EFFICIENT QUANTUM CIRCUITS FOR NON-QUBIT QUANTUM ERROR-CORRECTING CODES

MARKUS GRASSL, MARTIN ROTTELER, and THOMAS BETH
Institut für Algorithmen und Kognitive Systeme, Universität Karlsruhe,
Am Fasanengarten 5, 76128 Karlsruhe, Germany.

Received (November 4, 2002)
Revised
Communicated by Jozef Gruska

ABSTRACT

We present two methods for the construction of quantum circuits for quantum error-correcting codes (QECC). The underlying quantum systems are tensor products of subsystems (qudits) of equal dimension which is a prime power. For a QECC encoding $k$ qudits into $n$ qudits, the resulting quantum circuit has $O(n(n-k))$ gates. The running time of the classical algorithm to compute the quantum circuit is $O(n(n-k)^2)$.

Keywords: Quantum circuits, quantum error correction.

1. Introduction

Most quantum error-correction codes (QECC) have been constructed for quantum systems that are composed of two-dimensional subsystems, i.e., quantum bits or short qubits. At first glance, the size of the alphabet used to encode the information to be processed seems to be irrelevant. But already in the context of classical information processing and communication this is not true. For example, the entropy of a message depends on the alphabet used for the encoding, and increasing the size of the alphabet allows the construction of better error-correcting codes [22, 27].

In the context of quantum information, a quantum system consisting of two three-dimensional subsystems, i.e., qutrits, shows new features when compared to a two-qubit system. For example, bound entanglement exists only for the former [15], and for two qutrits there is even "nonlocality without entanglement" [3].

In this paper, we consider quantum systems which have subsystems of dimension $d = p^m$, where $p$ is prime. As a shorthand, we will use the term "qudit". Quantum codes for qudit systems have been studied, e.g., in [1, 2, 10, 24]. The question of encoding and decoding these codes, however, was not explicitly addressed. Here...
we present efficient (classical) algorithms to compute efficient quantum networks for the encoding process. The method applies to qubit codes as well, for which encoding algorithms have been discussed, e.g., in [6, 11, 12, 13, 14].

In the following section, we present the mathematical framework of quantum information processing using composed quantum systems. In Section 3 we recall the basic concepts and constructions of quantum error-correcting codes. A fast encoding algorithm for the class of CSS codes is presented in Section 4. The main result, an encoding algorithm for stabilizer codes over higher dimensional quantum systems, is derived in Section 5 and illustrated in Section 6.

2. Non-binary Quantum Systems

2.1. Quantum States and Registers

In the context of classical information processing, information is encoded using words over some finite alphabet \( A \), most often \( A = \{0, 1\} \). The elementary symbols of quantum information processing are states of a finite dimensional quantum system. Those can be modeled by normalized vectors in a complex Hilbert space \( \mathbb{C}^d \). Again, the simplest case of \( \mathbb{C}^2 \) is considered most often. Then the orthonormal basis states of such a quantum bit, or short qubit, are written as \( |0 \rangle \) and \( |1 \rangle \). (The notation of quantum states as ket vectors \( |i \rangle \) is attributed to Dirac [7]). A general state of a qubit is given by

\[ |i \rangle = a|0 \rangle + b|1 \rangle, \quad |a|^2 + |b|^2 = 1. \]

If both \( a \) and \( b \) are non-zero, the state \( |i \rangle \) is a so-called superposition of \( |0 \rangle \) and \( |1 \rangle \) with amplitudes \( a \) and \( b \). For a \( d \)-dimensional system, we label the orthonormal basis states by the elements of some alphabet of size \( d \), e.g., the numbers \( 0, 1, \ldots, d-1 \) or the elements of a finite field, if \( d \) is a prime power. The general state of a qudit is given by

\[ |i \rangle = \sum_{j=0}^{d-1} c_j |j \rangle, \quad \sum_{j=0}^{d-1} |c_j|^2 = 1. \]

Combining several qudits, we obtain a quantum register. The canonical basis states of a quantum register of length \( n \) are tensor products of the basis states of the single qudits. Hence we can label them by words \( x \in A^n \) of length \( n \). For the basis states of a quantum register we use the following notations:

\[ |k_1 \rangle |k_2 \rangle \cdots |k_n \rangle = |k_1 \rangle |k_2 \rangle |k_3 \rangle |k_4 \rangle \cdots |k_n \rangle = |k_1 \rangle |x_{k_2} \rangle \cdots |x_{k_n} \rangle = |x_{k_1} \rangle. \]

A general state of a quantum register of length \( n \) is a normalized vector in the exponentially large Hilbert space \( \mathbb{C}^d \times \mathbb{C}^d \times \cdots \times \mathbb{C}^d \), given by

\[ |i \rangle = \sum_{x \in A^n} c_x |x \rangle |x \rangle \cdots |x \rangle, \quad \sum_{x \in A^n} |c_x|^2 = 1. \]
When writing states of quantum registers, normalization factors may be omitted.

The elements of the dual space $H$ will be denoted by `bra vectors' $\langle h | i \rangle$. Then the inner product of two vectors $\langle h | i \rangle$ and $\langle j | k \rangle$ reads $\langle h | i \rangle \langle i | j \rangle$, and linear operators $M$ can be written in the form

$$M = \langle M |_{ij} \rangle = \sum_{ij} M_{ij} \langle i | j \rangle$$

As the operations have to preserve the normalization of the vectors, the admissible operations are unitary operators, which we will discuss next.

2.2. Elementary Gates

In the following we will consider qudit systems where each qudit corresponds to a $q$-dimensional Hilbert space where $q = p^n$ is a prime power.

**Definition 1 (Elementary Gates)** Let $q$ be a prime power, i.e., $q = p^n$ where $p$ is prime. By $\omega$ we denote a primitive complex $p$-th root of unity, i.e., $\omega = \exp(2\pi i/p)$. Furthermore, let $\text{tr}(z)$ denote the trace of an element $z \in \mathbb{F}_q = \mathbb{F}_{p^n}$ which is defined as $\text{tr}(z) = \sum_{i=0}^{m-1} p^i z \in \mathbb{F}_p$. Then we define the following operations:

1. $X = X^2 F_q \langle h | x \rangle$
2. $Z = Z^2 F_q \langle h | x \rangle$
3. $M = M^2 F_q \langle h | x \rangle$
4. $\text{DFT} = \frac{1}{\sqrt{q}} X \langle h | x \rangle$
5. $\text{ADD} = \langle h | x \rangle$
6. $\text{HORNER} = \langle h | x \rangle$

Here, when writing $\text{tr}(z)$, we identify $\mathbb{F}_p$ and $\mathbb{Z}/p\mathbb{Z}$, the integers modulo $p$. Then the function $z \mapsto \text{tr}(z)$ is an additive character of $\mathbb{F}_q$. Different values of $\text{tr}(z)$ yield the different additive characters of $\mathbb{F}_q$ (see, e.g., [17, 21]).

A graphical representation of these elementary operations is so-called quantum
gates is given in Figure 1. Each horizontal line corresponds to one qudit. The first three operations operate on single qudits. A superscript in brackets indicates on which subsystem the transformation acts, e.g., \(X^{(2)} = \text{id} X \text{id} \cdots \text{id}\). The operations \(X\) and \(M\) correspond to the addition of a fixed element \(2 \mathbb{F}_q\) and the multiplication with a fixed element \(6\), respectively. The operation \(Z\) has no direct classical analogue, it changes the phases of the basis states. The Fourier transformation DFT can be used to transform the state \(\psi_i\) to a superposition of all basis states with equal amplitudes, i.e.,

\[
\text{DFT } \psi_i = \frac{1}{\sqrt{q}} \sum_{j=0}^{q-1} X_i j \psi_i.
\]

Starting from this superposition, a quantum computation can, e.g., evaluate a function in parallel for all possible inputs. Arbitrary classical functions over \(\mathbb{F}_q\) can be implemented using the gates ADD\(^{(1,2)}\) and HORNER\(^{(1,2,3)}\). The former corresponds to the reversible implementation of the addition of two elements. The first qudit is called \{control\}, the second \{target\}. The latter is a universal reversible gate over \(\mathbb{F}_q\), as any function over \(\mathbb{F}_q\) corresponds to a polynomial which can be evaluated using the Horner scheme. It is the generalization of the so-called Toffoli gate [26] for qubits, which is universal gate for reversible boolean functions, as well as the Fredkin gate [8].

3. Quantum Error-Correcting Codes

3.1. Unitary Error Bases

In the following, we will recall the basic properties and constructions of quantum error-correcting codes. In order to construct an error-correcting code, one has to specify an error model. The error model can be specified by a finite set of error operators. In [20], the following characterization of error-correcting codes is given. Theorem 1 Let \(C\) be a subspace of the Hilbert space \(H\) with orthonormal basis \(\{\psi_i : \cdots : \psi_k\}\). Then \(C\) is a quantum error-correcting code for the error-operators \(E = E_1, \ldots, E \in \mathbb{F}_q\) if and only if there exists a \(2 \times 2\) such that for all \(\psi_i, \psi_j, \psi_k\) and \(E_1, E_2 \in E\)

\[
\sum_{i,j} E_{ij} E_{kl} = \delta_{ij} \delta_{kl}.
\]

Most quantum error-correcting codes are designed to correct local errors, i.e., errors that affect only some subsystems. The error acting on a single subsystem cannot be any linear transformation. It is sufficient that condition (1) holds for a basis of the linear space of error operators. For qudit systems of prime power dimension \(q\), we consider the following set of unitary operators:

\[
E = fX Z : 2 \mathbb{F}_q; \quad \text{(2)}
\]

It is not hard to show that those \(q^2\) operators are an orthogonal basis with respect to the inner product \(\langle A, B \rangle = \text{tr}(A^\dagger B)\). Further one, they generate an error group \(G_1\).
of size \( p^2 \) with center \((G_z) = \mathbb{H} I_i\) (see [18, 19]). Any element of \(G_z\) can uniquely be written as \( X \otimes Z \) where \( 2 \otimes 0; \ldots, p \otimes 1g \) and \( 2 \otimes F_q \). The commutation relations of two elements are derived from

\[
X \otimes Z X X = X \otimes j X + \imath h X \theta (z) \otimes j \theta X \gamma X + \imath j
\]

\[
X \otimes X = \imath \theta (z) \otimes X + \imath j \theta X \gamma X.
\]

Hence commuting two elements results in a phase factor, i.e.,

\[
(X \otimes Z) \otimes (X \otimes Z \otimes Z) = \imath \theta (z) \otimes \theta (z) \otimes \theta (z).
\] (3)

For an \( n \)-qudit system, the error basis and the error group are the \( n \)-fold tensor products \( E^n \) and \( G_n = G_z^n \), respectively. The weight of an error operator \( E \otimes E^n \) is the number of tensor factors that are different from identity. If a \( k \)-dimensional subspace \( C \) of \( (C^3)^n \) can correct all errors of weight no greater than \( t \), \( C \) is a \( t \)-error-correcting code with minimum distance \( 2t+1 \), denoted by \((n;K;2t+1)\).

### 3.2. Stabilizer Codes

A particular class of quantum error-correcting codes are so-called stabilizer codes (see [2, 4, 9]). The basic idea is to consider an Abelian subgroup \( S \) of the error group \( G_n \) such that its intersection with the center of \( G_n \) is trivial. The stabilizer code \( C \) is defined as the common eigenspace of the operators in \( S \). Stabilizer codes can be described in terms of certain classical codes over \( n \) qubits.

Any element \( E \) of the error group \( G_n \) can uniquely be written as

\[
E = \langle X \otimes Z \rangle \langle X \otimes Z \rangle \cdots \langle X \otimes Z \rangle = \langle X \otimes Z \rangle
\]

where \( 2 \otimes 0; \ldots, p \otimes 1g \) and \( = (\langle 1 \otimes 1 \rangle, \ldots, \langle n \otimes n \rangle)2\otimes F_q^n \). The weight of an element \( X \otimes Z \) is the number of indices \( i \) for which not both \( i \) and \( i \) are zero. From the commutation relation (3), it follows that for \( (\langle 1 \otimes 1 \rangle, \ldots, \langle 1 \otimes 1 \rangle)2\otimes F_q^n \)

\[
(X \otimes \otimes Z) \otimes (X \otimes Z \otimes Z) \otimes (X \otimes Z) = \langle 0 \rangle \langle 0 \rangle \langle 0 \rangle \langle 0 \rangle
\]

where the inner product is defined by

\[
\langle 0 \rangle \langle 0 \rangle = \langle 0 \rangle \langle 0 \rangle \\text{tr}(\langle 0 \rangle \langle 0 \rangle | I \rangle \langle 0 \rangle | I \rangle)
\]

This shows that the group \( G_n = G_z = \mathbb{H} I_i \) is isomorphic to \( F_q^n \). Further, one, two elements \( X \otimes Z \) and \( X \otimes Z \) commute if and only if \( \langle 0 \rangle \langle 0 \rangle = 0 \). Hence an Abelian subgroup \( S \) of \( G_n \) corresponds to a subspace \( C \) of \( F_q^n \) that is contained in its dual \( C^\ast \) with respect to the inner product (4).
Definition 2 (Stabilizer Matrix) Let \( S \) be an Abelian subgroup of \( G \), which has trivial intersection with the center of \( G \). Furtherm ore, let \( g_1, g_2, \ldots, g_k \) where \( g_i = X, Z \) with \( 2 F_0; \ldots; p \) and \( (i, j) \) \( 2 F_q^n \), \( F_q^n \) be a minimal set of generators for \( S \). Then a stabilizer matrix of the corresponding stabilizer code \( C \) is a generator matrix of the (classical) linear code \( C \). We will write this matrix in the form

\[
\begin{bmatrix}
0 & 1 \\
2 & C \\
\vdots & A \\
k & 2 F_q^n \end{bmatrix}
\]

Any error operator \( E \) that does not commute with all elements \( S \) will change the eigenvalue of an eigenstate \( j \) of \( S \) which can be detected by a measurement. Recall that \( E = X, Z \) does not commute with all \( S \) if and only if \( (i, j) \) \( C \).

Finally, the minimum distance of a stabilizer code is the minimum weight of the vectors \( (i, j) \) \( C \), since the errors corresponding to \( C \) are those that cannot be detected, and the operators corresponding to \( C \) have no effect on the code.

4. Encoding CSS Codes

A special class of stabilizer codes are so-called CSS codes named after Calderbank, Shor \( 5 \) and Steane \( 25 \). Originally, they have been designed for qubit systems, but they can easily generalized to any dimension (see, e.g., \( 1 \)). Given two linear codes \( C_1 = [n; k_1; d_1] \) and \( C_2 = [n; k_2; d_2] \) over \( F_q \) with \( C \), \( C_1 \), the basis states of the corresponding CSS code are

\[
X \frac{c_1}{c_2} \frac{w}{1} \quad \text{where} \quad w \in C_1. \tag{5}
\]

Two states \( j \) and \( j' \) are identical if and only if \( w \in C_1 \), otherwise they are orthogonal. As the dimension of \( C_2 \) is \( n \), there are \( q \) different cosets where \( k = k_1 + k_2 \). Hence the dimension of the CSS code is \( q \). Its minimum distance is \( d = \min(d_1, d_2) \).

We illustrate the encoding of a CSS code for the code \( [7; 3; 3] \) over seven 8-dimensional quantum systems. Let \( 2 F_q \) be a primitive element of \( F_q \) with minimal polynomial \( X = X^3 + X + 1 \). The code \( C = [7; 2; 6] \) with generator matrix

\[
G = \begin{bmatrix}
1 & 0 & 3 & 1 & 3 \\
0 & 1 & 4 & 1 & 5 & 5 & 4
\end{bmatrix}
\]

is contained in its dual \( C^7 = [7; 5; 3] \) with generator matrix

\[
H = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 3 & 1 & 3 & 1 & 1 \\
0 & 1 & 4 & 1 & 5 & 5 & 4 \\
0 & 0 & 0 & 1 & 0 & 0 & 0
\end{bmatrix}
\]
The first two rows of \( H \) are equal to \( G \). The cosets of \( C^7 = C \) are given by linear combinations of the last three rows. Denoting the rows of \( H \) by \( h_1; h_2; \ldots; h_6 \), we can rewrite equation (5) as

\[
\text{j}_{a;b;c} = \frac{1}{8} X \begin{bmatrix}
\text{j}h_1 + \text{j}h_2 + \text{ah}_3 + \text{bh}_4 + \text{ch}_5
\end{bmatrix}_{ij} F_i
\]

where \( a;b;c \in F_2 \). Applying a Fourier transformation to the first and second qudit of the initial state \( \text{j}0\text{i}j\text{i}j\text{i}j\text{i}0\text{i}j \), we obtain

\[
\frac{1}{8} X \begin{bmatrix}
\text{j}i\text{i}j\text{i}j\text{i}j\text{i}0\text{i}j
\end{bmatrix}_{ij} F_i
\]

Now we sequentially add the corresponding multiple of the rows of \( H \) in reverse order, i.e., starting with \( h_6 \). As \( H \) is in row echelon form, this corresponds to a sequence of modified ADD gates instead of adding the first qudit to the second, we have to add a multiple of it. The resulting encoding circuit is shown in Figure 2.

![Figure 2: Encoding circuit for the CSS code [7;3;3]. The Fourier transformation DFT is abbreviated by F, and for the gate M we give only the value of .](image)

For each row \( h_k \) of \( H \), we have a sequence of gates \( \text{ADD}^{(k;1)} \) corresponding to the non-zero entries \( H_{k1} \) for \( k \leq 6 \). The gate \( \text{ADD}^{(k;1)} \) is conjugated by a multiplication gate \( M \) where \( = H_{k1} \). Note that we have not combined adjacent multiplication gates. In general, \( G \) is a generator matrix of \( C_2^7 \), \( C_1 \), and \( H \) is its completion to a generator matrix of \( C_1 \). Using the construction illustrated above, we get

Proposition 1 Let \( C_1 = [h; k_1; d_1]_q \) and \( C_2 = [h; k_2; d_2]_q \) be linear codes over \( F_q \) with \( C_2^2 \), \( C_1 \). Then there exists a quantum circuit to encode the resulting CSS code \( C = [h; k; d]_q \) over qudits of dimension \( q \), where \( k = k_1 + k_2 \), \( n \), using \( n \) \( k_2 \) DFT gates, at most \( A = k_1 n \) \( k_2 + 1 \) ADD gates, and at most \( A + (n - 1) \) multiplication gates.

Proof. To create a state similar to (6), we need \( \text{dim} C_2^7 = n \) \( k_2 \) DFT gates. A generator matrix \( H \) for \( C_1 \) has \( k_1 \) rows. Hence the number \( A \) of ADD gates is at most \( k_1 n \). As \( H \) can be chosen to be in row echelon form, we have \( k_1 \) entries that are one. Those correspond to the controlled qudits. Furthermore, at least \( k_2 \) entries of \( H \) are zero, reducing the number of ADD gates further. Finally, each ADD gate is conjugated by a multiplication gate. Adjacent multiplication gates between the
ADD gates can be combined, so we count only the multiplication gates before the ADD gates, plus at most \( n \) multiplication gates at the end.

As the role of \( C_1 \) and \( C_2 \) in the construction of CSS codes is completely symmetric, one has the freedom to choose the matrices \( G \) and \( H \) to be generator matrices of either \( C_2^T \) and \( C_1 \), respectively \( C_1^T \) and \( C_2 \). This may reduce the number of gates by a constant factor, as well as optimizing the particular generator matrices.

5. Encoding Qudit Stabilizer Codes

In this section we derive an encoding algorithm for general stabilizer codes over qudit systems of prime power dimension \( q = p^s \). The main idea is to transform the Abelian stabilizer group \( S_n \) of the stabilizer code \( C = [n;k;d]_q \) into a stabilizer group \( S_0 \) for which encoding is particularly easy. Up to a permutation \( S_n \) of the qudits, which we may ignore in the sequel, that group is \( S_0 = \mathbb{Z}_p^{(1)} \otimes \mathbb{Z}_p^{(2)} \otimes \cdots \otimes \mathbb{Z}_p^{(n-k)} \). The corresponding stabilizer code \( C_0 \) cannot correct single qudit errors at arbitrary positions as the common eigenstates of \( S_0 \) are tensor products of the \( n \times (n-k) \)-qudit state \( |00:01 \rangle \) and the unencoded state \( j_{n} 12 (C^a)^k \), i.e., they are of the form

\[ |00:01ij_{n} 12 (C^a)^k \rangle. \]  

As the stabilizer groups \( S \) and \( S_0 \) are conjugated to each other there exists a transformation \( D \) such that

\[ D^1 S D = h\mathbb{Z}_p^{(1)} \otimes \mathbb{Z}_p^{(2)} \otimes \cdots \otimes \mathbb{Z}_p^{(n-k)} i = S_0; \]  

where = denotes equality up to the permutation of the qudits. Combining (7) and (8), it follows that the states \( D^1 (|00:01ij_{n} 12 (C^a)^k \rangle) \) are common eigenstates of \( S_0 \). Hence the transformation \( D^1 \) can be used to encode that states of \( C_0 \). In the following, we show that the transformation \( D \) can be efficiently decomposed into single qudit transformation and ADD-gates.

5.1. Conjugating the Error Basis

First we consider the operation of the Fourier transformation over the additive group of the \( \text{ed} \ F_q \) (see Definition 1 (iv)) on the error group modulo the center. The action on \( Z \) is given by

\[
\text{DFT}^1 Z \text{ DFT} = \frac{1}{q} \sum_{ij}^q X^{i \cdot \text{tr}(ij)} \sum_{kl}^q X^{k \cdot \text{tr}(kl)} \sum_{j} F_q \frac{1}{q} \sum_{l} F_q X^{j \cdot \text{tr}(jl)} \sum_{z} F_q X^{z \cdot \text{tr}(z)} \sum_{i} F_q X^{i \cdot \text{tr}(ij)} \sum_{j} F_q \frac{1}{q} \sum_{l} F_q X^{j \cdot \text{tr}(jl)}.
\]
Similarly,

\[ \text{DFT}^1 X \text{DFT} = \frac{1}{2} X^* \text{tr}(ij) \bar{x} \bar{y} \bar{z} + i \text{tr}(ij) \bar{x} \bar{y} \bar{z} \]

\[ = \frac{1}{2} X^* \text{tr}(ij) \bar{x} \bar{y} \bar{z} + i \text{tr}(ij) \bar{x} \bar{y} \bar{z} \]

\[ = \frac{1}{2} x^* \text{tr}(ij) \bar{x} \bar{y} \bar{z} + i \text{tr}(ij) \bar{x} \bar{y} \bar{z} \]

\[ = \frac{1}{2} X \text{tr}(ij) \bar{x} \bar{y} \bar{z} \]

As any element of \( G_1 \) corresponds to a row vector \((i,j)\), we can describe the action of DFT on \( G_1 \) as the linear transformation \( \text{DFT} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \).

Next we investigate the action of the matrix \( M \) corresponding to the multiplication with \( 2 F_q \) \((0, \text{see Definition 1 (iv))}. We compute

\[ M^1 X Z M = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} \text{tr}(z) & \text{tr}(v) \\ \text{tr}(v) & \text{tr}(z) \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

Hence, \( M \) acts on \((i,j)\) as \( M = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \).

For the next operation, we have to distinguish the cases whether \( q \) is odd or \( q \) is even. If \( q \) is odd, we define the operator

\[ P = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

which commutes with \( Z \). Furthermore, \( X \) acts on \( P^1 \) as follows:

\[ X^1 P^1 X = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} \text{tr}(z) & \text{tr}(v) \\ \text{tr}(v) & \text{tr}(z) \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]

\[ = \begin{pmatrix} x & x \\ x & x \end{pmatrix} \]
Hence $P$ acts on $(\mathbf{i}; \mathbf{j})$ as $P = \frac{1}{m} \mathbf{i}$. If $q$ is even, we have to use another definition for the operator $P$ as we cannot divide by two. For this, we can define a self-dual basis $B = b_1; \ldots ; b_m$ of $F_q = F_{2^m}$ over $F_2$ (see e.g., [17]). Hence, by definition, $\text{tr}(b_i b_j) = i_{ij}$. Any element $2 F_2$ can be uniquely written as $i_{1} \ldots i_{m}$ where $i_{2} F_2$. The coefficients are given by $i_{ij} = \text{tr}(b_i b_j)$ since

$$
X^{i} X^{j} = X^{i_{ij}}, \quad j_{1} \ldots j_{m} = \text{tr}(b_1 b_{2}) = \text{tr}(b_1 b_{3}) = \ldots = \text{tr}(b_1 b_{m}) = 1. \quad \text{for } i_{1} \ldots i_{m}.
$$

Lemma 1. Let $F$ be given by $2$ where $d$ divides $i_2$. For this, we define an arbitrary self-dual basis of $F_2$. Further, we define an integer-valued function on $F_2$ as

$$
\text{wgt}(F_q) = \mathbf{Z}; \quad \text{wgt}(b_j) = 2 F_2.
$$

Then the following holds for all $i_{1} F_2$:

$$
\text{wgt}(i_{1} F_2) = \text{wgt}(i_{2} F_2) = (i_{1} F_2) \text{tr}(b_j b_{k}) = i_{1} F_2 \text{tr}(b_j b_{k}) = \text{wgt}(i_{1} F_2).
$$

Proof. First, we observe that for any element $b_j$ of $F_2$, we have $i_{1} F_2 = \text{wgt}(i_{1} F_2) = \text{wgt}(i_{2} F_2)$

Second, the size of this set $m$ modulo two is given by

$$
\text{tr}(i_{1} F_2) = \text{tr}(0) X^{i_{1} F_2} = \text{tr}(0) X^{i_{1} F_2} = 1.
$$

which completes the proof.

Now we are ready to define an operator $P_1$ as

$$
P_1 = \begin{pmatrix} X & (i_{1} F_2) & X & (i_{1} F_2) \\ y_{1} F_2 & y_{1} F_2 & y_{1} F_2 & y_{1} F_2 \\ \end{pmatrix}
$$

Again, $P_1$ commutes with all matrices $Z$. The action on $X$ is derived from

$$
X^{i_{1} P_1} X = X^{i_{1} P_1} X = X^{i_{1} P_1} X = \begin{pmatrix} X & X & X & X \\ y_{1} F_2 & y_{1} F_2 & y_{1} F_2 & y_{1} F_2 \\ \end{pmatrix}
$$

Hence

$$
P_1 = (i_{1} F_2) Z P_1^{1}:
$$

$$
P_1^{1} X = (i_{1} F_2) Z X.
$$

Hence

$$
P_1^{1} X = (i_{1} F_2) X Z = (i_{1} F_2) X Z.
$$

Hence

$$
P_1^{1} X = (i_{1} F_2) X Z = (i_{1} F_2) X Z.
$$
Finally, for any \( 2 \mathbb{F}_q \) \( 6 \) 0, we define \( P \) as

\[
P := M_0^1 P_1 M_0^1 \quad \text{where} \quad M_0^1 = . \quad (10)
\]

Note that 0 is uniquely defined as \( x \rightarrow x^2 \) is an automorphism of \( \mathbb{F}_q = \mathbb{F}_{2^m} \). The matrix \( M \) is a permutation matrix, hence \( P \) is diagonal and commutes with \( Z \).

The action of \( P \) on \( X \) is given by

\[
P ^1 X P ^{-1} = M_0^1 P_1 X M_0^1 P_1 M_0^1 = M_0^1 P_1 X Z M_0^1 P_1 M_0^1 = Z \frac{x^\sigma}{x^\sigma} X Z^\sigma.
\]

Summarizing, the operators \( P \) (defined by (9) for \( q \) odd and (10) for \( q \) even) acts on \(( 1 ) \) always as \( P = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \).

Altogether, we have shown

**Theorem 2** The group \( \mathbb{J}_{\mathbb{F}_q} = \langle \text{DFT}, P, M \rangle : 2 \mathbb{F}_q \rangle \) acts on \( \mathbb{G}_1 \), the error group \( \mathbb{G}_1 \) mod the center, as \( \mathbb{SL}(2; \mathbb{F}_q) \).

**Proof.** The action of the matrices \( \text{DFT}, P, M \) on the error group \( \mathbb{G}_1 \) is given by

\[
\begin{bmatrix}
\text{DFT} & \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\
\overline{P} & \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \\
\overline{M} & \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}
\end{bmatrix}
\]

respectively. These matrices generate \( \mathbb{SL}(2; \mathbb{F}_q) \) (see, e.g., [23]).

As \( \mathbb{SL}(2; \mathbb{F}_q) \) acts transitively on the non-zero vectors (see, e.g., [16]), we obtain

**Corollary 1** The group \( \mathbb{J}_{\mathbb{F}_q} \) acts transitively on the non-trivial elements of \( \mathbb{G}_1 \).

So far, we have only considered the action of single qudit operations on the error group \( \mathbb{G}_n \). Our goal is to transform an arbitrary Abelian stabilizer group \( S \subset \mathbb{G}_n \) into \( S_0 \) which corresponds to the code \( C_0 = [n; k; 1] \). As \( C_0 \) cannot correct errors and single qudit operations do not change the error-correcting properties of a stabilizer code, we need additional transformations which are ADD-gates acting on pairs of qudits. Combining the definitions of \( X \) and \( \text{ADD}^{(1;2)} \) (see Definition 1 (i) and (v)), we can rewrite the \( \text{ADD} \)-gate as

\[
\text{ADD}^{(1;2)}(x) = \begin{cases} 
X & \text{if } j \in S \text{ and } h \in S \text{ and } j \neq h
\end{cases}
\]

Using (11), it is easy to show that \( \text{ADD}^{(1;2)} \) commutes with all matrices of the form \( Z \).

11
We use the stabilizer matrix representation. So our goal is to find a transformation \( D \) such that

\[
\begin{bmatrix}
X & Z
\end{bmatrix} = \begin{bmatrix}
D
\end{bmatrix} S
\]

Proof. We show that the algorithm computes a transformation \( D \) such that

\[
D^2 SD = S_0\quad \text{(see eq. (8))}
\]

instead of operating on the stabilizer group \( S \). Instead, we use the stabilizer matrix representation. So our goal is to find a transformation \( D \) such that

\[
\begin{bmatrix}
X & Z
\end{bmatrix} = \begin{bmatrix}
G & 0
\end{bmatrix};\quad \text{where } A \in \mathbb{F}_q^{(n,k)} \quad \text{has full rank.} \quad (12)
\]

In order to obtain a more compact graphical representation of the final quantum circuit, we will first transform the stabilizer matrix into the form

\[
\begin{bmatrix}
X & Z
\end{bmatrix} = \begin{bmatrix}
0 & 0
\end{bmatrix};\quad \text{where } A \in \mathbb{F}_q^{(n,k)} \quad \text{has full rank,} \quad (13)
\]

and use \((n,k)\) local Fourier transforms to exchange the \(X\)- and \(Z\)-part of the stabilizer matrix.
Input: a stabilizer matrix \((X;Z) \in \mathbb{F}_q^{2 \times 2n}\)

Output: a decoding transformation \(D = F^{(1)} A_1 T_1^{(1)} \cdots A_2 T_1^{(2)} \cdots A_n T_1^{(n)}\)

1. \(L\); 
2. foreach row \(i = 1\) to \(n\) do 
3.   foreach column \(j = 1\) to \(n\) do 
4.     if \((X_{ij};Z_{ij}) \notin \{(0,0)\}\) then 
5.       find a transformation \(T_{ij} \in J_{F_q}\) such that \((X_{ij};Z_{ij}) \rightarrow \overline{T_{ij}} = (1;0)\) 
6.       foreach row \(l = i\) to \(n\) do 
7.         \((X_{lj};Z_{lj})\) 
8.       end for 
9. else 
10.   \(T_{ij} = \text{id}\) 
11. end if 
12. end for 
13. \(T_i^{(1)} \cdots T_i^{(n)}\) 
14. end for 
15. find the first column \(l \in L\) where \(X_{il} \neq 0\) 
16. include \(l\) into \(L\) 
17. \(A_i = \text{id}\) 
18. foreach column \(j = 1\) to \(n\) do 
19.   if \(l \notin L\) and \(X_{ij} = 1\) then 
20.     \(A_i = A_i \text{ADD}^{(ij)}\) 
21.     foreach row \( = i\) to \(n\) do 
22.       \((\overline{X_{ij}Z_{ij}}) \rightarrow \overline{X_{ij}Z_{ij} + Z_{ij}}\) 
23.     end for 
24.   end if 
25. end for 
26. \(F_{2 \times L}^{(ii)}\) 
27. return \((F;A_1 T_1^{(1)} \cdots A_2 T_1^{(2)} \cdots A_n T_1^{(n)})\)

Fig. 3. A Algorithm to compute a decoding circuit for a qudit stabilizer code.

We prove the correctness of our algorithm by induction over \(i\), corresponding to the loop steps 2(25). The induction hypothesis is that the first \(i - 1\) rows of the stabilizer matrix are, again up to a permutation of the columns, in the form \((A_{i-1}, 0)\), where \(A\) is an \((i - 1) \times (i - 1)\) matrix of full rank.

After the loop steps 3(12), the \(Z\)-part of the \(i\)-th row of the stabilizer matrix will be zero. From Theorem 2 and Corollary 1 it follows that in step 5 we can always find a transformation \(T_{ij} \in J_{F_q}\) with the desired property. The loop steps 6(8) updates the stabilizer matrix. In step 13, we combine the transformations applied to each qudit to a transformation on all qudits. The definition of the stabilizer matrix (see Definition 2) implies that it has full rank. Hence in step 14, we will always find a column \(i\) with a non-zero entry \(X_{il}\). That column will be recorded in step 15. Then the loop steps 17(24) searches for column \(j \notin L\) where \(X_{ij}\) is non-zero. Applying an ADD-gate with control \(l\) and target \(j\) will change those positions \(X_{ij}\) to zero. The loop steps 20(22) updates the stabilizer matrix accordingly. So after step 24,
only the entries \(X_{ij}\) for \(12 \leq i\) may be non-zero, all other entries in the \(i\)-th row of the stabilizer matrix are zero. This completes the induction step from row \(i - 1\) to row \(i\).

A finer step 25, the 2-part of the stabilizer matrix is zero, and only the \(n \times k\) columns 12 \(\leq i\) of the \(n \times 2\) part are non-zero. Applying the transformation \(F\), i.e., local Fourier transformations at those positions, yields the stabilizer group \(S_0\), completing the proof of the correctness of our algorithm.

The running time of the algorithm follows directly from the fact that at most three for-loops are nested, two of them iterating over the \(n \times k\) rows, one over the \(n\) columns.

Each of the \(n \times k\) transformations \(A_j\) is a product of at most \(n \times j\) ADD-gates. Hence the total number of ADD-gates is at most \(\sum_{j=1}^{n} (n \times j) = n (n \times k) \frac{n k + 1}{2}\). Each of the \(n \times k\) transformations \(T_j\) is the product of \(O(n)\) single qubit operations. Together with the \(n \times k\) local Fourier transformations \(DFT\) in the transformation \(F\), the number of single qubit gates is \(O(n \times n \times k))\).

Note that, in contrast to the situation of qubits, for \(q = p^m\), \(p > 2\), the ADD-gate over \(F_q\) for \(q = p^m\), \(p > 2\), is not its own inverse. So a different graphical representation for \(ADD^1\) is required. For simplicity, we have not introduced such a representation. This is also the main reason why we do not directly compute the transformation \(D\) of (12), but decompose it as \(D = F \cdot D^0\) (see eq. (13)).

It is not difficult to modify our algorithm such we may save the main \(n \times k\) local Fourier transformations. In step 5, we have to find a transformation \(T_{ij} 2 J_{F_q}\) such that \(X_{ij} 2 Z_{ij} \cdot T_{ij} = (0; 1)\). Furthermore, we have to replace the ADD-gates by inverse ADD-gates. The modified steps of the algorithm are:

5' Find a transformation \(T_{ij} 2 J_{F_q}\) such that \((X_{ij}Z_{ij})^{-1} = (0; 1)\)

19' \(A_j = A_j (ADD^{-1})^{-1}\)

20' For each row \(i\) to \(n \times k\) do

21' \((X_jZ_j) = (X_j + X_{ij}Z_jZ_{ij})\)

22' End for

6. Example

We illustrate the algorithm of Figure 3 using a quantum code over qutrits. A stabilizer matrix of the code \(C = [9, 5; 3]\) is given by

\[
\begin{pmatrix}
0 & 1 & 0 & 2 & 1 & 2 & 2 & 0 & 1 & 1 \\
0 & 1 & 1 & 2 & 0 & 2 & 2 & 1 & 0 & 0 \\
0 & 0 & 2 & 1 & 2 & 2 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 2 & 1 & 1 & 0 & 2 & 2 & 2 \\
0 & 1 & 2 & 1 & 1 & 0 & 2 & 2 & 2 & 2
\end{pmatrix}
\]

In the first step, we transform each pair \((i; j)\) of the \(i\)-th row that is non-zero to \((1; 0)\). This is achieved by the transformation

\[
T_1 = id 2 M_2DFT 2 P_1M_2 2 P_1 2 P_2M_2 2 M_2DFT 2 P_2:
\]
The resulting stabilizer matrix is
\[
\begin{pmatrix}
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 2 & 1 & 0 & 1 & 1 & 2 & 0 & 0 \\
0 & 0 & 1 & 2 & 2 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 1 & 1 & 2 & 0 & 0 & 2 & 0 \\
1 & 0 & 2 & 2 & 2 & 1 & 1 & 2 & 1 & 0 \\
0 & 0 & 1 & 1 & 1 & 2 & 0 & 0 & 2 & 1 \\
0 & 1 & 1 & 0 & 2 & 1 & 1 & 1 & 0 & 0
\end{pmatrix}
\]
\[B = C A : \]

The first non-zero column is the first one. So using the transformation
\[A_1 = \text{ADD}^{1} \cdot \text{ADD}^{2} \cdot \text{ADD}^{3} \cdot \text{ADD}^{4} \cdot \text{ADD}^{5} \cdot \text{ADD}^{6} \cdot \text{ADD}^{7} \cdot \text{ADD}^{8} \cdot \text{ADD}^{9} \]
we obtain
\[
\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 2 & 1 & 0 & 1 & 1 & 2 & 0 & 0 \\
0 & 0 & 1 & 2 & 2 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 1 & 1 & 2 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 2 & 2 & 1 & 0 & 2 & 2 & 0 \\
0 & 0 & 1 & 1 & 1 & 2 & 0 & 0 & 2 & 0 \\
0 & 1 & 1 & 0 & 2 & 1 & 1 & 1 & 0 & 0
\end{pmatrix}
\[A : \]

Note that the first column of the Z-matrix is zero. This follows from the fact that the corresponding stabilizer elements commute with the X\(^{(1)}\) which corresponds to the first row.

In the next step, we use the transformations
\[T_2 = \text{id} \cdot \text{id} \cdot P_1 M_2 \cdot P_1 \cdot \text{DFT} \cdot P_2 \cdot \text{id} \cdot P_2 M_2 \cdot M_2 \cdot \text{DFT} \]
and
\[A_2 = \text{ADD}^{2} \cdot \text{ADD}^{3} \cdot \text{ADD}^{4} \cdot \text{ADD}^{5} \cdot \text{ADD}^{6} \cdot \text{ADD}^{7} \cdot \text{ADD}^{8} \cdot \text{ADD}^{9} \]
to obtain
\[
\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 2 & 1 & 0 & 1 & 1 & 2 & 0 & 0 \\
0 & 0 & 2 & 2 & 2 & 1 & 0 & 2 & 2 & 0 \\
0 & 0 & 2 & 1 & 2 & 2 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 2 & 2 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 2 & 2 & 1 & 0 & 2 & 2 & 0 \\
0 & 0 & 1 & 1 & 1 & 2 & 0 & 0 & 2 & 0 \\
0 & 1 & 1 & 0 & 2 & 1 & 1 & 1 & 0 & 0
\end{pmatrix}
\[A : \]

Using the transformations
\[T_3 = \text{id} \cdot \text{id} \cdot M_2 \cdot P_1 M_2 \cdot P_1 M_2 \cdot \text{id} \cdot \text{DFT} \cdot P_2 M_2 \cdot P_1 M_2 \]
and
\[A_3 = \text{ADD}^{1} \cdot \text{ADD}^{2} \cdot \text{ADD}^{3} \cdot \text{ADD}^{4} \cdot \text{ADD}^{5} \cdot \text{ADD}^{6} \cdot \text{ADD}^{7} \cdot \text{ADD}^{8} \cdot \text{ADD}^{9} \]
yields
\[
\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 & 2 & 2 & 0 \\
0 & 0 & 1 & 1 & 1 & 2 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 1 & 1 & 2 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 2 & 2 & 1 & 0 & 2 & 2 & 0 \\
0 & 1 & 1 & 0 & 2 & 1 & 1 & 1 & 0 & 0
\end{pmatrix}
\[A : \]

For the last row, we use the transformations
\[T_4 = \text{id} \cdot \text{id} \cdot P_2 \cdot M_2 \cdot \text{DFT} \cdot P_1 \cdot \text{id} \cdot P_1 M_2 \cdot P_1 M_2 \]

15
and
\[ A_4 = \text{ADD}^{(4;5)} \text{ADD}^{(4;6)} \text{ADD}^{(4;8)} \text{ADD}^{(4;9)} \]

and get
\[
\alpha(x) = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

Note that after the first three steps, we have \( L = f_1;2;3 \). Hence not the third, but the fourth qudit is the control qudit for the ADD gates of the last row. Equivalently, we could have subtracted the third row from the fourth to obtain \( X_{4;3} = 0 \). The stabilizer group would not have changed as the addition of one row to another corresponds to multiplying one generator by another.

Finally, a Fourier transformation on the first four positions transforms the stabilizer group into the form \( S_0 = hZ_1^{(1)};Z_1^{(2)};Z_1^{(3)};Z_1^{(4)} \).

![Diagram of decoding circuit](image)

The complete decoding circuit is shown in Figure 4. Reversing the order of the gates and replacing each gate by its inverse yields a quantum circuit for encoding.

7. Conclusions

The quantum circuits for both CSS codes and general stabilizer codes over qudit systems of prime power dimension resulting from our algorithms have a similar structure. They consist of an alternating sequence of single qudit gates and ADD gates with the same control qudit. The total number of gates is always at most quadratic in the number of qudits. For CSS codes, the single qudit gates are either Fourier transformation or multiplication gates, which are trivial for the case of qubits. Further, some optimizations are possible by interchanging the role of the two classical codes involved.

The algorithm of [6], respectively the modified version of [11, 12], to compute quantum circuits for the encoding of qudit stabilizer codes can also be generalized to non-qubit stabilizer codes. It turns out that the resulting quantum circuits have again the same structure and complexity as those presented in this paper. The
approach presented here, however, has the advantage that the transformation D used to conjugate the stabilizer group S to \( S_0 \) can also be used to compute "encoded gates" which preserve the code C. Conjugating gates from the error-group which preserve the code \( C_0 \) by D results in transformations that are also in the error-group and preserve the code C. As these transformations consist only of single qudit operations, they can be used for fault-tolerant quantum computation.

Acknowledgments

Markus Grassl and Martin Rotteler acknowledge the hospitality of the Mathematical Sciences Research Institute, Berkeley, USA. Part of this work was supported by Deutsche Forschungsgemeinschaft (DFG), Schwerpunktprogramm Quanten-Informationverarbeitung (SPP 1078), Projekt AQUA (Be 887/13) and the IST/FET project Q-ACTA (IST-1999-10596) funded by the European Commission.

References

1. D. Aharonov and M. Ben-Or, Fault-Tolerant Quantum Computation with Constant Error, in Proceedings of the 29th Annual ACM Symposium on Theory of Computing, Association for Computing Machinery, 1997, pp. 176-188. See also LANL preprint quant-ph/9611025.
2. A. Ashikhmin and E. Knill, Nonbinary quantum stabilizer codes, IEEE Transactions on Information Theory, 47 (2001), pp. 3065-3072. See also LANL preprint quant-ph/0005008.
3. C.H. Bennett, D.P. Divincenzo, C.A. Fuchs, T. Mor, E. Rains, P.W. Shor, J.A. Smolin, and W.K. Wootters, Quantum nonlocality without entanglement, Physical Review A, 59 (1999), pp. 1070-1091. See also LANL preprint quant-ph/9804053.
4. A.R. Calderbank, E.M. Rains, P.W. Shor, and N.J.A. Sloane, Quantum Error Correction Via Codes over GF(4), IEEE Transactions on Information Theory, 44 (1998), pp. 1369-1387. See also LANL preprint quant-ph/9608006.
5. A.R. Calderbank and P.W. Shor, Good quantum error-correcting codes exist, Physical Review A, 54 (1996), pp. 1098-1105. See also LANL preprint quant-ph/9512032.
6. R.C. Cleve and D. Gottesman, Efficient computations of encodings for quantum error correction, Physical Review A, 56 (1997), pp. 76-82. See also LANL preprint quant-ph/9607030.
7. P.M. A. Dirac, The Principles of Quantum Mechanics, Clarendon Press, Oxford, 4 ed., 1958.
8. E. Fredkin and T. Toffoli, Conservative Logic, International Journal of Theoretical Physics, 21 (1982), pp. 219-253.
9. D. Gottesman, A Class of Quantum Error-Correcting Codes Saturating the Quantum Hamming Bound, Physical Review A, 54 (1996), pp. 1862-1868. See also LANL preprint quant-ph/9604038.
10. ______, Fault-Tolerant Quantum Computation with Higher-Dimensional Systems, in Quantum Computing & Quantum Communications: First NASA International Conference; selected papers, QCQC'98, C.P. Williams ed., vol. 1509 of Lecture Notes in Computer Science, Palm Springs, February 17-20 1998, Springer, pp. 302-313.
11. M. Grassl, Algorithmic aspects of quantum error-correcting codes, in Mathematics of Quantum Computation, R. K. Brylinski and G. Chen, eds., CRC Press, 2002, pp. 223-252.

12. ————, Fehlerrkorrigierende Codes für Quantensysteme: Konstruktionen und Algorithmen, Shaker, Aachen, 2002. Zugl: Universität Karlsruhe, Dissertation, 2001.

13. M. Grassl and T. Beth, Cyclic quantum error-correcting codes and quantum shift registers, Proceedings of the Royal Society London A, 456 (2000), pp. 2689-2706. See also LANL preprint quant-ph/9910061.

14. M. Grassl, W. Geiselmann, and T. Beth, Quantum Reed-Solomon codes, in Proceedings of the IEEE International Symposium on Information Theory, 2002.

15. P. Horodecki, Separability criterion and inseparable mixed states with positive partial transpose, Physics Letters A, 232 (1997), pp. 333-339. See also LANL preprint quant-ph/9703004.

16. B. Huppert, Endliche Gruppen, Springer, Berlin, 1967.

17. D. Jungnickel, Finite Fields, BI Wissenschaftsverlag, Mannheim, 1993.

18. A. Klappenecker and M. Rotteler, Beyond Stabilizer Codes I: Nice Error Bases, IEEE Transactions on Information Theory, 48 (2002), pp. 2392-2395. See also LANL preprint quant-ph/0010082.

19. E. Knill, Non-Binary Unitary Error Bases and Quantum Codes, Tech. Rep. LAUR-96-2717, LANL, 1996. See also LANL preprint quant-ph/9608048.

20. E. Knill and R. Laflamme, Theory of quantum error-correcting codes, Physical Review A, 55 (1997), pp. 900-911. See also LANL preprint quant-ph/9604034.

21. R. Li and H. Niederreiter, Finite Fields, Encyclopedia of Mathematics and its Applications, Addison-Wesley, Reading, 1983.

22. F. J. MacWilliams and N. J. A. Sloane, The Theory of Error-Correcting Codes, North Holland, Amsterdam, 1977.

23. O. T. O'Meara, Symplectic Groups, American Mathematical Society, Providence, 1978.

24. E. M. Rains, Nonbinary Quantum Codes, IEEE Transactions on Information Theory, 45 (1999), pp. 1827-1832. See also LANL preprint quant-ph/9703048.

25. A. M. Steane, Error Correcting Codes in Quantum Theory, Physical Review Letters, 77 (1996), pp. 793-797.

26. T. Toffoli, Reversible Computing, in Automata, Languages and Programming, J. W. de Bakker and J. van Leeuwen, eds., vol. 85 of Lecture Notes in Computer Science, North-Holland, The Netherlands, 14-18 July 1980, Springer, pp. 632-644.

27. M. A. Tsfasman, S. G. Vladut, and T. Zink, Modular curves, Shimura curves, and Goppa codes, better than Varnamov-Gilbert bound, Mathematische Nachrichten, 109 (1982), pp. 21-28.