Encoding One Logical Qubit Into Six Physical Qubits

Bilal Shaw[1,4,5],* Mark M. Wilde[1,5], Ognyan Oreshkov[2,5], Isaac Kremsky[2,5], and Daniel A. Lidar[1,2,3,5]
[1]*Department of Electrical Engineering,* [2]*Department of Physics and Astronomy,*
[3]*Department of Chemistry,* [4]*Department of Computer Science,*
[5]*Center for Quantum Information Science and Technology,*
University of Southern California, Los Angeles, California 90089, USA
(Dated: May 26, 2013; Received May 26, 2013; Revised; Accepted; Published)

We discuss two methods to encode one qubit into six physical qubits. Each of our two examples corrects an arbitrary single-qubit error. Our first example is a degenerate six-qubit quantum error-correcting code. We explicitly provide the stabilizer generators, encoding circuit, codewords, logical Pauli operators, and logical CNOT operator for this code. We also show how to convert this code into a non-trivial subsystem code that saturates the subsystem Singleton bound. We then prove that a six-qubit code without entanglement assistance cannot simultaneously possess a Calderbank-Shor-Steane (CSS) stabilizer and correct an arbitrary single-qubit error. A corollary of this result is that the Steane seven-qubit code is the smallest single-error correcting CSS code. Our second example is the construction of a non-degenerate six-qubit CSS entanglement-assisted code. This code uses one bit of entanglement (an ebit) shared between the sender and the receiver and corrects an arbitrary single-qubit error. The code we obtain is globally equivalent to the Steane seven-qubit code and thus corrects an arbitrary error on the receiver's half of the ebit as well. We prove that this code is the smallest code with a CSS structure that uses only one ebit and corrects an arbitrary single-qubit error on the sender's side. We discuss the advantages and disadvantages for each of the two codes.

PACS numbers: 03.67.-a, 03.67.Hk, 03.67.Pp
Keywords: quantum error correction, stabilizer formalism, entanglement-assisted quantum error correction, CSS, fault-tolerance

I. INTRODUCTION

It has been more than a decade since Peter Shor's seminal paper on quantum error correction [1]. He showed how to protect one qubit against decoherence by encoding it into a subspace of a Hilbert space larger than its own. For the first time, it was possible to think about quantum computation from a practical standpoint.

Calderbank and Shor then provided asymptotic rates for the existence of quantum error-correcting codes and gave upper bounds for such rates [2]. They defined a quantum error-correcting code as an isometric map that encodes k qubits into a subspace of the Hilbert space of n qubits. As long as only t or fewer qubits in the encoded state undergo errors, we can decode the state correctly. The notation for describing such codes is $[[n, k, d]]$, where d represents the distance of the code, and the code encodes k logical qubits into n physical qubits.

These earlier codes are examples of additive or stabilizer codes. Additive codes encode quantum information into the $+1$ eigenstates of n-fold tensor products of Pauli operators [3, 4]. Gottesman developed an elegant theory, the stabilizer formalism, that describes error correction, detection, and recovery in terms of algebraic group theory.

Steane constructed a seven-qubit code that encodes one qubit, corrects an arbitrary single-qubit error, and is an example of a Calderbank-Shor-Steane (CSS) code [5]. The five-qubit quantum error-correcting code is a "perfect code" in the sense that it encodes one qubit with the smallest number of physical qubits while still correcting an arbitrary single-qubit error [6, 7].

Even though every stabilizer code is useful for fault-tolerant computation [3, 4], CSS codes allow for simpler fault-tolerant procedures. For example, an encoded CNOT gate admits a transversal implementation without the use of ancillas if and only if the code is of the CSS type [4]. The five-qubit code is not a CSS code and does not possess the simple fault-tolerant properties of CSS codes [8]. The Steane code is a CSS code and is well-suited for fault-tolerant computation because it has bitwise implementations of the Hadamard and the phase gates as well (the logical X and Z operators have bitwise implementations for any stabilizer code [3]). However, an experimental realization of the seven-qubit code may be more difficult to achieve than one for the five-qubit code because it uses two additional physical qubits for encoding.

Calderbank *et al.* discovered two distinct six-qubit quantum codes [9] which encode one qubit and correct an arbitrary single-qubit error. They discovered the first of these codes by trivially extending the five-qubit code and the other one through an exhaustive search of the encoding space. Neither of these codes is a CSS code.

The five-qubit code and the Steane code have been studied extensively [8], but the possibility for encoding one qubit into six has not received much attention except for the brief mention in Ref. [9]. In the current paper,

*Electronic address: bilalsha@usc.edu

we bridge the gap between the five-qubit code and the Steane code by discussing two examples of a six-qubit code. The first code we present is a standard stabilizer code and the second is an entanglement-assisted code. We have not explicitly checked whether our first example is equivalent to the non-trivial code of Calderbank *et al.*, but we provide a logical argument in a subsequent paragraph to show that they are equivalent. We also present several proofs concerning the existence of single-error-correcting CSS codes of a certain size. One of our proofs gives insight into why Calderbank *et al.* were unable to find a six-qubit CSS code. The other proofs use a technique similar to the first proof to show the non-existence of a CSS entanglement-assisted code that uses fewer than six local physical qubits where one of the local qubits is half of one ebit, and corrects an arbitrary single-qubit error.

We structure our work according to our four main results. We first present a degenerate six-qubit quantum code and show how to convert this code to a subsystem code. Our second result is a proof for the non-existence of a single-error-correcting CSS six-qubit code. Our third result is the construction of a six-qubit CSS entanglement-assisted quantum code. This code is globally equivalent to the Steane code. We finally show that the latter is the smallest example of an entanglement-assisted CSS code that corrects an arbitrary single-qubit error.

In Section II, we present a degenerate six-qubit quantum error-correcting code that corrects an arbitrary single-qubit error. We present the logical Pauli operators , CNOT and encoding circuit for this code. We also prove that a variation of this code gives us a non-trivial example of a subsystem code that saturates the subsystem Singleton bound [10].

In Section III, we present a proof that a single-error-correcting CSS six-qubit code does not exist. Our proof enumerates all possible CSS forms for the five stabilizer generators of the six-qubit code and shows that none of these forms corrects the set of all single-qubit errors.

Section IV describes the construction of a six-qubit non-degenerate entanglement-assisted CSS code and presents its stabilizer generators, encoding circuit, and logical Pauli operators. This code encodes one logical qubit into six local physical qubits. One of the physical qubits used for encoding is half of an ebit that the sender shares with the receiver. The six-qubit entanglement-assisted code is globally equivalent to the seven-qubit Steane code [5] and thus corrects an arbitrary single-qubit error on all of the qubits (including the receiver's half of the ebit). This ability to correct errors on the receiver's qubits in addition to the sender's qubits is not the usual case with codes in the entanglement-assisted paradigm, a model that assumes the receiver's halves of the ebits are noise free because they are already on the receiving end of the channel. We show that our example is a trivial case of a more general rule—every $[[n, 1, 3]]$ code is equivalent to a $[[n-1, 1, 3; 1]]$ entanglement-assisted code by using any qubit as Bob's half of the ebit.

h_1	Y	I	Z	X	X	Y
h_2	Z	X	I	I	X	Z
h_3	I	Z	X	X	X	X
h_4	I	I	I	Z	I	Z
h_5	Z	Z	Z	I	Z	I
$\overline{X}$	Z	I	X	I	X	I
$\overline{Z}$	I	Z	I	I	Z	Z

TABLE I: Stabilizer generators $h_1, \ldots, h_5$, and logical operators $\overline{X}$ and $\overline{Z}$ for the six-qubit code. The convention in the above generators is that $Y = ZX$.

Finally, in section V, we present a proof that the Steane code is an example of the smallest entanglement-assisted code that corrects an arbitrary single-qubit error on the sender's qubits, uses only one ebit, and possesses the CSS form.

The appendix gives a procedure to obtain the encoding circuit for the six-qubit CSS entanglement-assisted code. It also lists a table detailing the error-correcting properties for the degenerate six-qubit code.

II. DEGENERATE SIX-QUBIT QUANTUM CODE

This section details an example of a six-qubit code that corrects an arbitrary single-qubit error. We explicitly present the stabilizer generators, encoding circuit, logical codewords, logical Pauli operators and CNOT operator for this code. We also show how to convert this code into a subsystem code where one of the qubits is a gauge qubit. We finish this section by discussing the advantages and disadvantages of this code.

Calderbank *et al.* mention the existence of two non-equivalent six-qubit codes [9]. Their first example is a trivial extension of the five-qubit code. They append an ancilla qubit to the five-qubit code to obtain this code. Their second example is a non-trivial six-qubit code. They argue that there are no other codes "up to equivalence." Our example is not reducible to the trivial six-qubit code because every one of its qubits is entangled with the others. It therefore is equivalent to the second non-trivial six-qubit code in Ref. [9] according to the arguments of Calderbank *et al.*

Five generators specify the degenerate six-qubit code. Table I lists the generators $h_1, \ldots, h_5$ in the stabilizer $\mathcal{S}$, and the logical operators $\overline{X}$ and $\overline{Z}$ for the six-qubit code. Figure 1 illustrates an encoding circuit for the six-qubit code. The encoding circuit is not fault tolerant, but one can consult Refs. [3, 4] to determine fault-tolerant procedures for arbitrary stabilizer codes.

The quantum error-correcting conditions guarantee that the six-qubit code corrects an arbitrary single-qubit error [8]. Specifically, the error-correcting conditions

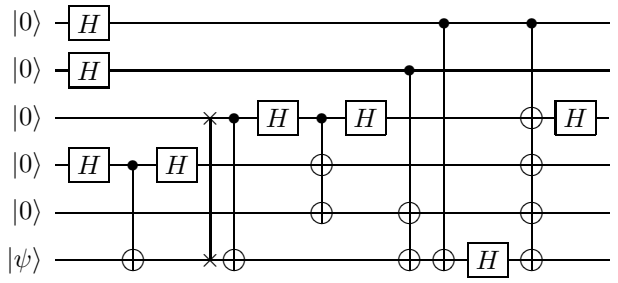

FIG. 1: Encoding circuit for the first six-qubit code. The H gate is a Hadamard gate. For example, we apply a Hadamard on qubit four followed by a CNOT with qubit four as the control qubit and qubit six as the target qubit.

are as follows: a stabilizer $\mathcal{S}$ with generators s_i where $i = 1, \ldots, n-k$ (in our case $n = 6$ and $k = 1$), corrects an error set $\mathcal{E}$ if every error pair $E_a^\dagger E_b \in \mathcal{E}$ either anticommutes with at least one stabilizer generator

$$\exists\ s_i \in \mathcal{S} : \left\{ s_i, E_a^\dagger E_b \right\} = 0, \tag{1}$$

or is in the stabilizer,

$$E_a^\dagger E_b \in \mathcal{S}. \tag{2}$$

These conditions imply the ability to correct any linear combination of errors in the set $\mathcal{E}$ [8, 11]. At least one generator from the six-qubit stabilizer anticommutes with each of the single-qubit Pauli errors, X_i, Y_i, Z_i where $i = 1, \ldots, 6$, because the generators have at least one Z and one X operator in all six positions. Additionally, at least one generator from the stabilizer anticommutes with each pair of two distinct Pauli errors (except $Z_4 Z_6$, which is in the stabilizer $\mathcal{S}$). Table IV in the appendix lists such a generator for every pair of distinct Pauli errors for the six-qubit code. These arguments and the table listings prove that the code can correct an arbitrary single-qubit error.

The logical basis states for the six-qubit code are as follows:

$$|\overline{0}\rangle = \begin{array}{c} |000000\rangle - |100111\rangle + |001111\rangle - |101000\rangle - \\ |010010\rangle + |110101\rangle + |011101\rangle - |111010\rangle \end{array},$$

$$|\overline{1}\rangle = \begin{array}{c} |001010\rangle + |101101\rangle + |000101\rangle + |100010\rangle - \\ |011000\rangle - |111111\rangle + |010111\rangle + |110000\rangle \end{array},$$

where we suppress the normalization factors of the above codewords.

A series of CNOT and controlled-Z operations implement the logical CNOT operation for the six-qubit code. Let $\mathrm{CN}(i, j)$ denote a CNOT acting on physical qubits i and j with qubit i as the control and qubit j as the target. Let $\mathrm{CZ}(i, j)$ denote controlled-Z operations. The logical CNOT for the six-qubit code is as follows:

$$\overline{\mathrm{CNOT}} = \mathrm{CZ}\,(2,7)\ \ \mathrm{CZ}\,(5,7)\ \ \mathrm{CZ}\,(6,7)\ \ \mathrm{CN}\,(1,9)$$
$$\mathrm{CN}\,(3,9)\ \ \mathrm{CN}\,(4,9)\ \ \mathrm{CN}\,(2,11)\ \ \mathrm{CN}\,(4,11)\ \ \mathrm{CN}\,(5,11)$$

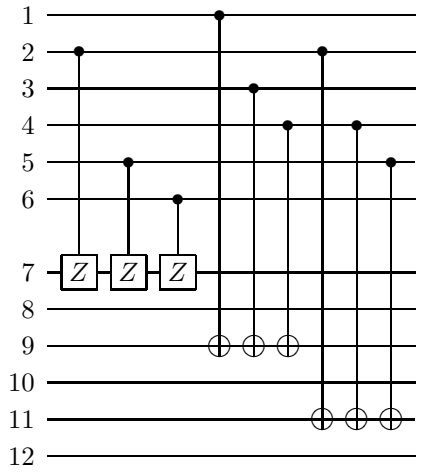

FIG. 2: Logical CNOT for the six-qubit quantum code. The first six qubits represent a logical source qubit and the last six represent a logical target qubit. For example we begin the circuit by applying a CZ (controlled-Z) gate from source qubit two to target qubit seven.

Figure 2 depicts the logical CNOT acting on two logical qubits encoded with the six-qubit code.

Both the six-qubit code and the five-qubit code correct an arbitrary single-qubit error. But the six-qubit code has the advantage that it corrects a larger set of errors than the five-qubit code. This error-correcting capability comes at the expense of a larger number of qubits—it corrects a larger set of errors because the Hilbert space for encoding is larger than that for the five-qubit code. In comparison to the Steane code, the six-qubit code uses a smaller number of qubits, but the disadvantage is that it does not admit a simple transversal implementation of the logical CNOT. In addition, the Steane code admits a bitwise implementation of all logical single-qubit Clifford gates whereas the six-qubit code does not.

A. Subsystem Code Construction

We convert the degenerate six-qubit code from the previous section into a subsystem code. The degeneracy inherent in the code allows us to perform this conversion. The code still corrects an arbitrary single-qubit error after we replace one of the unencoded ancilla qubits with a gauge qubit.

We briefly review the history of subsystem codes. The essential insight of Knill *et al.* was that the most general way to encode quantum information is into a subsystem rather than a subspace [12]. In the case when the information is encoded in a single subsystem, the Hilbert space decomposes as $\mathcal{H} = (\mathcal{H}_A \otimes \mathcal{H}_B) \oplus \mathcal{H}_C$ where the subsystem $\mathcal{H}_A$ stores the protected information. Errors that act on subsystem $\mathcal{H}_B$, also known as the gauge subsystem, do not require active correction be-

h_1	Y	I	Z	X	X	Y
h_2	Z	X	I	I	X	Z
h_3	I	Z	X	X	X	X
h_5	Z	Z	Z	I	Z	I
H_X	I	I	I	X	I	I
H_Z	I	I	I	Z	I	Z
$\overline{X}$	Z	I	X	I	X	I
$\overline{Z}$	I	Z	I	I	Z	Z

TABLE II: Stabilizer generators h_1, h_2, h_3 and h_5, gauge subgroup generators H_X and H_Z, and logical operators $\overline{X}$ and $\overline{Z}$ for the six-qubit code. The convention in the above generators is that $Y = ZX$.

cause $\mathcal{H}_B$ does not store any valuable information. This passive error-correction ability of a subsystem code may lead to a smaller number of stabilizer measurements during the recovery process and may lead to an improvement of the accuracy threshold for quantum computation [13]. Kribs *et al.* recognized in Ref. [14] that this subsystem structure of a Hilbert space is useful for active quantum error-correction as well (Knill *et al.* did not explicitly recognize this ability in Ref. [12].) See Ref. [15] for a discussion of all aspects of subsystem code constructions and a detailed theoretical comparison between subsystem and stabilizer codes.

We now detail how to convert the six-qubit code from the previous section into a subsystem code. The sixth unencoded qubit is the information qubit and the encoding operation transforms it into subsystem $\mathcal{H}_A$. We convert the fourth unencoded ancilla qubit to a gauge qubit. We simply consider it as a noisy qubit so that the operators X_4 and Z_4 have no effect on the quantum information stored in subsystem $\mathcal{H}_A$. The operators X_4 and Z_4 generate the unencoded gauge group. The encoding circuit in Figure 1 transforms these unencoded operators into X_4 and Z_4Z_6 respectively. These operators together generate the encoded gauge subgroup $H = \langle X_4, Z_4Z_6\rangle$. Errors in this subgroup do not affect the encoded quantum information. The code is still able to correct an arbitrary single-qubit error because each one of the single-qubit Pauli error pairs anticommutes with at least one of the generators from the new stabilizer $\widetilde{\mathcal{S}} = \langle h_1, h_2, h_3, h_5\rangle$, or belong to H [16]. Table IV shows this property for all error pairs. The code passively corrects the error pairs X_4, Z_4Z_6, Y_4Z_6 because they belong to the gauge subgroup.

The six-qubit single-error-correcting subsystem code discussed above saturates the Singleton bound for subsystem codes [10],

$$n - k - r \geq 2(d-1), \tag{3}$$

where for our code, $n = 6$, $k = 1$, $r = 1$, and $d = 3$. This code is the smallest non-trivial subsystem code that corrects an arbitrary single-qubit error and is a code that satisfies the conjecture at the end of Ref. [17]. A trivial way to saturate this bound is to add a noisy qubit to the five-qubit code! One of the advantages of using the subsystem construction is that we only need to perform four stabilizer measurements instead of five during the recovery process.

III. NON-EXISTENCE OF A $[[6, 1, 3]]$ CSS CODE

Our proposition below proves that it is impossible for a six-qubit code to possess the CSS structure while correcting an arbitrary single-qubit error. An immediate corollary of this proposition is that the seven-qubit code is the smallest single-error-correcting CSS code.

Proposition. *There is no six-qubit code that encodes one qubit, possesses the CSS structure, and corrects an arbitrary single-qubit error.*

Proof. We first suppose that a code with the above properties exists. If a $[[6, 1, 3]]$ CSS code exists, its stabilizer $\mathcal{S}$ must have five generators:

$$\mathcal{S} = \langle g_1, ..., g_5\rangle. \tag{4}$$

The CSS structure implies that each of these generators includes X operators only or Z operators only (except for the identity). The set of correctable Pauli errors $\{E_j\}$ in the Pauli group acting on six qubits satisfies $\{E_iE_j, \mathcal{S}\} = 0$ unless $E_iE_j \in \mathcal{S}$, for all i, j. We show below that no set of five CSS stabilizer generators acting on six qubits can correct an arbitrary single-qubit error and possess the CSS structure.

First assume that such generators exist. It is not possible that all generators consist of the same type of operators (all X or all Z) because single-qubit errors of the same type (X or Z) are then not correctable. Consider the possibility that there is one generator of one type, say X, and four generators of the other type, say Z. If the generator of type X has an identity acting on any qubit, say the first one, then the error Z_1 commutes with all generators. This error is not correctable unless it belongs to the stabilizer. But if it belongs to the stabilizer, the first qubit of the code must be fixed in the state $|0\rangle$, which makes for a trivial code. The other possibility is that the X-type generator has the form $g_1 = XXXXXX$. But then any combination of two Z-errors (Z_iZ_j) commutes with it, and so they have to belong to the stabilizer. But there are five independent such combinations of errors (Z_1Z_2, Z_1Z_3, Z_1Z_4, Z_1Z_5, Z_1Z_6) and only four generators of the Z type. Therefore, it is impossible for the code to have four generators of one type and one generator of the other type.

The only possibility left is that there are two generators of one type, say X, and three generators of the other type, say Z. The two X-type generators should not both have identity acting on any given qubit because a Z error on that qubit commutes with all generators. Such an error cannot belong to the stabilizer because it would again make for a trivial code. Specifically, we write the two

X-type generators (g_1 and g_2) one above the other

$$\begin{array}{cc} g_1 = & - \ - \ - \ - \ - \ - \\ g_2 & - \ - \ - \ - \ - \ - \end{array}, \tag{5}$$

where we leave the entries unspecified in the above equation, but they are either X or I. Both generators cannot have the column

$$\begin{array}{c} I \\ I \end{array}$$

in (5) because both generators cannot have identities acting on the same qubit. Thus, only three different columns can build up the generators in (5):

$$\begin{array}{ccccc} I & & X & & X \\ X & , & I & , & X \end{array}$$

We distinguish the following cases:

1. Each column appears twice.
2. One column appears three times, another column appears twice, and the third column appears once.
3. One column appears three times and another column appears three times.
4. At least one column appears more than three times.

If one and the same column appears on two different places, say qubit one and qubit two as in the following example,

$$\begin{array}{cc} g_1 = & X \ X \ - \ - \ - \ - \\ g_2 & I \ I \ - \ - \ - \ - \end{array}, \tag{6}$$

then a pair of Z errors on these qubits (Z_1Z_2) commutes with all generators, and therefore belongs to the stabilizer.

In the first case considered above, there are three such pairs of errors, which up to a relabeling of the qubits can be taken to be Z_1Z_2, Z_3Z_4, Z_5Z_6. They can be used as stabilizer generators because these operators are independent. But then the following pairs of single-qubit X errors commute with all generators: X_1X_2, X_3X_4, X_5X_6. This possibility is ruled out because the latter cannot be part of the stabilizer generators.

In the second case, up to a relabeling of the qubits, we have the following pairs of Z errors that commute with the stabilizer: Z_1Z_2, Z_1Z_3, Z_2Z_3, Z_4Z_5. Only three of all four are independent, and they can be taken to be stabilizer generators. But then all three generators of Z-type have the identity acting on the sixth qubit, and therefore the error X_6 is not correctable (and it cannot be a stabilizer generator because it would make for a trivial code).

In the third case, the pairs Z_1Z_2, Z_1Z_3, Z_2Z_3, Z_4Z_5, Z_4Z_6, Z_5Z_6 (up to a relabeling), four of which are independent, commute with the stabilizer. But they cannot all belong to the stabilizer because there are only three possible generators of the Z-type.

Finally, in the fourth case, we have three or more independent pairs of Z errors commuting with the stabilizer (for example Z_1Z_2, Z_1Z_3, Z_1Z_4, which corresponds to the first four columns being identical). If the independent pairs are more than three, then their number is more than the possible number of generators. If they are exactly three, we can take them as generators. But then Z-type generators act trivially upon two qubits, and therefore X errors on these qubits are not correctable. This last step completes the proof. ■

IV. NON-DEGENERATE SIX-QUBIT CSS ENTANGLEMENT-ASSISTED QUANTUM CODE

We detail the construction of a six-qubit CSS entanglement-assisted quantum code in this section. We first review the history of entanglement-assisted quantum coding and discuss the operation of an entanglement-assisted code. We then describe our construction. It turns out that the code we obtain is equivalent to the Steane code [5] when including Bob's qubit, and therefore is not a new code. It suggests, however, a general rule for which we present a proof—every $[[n, 1, 3]]$ code is equivalent to a $[[n-1, 1, 3; 1]]$ entanglement-assisted code with any qubit serving as Bob's half of the ebit. Even though our code is a trivial example of this rule, it is instructive to present its derivation from the perspective of the theory of entanglement-assisted codes.

Bowen constructed an example of a quantum error-correcting code that exploits shared entanglement between sender and receiver [18]. Brun, Devetak, and Hsieh later generalized Bowen's example and developed the entanglement-assisted stabilizer formalism [19, 20]. This theory is an extension of the standard stabilizer formalism and uses shared entanglement to formulate stabilizer codes. Several references provide a review [19, 20, 21] and generalizations of the theory to entanglement-assisted operator codes [21, 22], convolutional entanglement distillation protocols [23], continuous-variable codes [24], and entanglement-assisted quantum convolutional codes [25, 26]. Gilbert *et al.* also generalized their "quantum computer condition" for fault tolerance to the entanglement-assisted case [27]. Entanglement-assisted codes are a special case of "correlation-assisted codes", where Bob's qubit is also allowed to be noisy. Such codes are in turn instances of general linear quantum error-correcting codes [28].

An entanglement-assisted quantum error-correcting code operates as follows. A sender and receiver share c ebits before communication takes place. The sender possesses her half of the c ebits, $n-k-c$ ancilla qubits, and k information qubits. She performs an encoding unitary on her n qubits and sends them over a noisy quantum communication channel. The receiver combines his half of the c ebits with the n encoded qubits and performs

measurements on all of the qubits to diagnose the errors from the noisy channel. The generators corresponding to the measurements on all of the qubits form a commuting set. The generators thus form a valid stabilizer, they do not disturb the encoded quantum information, and they learn only about the errors from the noisy channel. The notation for such a code is $[[n, k, d; c]]$, where d is the distance of the code.

The typical assumption for an entanglement-assisted quantum code is that noise does not affect Bob's half of the ebits because they reside on the other side of a noisy quantum communication channel between Alice and Bob. Our $[[6, 1, 3; 1]]$ entanglement-assisted code is globally equivalent to the $[[7, 1, 3]]$ Steane code and thus corrects errors on Bob's side as well. From a computational perspective, a code that corrects errors on all qubits is more powerful than a code that does not. From the perspective of the entanglement-assisted paradigm, however, this feature is unnecessary and may result in decreased error-correcting capabilities of the code with respect to errors on Alice's side.

We construct our code using the parity check matrix of a classical code. Consider the parity check matrix for the $[7, 4, 3]$ Hamming code:

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} \tag{7}$$

The Hamming code encodes four classical bits and corrects a single-bit error. We remove one column of the above parity check matrix to form a new parity check matrix H as follows:

$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}. \tag{8}$$

The code corresponding to H encodes three bits and still corrects a single-bit error. We begin constructing the stabilizer for an entanglement-assisted quantum code by using the CSS construction [21, 22]:

$$\left[\begin{array}{c|c} H & 0 \\ 0 & H \end{array}\right]. \tag{9}$$

The left side of the above matrix is the "Z" side and the right side of the above matrix is the "X" side. The isomorphism between n-fold tensor products of Pauli matrices and n-dimensional binary vectors gives a correspondence between the matrix in (9) and the set of Pauli generators below [4, 8, 19]:

$$\begin{array}{cccccc} Z & I & I & Z & I & Z \\ I & Z & I & Z & Z & I \\ I & I & Z & I & Z & Z \\ X & I & I & X & I & X \\ I & X & I & X & X & I \\ I & I & X & I & X & X \end{array} \tag{10}$$

	Bob	Alice				
g_1'	I	I	Z	I	I	I	I
g_2'	I	I	I	Z	I	I	I
g_3'	Z	Z	I	I	I	I	I
g_4'	I	I	I	I	Z	I	I
g_5'	I	I	I	I	I	Z	I
g_6'	X	X	I	I	I	I	I
$\overline{X}'$	I	I	I	I	I	I	X
$\overline{Z}'$	I	I	I	I	I	I	Z

(a)

	Bob	Alice				
g_1	I	Z	I	Z	Z	Z	I
g_2	I	Z	Z	I	I	Z	Z
g_3	Z	Z	I	I	Z	I	Z
g_4	I	X	X	I	I	X	X
g_5	I	I	X	X	X	I	X
g_6	X	X	I	I	X	I	X
$\overline{X}$	I	I	I	I	X	X	X
$\overline{Z}$	I	I	Z	Z	I	Z	I

(b)

TABLE III: (a) The generators and logical operators for the unencoded state. Generators g_3' and g_6' indicate that Alice and Bob share an ebit. Alice's half of the ebit is her first qubit and Bob's qubit is the other half of the ebit. Generators g_1', g_2', g_4', and g_5' indicate that Alice's second, third, fourth, and fifth respective qubits are ancilla qubits in the state $|0\rangle$. The unencoded logical operators $\overline{X}'$ and $\overline{Z}'$ act on the sixth qubit and indicate that the sixth qubit is the information qubit. (b) The encoded generators and logical operators for the $[[6, 1, 3; 1]]$ entanglement-assisted quantum error-correcting code.

The above set of generators have good quantum error-correcting properties because they correct an arbitrary single-qubit error. These properties follow directly from the properties of the classical code. The problem with the above generators is that they do not form a commuting set and thus do not correspond to a valid quantum code. We use entanglement to resolve this problem by employing the method outlined in Ref. [19, 20, 21].

Three different but related methods determine the minimum number of ebits that the entanglement-assisted quantum code requires:

1. Multiplication of the above generators with one another according to the "symplectic Gram-Schmidt orthogonalization algorithm" forms a new set of generators [19, 20]. The error-correcting properties of the code are invariant under these multiplications because the code is an additive code. The resulting code has equivalent error-correcting properties and uses the minimum number of ebits. We employ this technique in this work.

2. A slightly different algorithm in the appendix of Ref. [23] determines the minimum number of ebits required, the stabilizer measurements to perform, and the local encoding unitary that Alice performs to rotate the unencoded state to the encoded state. This algorithm is the most useful because it "kills three birds with one stone."

3. The minimum number of ebits for a CSS entanglement-assisted code is equal to the rank of HH^T [21, 22, 29]. This simple formula is useful if we are only concerned with computing the minimum number of ebits. It does not determine the

stabilizer generators or the encoding circuit. Our code requires one ebit to form a valid stabilizer code because the rank of HH^T for our code is equal to one.

Table III(b) gives the final form of the stabilizer for our entanglement-assisted six-qubit code. We list both the unencoded and the encoded generators for this code in Table III.

Our code uses one ebit shared between sender and receiver in the encoding process. The sender performs a local encoding unitary that encodes one qubit with the help of four ancilla qubits and one ebit.

The symplectic Gram-Schmidt algorithm yields a symplectic matrix that rotates the unencoded symplectic vectors to the encoded symplectic vectors. The symplectic matrix corresponds to an encoding unitary acting on the unencoded quantum state [19, 20]. This correspondence results from the Stone-von Neumann Theorem and unifies the Schrödinger and Heisenberg pictures for quantum error correction [30].

The symplectic Gram-Schmidt algorithm also determines the logical operators for the code. Some of the vectors in the symplectic matrix that do not correspond to a stabilizer generator are equivalent to the logical operators for the code. It is straightforward to determine which symplectic vector corresponds to which logical operator (X or Z) by observing the action of the symplectic matrix on vectors that correspond to the unencoded X or Z logical operators.

For our code, the symplectic matrix is as follows:

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&1&0&1&0&0&0&0&0&0\\ 1&0&1&1&1&0&0&0&0&0&0&0\\ 1&1&0&0&1&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&1&0&0&1&1\\ 0&0&0&0&0&0&0&1&1&1&0&1\\ 0&1&1&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&1&0&1\\ 0&0&0&0&0&0&0&0&1&1&1&1\\ 0&0&0&0&0&0&0&0&1&0&1&0\\ 0&0&0&1&0&1&0&0&0&0&0&0\\ 0&1&0&1&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&1&1 \end{array}\right] \tag{11}$$

The index of the rows of the above matrix corresponds to the operators in the unencoded stabilizer in Table III(a). Therefore, the first five rows correspond to the encoded Z operators in the stabilizer and the sixth row corresponds to the logical $\overline{Z}$ operator. As an example, we can represent the unencoded logical $\overline{Z}$ operator in Table III(a) as the following binary vector:

$$\left[\begin{array}{cccccc|cccccc} 0&0&0&0&0&1&0&0&0&0&0&0 \end{array}\right]. \tag{12}$$

Premultiplying the above matrix by the above row vector gives the binary form for the encoded logical $\overline{Z}$ operator.

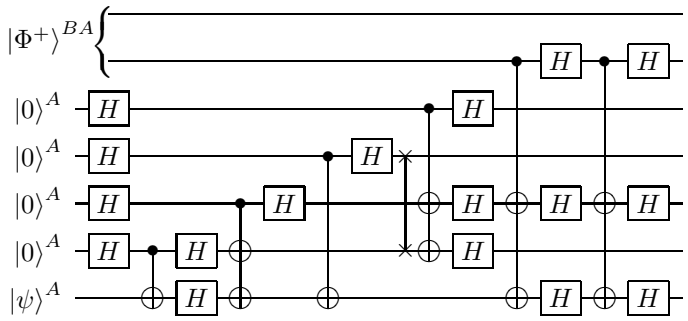

FIG. 3: Encoding circuit for the [[6,1,3;1]] code. The "H" gate is a Hadamard gate.

We can then translate this binary vector to a six-fold tensor product of Paulis equivalent to the logical $\overline{Z}$ operator in Table III(b). Using this same idea, the first row of the above matrix corresponds to Alice's Paulis in g_3, the second row to g_1, the third row to g_2, the fourth row to g_4, the fifth row to g_5, and the seventh row to g_6. The last six rows in the above matrix correspond to encoded X operators and it is only the last row that is interesting because it acts as a logical X operator.

Figure 3 gives the encoding circuit for the code.

We now detail the operations that give the equivalence of this code to the seven-qubit Steane code. Consider the generators in Table III(b). Label the columns from left to right as 1, 2, ..., 7 where "1" corresponds to Bob's column. Replace the generator g_1 by $g_1g_2g_3$, and the generator g_5 by g_5g_6. Switch the new generators g_4 and g_5. Switch columns 2 and 3. Switch columns 1 and 5. Cyclically permute the columns once so that 1 becomes 7, 2 becomes 1, 3 becomes 2, ..., 7 becomes 6. The resulting code is exactly the Steane code if one reads it from right to left (i.e., up to the permutation $1 \leftrightarrow 7$, $2 \leftrightarrow 6$, $3 \leftrightarrow 5$).

Inspection of the encoded logical operators in Table III(b) reveals that Alice can perform logical $\overline{X}$ and $\overline{Z}$ operations locally. Since the CNOT has a transversal implementation for the Steane code, if Alice and Bob possess two logical qubits each encoded with this entanglement-assisted code, they can apply an encoded CNOT transversally by the use of classical communication to coordinate their actions. We point out, however, that the idea of computation in the entanglement-assisted paradigm is not well motivated, since if classical communication is allowed, Alice could send the initial state to Bob and inform him of the operations that need to be applied. An interesting open question is if there exist codes that allow fault-tolerant computation on Alice's side only.

From this example, we observe that some subset of the entanglement-assisted codes correct errors on Bob's side. This phenomenon can be understood as an instance of the more general correlation-assisted codes and linear quantum error-correction theory detailed in Ref. [28]. It may be useful from a practical standpoint to determine

which entanglement-assisted codes satisfy this property. Here we provide an answer for the case of single-error-correcting codes that use one bit of entanglement.

Proposition. *Every* $[[n,1,3]]$ *code is equivalent to a* $[[n-1,1,3;1]]$ *code with any qubit serving as Bob's half of the ebit.*

Proof. We prove this proposition by showing that any column in the table of stabilizer generators for such a code can be reduced to the standard form of Bob's column in an entanglement-assisted code (containing exactly one X and one Z operator). Without loss of generality, consider the column corresponding to the first qubit. This column generally may contain X, Y, Z, or I operators, but if the code corrects any error on the first qubit, there must be at least two different Pauli operators in this column. We can reduce this column to the desired form as follows. Choose one of the generators that contains X on the first qubit, and replace each of the other generators that contain an X there by its product with the chosen generator. Do the same for Y and Z. Thus we are left with at most one generator with X, one with Y and one with Z. To eliminate Y, we replace it by its product with the X and Z generators. If either X or Z is missing, we replace the Y generator with its product with the other non-trivial generator. ■

This result can be understood as a reflection of the fact that in a code that corrects arbitrary single-qubit errors, every qubit is maximally entangled with the rest and therefore can be thought of as part of an ebit. The latter can also be seen to follow from the property that every single-qubit error must send the code space to an orthogonal subspace.

Note that for the case of $[[n,1,3;c]]$ codes with $c>1$, the relation could be more complicated. If such a code corrects an arbitrary single-qubit error, it is equivalent to an $[[n+c,1,3]]$ code, but it is not obvious whether a $[[n+c,1,3]]$ code can be interpreted as a $[[n,1,3;c]]$ code because the type of entanglement that exists between c qubits and the rest n qubits may not be the same as that of c e-bits.

V. NON-EXISTENCE OF $[[n,1,3;1]]$ CSS CODES FOR $n \leq 5$

We now show that there does not exist a smaller entanglement-assisted CSS code that uses only one ebit and corrects an arbitrary single-qubit error on Alice's side. The proof is similar to that for the non-existence of a $[[6,1,3]]$ CSS code.

Proposition. *There does not exist an* $[[n,1,3;1]]$ *entanglement-assisted CSS code for* $n \leq 5$.

Proof. We being this proof by giving a dimensionality argument for the non-existence of quantum codes (CSS or non-CSS) with $n<4$. This can be easily seen as follows. Assume that the code is non-degenerate. There are $3n$ different single-qubit errors on Alice's side, which means that there must exist $3n+1$ orthogonal subspaces of dimension 2 inside the entire 2^{n+1}-dimensional Hilbert space, i.e., $(3n+1)2 \leq 2^{n+1}$. This is impossible for $n<4$. Since for $n \leq 3$ the number of generators is at most 3, and two of the generators have to act non-trivially on Bob's side, we can have degeneracy with respect to errors on Alice's side only for $n=3$ with exactly one of the generators being equal to a pair of errors on Alice's side. These two errors would be the only indistinguishable single-qubit errors on Alice's side (no other pair of errors on Alice's side can belong to the stabilizer), which reduces the number of required orthogonal subspaces from $3\times 3+1=10$ to 9. The required dimensions are $2\times 9=18$ and they cannot fit in the $2^4=16$-dimensional Hilbert space.

Suppose that there exists a $[[5,1,3;1]]$ CSS code. Its stabilizer must have 5 generators ($S=\langle g_1,...,g_5\rangle$), each consisting of only X and I operators or Z and I operators. For an entanglement-assisted code, the generators must be of the form

$$\begin{array}{l} g_1 = - - - - - \,|\, X \\ g_2 = - - - - - \,|\, Z \\ g_3 = - - - - - \,|\, I \\ g_4 = - - - - - \,|\, I \\ g_5 = - - - - - \,|\, I \end{array} \qquad (13)$$

where we have left the entries on Alice's side unspecified. The set of correctable Pauli errors on Alice's side $\{E_j \in \mathcal{P}_5\}$ (where $\mathcal{P}_5$ is the five-qubit Pauli group) must satisfy $\{E_iE_j, S\}=0$ unless $E_iE_j \in S$, for all $i,j=1,2,3,4,5$. All generators cannot be of the same type (X or Z). The possibility that there is one generator of one type, say X, and four generators of the other (Z) type, is also ruled out because the X-type generator would have to be of the form $g_1 = XXXXX|X$ in order that every qubit is acted upon non-trivially by at least one X operator from the stabilizer. This would mean, however, that any combination of two Z-errors (Z_iZ_j, $i,j=1,2,3,4,5$) would commute with the stabilizer, and so it would have to belong to the stabilizer. There are four independent such combinations of errors (Z_1Z_2,Z_1Z_3,Z_1Z_4,Z_1Z_5) which would have to be the other four generators. But then there would be no possibility for a Z operator on Bob's side (as in g_2). Therefore, this is impossible.

The only possibility is that there are 2 generators of one type, say X, and 3 generators of the other type (Z). The two X-type generators should not both have identity acting on any given qubit on Alice's side because a Z error on that qubit would commute with all generators. Consider the following form for the two X-type generators:

$$\begin{array}{l} g_1 = - - - - - \,|\, X \\ g_3 = - - - - - \,|\, I \end{array} \qquad (14)$$

There are three different columns that can fill the un-

specified entries in the above table:

$$\begin{matrix} I \\ X \end{matrix} \quad , \quad \begin{matrix} X \\ I \end{matrix} \quad , \quad \begin{matrix} X \\ X \end{matrix} \quad .$$

We distinguish the following cases: two columns appear twice and one column appears once, one column appears three times and another column appears twice, one column appears three times and each of the other columns appears once, at least one column appears more than three times.

In the first case, up to relabeling of the qubits, we distinguish the following possibilities:

$$\begin{aligned} g_1' &= I\ \ I\ \ X\ \ X\ \ X \big| X \\ g_3' &= X\ \ X\ \ I\ \ I\ \ X \big| I \end{aligned} \tag{15}$$

$$\begin{aligned} g_1'' &= X\ \ X\ \ I\ \ I\ \ X \big| X \\ g_3'' &= X\ \ X\ \ X\ \ X\ \ I \big| I \end{aligned} \tag{16}$$

$$\begin{aligned} g_1''' &= X\ \ X\ \ X\ \ X\ \ I \big| X \\ g_3''' &= X\ \ X\ \ I\ \ I\ \ X \big| I \end{aligned} \tag{17}$$

For each possibility, the pairs of errors Z_1Z_2 and Z_3Z_4 commute with the stabilizer and therefore they would have to be equal to the stabilizer generators g_4 and g_5. But the pairs of errors X_1X_2 and X_3X_4 would commute with g_1, g_3, g_4 and g_5. Since these errors do not belong to the stabilizer, they would have to anti-commute with g_3. Therefore, up to interchanging the first and second, or the third and fourth qubits, the generator g_2 must have the form

$$g_3 = Z\ I\ Z\ I\ Z \big| Z\,. \tag{18}$$

(Note that the fifth entry must be Z because there must be at least one generator that has a Z acting on that qubit.) But it can be verified that for each of the possibilities (15), (16) and (17), g_3 anti-commutes with one of the X-type generators. Therefore, the first case is impossible.

In the second case, one of the possible columns appears three times and another column appears twice, e.g.,

$$\begin{aligned} g_1 &= X\ \ X\ \ X\ \ X\ \ X \big| X \\ g_3 &= X\ \ X\ \ X\ \ I\ \ I \big| I \end{aligned} \tag{19}$$

In such a case we would have three independent pairs of Z errors (Z_1Z_2, Z_1Z_3 and Z_4Z_5) which commute with the stabilizer and therefore have to belong to it. But then there would be no possibility for a Z operator on Bob's side (the generator g_2). Therefore this case is impossible.

In the third case, one column appears three times and each other column appears once, as in

$$\begin{aligned} g_1 &= X\ \ X\ \ X\ \ X\ \ I \big| X \\ g_3 &= X\ \ X\ \ X\ \ I\ \ X \big| I \end{aligned} \tag{20}$$

In this case, the pairs of errors Z_1Z_2 and Z_1Z_3 commute with the stabilizer and must be equal to g_4 and g_5. But in order for the fourth and fifth qubits to be each acted upon by at least one Z operator from the stabilizer, the generator g_2 would have to be of the form

$$g_2 = -\ -\ -\ Z\ Z \big| Z \tag{21}$$

This means that the pair of errors X_4X_5 commutes with the stabilizer, and since it is not part of the stabilizer, this case is also impossible.

Finally, if one column appears more than three times, there would be at least three independent pairs of Z errors on Alice's side which have to belong to the stabilizer. This leaves no possibility for a Z operator on Bob's side, i.e., this case is also ruled out. Therefore, a $[[5, 1, 3; 1]]$ CSS code does not exist.

In a similar way we can show that a $[[4, 1, 3; 1]]$ CSS code does not exist. Such a code would have 4 generators of the form

$$\begin{aligned} g_1 &= -\ -\ -\ - \big| X \\ g_2 &= -\ -\ -\ - \big| Z \\ g_3 &= -\ -\ -\ - \big| I \\ g_4 &= -\ -\ -\ - \big| I \end{aligned} \tag{22}$$

The possibilities that all of the generators are of the same type, or that one generator is of one type and the other three are of the other type, are readily ruled out by arguments similar to those for the $[[5, 1, 3; 1]]$ code. The only possibility is two X-type generators and two Z-type generators. The table of the X-type generators

$$\begin{aligned} g_1 &= -\ -\ -\ - \big| X \\ g_3 &= -\ -\ -\ - \big| I \end{aligned} \tag{23}$$

has to be filled by the same three columns we discussed before. As we saw in our previous arguments, in the case when one column appears three or more times there are at least two independent pairs of errors on Alice's side which commute with the stabilizer. These errors would have to belong to the stabilizer, but this leaves no possibility for a Z operator on Bob's side. In the case when one column appears twice and another column appears twice, the situation is analogous. The only other case is when one column appears twice and each of the other two columns appears once, as in

$$\begin{aligned} g_1 &= X\ \ X\ \ I\ \ X \big| X \\ g_3 &= X\ \ X\ \ X\ \ I \big| I \end{aligned} \tag{24}$$

Since in this case the pair of errors Z_1Z_2 would commute with the stabilizer, this pair would have to be equal to the generator g_4. The third and fourth qubits each have to be acted upon by at least one Z operators from the stabilizer. Thus the generator g_2 would have to have the form

$$g_2 = -\ -\ Z\ Z \big| Z\,. \tag{25}$$

But then the pair X_3X_4 which does not belong to the stabilizer would commute with all stabilizer generators. Therefore a $[[4, 1, 3; 1]]$ CSS code does not exist. ■

We point out that a $[[4, 1, 3; 1]]$ non-CSS code was found in Ref. [20]. This is the smallest possible code that can encode one qubit with the use of only one ebit, and at the same time correct an arbitrary single-qubit error on Alice's side. Here we have identified an example of the smallest possible CSS code with these characteristics.

VI. SUMMARY AND CONCLUSION

We have discussed two different examples of a six-qubit code and have included a subsystem construction for the degenerate six-qubit code. Our proof explains why a six-qubit CSS code does not exist and clarifies earlier results in Ref. [9] based on a search algorithm. An immediate corollary of our result is that the seven-qubit Steane code is the smallest CSS code capable of correcting an arbitrary single-qubit error. An interesting open problem is to generalize this tight lower bound to the setting of CSS codes with a higher distance. We expect that our proof technique may be useful for this purpose.

Our first example is a degenerate six-qubit code that corrects an arbitrary single-qubit error. The presentation of the encoding circuit and the operations required for a logical X, Z, and CNOT should aid in the implementation and operation of this code. We have converted this code into a subsystem code that is non-trivial and saturates the subsystem Singleton bound. Our six-qubit subsystem code requires only four stabilizer measurements during the recovery process.

Our second example is an entanglement-assisted $[[6, 1, 3; 1]]$ CSS code that is globally equivalent to the Steane seven-qubit code. We have presented the construction of this code from a set of six non-commuting generators on six qubits. We have further shown that every $[[n, 1, 3]]$ code can be used as a $[[n - 1, 1, 3; 1]]$ entanglement-assisted code.

Based on the proof technique that we used for the earlier six-qubit code, we have shown that the Steane code is an example of the smallest entanglement-assisted code that possesses the CSS structure and uses exactly one ebit. Here too, an interesting open problem is the generalization of this tight lower bound to higher distance entanglement-assisted codes or to codes that use more than one ebit.

VII. ACKNOWLEDGMENTS

B.A.S. acknowledges support from NSF grant No. CCF-0448658, M.M.W. from NSF grant No. CCF-0545845, and O.O. from NSF Grant No. CCF-0524822. D.A.L. was sponsored by NSF under grants CCF-0523675 and CCF-0726439, and by the United States Department of Defense. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of the U.S. Government. The authors thank Todd Brun for useful discussions.

VIII. APPENDIX

A. Tables

The tables in the appendix detail the error-correcting properties of both of the $[[6, 1, 3]]$ codes. Each table lists all possible pairs of single-qubit errors and a corresponding generator of the code that anticommutes with the pair.

B. Entanglement-Assisted Encoding Circuit

Here we detail an algorithm that generates the encoding circuit for the $[[6, 1, 3; 1]]$ code. We follow the recipe outlined in the appendix of Ref. [23]. We begin by first converting the stabilizer generators in Table III(b) into a binary form which we refer to as a $Z|X$ matrix. We obtain the the left Z submatrix by inserting a "1" wherever we see a Z in the stabilizer generators. We obtain the X submatrix by inserting a "1" wherever we see a corresponding X in the stabilizer generator. If there is a Y in the generator, we insert a "1" in the corresponding row and column of both the Z and X submatrices.

The idea is to convert (26) to (42) through a series of row and column operations. The binary form of the matrix in (26) corresponds to the stabilizer generators in Table III(b) by employing the Pauli-to-binary isomorphism outlined in Ref. [8]. We can use CNOT, Hadamard, Phase, and SWAP gates.

1. When we apply a CNOT gate from qubit i to qubit j, it adds column i to column j in the X submatrix, and in the Z submatrix it adds column j to column i.

2. A Hadamard on qubit i swaps column i in the Z submatrix with column i in the X submatrix.

3. A Phase gate on qubit i adds column i in the X submatrix to column i in the Z submatrix.

4. When we apply a SWAP gate from qubit i to qubit j, we exchange column i with column j in Z submatrix and column i and column j in the X submatrix.

Row operations do not change the error-correcting properties of the code. They do not cost us in terms of gates. They are also crucial in determining the minimum number of ebits for the code.

Error	AG	Error	AG	Error	AG	Error	AG	Error	AG	Error	AG	Error	AG	Error	AG	Error	AG	Error	AG	Error	AG
X_1X_2	h_1	X_1X_3	h_2	X_1X_4	h_1	X_1X_5	h_1	X_1X_6	h_5	X_1Y_2	h_1	X_1Y_3	h_2	X_1Y_4	h_2	X_1Y_5	h_3	X_1Y_6	h_1	X_1Z_2	h_1
X_1Z_3	h_1	X_1Z_4	h_2	X_1Z_5	h_3	X_1Z_6	h_2	X_2X_3	h_1	X_2X_4	h_3	X_2X_5	h_3	X_2X_6	h_1	X_2Y_3	h_1	X_2Y_4	h_1	X_2Y_5	h_1
X_2Y_6	h_2	X_2Z_3	h_5	X_2Z_4	h_1	X_2Z_5	h_1	X_2Z_6	h_1	X_3X_4	h_1	X_3X_5	h_1	X_3X_6	h_2	X_3Y_4	h_3	X_3Y_5	h_2	X_3Y_6	h_1
X_3Z_4	h_3	X_3Z_5	h_2	X_3Z_6	h_3	X_4X_5	h_5	X_4X_6	h_1	X_4Y_5	h_1	X_4Y_6	h_2	X_4Z_5	h_1	X_4Z_6	h_1	X_5X_6	h_1	X_5Y_6	h_2
X_5Z_6	h_1	Y_1X_2	h_2	Y_1X_3	h_1	Y_1X_4	h_2	Y_1X_5	h_2	Y_1X_6	h_1	Y_1Y_2	h_3	Y_1Y_3	h_1	Y_1Y_4	h_1	Y_1Y_5	h_1	Y_1Y_6	h_3
Y_1Z_2	h_5	Y_1Z_3	h_2	Y_1Z_4	h_1	Y_1Z_5	h_1	Y_1Z_6	h_1	Y_2X_3	h_1	y_2X_4	h_2	Y_2X_5	h_2	Y_2X_6	h_1	Y_2Y_3	h_1	Y_2Y_4	h_1
Y_2Y_5	h_1	Y_2Y_6	h_5	Y_2Z_3	h_5	Y_2Z_4	h_1	Y_2Z_5	h_1	Y_2Z_6	h_1	Y_3X_4	h_1	Y_3X_5	h_1	Y_3X_6	h_2	Y_3Y_4	h_5	Y_3Y_5	h_2
Y_3Y_6	h_1	Y_3Z_4	h_5	Y_3Z_5	h_2	Y_3Z_6	h_5	Y_4X_5	h_1	Y_4X_6	h_2	Y_4Y_5	h_2	Y_4Y_6	h_1	Y_4Z_5	h_2	Y_4Z_6	h_4	Y_5X_6	h_3
Y_5Y_6	h_1	Y_5Z_6	h_2	Z_1X_2	h_1	Z_1X_3	h_5	Z_1X_4	h_1	Z_1X_5	h_1	Z_1X_6	h_2	Z_1Y_2	h_1	Z_1Y_3	h_3	Z_1Y_4	h_3	Z_1Y_5	h_2
Z_1Y_6	h_1	Z_1Z_2	h_1	Z_1Z_3	h_1	Z_1Z_4	h_3	Z_1Z_5	h_2	Z_1Z_6	h_3	Z_2X_3	h_1	Z_2X_4	h_2	Z_2X_5	h_2	Z_2X_6	h_1	Z_2Y_3	h_1
Z_2Y_4	h_1	Z_2Y_5	h_1	Z_2Y_6	h_3	Z_2Z_3	h_2	Z_2Z_4	h_1	Z_2Z_5	h_1	Z_2Z_6	h_1	Z_3X_4	h_3	Z_3X_5	h_3	Z_3X_6	h_1	Z_3Y_4	h_1
Z_3Y_5	h_1	Z_3Y_6	h_2	Z_3Z_4	h_1	Z_3Z_5	h_1	Z_3Z_6	h_1	Z_4X_5	h_1	Z_4X_6	h_2	Z_4Y_5	h_2	Z_4Y_6	h_1	Z_4Z_5	h_2	Z_4Z_6	h_4
Z_5X_6	h_3	Z_5Y_6	h_1	Z_5Z_6	h_2	X_1	h_1	X_2	h_3	X_3	h_1	X_4	h_4	X_5	h_5	X_6	h_1	Y_1	h_2	Y_2	h_2
Y_3	h_3	Y_4	h_3	Y_5	h_3	Y_6	h_3	Z_1	h_1	Z_2	h_2	Z_3	h_3	Z_4	h_3	Z_5	h_3	Z_6	h_1		

TABLE IV: Distinct pairs of single-qubit Pauli errors for the $[[6, 1, 3]]$ quantum code. Each double-lined column lists a pair of single-qubit errors and a corresponding anticommuting generator (AG) for the code. X_4 and Z_4Z_6 lie in the gauge subgroup H.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&1&0&1&0&0&0&0&0&0\\ 1&0&1&1&1&0&0&0&0&0&0&0\\ 1&1&0&0&1&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&1&0&0&1&1\\ 0&0&0&0&0&0&1&0&1&1&1&0\\ 0&0&0&0&0&0&1&0&0&1&0&1 \end{array}\right] \tag{26}$$

We begin the algorithm by computing the symplectic product [19] between the various rows of the matrix. The first row is symplectically orthogonal to the second row. Moreover, it is symplectically orthogonal to all the rows except row six. So we swap the second row with the sixth row.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&1&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&1&0&1\\ 1&1&0&0&1&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&1&0&0&1&1\\ 0&0&0&0&0&0&1&0&1&1&1&0\\ 1&0&1&1&1&0&0&0&0&0&0&0 \end{array}\right] \tag{27}$$

Now apply Hadamard gates to qubits, one, four and six. This operation swaps the columns one, four and six on the Z side with columns one, four and six on the X side.

$$\left[\begin{array}{cccccc|cccccc} 0&0&0&0&0&0&1&0&0&1&0&1\\ 1&0&0&1&0&1&0&0&0&0&0&0\\ 0&1&0&0&1&0&1&0&0&0&0&1\\ 1&0&0&0&0&1&0&1&0&0&1&0\\ 1&0&0&1&0&0&0&0&1&0&1&0\\ 0&0&1&0&1&0&1&0&0&1&0&0 \end{array}\right] \tag{28}$$

Apply a CNOT from qubit one to qubit four and a CNOT from qubit one to qubit six. This operation adds column one to four and column one to column six on the X side. On the Z side of the matrix, the CNOT operation adds column four to column one and column six to column one.

$$\left[\begin{array}{cccccc|cccccc} 0&0&0&0&0&0&1&0&0&0&0&0\\ 1&0&0&1&0&1&0&0&0&0&0&0\\ 0&1&0&0&1&0&1&0&0&1&0&0\\ 0&0&0&0&0&1&0&1&0&0&1&0\\ 0&0&0&1&0&0&0&0&1&0&1&0\\ 0&0&1&0&1&0&1&0&0&0&0&1 \end{array}\right] \tag{29}$$

Now apply a Hadamard gate on qubit one.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&1&1&0&0&0&0&0\\ 0&1&0&0&1&0&0&0&0&1&0&0\\ 0&0&0&0&0&1&0&1&0&0&1&0\\ 0&0&0&1&0&0&0&0&1&0&1&0\\ 0&0&1&0&1&0&0&0&0&0&0&1 \end{array}\right] \tag{30}$$

Apply a Hadamard gate on qubit four and qubit six. This operation swaps columns four and six on Z side with respective columns on the X side.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&1&0&1\\ 0&1&0&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&1&1\\ 0&0&0&0&0&0&0&0&1&1&1&0\\ 0&0&1&0&1&1&0&0&0&0&0&0 \end{array}\right] \tag{31}$$

Finally, we apply a CNOT gate from qubit one to qubit

four and another CNOT gate from qubit one to qubit six.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&1&0&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&1&1\\ 0&0&0&0&0&0&0&0&1&1&1&0\\ 0&0&1&0&1&1&0&0&0&0&0&0 \end{array}\right] \tag{32}$$

At this point we are done processing qubit one and qubit two. We now proceed to manipulate columns two through six on the Z and X side. We apply a Hadamard gate on qubit two, four and five.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&1&1&0\\ 0&1&0&0&1&0&0&0&0&0&0&1\\ 0&0&0&1&1&0&0&0&1&0&0&0\\ 0&0&1&0&0&1&0&0&0&0&1&0 \end{array}\right] \tag{33}$$

Perform a CNOT gate from qubit two to qubit four and from qubit two to qubit five.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&1\\ 0&0&0&1&1&0&0&0&1&0&0&0\\ 0&0&1&0&0&1&0&0&0&0&1&0 \end{array}\right] \tag{34}$$

Perform a Hadamard on qubit two.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&1\\ 0&0&0&1&1&0&0&0&1&0&0&0\\ 0&0&1&0&0&1&0&0&0&0&1&0 \end{array}\right] \tag{35}$$

We have processed qubit three. Now look at the submatrix from columns three to six on the Z and X side. Perform a SWAP gate between qubit three and qubit five. This operation swaps column three with five in the Z submatrix and column three and five in the X submatrix.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&1\\ 0&0&0&1&1&0&0&0&0&0&1&0\\ 0&0&1&0&0&1&0&0&1&0&0&0 \end{array}\right] \tag{36}$$

Perform a Hadamard gate on qubit three, followed by a CNOT gate from qubit three to qubit six, and another Hadamard on qubit three.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&1&1&0&0&0&0&0&0&1&1\\ 0&0&0&0&1&1&0&0&0&0&0&0 \end{array}\right] \tag{37}$$

Add row four to five.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&1&1\\ 0&0&0&0&1&1&0&0&0&0&0&0 \end{array}\right] \tag{38}$$

We have completed processing qubit four. Now focus on columns four to six. Apply a Hadamard gate on qubit four, followed by CNOT gate from qubit four to qubit five, and again from qubit four to qubit six.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&1&1&0&0&0&0&0&0 \end{array}\right] \tag{39}$$

Perform a Hadamard gate on qubit four.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&1&0&0&0&0&0&0 \end{array}\right] \tag{40}$$

Now look at columns five and six. Apply a Hadamard gate on qubit five and qubit six, followed by a CNOT gate from qubit five to qubit six.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&1&0 \end{array}\right] \tag{41}$$

Perform a Hadamard on qubit five.

$$\left[\begin{array}{cccccc|cccccc} 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0 \end{array}\right] \tag{42}$$

We have finally obtained a binary matrix that corresponds to the canonical stabilizer generators in Table III(a). Figure 3 gives the encoding circuit for the all the quantum operations that we performed above. Multiplying the above operations in reverse takes us from the unencoded canonical stabilizers to the encoded ones.

[1] P. W. Shor, Phys. Rev. A **52**, R2493 (1995).
[2] A. R. Calderbank and P. W. Shor, Phys. Rev. A **54**, 1098 (1996).
[3] D. Gottesman, Phys. Rev. A **54**, 1862 (1996).
[4] D. Gottesman, Ph.D. thesis, California Institue of Technology (1997).
[5] A. M. Steane, Phys. Rev. Lett. **77**, 793 (1996).
[6] R. Laflamme, C. Miquel, J. P. Paz, and H. W. Zurek, Phys. Rev. Lett. **77**, 198 (1996).
[7] C. H. Bennett, D. P. DiVincenzo, J. A. Smolin, and W. K. Wootters, Phys. Rev. A **54**, 3824 (1996).
[8] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, 2000).
[9] A. Calderbank, E. Rains, P. Shor, and N. Sloane, IEEE Trans. Inf. Theory **44**, 1369 (1998).
[10] A. Klappenecker and P. K. Sarvepalli, arXiv:quant-ph/0703213 (2007).
[11] N. D. Mermin, *Quantum Computer Science* (Cambridge University Press, 2007).
[12] E. Knill, R. Laflamme, and L. Viola, Phys. Rev. Lett. **84**, 2525 (2000).
[13] P. Aliferis and A. W. Cross, Phys. Rev. Lett. **98**, 220502 (2007).
[14] D. Kribs, R. Laflamme, and D. Poulin, Phys. Rev. Lett. **94**, 180501 (2005).
[15] S. A. Aly and A. Klappenecker, in *Proceedings of the IEEE International Symposium on Information Theory (arXiv:0712.4321)* (2008).
[16] D. Poulin, Phys. Rev. Lett. **95**, 230504 (2005).
[17] S. A. Aly, arXiv:0802.4270 (2008).
[18] G. Bowen, Phys. Rev. A **66**, 052313 (2002).
[19] T. A. Brun, I. Devetak, and M.-H. Hsieh, arXiv:quant-ph/0608027 (2006).
[20] T. A. Brun, I. Devetak, and M.-H. Hsieh, Science **314**, pp. 436 (2006).
[21] M.-H. Hsieh, I. Devetak, and T. A. Brun, arXiv:0708.2142 (2007).
[22] T. Brun, I. Devetak, and M.-H. Hsieh, in *Proceedings of the IEEE International Symposium on Information Theory* (2007).
[23] M. M. Wilde, H. Krovi, and T. A. Brun, arXiv:0708.3699 (2007).
[24] M. M. Wilde, H. Krovi, and T. A. Brun, Phys. Rev. A **76**, 052308 (2007).
[25] M. M. Wilde and T. A. Brun, arXiv:0712.2223 (2007).
[26] M. M. Wilde and T. A. Brun, in *IEEE International Symposium on Information Theory (arXiv:0801.0821)* (2008).
[27] G. Gilbert, M. Hamrick, Y. S. Weinstein, V. Aggarwal, and A. R. Calderbank, arXiv:0709.0128 (2007).
[28] A. Shabani and D. A. Lidar, arXiv:0708.1953 (2007).
[29] M. M. Wilde and T. A. Brun, Phys. Rev. A **77**, 064302 (2008).
[30] J. Eisert and M. B. Plenio, International Journal of Quantum Information **1**, 479 (2003).