# Cyclic Quantum Error-Correcting Codes and Quantum Shift Registers

BY MARKUS GRASSL AND THOMAS BETH

Institut für Algorithmen und Kognitive Systeme Universität Karlsruhe, Am Fasanengarten 5, 76 128 Karlsruhe, Germany.

We transfer the concept of linear feed-back shift registers to quantum circuits. It is shown how to use these quantum linear shift registers for encoding and decoding cyclic quantum error-correcting codes.

Keywords: Quantum error-correcting codes, linear shift registers, quantum computing

## 1. Introduction

Quantum error-correction will be an essential building-block for the physical implementation of a quantum computer since it is unlikely that the coherence time of a quantum mechanical system is long enough to perform any computation of interest, such as factoring large numbers (see Shor 1994). The last years have seen a great progress in the theory of quantum error-correcting codes (see, e.g., Knill & Laflamme 1997; Calderbank et al. 1998). The algorithmic aspect of encoding and decoding, however, has hardly been addressed, yet.

Cleve & Gottesman (1997) gave a general construction for encoding circuits, but not for decoding. Beth & Grassl (1998) illustrated how to derive decoding circuits for quantum error-correcting codes in general. In this paper, we present a technique for encoding and decoding tailored to cyclic quantum error-correcting codes. The resulting quantum circuits are based on the quantum version of linear feed-back shift registers. Hence, these circuits possess a highly regular structure and are especially suited for systems with inherent cyclic symmetries, e.g., circular ion traps. Linear feed-back shift registers fit also to a heterogeneous system—such as optically trapped atoms combined with a cavity—where one part of the system—e.g., the cavity—acts as bus for the feed-back.

The paper is organised as follows: Assuming that the reader is familiar with the concept of quantum computation in general (see, e.g., Berthiaume (1997); Steane (1998a)), we start with an introduction to (classical) cyclic error-correcting codes. Then we present linear shift registers, firstly in their classical, secondly in their quantum version. In §5 quantum circuits for encoding and decoding cyclic quantum-error correcting codes are presented. We conclude with an illustrating example and final remarks.

## 2. Cyclic Codes

In this section we recall some properties of (classical) cyclic codes. A good reference is, e.g., MacWilliams & Sloane (1977).

## (a) Polynomial Description

A cyclic code  $C = [N, K]_q$  of length N and dimension K over a finite field  $\mathbb{F}_q = GF(q)$  is a K-dimensional subspace of  $\mathbb{F}_q^N$  that is invariant under cyclic shifting the coordinates, i.e., for a codeword  $\mathbf{c} = (c_0, \dots, c_{N-1})$ , the cyclic shift  $(c_{N-1}, c_0, \dots, c_{N-2})$  is again a codeword. To any codeword  $\mathbf{c} = (c_0, \dots, c_{N-1})$  we associate the code polynomial  $\mathbf{c}(X) := c_0 + c_1 X + \dots + c_{N-1} X^{N-1} = \sum_i c_i X^i$ . Cyclic shifting the codeword  $\mathbf{c}$  corresponds to multiplication of the polynomial  $\mathbf{c}(X)$  by X and reducing it modulo  $X^N - 1$ . Furthermore, any linear combination of codewords—and thus code polynomials—is again a codeword. Altogether, the code corresponds to an ideal in the ring  $\mathbb{F}_q[X]/(X^N - 1)$ . This ideal is generated by (the residue class of) a polynomial  $\mathbf{c}(X)$  of degree N - K, the generator polynomial of C. Hence, any code polynomial  $\mathbf{c}(X)$  can be written as

$$c(X) = i(X)g(X) \bmod X^{N} - 1.$$
(2.1)

It can be shown that g(X) may be chosen as the unique monic non-zero polynomial of least degree in the code and that g(X) divides  $X^N - 1$ , thus  $g_0 = g(0) \neq 0$ . The set of code polynomials is given by

$$\{c(X) : c \in C\} = \{i(X)g(X) \mid \deg i(X) < K\}.$$
 (2.2)

(b) The Dual of Cyclic Codes

For a linear block code C of length N over a field  $\mathbb{F}_q$ , the dual code  $C^{\perp}$  is given by

$$C^{\perp} := \{ \boldsymbol{v} \in \mathbb{F}_q^N \mid \forall \boldsymbol{c} \in C : \boldsymbol{c} \cdot \boldsymbol{v} = 0 \}.$$

Here  $c \cdot v := \sum_{i} c_i v_i$  is the usual inner product of the vectors c and v.

Obviously, the dual of a cyclic code is cyclic, too. The generator polynomial  $g^{\perp}(X)$  of the dual code is given by

$$g^{\perp}(X) = h_0^{-1} h^{\text{rev}}(X)$$
 where  $g(X)h(X) = X^N - 1$ . (2.3)

(Note that  $h_0 \neq 0$  since  $\boldsymbol{h}(X)|X^N-1$ .) Here  $\boldsymbol{h}^{\text{rev}}(X)$  denotes the reciprocal polynomial of  $\boldsymbol{h}(X) = \sum_i h_i X^i$  obtained by reversing the sequence of coefficients, i.e.,

$$\mathbf{h}^{\text{rev}}(X) := h_0 X^{\operatorname{deg} \mathbf{h}(X)} + \ldots + h_{\operatorname{deg} \mathbf{h}(X)} X^0$$

$$= X^{\operatorname{deg} \mathbf{h}(X)} \mathbf{h}(1/X).$$

(c) The Syndrome of Cyclic Codes

There are several ways to check whether a given vector  $\mathbf{r}$  resp. polynomial  $\mathbf{r}(X)$  is an element of a cyclic code C. From equation (2.1), any code polynomial is a multiple of the generator polynomial  $\mathbf{g}(X)$ . Therefore the syndrome polynomial  $\mathbf{s}(X)$  can be defined as

$$s(X) := r(X) \bmod g(X). \tag{2.4}$$

The syndrome polynomial is zero if and only if  $r \in C$ , and its degree is less than N - K otherwise.

Another way to check whether a polynomial r(X) belongs to a code C generated by g(X) is the following: Recall that  $h(X) = (X^N - 1)/g(X)$  and that every codeword is a multiple of g(X). Hence h(X) can be used as a *check polynomial* with

$$r(X) \in C \iff r(X)h(X) = 0 \mod (X^N - 1).$$
 (2.5)

(d) Weakly Self-Dual Cyclic Codes

The construction of quantum error-correcting codes presented in §5 is based on weakly self-dual classical codes, i.e., codes C with  $C \leq C^{\perp}$ . For cyclic codes, a code  $C_1$  with generator polynomial  $\mathbf{g}_1(X)$  is contained in the code  $C_2$  with generator polynomial  $\mathbf{g}_2(X)$  iff  $\mathbf{g}_2(X)$  divides  $\mathbf{g}_1(X)$ . Thus a cyclic code with generator polynomial  $\mathbf{g}(X)$  is weakly self-dual iff the generator polynomial  $\mathbf{g}^{\perp}(X) = h_0^{-1} \mathbf{h}^{\text{rev}}(X)$  of the dual code divides  $\mathbf{g}(X)$ . In combination with equation (2.3) we get the following identities:

$$\mathbf{g}^{\perp}(X) = h_0^{-1} \mathbf{h}^{\text{rev}}(X); 
\mathbf{g}(X) = \mathbf{g}^{\perp}(X) \tilde{\mathbf{g}}(X); 
X^N - 1 = h_0^{-1} \mathbf{h}^{\text{rev}}(X) \tilde{\mathbf{g}}(X) \mathbf{h}(X).$$
(2.6)

For a cyclic code C = [N, K] of length N and dimension K, the degrees of the polynomials are as follows:

$$\deg \mathbf{g}^{\perp}(X) = K;$$
  

$$\deg \mathbf{g}(X) = N - K;$$
  

$$\deg \tilde{\mathbf{g}}(X) = N - 2K.$$

Next we characterise weakly self-dual cyclic codes in terms of the factorisation of  $X^N-1$  into irreducible polynomials over the field  $\mathbb{F}_q$ . As  $X^N-1$  is (up to a constant) a self-reciprocal polynomial, for any factor f(X) of  $X^N-1$ ,  $f^{\text{rev}}(X)$  is a factor as well. Hence we can write the factorisation of  $X^N-1$  as

$$X^N - 1 = \prod_j \boldsymbol{r}_j(X) \prod_i \boldsymbol{p}_i(X) \prod_i \boldsymbol{p}_i^{\text{rev}}(X)$$

where the polynomials  $r_j(X)$  are the (up to a constant) self-reciprocal factors. From equation (2.7) follows that h(X) and  $h^{\text{rev}}(X)$  have no common factor, hence each of the self-reciprocal polynomials  $r_j(X)$  is a factor of  $\tilde{g}(X)$ , i.e.,

$$\prod_{j} \mathbf{r}_{j}(X) := \mathbf{r}(X) \quad \text{and} \quad \mathbf{r}(X) | \tilde{\mathbf{g}}(X). \tag{2.8}$$

Furthermore, for each i at least one of the polynomials  $p_i(X)$  and  $p_i^{\text{rev}}(X)$  is a factor of g(X).

We conclude this section by a statement about the weights of the codewords of weakly-self dual cyclic binary codes.

**Theorem 2.1.** Any weakly self-dual cyclic binary code of odd length is doubly even, i.e., the weight of any codeword is divisible by four.

*Proof.* The generator polynomial of the code C can be written as

$$g(X) = \sum_{i=1}^{w} X^{d_i}$$
 where  $d_1 = 0 < d_2 < \dots < d_w < N$ . (2.9)

The dual code  $C^{\perp}$  has generator polynomial  $\mathbf{g}^{\perp}(X) = \mathbf{h}^{\text{rev}}(X)$ , and its check polynomial is  $(X^N - 1)/\mathbf{h}^{\text{rev}}(X) = \mathbf{g}^{\text{rev}}(X)$ . From  $C \leq C^{\perp}$  and equation (2.5) we obtain

$$f(X) := g(X)g^{rev}(X) = a(X)(X^N - 1).$$

From equation (2.8) follows that  $r(X)^2|g(X)g^{\text{rev}}(X)$  and thus r(X)|a(X), in particular, (X+1)|a(X). Hence the number of terms in a(X) is even. The degree of g(X) is less than N, and therefore the degree of a(X) is less than N, too. This implies that in the summation  $X^N a(X) - a(X)$  no terms cancel each other, showing that the number of terms in f(X), denoted by #f, is divisible by four.

On the other hand, from equation (2.9), f(X) can be written as

$$f(X) = \sum_{i=1}^{w} X^{d_i} \left( X^{d_w} \sum_{j=1}^{w} X^{-d_j} \right) = X^{d_w} \sum_{i,j=1}^{w} X^{d_i - d_j}.$$
 (2.10)

Again from equation (2.8), we conclude that (X+1)|g(X), and thus the number of terms w of g(X) is even. Hence for i=j all terms  $X^{d_i-d_j}$  in the summation (2.10) cancel each other. For the remaining w(w-1) terms, two terms cancel each other iff  $d_i - d_j = d_k - d_l$ . But then we have also  $d_j - d_i = d_l - d_k$ , so in total four terms are cancelled. Hence  $\#\mathbf{f} = w(w-1) - 4m$  for some integer m. We already know that  $\#\mathbf{f}$  is divisible by four. Therefore w(w-1) must also be divisible by four which implies that w is divisible by four since w-1 is odd.

From equation (2.2) follows that  $\{X^i g(X) : i = 0, \dots, N - d_w - 1\}$  is a vector space basis of the code. The weight of each of these vectors is divisible by four. Being a weakly self-dual code, the inner product of any two codewords is zero, i.e., the number of common ones is even. This implies that the weight of the sum of two codewords which are doubly-even is again divisible by four. (For the last implication see also MacWilliams & Sloane (1977), Ch. 1, §8, Problem (38).)

This theorem shows that all† quantum error-correcting codes derived from weakly self-dual cyclic binary codes are well suited for fault-tolerant quantum computing (cf. Gottesman 1998). This is reflected by the fact that these codes admit the bitwise implementation of the operation  $P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$  (see Steane (1998b), Lemma 4).

### 3. Cyclic Codes and Linear Shift Registers

The basic operations related to cyclic codes are polynomial multiplication and division. Both can be done using linear shift registers.

 $\dagger$  Steane (1998b) observed that the dual of some primitive narrow sense BCH codes turn out to be doubly even. At the CCP workshop at the Isaac Newton Institute, Cambridge, July 1999, he discussed with us the question when a cyclic code is doubly even.

## (a) Polynomial Multiplication

From Horner's rule, the multiplication of a polynomial  $i(X) = \sum_{j=0}^{\mu} i_j X^j$  by the (fixed) polynomial  $g(X) = \sum_{j=0}^{d} g_j X^j$  can be written as

$$i(X)g(X) = \left( \left( (i_{\mu}X + i_{\mu-1})X + \dots \right) X + i_0 \right) g(X)$$
$$= \left( \left( i_{\mu}g(X)X + i_{\mu-1}g(X) \right) X + \dots \right) X + i_0 g(X).$$

Feeding the sequence  $i_{\mu}, i_{\mu-1}, \ldots, i_0, 0, 0, \ldots$  (starting with  $i_{\mu}$ ) into the shift register shown in figure 1 with the register cells initialised with zero, it outputs the coefficients of i(X)g(X), starting with the coefficient of  $X^{d+\mu}$ .

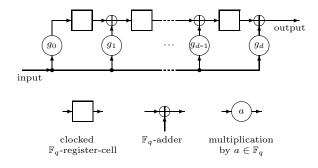


Figure 1. Circuit diagram for a linear feed-forward shift register to multiply the input by g(X).

From equation (2.1) we see that in order to generate a codeword of a cyclic code with generating polynomial g(X), we just multiply a polynomial i(X) by g(X) modulo  $X^N - 1$ . From equation (2.2) follows that the degree of i(X) can be chosen to be less than K. Then, reduction modulo  $X^N - 1$  is not necessary since the degree of the product is less than N. Thus from the circuit shown in figure 1 we can construct a circuit with N register cells that computes c(X) = i(X)g(X) in K steps starting with the initialisation shown in figure 2.

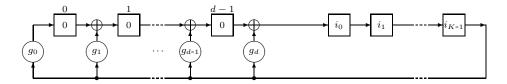


Figure 2. Circuit diagram for encoding a cyclic code of length N and dimension K with generator polynomial g(X).

One single step of the shift register corresponds to the linear mapping given by

$$(r'_0,\ldots,r'_{N-1})=(r_0,\ldots,r_{N-1})\cdot E$$

where

$$E = \begin{pmatrix} 0 & 1 & 0 & \cdots & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ g_0 & g_1 & \dots & g_d & 0 & \dots & 0 & 0 \end{pmatrix}.$$

The matrix E can be factored into a cyclic shift and adding multiples of the first element to several others as follows:

$$E = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} g_0 & g_1 & \cdots & g_d & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & \cdots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & 1 \end{pmatrix}$$

Since the code does not change if we multiply the generator polynomial by a non-zero constant, we can assume without loss of generality  $g_0 = 1$  (note that  $g_0 \neq 0$ ) thereby simplifying the second factor.

The  $K^{\text{th}}$  power of E is given by

$$E^{K} = \begin{pmatrix} & & 1 & & & \\ & & & \ddots & & \\ g_{0} & g_{1} & \cdots & g_{d} & & & \\ g_{0} & g_{1} & \cdots & g_{d} & & & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & g_{0} & g_{1} & \cdots & g_{d} \end{pmatrix}$$

showing that indeed  $(0, \ldots, 0, i_0, \ldots, i_{K-1})E^K = \mathbf{c}$  with  $\mathbf{c}(X) = \mathbf{i}(X)\mathbf{g}(X)$  and thus  $\mathbf{c} \in C$ .

Similarly, it can be shown that for the initialisation  $(j_0, \ldots, j_{d-1}, i_0, \ldots, i_{K-1})$ , after K steps the state of the shift register corresponds to

$$p(X) = i(X)g(X) + X^{K}j(X)$$
(3.1)

where  $j(X) = j_{d-1}X^{d-1} + \ldots + j_0$ .

### (b) Polynomial Division

Similar to shift registers for polynomial multiplication, shift registers can be constructed for polynomial division. The circuit shown in figure 3 implements a polynomial division by a monic polynomial g(X) of degree d. Feeding the sequence  $f_{\mu}, f_{\mu-1}, \ldots, f_0$  (starting with  $f_{\mu}$ ) into the shift register shown in figure 3 with the register cells initialised with zero, it outputs the coefficients of f(X) div g(X),

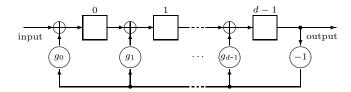


Figure 3. Circuit diagram for a linear feed-back shift register to divide the input by the monic polynomial g(X).

starting with the coefficient of  $X^{\mu-d}$ . After  $\mu+1$  steps, the contents of the register cells are the coefficients of  $f(X) \mod g(X)$ .

To obtain the syndrome of a cyclic code (cf. equation (2.4)), we have to compute the remainder of the polynomial r(X) modulo g(X). Since the degree of r(X) is less than N, we can use the circuit shown in figure 4 with N register cells initialised with  $(r_0, \ldots, r_{N-1})$ . After N steps, the first d = N - K register cells contain the remainder r(X) mod g(X), and the last K registers contain r(X) div g(X).

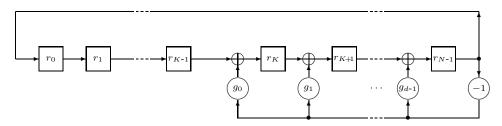


Figure 4. Circuit diagram for computing the quotient r(X) div g(X) and the remainder r(X) mod g(X) of the polynomials r(X) and g(X) of degree less than N and d, resp.

The corresponding matrix is given by

$$S = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 1 \\ 1 & 0 & \dots & 0 & 0 \end{pmatrix} \cdot S_2$$

where

If we are only interested in the remainder and want to keep the original polynomial r(X), a slightly modified version of the previous circuit can be used (cf.

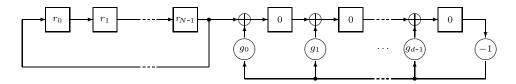


Figure 5. Circuit diagram for syndrome computation for a cyclic code of length N with (monic) generator polynomial g(X) of degree d.

figure 5). After N steps, the first N register cells contain again r(X), and the last d register cells contain  $r(X) \mod g(X)$ . As before, this transformation can be factored into a shift operation—with two disjoint cycles of length N and d—and a simple linear mapping.

# 4. Quantum Shift Registers

In this section we show how the linear shift registers presented in the previous section can be transformed into quantum circuits. For both linear feed-forward shift registers (for polynomial multiplication) and linear feed-back shift registers (for polynomial division) a single basic step can be decomposed into a cyclic shift followed by a linear mapping of the form

First we consider how two implement these mappings for shift registers over the binary field  $\mathbb{F}_2$ , then for shift registers over any field of characteristic two, i.e., over  $\mathbb{F}_{2^k}$ . In this paper, we restrict ourselves to fields of characteristic two—corresponding to qubits—, but the results can easily be generalised to any characteristic p > 0.

#### (a) Binary Quantum Shift Registers

### (i) Cyclic Shifting

For binary shift register in each cell we have the values zero or one. Thus we replace each cell by one quantum bit (qubit). The shift register circuits shown in figures 2, figure 4, and 5 do all operations in place, i.e., have no input and output. Therefore, the state of the whole shift register can be represented by N (resp. N+d) qubits.

The first part of the basic step of a linear shift register is a cyclic shift of the qubits. This corresponds to the permutation  $\pi = (1 \ 2 \dots N)$  which can be written as product of transpositions

$$(1 \ 2 \dots N) = (N-1 \ N) \dots (2 \ 3)(1 \ 2)$$

$$= (1 \ N-1)(2 \ N-2) \dots (i \ N-i) \dots$$

$$\cdot (1 \ N)(2 \ N-1) \dots (i \ N+1-i) \dots$$

$$(4.2)$$

(here the leftmost transposition is applied first). While in the first factorisation there are only transpositions of neighbouring numbers, the second factorisation is a product of two permutations each of which is a product of disjoint transpositions. A transposition of two qubits—a SWAP gate—can be implemented with three controlled not (CNOT) gates as shown in figure 6. (For the graphical notation of quantum operations see, e.g., Barenco et al. 1995.)

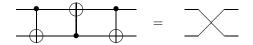


Figure 6. Quantum circuit to swap two qubits.

In figure 7 the circuits corresponding to the factorisations in equation (4.2) and equation (4.3) resp. are presented for seven qubits. Both circuits have the same number of CNOT gates, namely 3(N-1), but the second one has only (constant) depth six if CNOT gates on disjoint sets of qubits can be performed in parallel.

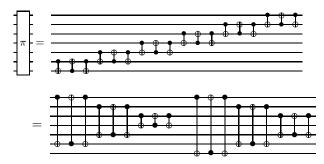


Figure 7. Quantum circuits for cyclic shifting, corresponding to the different factorisations of the permutation  $\pi = (1\ 2\ 3\ 4\ 5\ 6\ 7)$  given in equation (4.2) and equation (4.3) resp.

Note that particular systems may admit simpler implementations of a single SWAP gate (Sanders *et al.* 1999) or the complete cyclic shift.

#### (ii) Linear Feed-Forward/Feed-Back

The second part of the basic step of a shift register is the linear transformation given in equation (4.1). The first register cell is unchanged, while multiples of the contents of the first register cell are added to the other cells. For binary shift registers, either the value of the first register is added or nothing is done. The addition of a binary value can be implemented easily, it corresponds to a CNOT gate. The quantum circuit corresponding to the linear feed-forward shift register for multiplication by the polynomial  $g(X) = X^3 + X + 1$  is shown in figure 8. The shift operation is depicted as a black-box (see figure 7). The two CNOT gates after each shift correspond to the terms  $X^3$  and X in g(X).

An alternate version of this circuit can be obtained if instead of cyclic shifting the qubits, the other operations are shifted and the output qubits are re-labelled, as shown in figure 9. Furthermore, we have combined CNOT gates with the same

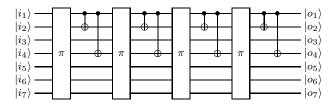


Figure 8. Quantum circuit corresponding to K=4 steps of a quantum linear feed-forward shift register for multiplication by the polynomial  $g(X)=X^3+X+1$ .

control qubit since these gates could be realised with fewer operations, e.g., in a linear ion trap where the control qubit is put on the phonon bus.

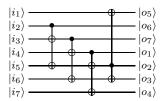


Figure 9. Alternate version of the quantum circuit shown in figure 8.

#### (b) Quantum Shift Registers over Extension Fields

## (i) Finite Fields of Characteristic Two

First, we recall some facts about finite fields (see, e.g., Jungnickel 1993).

Any finite field  $\mathbb{F}_q$  has  $q=p^k$  elements where p is a prime number, the *characteristic* of the field. The smallest subset of  $\mathbb{F}_q$  that is a field is called the *prime field* of  $\mathbb{F}_q$  and has p elements. Conversely, the field  $\mathbb{F}_q$  is an *extension field* of  $\mathbb{F}_p$ . It can be constructed as  $\mathbb{F}_p[X]/(f(X))$  where  $f(X) \in \mathbb{F}_p[x]$  is an irreducible polynomial of degree k. The extension field  $\mathbb{F}_q$  is a vector space of dimension k over  $\mathbb{F}_p$ , and thus possesses a basis of k linearly independent elements. For a fixed basis  $\mathcal{B}$ , any element of  $\mathbb{F}_q$  can be represented by a vector of length k over  $\mathbb{F}_p$ . The multiplication by a fixed element  $a \in \mathbb{F}_q$  is a linear mapping and can thus be written as a  $k \times k$  matrix  $M_{\mathcal{B}}(a)$  over  $\mathbb{F}_p$ . The trace of  $M_{\mathcal{B}}(a)$  is independent of the choice of the basis and defines an  $\mathbb{F}_p$ -linear mapping

$$\operatorname{tr} : \mathbb{F}_q \to \mathbb{F}_p, \quad x \mapsto \operatorname{tr}(x) := \operatorname{tr}(M_{\mathcal{B}}(x)) = \sum_{i=0}^{k-1} x^{p^i}$$

(for the last equality see, e.g., Geiselmann (1994), Satz 1.24).

Finally, we need the definition of the dual basis. Given a basis  $\mathcal{B} = (b_1, \ldots, b_k)$  of a finite field  $\mathbb{F}_q$  as  $\mathbb{F}_p$ -vector space, the *dual basis* is another basis  $\mathcal{B}^{\perp} = (b'_1, \ldots, b'_k)$  with

$$\forall i, j \colon \operatorname{tr}(b_i b'_j) = \delta_{ij}.$$

Such a dual basis exists for any basis, and the dual basis is unique (see Jungnickel (1993), Theorem 4.1.1). A basis that equals its dual basis is called self-dual.

#### (ii) Cyclic Shifting

For binary shift register each cell was represented by one qubit. Fixing a basis  $\mathcal{B}$ , each element of the field  $\mathbb{F}_{2^k}$  can be represented by a binary vector of length k. Hence each cell of the quantum shift register over the field  $\mathbb{F}_{2^k}$  is represented by k qubits. Cyclic shifting over the extension field is implemented similarly to the binary case, but now shifting is performed in parallel in blocks of size k. The complexity increases only by the factor k, i.e., shifting can be done with 3k(N-1) CNOT gates. The parallelised version has again constant depth six.

# (iii) Linear Feed-Forward/Feed-Back

For the second part of the basic step of a shift register we have to implement the linear transformation given in equation (4.1). Multiples of the contents of the first register cell are added to the other cells, i.e., we have to implement the transformations

$$|x\rangle_1|y\rangle_i \mapsto |x\rangle_1|m_ix+y\rangle_i$$

for fixed values  $m_i \in \mathbb{F}_{2^k}$ . Writing the field elements x and y as binary vectors of length k with respect to the basis  $\mathcal{B} = (b_1, \ldots, b_k)$ , the multiplication by  $m_i$  is a linear transformation given by the matrix  $M := M_{\mathcal{B}}(m_i)$ . Now the transformation can be written as

$$m_i x + y = \sum_{j=1}^k \left( \sum_{l=1}^k M_{jl} x_l + y_j \right) b_j$$

where all operations in parentheses are over the binary field. This translates directly into a quantum circuit as demonstrated by the following example.

We consider the field  $\mathbb{F}_{2^3}$  with basis  $\mathcal{B} = (\alpha^3, \alpha^6, \alpha^5)$  where  $\alpha^3 + \alpha + 1 = 0$ . Elements of  $\mathbb{F}_8$  are written as binary column vectors. Multiplication by  $m_2 := \alpha$  corresponds to (left) multiplication of the column vectors by

$$M_{\mathcal{B}}(\alpha) = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}. \tag{4.4}$$

The quantum circuit for the transformation  $|x\rangle|y\rangle \mapsto |x\rangle|\alpha x + y\rangle$  is shown in figure 10. Conditioned on  $x_i$ , the *i*th column of  $M_{\mathcal{B}}(\alpha)$  is added to the vector  $y = (y_1, y_2, y_3)^t$ . The total number of CNOT gates in the circuit is at most  $k + (k-1)^2 = k^2 - k + 1$  since the matrix M is either zero or has full rank which implies that at most one column (resp. row) contains no zero.

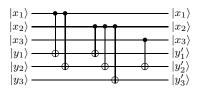


Figure 10. Quantum circuit implementing the transformation  $|x\rangle|y\rangle \mapsto |x\rangle|\alpha x + y\rangle$ .

# 5. Cyclic Quantum Codes

## (a) Binary Codes

We follow the construction of quantum error-correcting codes from weakly self-dual binary codes presented by Calderbank & Shor (1996) and Steane (1996a,b). In the literature, these codes are also referred to as CSS codes.

Given a weakly self-dual linear binary code C = [N, K], the basis states of the corresponding quantum code are given by

$$|\psi_j\rangle = \frac{1}{\sqrt{|C|}} \sum_{c \in C} |c + w_j\rangle$$
 (5.1)

where  $\{\boldsymbol{w}_j: j=1,\ldots,2^{N-2K}\}$  is a system of representatives of the cosets  $C^{\perp}/C$ . For cyclic codes, the vector  $\boldsymbol{c}+\boldsymbol{w}_j$  corresponds to the polynomial  $\boldsymbol{c}(X)+\boldsymbol{w}_j(X)$ . Since  $\boldsymbol{c}\in C$  and  $\boldsymbol{w}_j\in C^{\perp}$ , we have

$$oldsymbol{c}(X) = oldsymbol{i}(X) oldsymbol{g}(X)$$
 and  $oldsymbol{w}_j(X) = oldsymbol{j}(X) oldsymbol{g}^{\perp}(X)$ 

for suitably chosen  $\boldsymbol{i}(X)$  and  $\boldsymbol{j}(X)$ . From equation (2.6) we get  $\boldsymbol{g}(X) = \tilde{\boldsymbol{g}}(X)\boldsymbol{g}^{\perp}(X)$  and thus

$$c(X) + w_j(X) = (i(X)\tilde{g}(X) + j(X))g^{\perp}(X).$$
(5.2)

Combining equations (5.1) and (5.2), we obtain

$$|\psi_j\rangle = \frac{1}{\sqrt{2^K}} \sum_{\text{deg } \boldsymbol{i}(X) \leq K} \left| \left( \boldsymbol{i}(X)\tilde{\boldsymbol{g}}(X) + \boldsymbol{j}(X) \right) \boldsymbol{g}^{\perp}(X) \right\rangle$$
 (5.3)

where for the polynomial  $\mathbf{f}(X) = f_0 + f_1 X + \ldots + f_{N-1} X^{N-1}$ ,  $|\mathbf{f}(X)\rangle$  denotes the state  $|f_0\rangle|f_1\rangle\ldots|f_{N-1}\rangle$ .

As j(X) is a representative of a coset of the code generated by  $\tilde{\boldsymbol{g}}(X)$ , without loss of generality we can reduce j(X) modulo  $\tilde{\boldsymbol{g}}(X)$  and obtain  $\deg j(X) < \deg \tilde{\boldsymbol{g}}(X) = N-2K$ . Hence we get orthogonal basis states  $|\psi_j\rangle$  of the code parameterised by all polynomials j(X) with  $\deg j(X) < N-2K$ . The polynomials  $i(X)\tilde{\boldsymbol{g}}(X)+j(X)$  correspond to elements of the cosets of the cyclic code C. Thus the state  $|\psi_j\rangle$  does not change if we cyclically shift the qubits, i.e., multiply the polynomial by  $X^{N-2K} \mod (X^N-1)$ . Hence equation (5.3) can be written in the (unnormalised) form

$$|\psi_j\rangle = \sum_{\text{deg } \boldsymbol{i}(X) < K} \left| (\boldsymbol{i}(X)\tilde{\boldsymbol{g}}(X) + X^{N-2K}\boldsymbol{j}(X))\boldsymbol{g}^{\perp}(X) \right\rangle$$
 (5.4)

which can be directly translated into an encoding algorithm.

# (b) Encoding and Decoding

# (i) Encoding

First, we show how to encode quantum information using quantum shift registers. The initial state of N-2K qubits is embedded into N qubits as follows:

$$|\phi_0\rangle = \sum_{\deg \mathbf{j}(X) < N-2K} \alpha_j \underbrace{|0\rangle \dots |0\rangle}_K \underbrace{|\mathbf{j}(X)\rangle}_{N-2K} \underbrace{|0\rangle \dots |0\rangle}_K.$$

Hadamard transformation of the last K qubits yields the state

$$|\phi_1\rangle = \sum_{\substack{\deg i(X) < K \\ \deg j(X) < N - 2K}} \alpha_j |\mathbf{0}\rangle |j(X)\rangle |i(X)\rangle,$$

where we have omitted the overall normalisation factor. Using a quantum linear shift register of length N-K (on the last N-K qubits) for the multiplication by  $\tilde{g}(X)$ , we get (cf. equation (3.1))

$$|\phi_2\rangle = \sum_{\substack{\deg i(X) < K \\ \deg j(X) < N-2K}} \alpha_j |\mathbf{0}\rangle |i(X)\tilde{g}(X) + X^{N-2K}j(X)\rangle.$$

Finally, in order to multiply by  $g^{\perp}(X)$  we use a quantum shift register of length N and obtain the desired state (cf. equation (5.4))

$$|\phi_3\rangle = \sum_{\substack{\text{deg } i(X) < K \\ \text{deg } j(X) < N - 2K}} \alpha_j \left| \left( i(X) \tilde{\boldsymbol{g}}(X) + X^{N-2K} \boldsymbol{j}(X) \right) \boldsymbol{g}^{\perp}(X) \right\rangle.$$
 (5.5)

The whole encoding process is sketched in figure 11.

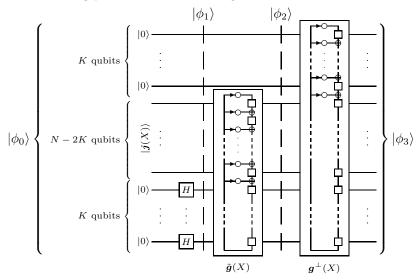


Figure 11. Quantum circuit for encoding a cyclic quantum-error correcting code using quantum linear shift registers for multiplication by  $\tilde{g}(X)$  and  $g^{\perp}(X)$ .

#### (ii) Decoding

The general outline of the decoding procedure for CSS codes is shown in figure 12. First, errors corresponding to tensor products of identity and the Pauli matrix  $\sigma_x$  (bit-flip errors) are corrected. Then, a Hadamard transformation interchanges phase-flip errors (corresponding to  $\sigma_z$ ) with respect to the original basis and bit-flip errors with respect to the transformed basis. For quantum error-correcting

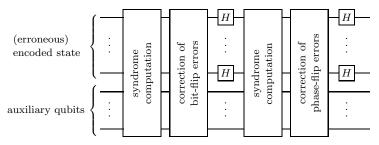


Figure 12. General decoding scheme for a quantum error-correcting code constructed from a weakly self-dual binary code.

codes derived from weakly self-dual binary codes, both steps are essentially the same. Therefore, we describe only the first step.

The error-free state (5.5) is a superposition of codewords of the cyclic code generated by  $\mathbf{g}^{\perp}(X)$ . Hence computing the syndrome  $\mathbf{s}(X) = \mathbf{r}(X) \mod \mathbf{g}^{\perp}(X)$  (cf. equation (2.4)) yields information about the error. For the computation of the remainder  $\mathbf{r}(X) \mod \mathbf{g}^{\perp}(X)$ , we use the quantum version of the linear feed-back shift register shown in figure 5. The degree of  $\mathbf{g}^{\perp}(X)$  is K, therefore we need K auxiliary qubits for the syndrome. The N qubits of the (erroneous) encoded state are successively fed into the shift register, as depicted in figure 13. After N steps, the K auxiliary qubits contain the syndrome of the bit-flip errors. At this point, a classical binary syndrome can be obtained by measuring the K syndrome qubits. Then, the corresponding error can be determined using classical algorithms (e.g., the Berlekamp-Massey algorithm, see MacWilliams & Sloane (1977)). Alternatively, the error may be corrected using quantum operations that are conditioned on the state of the syndrome qubits.

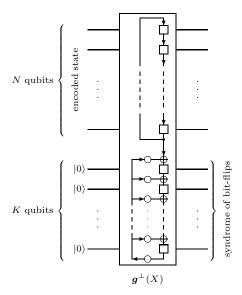


Figure 13. Computing the syndrome of a cyclic quantum error-correcting code using a quantum linear feed-back shift register.

## (c) Codes over Fields of Characteristic Two

Grassl et al. (1999) showed that CSS codes can also be constructed using non-binary classical codes. The main idea is to map a code over an extension field  $\mathbb{F}_{2^k}$  to a code over the prime field  $\mathbb{F}_2$ , as described in the following definition.

**Definition 5.1.** Let  $C = [N, K]_{2^k}$  be a linear code over the field  $\mathbb{F}_{2^k}$  with basis  $\mathcal{B} = (b_1, \dots, b_k)$ .

Then the binary expansion of C with respect to the basis  $\mathcal{B}$  is the linear binary code  $C_2 = [kN, kK]_2$  given by

$$C_2 := \left\{ \left( c_{ij} \right)_{i,j} \in \mathbb{F}_2^{kN} \mid \boldsymbol{c} = \left( \sum_j c_{ij} b_j \right)_i \in C \right\}.$$

The relations between the codes in the previous definition and their duals are reflected by the following theorem.

**Theorem 5.2 (see Grassl et al. 1999).** Let  $C = [N, K]_2$  be a linear code over the field  $\mathbb{F}_{2^k}$  and let  $C^{\perp}$  be its dual. Then the dual code of the binary expansion of C with respect to the basis  $\mathcal{B}$  is the binary expansion of the dual code  $C^{\perp}$  with respect to the dual basis  $\mathcal{B}^{\perp}$ , i.e., the following diagram commutes:

$$\begin{array}{ccc} C & \longrightarrow & C^{\perp} \\ basis \ \mathcal{B} & & & dual \ basis \ \mathcal{B}^{\perp} \\ C_2 & \longrightarrow & C_2^{\perp} \end{array}$$

This theorem shows in particular that the binary code inherits the property of being weakly self-dual from the code over the extension field if the binary expansion is with respect to a self-dual basis (see  $\S4b(i)$ ).

If we start with a weakly self-dual cyclic code over the extension field, the same principles as for cyclic binary codes can be used for encoding and decoding. We just have to replace the quantum linear shift registers over the binary field by shift registers over extension fields (see  $\S4b$  (ii) and (iii)).

### 6. Example

To illustrate the preceding, we present quantum circuits based on quantum shift-registers for quantum Reed-Solomon (QRS) codes (Grassl *et al.* 1999).

We construct a QRS code from a Reed-Solomon code  $C = [7,3,5]_8$  over the field  $\mathbb{F}_8$ . The generator polynomial is

$$g(X) = (X - \alpha^0)(X - \alpha^1)(X - \alpha^2)(X - \alpha^3),$$

where  $\alpha^3 + \alpha + 1 = 0$  as above. The dual code  $C^{\perp} = [7, 4, 4]_8$  is generated by

$$g^{\perp}(X) = (X - \alpha^{-4})(X - \alpha^{-5})(X - \alpha^{-6})$$
  
=  $(X - \alpha^3)(X - \alpha^2)(X - \alpha^1)$   
=  $\alpha^6(\alpha X^3 + X^2 + \alpha^2 X + 1).$ 

Hence  $C \leq C^{\perp}$  and  $g(X) = (X - 1)g^{\perp}(X)$ .

As self-dual basis of  $\mathbb{F}_8$  over  $\mathbb{F}_2$  we choose  $\mathcal{B} = (\alpha^3, \alpha^6, \alpha^5)$ . The binary expansions of C and  $C^{\perp}$  yield binary codes  $C_2 = [21, 9, 8]_2$  and  $C_2^{\perp} = [21, 12, 5]_2$ . Thus the  $\mathcal{QRS}$  code has parameters  $\mathcal{C} = [[21, 3, 5]]$ .

The encoding circuit shown in figure 14 has the same structure as that in figure 11. First, the 3-qubit state ('q-octet')  $|\phi\rangle$  is embedded into 21 qubits (or 7 q-octets) forming the state  $|\phi_0\rangle$ . Next, three steps of the quantum shift register for the multiplication by  $\tilde{\boldsymbol{g}}(X) = X + 1$  follow. In figure 14, the shift operation is depicted by a permutation of the lines representing the qubits. Finally, we have four steps of the quantum shift register for the multiplication by  $\boldsymbol{g}^{\perp}(X)$ . We make the normalisation  $g_0 = 1$  and obtain  $\boldsymbol{g}^{\perp}(X) = \alpha X^3 + X^2 + \alpha^2 X + 1$ . The matrices corresponding the multiplication by the non-trivial coefficients of  $\boldsymbol{g}^{\perp}(X)$  are given by

$$M_{\mathcal{B}}(\alpha) = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$
 and  $M_{\mathcal{B}}(\alpha^2) = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$ .

As in equation (4.4) and figure 10, the structure of this matrices is reflected by the quantum circuit.

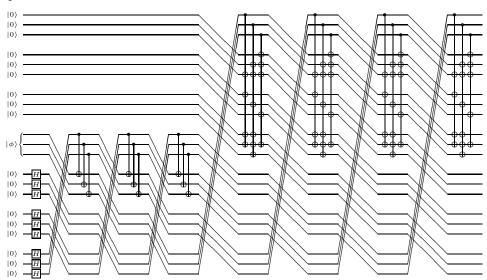


Figure 14. Encoder for the quantum Reed-Solomon code [[21, 3, 5]] using quantum shift registers for the multiplication by  $\tilde{g}(X) = X + 1$  and  $g^{\perp} = \alpha X^3 + X^2 + \alpha^2 X + 1$ .

The quantum circuit in figure 14 strictly follows the concept of cyclic shifting and linear feed-back. Hence it is highly structured. On the other hand, if shifting cannot be implemented easily, we can re-shuffle the circuit and simplify it by combining all shift operations to a permutation of the input (see figure 15).

## 7. Conclusion

In this paper, we presented new methods for encoding and decoding cyclic quantum error-correcting codes based on quantum linear shift registers. They may ease the physical implementation of quantum computers.

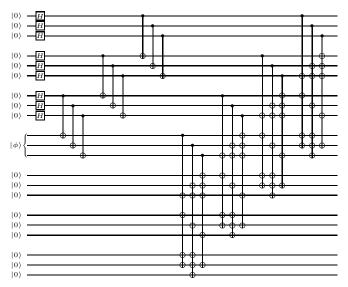


Figure 15. Alternate version of the encoder shown in figure 14.

Classically, linear feed-back shift registers are also used to produce pseudo random sequences for cryptographic purposes. Hence, it is worthwhile to investigate the cryptographic properties of quantum states produced by quantum linear feed-back shift registers (QLFSR).

Another application of (classical) linear shift registers is the area of convolutional codes. Therefore, the quantum version of linear shift registers might prove useful in the context of quantum convolutional codes (Chau 1998), too.

The authors would like to thank Willi Geiselmann for numerous stimulating discussions during the process of writing this paper. We are indebted to Rainer Steinwandt for his critical comments to preliminary versions of this paper. Part of this work was supported by Deutsche Forschungsgemeinschaft (DFG), Schwerpunktprogramm Quanten-Informationsverarbeitung (SPP 1078), Projekt AQUA (Be 887/13-1).

## References

Barenco, A., Bennett, C. H., Cleve, R., DiVincenzo, D. P., Margolus, N., Shor, P., Sleator, T., Smolin, J. A. & Weinfurter, H. 1995 Elementary gates for quantum computation. Phys. Rev. A 52, 3457–3467. (See also LANL preprint quant-ph/9503016).

Berthiaume, A. 1997 Quantum computation. In *Complexity Theory Retrospective II* (ed. L. A. Hemaspaandra & A. L. Selman), pp. 23–51. New York: Springer.

Beth, T. & Grassl, M. 1998 The quantum Hamming and hexacodes. Fortschr. Phys. 46, 459–491.

Calderbank, A. R. & Shor, P. W. 1996 Good quantum error-correcting codes exist. *Phys. Rev. A* 54, 1098–1105. (See also LANL preprint quant-ph/9512032).

Calderbank, A. R., Rains, E. M., Shor, P. W. & Sloane, N. J. A. 1998 Quantum error correction via codes over GF(4). *IEEE Trans. Inf. Theory* **IT-44**, 1369–1387. (See also LANL preprint quant-ph/9608006).

Chau, H. F. 1998 Quantum convolutional codes. *Phys. Rev. A* 58, 905–909. (See also LANL preprint quant-ph/9712029).

- Cleve, R. & Gottesman, D. 1997 Efficient computations of encodings for quantum error correction. *Phys. Rev. A* **56**, 76–82.
- Geiselmann, W. 1994 Algebraische Algorithmenentwicklung am Beispiel der Arithmetik in endlichen Körpern. Aachen: Shaker. (Zugleich Dissertation, Universität Karlsruhe, 1993.).
- Gottesman, D. 1998 Theory of fault-tolerant quantum computation. *Phys. Rev. A* 57, 127–137. (See also LANL preprint quant-ph/9702029.).
- Grassl, M., Geiselmann, W. & Beth, T. 1999 Quantum Reed-Solomon codes. In *Proceedings Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC-13)* (ed. M. Fossorier, H. Imai, S. Lin & A. Poli), Lecture Notes in Computer Science, no. 1719, Honolulu, Hawaii, November 15–19 1999.
- Jungnickel, D. 1993 Finite fields. Mannheim: BI-Wissenschaftsverlag.
- Knill, E. & Laflamme, R. 1997 Theory of quantum error-correcting codes. *Phys. Rev.*  $A\,55,\,900-911.$  (See also LANL preprint quant-ph/9604034).
- MacWilliams, F. J. & Sloane, N. J. A. 1977 The theory of error-correcting codes. Amsterdam: North-Holland.
- Sanders, G. D., Kim, K. W. & Holton, W. C. 1999 Quantum computing with complex instruction sets. Phys. Rev. A 59, 1098–1101.
- Shor, P. W. 1994 Algorithms for quantum computation: discrete logarithm and factoring. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134. IEEE Computer Society Press, November 1994. (See also LANL preprint quant-ph/9508027).
- Steane, A. 1996a Error correcting codes in quantum theory. *Phys. Rev. Lett.* **77**, 793–797. Steane, A. 1996b Multiple particle interference and quantum error correction. *Proc. R. Soc. A* **452**, 2551–2577. (See also LANL preprint quant-ph/9601029).
- Steane, A. 1998a Quantum computing. Rep. Progr. Phys. 61, 117–173. (See also LANL preprint quant-ph/9708022).
- Steane, A. M. 1998b Efficient fault-tolerant quantum computing. LANL preprint quant-ph/9809054.