

Efficient algorithm for computing syndrome polynomial in Reed-Solomon decoder

Elena Costa¹, Sergei Fedorenko², Peter Trifonov²

¹ Siemens AG, Germany, elena.costa@siemens.com

² St.Petersburg State Polytechnic University, Russia, {sfedorenko,ptrifonov}@iee.org

Abstract

Application of the cyclotomic Fast Fourier Transform algorithm to the syndrome evaluation problem in classical Reed-Solomon decoders is described. A number of complexity reduction tricks is suggested. Application of the algorithm leads to significant reductions in the complexity of syndrome evaluation. Moreover, automatic generation of the program code implementing the described algorithm is possible.

1 Introduction

One of the most time-consuming steps in classical decoding of Reed-Solomon codes is evaluation of the syndrome vector. It is well-known that the Fast Fourier Transform (FFT) can be used to compute it [3], but practical implementation of this idea meets certain difficulties. The problem is that most existing FFT algorithms are inefficient if only a small fraction of Discrete Fourier Transform (DFT) components needs to be computed, which is the case of syndrome evaluation.

In this paper we propose the application of the cyclotomic FFT algorithm [5] to this problem. The structure of the cyclotomic FFT allows one to efficiently evaluate partial Fourier transforms leading to dramatic reductions in the complexity. However, it must be recognized, that the application of the suggested algorithm makes sense only if the whole word to be decoded is fed into the decoder simultaneously, not symbol-by-symbol. Such situation occurs, for example, in the decoding of a Reed-Solomon outer code concatenated with some sufficiently long inner code, as specified in e.g. CCSDS 101.0-B-4 and IEEE 802.16 standard [4].

The paper is organized as follows. In section 2 the cyclotomic FFT algorithm is reviewed. Section 3 describes its application to the syndrome evaluation problem. Section 4 presents an example illustrating the developed techniques. Finally, conclusions are drawn in Section 5.

2 Cyclotomic FFT

The cyclotomic FFT algorithm [5] is based on some properties of linearized polynomials.

Definition 1: A polynomial $L(y)$ over $GF(2^m)$ is called linearized if

$$L(y) = \sum_i L_i y^{2^i}, \quad L_i \in GF(2^m).$$

It can be easily seen that $L(a+b) = L(a) + L(b)$ holds for linearized polynomials. This property leads to the following Lemma, presented here in a slightly modified form with respect to that in [2].

Lemma 1: Let $x \in GF(2^m)$ and let $\beta_0, \beta_1, \dots, \beta_{m-1}$ be a basis of the field.

$$\text{If } x = \sum_{i=0}^{m-1} x_i \beta_i, \quad x_i \in GF(2), \quad \text{then}$$

$$L(x) = \sum_{i=0}^{m-1} x_i L(\beta_i).$$

Let us consider cyclotomic cosets modulo $n = 2^m - 1$ over $GF(2)$:

$$\begin{aligned} & \{0\} \\ & \{k_1, k_1 2, k_1 2^2, \dots, k_1 2^{m_1-1}\}, \\ & \dots, \\ & \{k_l, k_l 2, k_l 2^2, \dots, k_l 2^{m_l-1}\}, \end{aligned}$$

where $k_i \equiv k_i 2^{m_i} \pmod{n}$.

Then any polynomial $f(x) = \sum_{i=0}^{n-1} f_i x^i, f_i \in GF(2^m)$ can be decomposed as

$$f(x) = \sum_{i=0}^l L_i(x^{k_i}), \quad \text{where } L_i(y) = \sum_{j=0}^{m_i-1} f_{k_i 2^j \pmod{n}} y^{2^j}. \quad (1)$$

In fact, (1) represents a way of grouping indices $0 \leq i < n$ of $f(x)$ terms into cyclotomic cosets: $i \equiv k_s 2^j \pmod{n}$. Obviously, this decomposition is always possible. Note, that the term f_0 can be represented as $L_0(x^0)$, where $L_0(y) = f_0 y$.

Let us now consider the problem of computing the DFT of a polynomial $f(x)$, i.e. computing values $f(\alpha^j) = \sum_{i=0}^{n-1} f_i \alpha^{ij}, j = 0..n-1$, where α is a primitive element of $GF(2^m)$. According to (1), $f(\alpha^j)$ can be represented as $f(\alpha^j) = \sum_{i=0}^l L_i(\alpha^{j k_i})$. It is known

[2], that α^{k_i} is a root of a minimal polynomial of degree $m_i \mid m$, and thus belongs to a subfield $GF(2^{m_i})$. Thus all the values $(\alpha^{k_i})^j$ lie in $GF(2^{m_i})$ and so they can be decomposed in some basis $(\beta_{i,0}, \dots, \beta_{i,m_i-1})$ of the subfield: $\alpha^{j k_i} = \sum_{s=0}^{m_i-1} a_{ijs} \beta_{i,s}$, $a_{ijs} \in GF(2)$. Then, according to the Lemma 1,

$$\begin{aligned} F_j &= f(\alpha^j) = \sum_{i=0}^l \sum_{s=0}^{m_i-1} a_{ijs} L_i(\beta_{i,s}) \\ &= \sum_{i=0}^l \sum_{s=0}^{m_i-1} a_{ijs} \left(\sum_{p=0}^{m_i-1} \beta_{i,s}^{2^p} f_{k_i 2^p} \right). \end{aligned} \quad (2)$$

This equation can be represented in matrix form as

$$F = ALf, \quad (3)$$

where F and f are vectors consisting of some permutations of elements F_j and f_i , respectively, A is a matrix with elements $a_{ijs} \in GF(2)$ and L is a block diagonal matrix with elements $\beta_{i,s}^{2^p}$.

It is possible to choose the same basis for all the linearized polynomials of the same degree m_i in (1) and obtain a very small amount of different blocks in the matrix L . This can simplify the problem of constructing of a fast algorithm for multiplication of the matrix L by a vector f over $GF(2^m)$. Moreover, if one chooses the normal basis in (2), then all the blocks of the matrix L are circulant matrices. Thus, the multiplication by this matrix can be considered as a problem of computing a set of cyclic convolutions of degree $m_i \mid m$. Since a lot of efficient algorithms for computing cyclic convolutions of various lengths are known, the complexity of the FFT algorithm is significantly reduced. For computing the product Af one can use either the "Four Russians" algorithm [1] or a computer-optimized sequence of additions.

3 Computing the syndrome polynomial

In this section, we will solve the problem of computing the syndrome polynomial for classical RS codes by applying the cyclotomic FFT algorithm in an efficient way. Let $S(x) = \sum_{i=0}^{2t-1} S_i x^i$ be the syndrome polynomial, where t is the number of errors correctable by the code. It is well-known [3], that its coefficients can be computed as

$$S_i = D(\alpha^i), i = 0..2t - 1$$

where $D(x) = \sum_{i=0}^{n-1} D_i x^i$ is the polynomial corresponding to the data vector to be decoded.

One clearly recognize that computing S_i corresponds to evaluating the partial Discrete Fourier Transform of $D(x)$. However, the direct application of the cyclotomic algorithm (3) would require evaluation of all cyclic convolutions, so that the algorithm would have the same number of multiplications as for the case of complete DFT.

Since both matrices A and L are invertible, one can derive from (3) the following representation of the inverse DFT:

$$f = L^{-1} A^{-1} F. \quad (4)$$

By recalling that the direct and inverse Fourier transforms differ only by a fixed permutation and by observing that the inverse of the matrix L is also a block diagonal matrix consisting of circulants, we can derive that (4) does also represent an FFT algorithm. However, in this case if one needs to evaluate only a fraction of the vector f components, it is sufficient to perform multiplications only by those blocks of L^{-1} which occupy the corresponding rows of this matrix. This dramatically reduces the overall number of multiplications. Moreover, one does not need to compute the whole product $A^{-1}F$, but only the elements corresponding to the required blocks of L^{-1} should be evaluated. This is equivalent to truncating the matrix A^{-1} , thus reducing the overall number of additions.

Most existing cyclic convolution algorithms for computing $c(x) = a(x)b(x) \bmod x^m - 1$ can be represented as

$$\begin{pmatrix} c_0 \\ c_1 \\ \dots \\ c_{m-1} \end{pmatrix} = P \left[S_1 \begin{pmatrix} a_0 \\ a_1 \\ \dots \\ a_{m-1} \end{pmatrix} \cdot S_2 \begin{pmatrix} b_0 \\ b_1 \\ \dots \\ b_{m-1} \end{pmatrix} \right],$$

where P, S_1, S_2 are some binary *postsummations* and *presummations* matrices and $x \cdot y$ denotes component-wise multiplication of vectors x and y . Hence, (4) can be rewritten as

$$f = P' [(S'_1 \Gamma) \cdot (S'_2 A^{-1} f)], \quad (5)$$

where P', S'_1 and S'_2 are combined post- and presummation matrices, and $\Gamma = (\beta_{0,0}, \dots, \beta_{0,m_0-1}, \beta_{1,0}, \dots, \beta_{1,m_1-1}, \dots)^T$ is combined vector of basis elements $\beta_{i,s}$. Clearly, vector $C = S'_1 \Gamma$ can be computed beforehand. Since most cyclic convolution algorithms have a number of rows in S_1 consisting only of 1's, multiplication of Γ by these rows would lead to $\sum_{s=0}^{m_i-1} \beta_{i,s}$. If $\{\beta_{i,s}\}$ represent a normal basis of the finite field $GF(2^{m_i})$, this quantity is always equal to 1, so that one does not actually need to perform some multiplications in (5). Moreover, if one computes partial DFT, it is not necessary to perform multiplications by all rows of P' . By striking out these rows, a number of columns in P' become zero columns, which implies in turn that one does not need to compute some products in square brackets in (5). Furthermore, by changing the order of the basis elements $\beta_{i,s}$ one can alter the number of non-zero columns remaining after striking out unused rows and, thus, the number of multiplications to be eliminated. Note that one can easily check all basis reorderings and find the best one minimizing the number of multiplications.

This optimization is possible due to the fact that cyclic convolution algorithms which are proved to be optimal (such as Winograd ones [3]) are not optimal anymore if one computes only a fraction of cyclic convolution components. For example, the 4-point cyclic convolution algorithm in [3] has the following postsumations matrix:

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

If one strikes out the two middle rows, only two zero columns are obtained, but by striking out the first two rows it is possible to obtain three zero columns.

Note that the described method for constructing FFT (or syndrome evaluation) algorithm does not require any manual optimizations and thus can be implemented in a computer program. In fact, all examples presented in this paper were constructed automatically by such a program.

4 Example

This section presents a small example illustrating the techniques described above.

A polynomial $f(x) = \sum_{i=0}^6 f_i x^i$, $f_i \in GF(2^3)$ can be represented as

$$\begin{aligned} f(x) &= L_0(x^0) + L_1(x) + L_2(x^3) \\ L_0(y) &= f_0 y \\ L_1(y) &= f_1 y + f_2 y^2 + f_4 y^4 \\ L_2(y) &= f_3 y + f_6 y^2 + f_5 y^4. \end{aligned}$$

Let $(\gamma, \gamma^2, \gamma^4), \gamma = \alpha^3$ be a normal basis of $GF(2^3)$, where α is a root of the primitive polynomial $x^3 + x + 1$. Then the Discrete Fourier Transform of $f(x)$ can be represented as

$$\begin{aligned} f(\alpha^0) &= L_0(\alpha^0) + L_1(\alpha^0) + L_2(\alpha^0) \\ &= L_0(1) + L_1(\gamma) + L_1(\gamma^2) + L_1(\gamma^4) + \\ &\quad L_2(\gamma) + L_2(\gamma^2) + L_2(\gamma^4) \\ f(\alpha^1) &= L_0(\alpha^0) + L_1(\alpha) + L_2(\alpha^3) \\ &= L_0(1) + L_1(\gamma^2) + L_1(\gamma^4) + L_2(\gamma) \\ f(\alpha^2) &= L_0(\alpha^0) + L_1(\alpha^2) + L_2(\alpha^6) \\ &= L_0(1) + L_1(\gamma) + L_1(\gamma^4) + L_2(\gamma^2) \\ f(\alpha^3) &= L_0(\alpha^0) + L_1(\alpha^3) + L_2(\alpha^2) \\ &= L_0(1) + L_1(\gamma) + L_2(\gamma) + L_2(\gamma^4) \\ f(\alpha^4) &= L_0(\alpha^0) + L_1(\alpha^4) + L_2(\alpha^5) \\ &= L_0(1) + L_1(\gamma) + L_1(\gamma^2) + L_2(\gamma^4) \\ f(\alpha^5) &= L_0(\alpha^0) + L_1(\alpha^5) + L_2(\alpha) \\ &= L_0(1) + L_1(\gamma^4) + L_2(\gamma^2) + L_2(\gamma^4) \\ f(\alpha^6) &= L_0(\alpha^0) + L_1(\alpha^6) + L_2(\alpha^4) \\ &= L_0(1) + L_1(\gamma^2) + L_2(\gamma) + L_2(\gamma^2). \end{aligned}$$

These equations can be rewritten in matrix form as

$$\begin{aligned} F &= \begin{pmatrix} F_0 \\ F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} L_0(1) \\ L_1(\gamma) \\ L_1(\gamma^2) \\ L_1(\gamma^4) \\ L_2(\gamma) \\ L_2(\gamma^2) \\ L_2(\gamma^4) \end{pmatrix} = \\ A &\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \gamma^1 & \gamma^2 & \gamma^4 & 0 & 0 & 0 \\ 0 & \gamma^2 & \gamma^4 & \gamma^1 & 0 & 0 & 0 \\ 0 & \gamma^4 & \gamma^1 & \gamma^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \gamma^1 & \gamma^2 & \gamma^4 \\ 0 & 0 & 0 & 0 & \gamma^2 & \gamma^4 & \gamma^1 \\ 0 & 0 & 0 & 0 & \gamma^4 & \gamma^1 & \gamma^2 \end{pmatrix} \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_4 \\ f_3 \\ f_6 \\ f_5 \end{pmatrix} = ALf. \end{aligned}$$

Note that each non-zero block of the second matrix is circulant.

By inverting matrices A and L , the following Inverse Fourier Transform algorithm can be obtained:

$$\begin{aligned} f &= \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_4 \\ f_3 \\ f_6 \\ f_5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \gamma^1 & \gamma^2 & \gamma^4 & 0 & 0 & 0 \\ 0 & \gamma^2 & \gamma^4 & \gamma^1 & 0 & 0 & 0 \\ 0 & \gamma^4 & \gamma^1 & \gamma^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \gamma^1 & \gamma^2 & \gamma^4 \\ 0 & 0 & 0 & 0 & \gamma^2 & \gamma^4 & \gamma^1 \\ 0 & 0 & 0 & 0 & \gamma^4 & \gamma^1 & \gamma^2 \end{pmatrix} \times \\ &\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} F_0 \\ F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \end{pmatrix} = L^{-1}A^{-1}F. \end{aligned}$$

Since direct and inverse DFT are symmetrical, we obtain the following expression for computing the DFT:

$$\tilde{F} = \begin{pmatrix} F_0 \\ F_6 \\ F_5 \\ F_3 \\ F_4 \\ F_1 \\ F_2 \end{pmatrix} = L^{-1}A^{-1} \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \end{pmatrix} = L^{-1}A^{-1}\tilde{f}, \quad (6)$$

where \tilde{F} and \tilde{f} are some permutations of F_j and f_i . The following algorithm [3] can be used to compute the 3-point cyclic convolution $b_i(x) = b_{i,0} + b_{i,2}x + b_{i,1}x^2 =$

$(\gamma + \gamma^4 x + \gamma^2 x^2)(a_{i,0} + a_{i,1}x + a_{i,2}x^2) \bmod (x^3 - 1)$:

$$b_i = \begin{pmatrix} b_{i,0} \\ b_{i,1} \\ b_{i,2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix} \times \left(\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} \gamma \\ \gamma^4 \\ \gamma^2 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} a_{i,0} \\ a_{i,1} \\ a_{i,2} \end{pmatrix} \right) \\ = P(C_i \cdot (S_2 a_i)), \quad i = 1, 2.$$

Since $\gamma + \gamma^2 + \gamma^4 = 1$, one can clearly see that multiplication of a vector by each block $\begin{pmatrix} \gamma^1 & \gamma^2 & \gamma^4 \\ \gamma^2 & \gamma^4 & \gamma^1 \\ \gamma^4 & \gamma^1 & \gamma^2 \end{pmatrix}$ requires 3 multiplications, 4 pre- and 5 postsummations over $GF(2^3)$. By combining this algorithm with (6) one can obtain

$$\begin{pmatrix} F_0 \\ F_6 \\ F_5 \\ F_3 \\ F_4 \\ F_1 \\ F_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \times \left(\begin{pmatrix} 1 \\ 1 \\ \gamma^2 + \gamma^4 \\ \gamma + \gamma^4 \\ \gamma + \gamma^2 \\ 1 \\ \gamma^2 + \gamma^4 \\ \gamma + \gamma^4 \\ \gamma + \gamma^2 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \end{pmatrix} \right).$$

It can be easily shown that this algorithm requires 6 multiplications and 24 additions and appears to be the best known 7-point FFT for $GF(2^3)$.

If one needs to evaluate only F_0 and F_1 (syndrome components for (7, 5, 3) Reed-Solomon code over $GF(2^3)$), then the above expression reduces to

$$\begin{pmatrix} F_0 \\ F_1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \times \left(\begin{pmatrix} 1 \\ 1 \\ \gamma^2 + \gamma^4 \\ \gamma + \gamma^4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \end{pmatrix} \right).$$

This leads to the following algorithm for computing

two components of DFT:

$$\begin{aligned} T_0 &:= f_5 + f_6; & T_7 &:= f_3 + T_3 \\ T_1 &:= f_2 + f_4; & T_8 &:= T_1 + T_6 \\ T_2 &:= f_0 + f_3; & T_9 &:= f_5 + T_7 \\ T_3 &:= f_1 + f_4; & T_{10} &:= \alpha T_9 \\ T_4 &:= T_0 + T_2; & T_{11} &:= \alpha^2 T_5 \\ T_5 &:= T_0 + T_1; & T_{12} &:= T_{10} + T_{11} \\ T_6 &:= f_1 + T_4; \end{aligned}$$

$$F_0 := T_8; \quad F_1 := T_4 + T_{12}$$

Note that if we change the order of normal basis elements (e.g., $(\gamma^2, \gamma^4, \gamma)$) this would cause the order of rows in matrix P to change. Since its rows are in general not symmetric, it is possible to find an order of elements for which the rows in the required positions are such that the number of zero columns in them is maximal, thus minimizing the total number of multiplications. The above example does not illustrate this effect, but by studying matrix representation of cyclic convolution algorithms presented in [3] one can find that complexity savings due to this effect may be very significant.

Since this is very simple example, it does not show any advantage compared to the conventional syndrome evaluation methods. However, it demonstrates the main ideas of the proposed method:

- 1) Construction of the inverse cyclotomic FFT algorithm.
- 2) Elimination of some multiplications by appropriate selection of normal basis ordering.

This method has been used to construct an algorithm for computing the syndrome polynomial for (255, 239, 17) Reed-Solomon code (this code is specified in e.g. IEEE 802.16a standard) with 75 multiplications and 2938 additions, while the application of the Horner rule to the same problem requires $15 \times 254 = 3810$ multiplications and $15 \times 254 + 254 = 4064$ additions. FFTDesigner program by T. Zakharova, which is based on the development presented in [6], produces for the same problem an algorithm with 85 multiplications and 2989 additions. Simulation results show that the algorithm based on the approach described in this paper is 8 times faster than the classical Horner rule applied to syndrome evaluation problem and about 7% faster than the algorithm based on [6].

5 Conclusions

In this paper we have presented an algorithm for computing syndrome polynomial required by classical Reed-Solomon decoders. The algorithm is based on cyclotomic FFT and has much smaller complexity than conventional syndrome evaluation techniques. It also

allows automatic construction of highly-optimized program code. It must be recognized that the application of the algorithm makes sense only if all symbols of the word to be decoded are supplied simultaneously to the syndrome evaluation block. We have to note that this is a quite common situation, e.g. occurring in the decoders of Reed Solomon codes concatenated with some other codes.

References

- [1] A. Aho, J. Hopcroft, and J. Ullman. *The Design and analysis of computer algorithms*. Addison-Wesley, 1976.
- [2] E.R. Berlekamp. *Algebraic coding theory*. New York: McGraw-Hill, 1968.
- [3] R. Blahut. *Theory and Practice of Error Control Codes*. Addison-Wesley, 1984.
- [4] C. Eklund, R. B. Marks, K. L. Stanwood, and S. Wang. IEEE standard 802.16: a technical overview of the WirelessMAN air interface for broadband wireless access. *IEEE Communications Magazine*, 40(6):98–107, June 2002.
- [5] P.V. Trifonov and S.V. Fedorenko. A method for fast computation of the Fourier transform over a finite field. *Problems of Information Transmission*, 39(3):231–238, July-September 2003. Translation of Problemy Peredachi Informatsii.
- [6] T. G. Zakharova. Application of the Fourier transform in Reed-Solomon decoding. *Radiotechnics*, (12):55–57, 1996. In Russian.