathrm{S}_5 < \mathrm{Sp}_{16}(\mathbb{F}_2)$. This can be verified using a computer algebra system, such as GAP [31].

3.4 ZX-Dualities

The group of automorphisms of Bring's code is $\mathrm{Aut}(C) = \mathrm{S}_5$ which has order $5! = 120$. Furthermore, Bring's code is self-ZX-dual, by Theorem 5, with self-ZX-duality

$$\tau_0 := f \in R_{5,4}.$$

By Corollary 2 we know that there are 120 ZX-dualities exchanging X- and Z-checks of Bring's code. Among these, the number of self-inverse dualities ($\tau^2 = \mathrm{id}$) is 20. For the implementation of the Hadamard- and phase-type gates in Section 3.5 we will consider the ZX-duality τ_0.

Figure 6 shows the orbits of edges under the action of τ_0. As τ_0 is self-inverse each edge is either mapped onto a partner ($\tau_0(e_1) = e_2$ and $\tau_0(e_2) = e_1$), or it is fixed ($\tau_0(e) = e$). In the figure, the pairs are given the same color, while the fixed edges are drawn in black. The same information is shown in Figure 7 in the planar layout.

We have picked the ordered bases $\mathfrak{B}_{\mathcal{H}_1}$ and $\mathfrak{B}_{\mathcal{H}^1}$ in Section 3.2 such that τ_0 maps be-

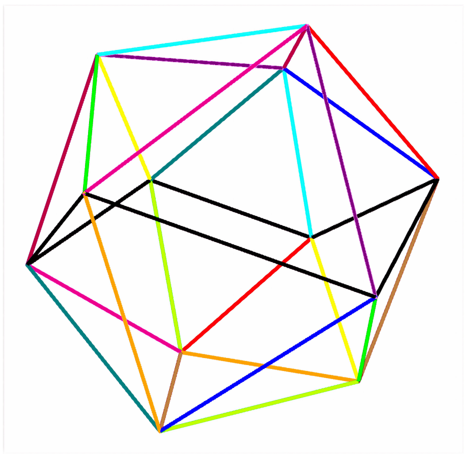


Figure 6: The ZX-duality τ_0 on Bring's code. Qubits are associated with edges. Qubits colored in black are left invariant under the ZX-duality. Qubits not left invariant come in pairs, as $\tau_0^2 = \mathrm{id}$, and given the same color.

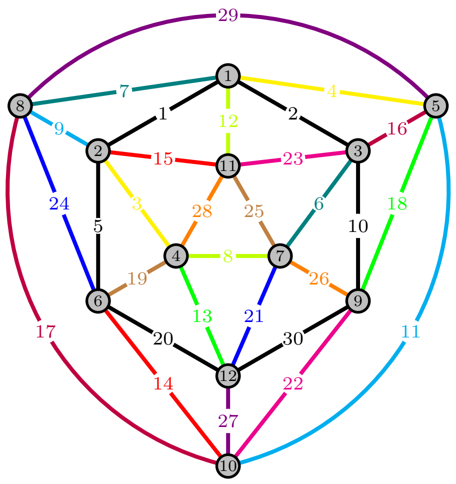


Figure 7: The orbits of τ_0 shown in the planar layout.

tween the pairs of basis elements, i.e. $\tau_0(b_i) = b^i$ for $b_i \in \mathfrak{B}_{\mathcal{H}_1}$ and $\tau_0(b^i) = b_i$ for $b^i \in \mathfrak{B}_{\mathcal{H}^1}$.

3.5 Fold-transversal gates

We will now use the ZX-duality τ_0 from the previous section to construct fold-transversal Clifford gates for Bring's code. This is straightforward, as τ_0 and the bases of $\mathcal{H}_1 \oplus \mathcal{H}^1$ were carefully chosen.

3.5.1 Hadamard-type gate

We have picked the bases $\mathfrak{B}_{\mathcal{H}_1}$ and $\mathfrak{B}_{\mathcal{H}^1}$ such that $\tau_0(b_i) = b^i$ and $\tau_0(b^i) = b_i$ for each $b_i \in \mathfrak{B}_{\mathcal{H}_1}$ and $b^i \in \mathfrak{B}_{\mathcal{H}^1}$. Therefore, the representation of $\tilde{H}_{\tau_0}$ in the basis of $\mathfrak{B}_{\mathcal{H}_1 \oplus \mathcal{H}^1}$ takes the simple form

$$\tilde{H}_{\tau_0} = \left(\begin{array}{c|c} 0 & I_8 \\ \hline I_8 & 0 \end{array}\right).$$

Changing to the symplectic basis $\mathfrak{B}'_{\mathcal{H}_1 \oplus \mathcal{H}^1}$ yields

$$\tilde{H}'_{\tau_0} = \left(\begin{array}{ccc|ccc} & & & I_6 & & \\ & 0 & & & 0 & 1 \\ & & & & 1 & 1 \\ \hline I_6 & & & & & \\ & 1 & 1 & & 0 & \\ & 1 & 0 & & & \end{array}\right).$$

3.5.2 Phase-type gate

First we note that τ_0 fulfills all requirements listed in Section 2.4.2: it is self-dual, the number of stabilized qubits is six, there are either none or two stabilized qubits in the support of each X-check and no X-check contains an orbit of cardinality two. All of this can be verified by inspecting Figure 7.

The representation of $\tilde{S}_{\tau_0}$ in the basis $\mathfrak{B}_{\mathcal{H}_1 \oplus \mathcal{H}^1}$ takes the simple form

$$\tilde{S}_{\tau_0} = \left(\begin{array}{c|c} I_8 & I_8 \\ \hline I_8 & 0 \end{array}\right).$$

Changing to the symplectic basis $\mathfrak{B}'_{\mathcal{H}_1 \oplus \mathcal{H}^1}$ yields

$$\tilde{S}'_{\tau_0} = \left(\begin{array}{ccc|c} & I_8 & & 0 \\ \hline I_6 & & & \\ & 1 & 1 & I_8 \\ & 1 & 0 & \end{array}\right).$$

3.6 Achievable gates

Let G_{τ_0} be the group generated by the permutation gates and fold-transversal gates, i.e. it is generated by the set $\{\tilde{\sigma}_a, \tilde{\sigma}_b, \tilde{\sigma}_c, \tilde{H}_{\tau_0}, \tilde{S}_{\tau_0}\}$. We observe, using GAP [31], that the group G_{τ_0} is isomorphic to $\mathrm{Sp}_8(\mathbb{F}_2) \times \mathrm{C}_2$. We can also verify that G_{τ_0} does in fact not depend on the choice of ZX-duality τ_0. Further, a single ZX-duality generates the same group as taking all possible Hadamard-type and phase-type gates of Bring's code.

3.6.1 Full Clifford group

Adding the symplectic equivalent of a single CNOT- or CZ-gate between any of the logical qubits to the set $\{\tilde{\sigma}_a, \tilde{\sigma}_b, \tilde{\sigma}_c, \tilde{H}_{\tau_0}, \tilde{S}_{\tau_0}\}$ elevates it to a generating set of $\mathrm{Sp}_{16}(\mathbb{F}_2)$. Hence, we can obtain the full Clifford group $\mathcal{C}_8$.

This can be done in several ways: In [10] the authors show how to use Dehn twists to implement logical CNOT-gates. Their technique requires either an increase of the qubit degree[1] up to $O(\log(n))$ or, alternatively, an increase in overhead due to ancilla qubits. The temporal overhead of this technique is $O(d^2) = O(\log(n)^2)$. Alternatively, the authors of [14] suggest a generalization of lattice surgery facilitated by an ancillary surface code. This results in an overhead in ancilla qubits scaling as $O(d^2)$.

3.6.2 Fold-transversal Clifford group

Implementing entangling gates can certainly be done using lattice surgery and code deformation techniques. However, it would be much more appealing to achieve all logical Clifford gates fold-transversally. It turns out that this can indeed be done: Let $\tilde{\sigma}_r = \tilde{\sigma}_a\tilde{\sigma}_b$ and $\tilde{\sigma}_s = \tilde{\sigma}_b\tilde{\sigma}_c$.

Let $\tilde{\sigma}_r = \tilde{\sigma}_a\tilde{\sigma}_b$ and $\tilde{\sigma}_s = \tilde{\sigma}_b\tilde{\sigma}_c$. Under the action of G_{τ_0} the symplectic space $\mathcal{H}_1 \oplus \mathcal{H}^1$ decomposes into two invariant subspaces V and W. Both, V and W have dimension eight. The subspace V is symplectic and gives rise to a set of Pauli-operators acting on four qubits. The subspace W, on the other hand, is maximally isotropic ($W^\perp = W$), in other words, it is a Lagrangian subspace. A basis of both subspaces can be found in Appendix A.2. The group G_{τ_0} acts faithfully as $\mathrm{Sp}_8(\mathbb{F}_2)$ on V. Therefore, we obtain the full Clifford group $\mathcal{C}_4$ when we restrict the logical subspace to the one induced by V.

[1]Meaning the cumulative total number of other qubits a single qubit has to be connected with. The instantaneous qubit degree is constant throughout the procedure.

4 Conclusion

We have introduced a method to implement Clifford gates in CSS quantum codes using symmetries that we call ZX-dualities. A ZX-duality defines a fold-transversal operator, which is a quantum circuit with gates supported on the orbits of τ. In particular, we defined Hadamard-type gates H_τ and phase-type gates S_τ, derived from suitable ZX-dualities τ. For the surface code with τ being a literal fold (see Figure 1i), this is equivalent to the construction of Moussa [1]. Further, we explicitly discussed Bring's code as an example, showing that it is possible to generate the Clifford group $\mathcal{C}_4$ by restricting the logical subspace.

It would be interesting to understand the set of fold-transversal gates for a family of LDPC quantum codes, such as hyperbolic codes or those discussed in Appendix B. While one can verify the construction using a computer algebra system, as we have done here, this has its limitations. This is because the groups generated by the fold-transversal and permutation gates become very large, even for a moderate number of encoded qubits k.

Finally, we expect our results to generalize to other code families that support a notion of Fourier duality: non-CSS qubit stabilizer codes, qudit stabilizer codes, and subsystem codes.

Acknowledgements

N.P.B. acknowledges support through the EPSRC Prosperity Partnership in Quantum Software for Simulation and Modelling (Grant No. EP/S005021/1). S.B. acknowledges financial support by the Foundation for Polish Science through TEAM-NET project (contract no. POIR.04.04.00-00-17C1/18-00).

References

[1] Jonathan E. Moussa. Transversal Clifford gates on folded surface codes. *Phys. Rev. A*, 94:042316, Oct 2016.

[2] Aleksander Kubica, Beni Yoshida, and Fernando Pastawski. Unfolding the color code. *New Journal of Physics*, 17(8):083026, aug 2015.

[3] Nikolas P. Breuckmann and Jens Niklas Eberhardt. Quantum Low-Density Parity-Check Codes. *PRX Quantum*, 2:040101, Oct 2021.

[4] Daniel Gottesman. Fault-tolerant quantum computation with constant overhead. *Quantum Information and Computation*, (15-16):1338–1372, 2014.

[5] Omar Fawzi, Antoine Grospellier, and Anthony Leverrier. Constant overhead quantum fault tolerance with quantum expander codes. *Commun. ACM*, 64(1):106–114, dec 2020.

[6] Matthew B. Hastings, Jeongwan Haah, and Ryan O'Donnell. *Fiber Bundle Codes*, page 1276–1288. Association for Computing Machinery, New York, NY, USA, 2021.

[7] Pavel Panteleev and Gleb Kalachev. Quantum LDPC Codes With Almost Linear Minimum Distance. *IEEE Transactions on Information Theory*, 68(1):213–229, 2022.

[8] Nikolas P. Breuckmann and Jens N. Eberhardt. Balanced Product Quantum Codes. *IEEE Transactions on Information Theory*, 67(10):6653–6674, 2021.

[9] Pavel Panteleev and Gleb Kalachev. Asymptotically good quantum and locally testable classical LDPC codes. *arXiv preprint arXiv:2111.03654*, 2021.

[10] Nikolas P. Breuckmann, Christophe Vuillot, Earl Campbell, Anirudh Krishna, and Barbara M. Terhal. Hyperbolic and semi-hyperbolic surface codes for quantum storage. *Quantum Science and Technology*, 2(3):035007, aug 2017.

[11] Ali Lavasani and Maissam Barkeshli. Low overhead Clifford gates from joint measurements in surface, color, and hyperbolic codes. *Phys. Rev. A*, 98:052319, Nov 2018.

[12] Anirudh Krishna and David Poulin. Fault-tolerant gates on hypergraph product codes. *Phys. Rev. X*, 11:011023, Feb 2021.

[13] Simon Burton and Dan Browne. Limitations on transversal gates for hypergraph product codes. *IEEE Transactions on Information Theory*, pages 1–10, 2021.

[14] Lawrence Z. Cohen, Isaac H. Kim, Stephen D. Bartlett, and Benjamin J. Brown. Low-overhead fault-tolerant quantum computing using long-range connectivity. *arXiv preprint arXiv:2110.10794*, 2021.

[15] Terry Rudolph. Why I am optimistic about the silicon-photonic route to quantum computing. *APL Photonics*, 2(3):030901, 2017.

[16] Hector Bombin, Isaac H Kim, Daniel Litinski, Naomi Nickerson, Mihir Pant, Fernando Pastawski, Sam Roberts, and Terry Rudolph. Interleaving: Modular architectures for fault-tolerant photonic quantum computing. *arXiv preprint arXiv:2103.08612*, 2021.

[17] Sara Bartolucci, Patrick Birchall, Hector Bombin, Hugo Cable, Chris Dawson, Mercedes Gimeno-Segovia, Eric Johnston, Konrad Kieling, Naomi Nickerson, Mihir Pant, et al. Fusion-based quantum computation. *arXiv preprint arXiv:2101.09310*, 2021.

[18] Isaac H Kim, Eunseok Lee, Ye-Hua Liu, Sam Pallister, William Pol, and Sam Roberts. Fault-tolerant resource estimate for quantum chemical simulations: Case study on Li-ion battery electrolyte molecules. *arXiv preprint arXiv:2104.10653*, 2021.

[19] Christopher Monroe and Jungsang Kim. Scaling the ion trap quantum processor. *Science*, 339(6124):1164–1169, 2013.

[20] Ramil Nigmatullin, Christopher J Ballance, Niel De Beaudrap, and Simon C Benjamin. Minimally complex ion traps as modules for quantum communication

and computing. *New Journal of Physics*, 18(10):103028, 2016.

[21] Naomi H. Nickerson, Joseph F. Fitzsimons, and Simon C. Benjamin. Freely scalable quantum technologies using cells of 5-to-50 qubits with very lossy and noisy photonic links. *Phys. Rev. X*, 4:041041, Dec 2014.

[22] Dolev Bluvstein, Harry Levine, Giulia Semeghini, Tout T Wang, Sepehr Ebadi, Marcin Kalinowski, Alexander Keesling, Nishad Maskara, Hannes Pichler, Markus Greiner, et al. A quantum processor based on coherent transport of entangled atom arrays. *arXiv preprint arXiv:2112.03923*, 2021.

[23] T. Pellizzari, S. A. Gardiner, J. I. Cirac, and P. Zoller. Decoherence, continuous observation, and quantum computing: A cavity qed model. *Phys. Rev. Lett.*, 75:3788–3791, Nov 1995.

[24] Andrew C. J. Wade, Marco Mattioli, and Klaus Mølmer. Single-atom single-photon coupling facilitated by atomic-ensemble dark-state mechanisms. *Phys. Rev. A*, 94:053830, Nov 2016.

[25] Joshua Ramette, Josiah Sinclair, Zachary Vendeiro, Alyssa Rudelis, Marko Cetina, and Vladan Vuletić. Any-to-any connected cavity-mediated architecture for quantum computing with trapped ions or Rydberg arrays. *arXiv preprint arXiv:2109.11551*, 2021.

[26] Ernesto Girondo and Gabino González-Diez. *Introduction to Compact Riemann Surfaces and Dessins d'Enfants*. London Mathematical Society Student Texts. Cambridge University Press, 2011.

[27] Nikolas P. Breuckmann and Barbara M. Terhal. Constructions and Noise Threshold of Hyperbolic Surface Codes. *IEEE Transactions on Information Theory*, 62(6):3731–3744, 2016.

[28] Nikolas P. Breuckmann. *Homological Quantum Codes Beyond the Toric Code*. PhD thesis, RWTH Aachen University, 2017.

[29] Beverley Bolt, TG Room, and GE Wall. On the Clifford collineation, transform and similarity groups. II. *Journal of the Australian Mathematical Society*, 2(1):80–96, 1961.

[30] Robert Koenig and John A. Smolin. How to efficiently select an arbitrary Clifford group element. *Journal of Mathematical Physics*, 55(12):122202, 2014.

[31] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.11.1*, 2021.

[32] Robert Webb. Stella: polyhedron navigator. *Symmetry: Culture and Science*, 11(1-4):231–268, 2003.

[33] Jonathan Conrad, Christopher Chamberland, Nikolas P. Breuckmann, and Barbara M. Terhal. The small stellated dodecahedron code and friends. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 376(2123):20170323, 2018.

[34] Markus Grassl and Martin Roetteler. Leveraging automorphisms of quantum codes for fault-tolerant quantum computation. In *2013 IEEE International Symposium on Information Theory*, pages 534–538, 2013.

[35] John Baez. *Golay Code*. American Mathematical Society, Visual Insight Blog, 2015. https://blogs.ams.org/visualinsight/2015/12/01/golay-code/.

A Matrices

Here we give further details on associated matrices coming from the construction of Bring's code in Section 3.

A.1 Qubit permutation

In the basis $\mathfrak{B}'_{\mathcal{H}_1}$ we have

$$\tilde{\sigma}_{\mathcal{H}_1}(a) = \begin{pmatrix} 1&0&0&0&0&0&0&0 \\ 0&0&0&0&1&1&1&0 \\ 0&1&1&0&0&0&0&1 \\ 0&0&0&1&0&1&1&1 \\ 0&0&0&0&1&0&0&0 \\ 1&1&0&0&0&0&1&0 \\ 1&0&0&0&1&0&1&0 \\ 0&1&0&0&1&1&1&1 \end{pmatrix} \qquad \tilde{\sigma}_{\mathcal{H}_1}(b) = \begin{pmatrix} 0&1&1&0&0&0&0&1 \\ 1&0&0&0&0&1&1&1 \\ 1&0&0&1&0&1&0&0 \\ 0&0&1&0&1&0&0&1 \\ 0&0&0&1&0&1&1&1 \\ 0&1&1&0&1&0&0&0 \\ 1&1&0&1&1&0&1&0 \\ 1&0&0&1&0&0&1&1 \end{pmatrix}$$

$$\tilde{\sigma}_{\mathcal{H}_1}(c) = \begin{pmatrix} 1&0&0&0&0&1&1&1 \\ 0&0&1&0&0&1&1&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&1&0&0&1&0&0&1 \\ 0&1&0&1&0&0&1&0 \\ 0&0&1&1&0&0&1&0 \\ 0&1&1&0&0&1&1&1 \end{pmatrix}$$

and in the basis $\mathfrak{B}'_{\mathcal{H}^1}$ we have $\tilde{\sigma}_{\mathcal{H}^1}(x) = \tilde{\sigma}_{\mathcal{H}_1}(x)^\top$, for $x \in \{a, b, c\}$.

Note that the matrices operate on *row vectors* from the *right*.

A.2 G_{τ_0}-Invariant subspaces of $\mathcal{H}_1 \oplus \mathcal{H}^1$

A symplectic basis of the symplectic space $V \subset \mathcal{H}_1 \oplus \mathcal{H}^1$ in the basis $\mathfrak{B}'_{\mathcal{H}_1 \oplus \mathcal{H}^1}$:

$$\left(\begin{array}{cccccccc|cccccccc} 0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&1&1&1&1&0&1&0&0&0&0&0&0&0&0&0 \\ 1&0&0&1&0&1&1&1&0&0&0&0&0&0&0&0 \\ \hline 0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0&0&1&0&1&1&0 \\ 0&0&0&0&0&0&0&0&0&1&1&0&1&1&1&1 \end{array}\right)$$

A basis of the Lagrangian space $W \subset \mathcal{H}_1 \oplus \mathcal{H}^1$ in the basis $\mathfrak{B}'_{\mathcal{H}_1 \oplus \mathcal{H}^1}$:

$$\left(\begin{array}{cccccccc|cccccccc}
1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0
\end{array}\right)$$

In the basis of V above, the action of the generators of $G_{\tau_0} \cong \mathrm{Sp}_8(\mathbb{F}_2)$ restricted to V is given by the following matrices:

$$\tilde{\sigma}_r|_V = \left(\begin{array}{cccc|cccc}
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0
\end{array}\right) \qquad
\tilde{\sigma}_s|_V = \left(\begin{array}{cccc|cccc}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1
\end{array}\right)$$

$$\tilde{H}_{\tau_0}|_V = \left(\begin{array}{cccc|cccc}
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
\hline
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{array}\right) \qquad
\tilde{S}_{\tau_0}|_V = \left(\begin{array}{cccc|cccc}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
\hline
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1
\end{array}\right)$$

Note that matrices operate on *row vectors* from the *right*.

B Further Examples

Here we discus some further examples of fold-transversal gates and the groups they generate. We do not go through all the details, as we did for Bring's code. However, the derivation of these results follows along the same steps.

B.1 Hyperbolic Surface Codes

We list some more examples of $\{5, 5\}$ hyperbolic surface codes obtained from the construction in Section 2.3, the first of which is Bring's code. The table shows the parameters $[[n, k, d]]$ for the code C, and the order of the automorphism group $\mathrm{Aut}(C)$ which is equal to the number of ZX-dualities.

Code parameters	$\lvert\mathcal{D}_{ZX}\rvert = \lvert\mathrm{Aut}(C)\rvert$
$[[30, 8, 3]]$	120
$[[40, 10, 4]]$	160
$[[80, 18, 5]]$	320
$[[150, 32, 6]]$	600
$[[180, 38, 4]]$	720
$[[330, 68, 6]]$	1320
$[[480, 95, ?]]$	1920

B.2 Hypergraph Product Code

Consider the $[6, 2, 4]$ code defined by the parity check matrix

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

acting on row vectors. The hypergraph product of this code with its dual gives a $[[52, 4, 4]]$ quantum code with checks of weights 3, 4 and 5. The automorphism group of the code, as defined in Section 2.2, is $(\mathrm{S}_3 \times \mathrm{S}_3) \rtimes \mathrm{C}_2$. This can be seen from the fact that S_3 is the symmetry group of the input code and we have the exchange of X- and Z-checks, giving the semi-direct product with C_2. There are four ZX-dualities and all of them fulfill the requirements of the theorems in Section 2.4. Using a suitable choice of basis for $\mathcal{H}_1 \oplus \mathcal{H}^1$, we can construct eight Hadamard- and phase-type gates. Together with the permutation gates they generate the group $(\mathrm{A}_6 \times \mathrm{A}_6) \rtimes \mathrm{D}_4$. This is an index-45696 subgroup of $\mathrm{Sp}_8(\mathbb{F}_2)$.

As for Bring's code, we can obtain the full group symplectic group $\mathrm{Sp}_8(\mathbb{F}_2)$ by adding a single entangling gate, e.g. by using surgery with an ancillary surface code [14]. Alternatively, the largest symplectic subgroup of $(\mathrm{A}_6 \times \mathrm{A}_6) \rtimes \mathrm{D}_4$ is $\mathrm{Sp}_4(\mathbb{F}_2)$, so we can obtain the full Clifford group on two logical qubits when restricting the code space accordingly.

B.3 Balanced Product Code

Balanced product quantum codes are a generalization of the hypergraph product codes, allowing for the construction of quantum codes that can achieve higher encoding rates and larger distances [8]. The balanced product between two classical codes, that share a common symmetry, can be understood as the hypergraph product between the two, with the symmetry subsequently factored out. We refer to [3, 8] for more details. Symmetry groups play a prominent role in the construction of balanced product codes, so that they are amenable to our scheme.

For symmetric classical codes, let us consider the extended binary Golay code $\mathcal{G}_{24}$ with parameters $[24, 12, 8]$ and its dual $\mathcal{G}_{24}^{\top}$. It turns out that, coincidentally, the parity checks of the Golay code can also be derived from the great dodecahedron (Figure 3), see [35] for details. The automorphism group of this parity check matrix of $\mathcal{G}_{24}$ is thus the same as that of Bring's code $\mathrm{S}_5 \times \mathrm{C}_2$ (see Section 3.1). The alternating group A_5 is an (up to conjugation) unique subgroup. The balanced product code $\mathcal{G}_{24} \otimes_{\mathrm{A}_5} \mathcal{G}_{24}^{\top}$ is a $[[20, 4, 3]]$ code with check weights 4, 5 and 7. It has 10 ZX-dualities, giving rise to Hadamard-type gates. However, only one of the ZX-dualities satisfies the requirements of Theorem 7, giving rise to a single phase-type gate.

The group of permissible qubit permutations is $\mathrm{C}_2 \times \mathrm{S}_4$ and it has a faithful representation on $\mathcal{H}_1 \oplus \mathcal{H}^1$. The permutation gates, the Hadamard-type gates and the phase-type gate generate the group $\mathrm{S}_3 \times \mathrm{S}_4$, which is an index-329011200 subgroup of $\mathrm{Sp}_8(\mathbb{F}_4)$. Adding a single entangling gate generates the full symplectic group $\mathrm{Sp}_8(\mathbb{F}_2)$.

Alternatively, consider the smaller subgroup $H = \mathrm{C}_2 \times (\mathrm{C}_5 \rtimes \mathrm{C}_4)$. The balanced product code $\mathcal{G}_{24} \otimes_H \mathcal{G}_{24}^\top$ is a $[[30, 6, 3]]$ code. Similarly, the permutation gates, the Hadamard-type gates and the phase-type gate generate an index-135491300609753088000 subgroup of $\mathrm{Sp}_{12}(\mathbb{F}_4)$. Adding a single entangling gate generates the full symplectic group $\mathrm{Sp}_{12}(\mathbb{F}_2)$.

A $[[60, 12, 5]]$ code can be obtained by taking the balanced product over the even smaller subgroup $\mathrm{C}_3 \rtimes \mathrm{C}_4$. Unfortunately, the space $\mathcal{H}_1 \oplus \mathcal{H}^1$ is 24-dimensional and thus too large for our computational methods.

B.4 Block Code

The following example shows that the fold-transversal gates do not always suffice to generate large groups. Consider the $[[16, 4, 4]]$ code, obtained by concatenating the $[[4, 2, 2]]$ code with itself. It is a weakly self-dual CSS code with classical code defined by the check matrix

$$\begin{pmatrix}
1 & 1 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 1
\end{pmatrix}$$

A natural basis of H_1 and H^1 is

$$\begin{pmatrix}
1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{pmatrix}$$

There exist 20 ZX-dualities, giving rise to 8 distinct Hadamard-type gates on $\mathcal{H}_1 \oplus \mathcal{H}^1$. Only one of the ZX-dualities satisfies the requirements of Theorem 7, giving rise to a phase-type gate. The group of permissible qubit permutations is $\mathrm{C}_2 \times \mathrm{S}_4$. However, unlike in the previous examples, the induced representation on $\mathcal{H}_1 \oplus \mathcal{H}^1$ is not faithful, as its image is isomorphic to the dihedral group D_6. The permutation gates, the Hadamard-type gates and the phase-type gate generate the group $\mathrm{C}_2 \times \mathrm{S}_3 \times \mathrm{S}_3$. Unlike in the previous examples, note even adding all $\tilde{\mathrm{CZ}}_{i,j}$-gates does not give rise to the full symplectic group $\mathrm{Sp}_8(\mathbb{F}_2)$, but to the orthogonal group $\mathrm{O}_8^+(\mathbb{F}_2)$, which is an index-272 subgroup.