On Efficient Recovery of Erased Symbols in Generalized Reed–Solomon Codes

Joschi Brauchle
Institute for Communications Engineering
Technische Universität München
80290 Munich, Germany
Email: joschi.brauchle@tum.de

***Abstract*—A method for efficient recovery of erased symbols in (generalized) Reed–Solomon, BCH and alternant codes is presented. By exploiting the structure of Vandermonde-based parity-check matrices, an expression for direct calculation of the erased code symbols is found as well as a practical circuitry of the proposed method is designed. Applications of this method include systematic encoding with arbitrary parity positions as well as erasure decoding of such codes.**

I. Introduction

More than half a century after their invention by Irving Reed and Gus Solomon [1] in 1960, Reed–Solomon (RS) codes are still being used in many modern communication systems and standards. In the last decade, joining their countless applications in lower layer FEC coding, most notable in deep space communications of NASA and ESA space exploration missions, magnetic and optical storage systems like hard-disks and CDs/DVDs/Blu-ray and digital broadcasting systems like DVB, they are also more and more applied in higher layer applications to provide strong protection against packet loss due to their MDS property and non-binary nature. Some of these new applications include multimedia multicast communications, e.g., in DVB-H, DVB-SH [2] and ATSC–M/H [3], and fault tolerant distributed storage systems like RAID-6 [4] or ClusterRAID deployed at the CERN Large Hadron Collider (LHC) [5].

In DVB-H and DVB-SH systems for example, Reed–Solomon codes are used as a part of the multiprotocol encapsulation forward error correction (MPE-FEC standard [2]) to code over interleaved IP datagrams on the transmitter side. If IP datagram packets are lost during transmission, e.g., when lower level FEC fails due to a deep signal fade, the receiver can apply erasure decoding in the application layer to easily recover these packets up to the theoretical limit.

In another application, the RS encoder of the ATSC Mobile/Handheld Digital Television Standard (ATSC–M/H) [3] features a special mode of operation, where parity symbols are no longer restricted to form a consecutive block at the beginning or end of the codeword, but may appear at arbitrary positions within the codeword. Due to backward compatibility requirements to ATSC legacy receivers, symbol interleaving after traditional systematic encoding cannot be applied, as the resulting vectors are not valid codewords of the original RS code. Thus, the Reed–Solomon encoder must be able to directly produce these codewords according to any arbitrary pattern of parity positions.

The underlying problem is the same in both applications: Efficient recovery of erased or unknown symbols at arbitrary positions within the codeword. In this report, a method that solves this problem efficiently for codes based on Vandermonde parity-check matrices, including generalized Reed–Solomon, BCH and alternant codes is presented. Hence, it can be applied both in encoding with arbitrary parity positions and erasure decoding of such codes. First, a description of generalized Reed–Solomon codes is given in Section II. Then in Section III a modified parity-check matrix is derived, which then helps in finding an expression to efficiently calculate the unknown or erased symbols. The resulting method is algorithmically described in Section IV and an efficient implementation is presented in Section V.

II. Definition of Generalized Reed–Solomon Codes

Let us denote by $\underline{a} = \{a_0, a_1, \ldots, a_{n-1}\} \in \mathbb{F}_q$ a set of $n < q$ distinct and nonzero elements of a finite field $\mathbb{F}_q$, where $q = |\mathbb{F}_q|$ represents the cardinality of said field. Also, let $\underline{y} = \{y_0, y_1, \ldots, y_{n-1}\}$ denote another set of n nonzero (but not necessarily distinct) elements of the same field $\mathbb{F}_q$. A generalized Reed–Solomon (GRS) code of length n, dimension $k = n - r$ and r redundancy symbols over the field $\mathbb{F}_q$ may now be defined as

$$\mathrm{GRS}\left(\mathbb{F}_q, n, k, \underline{a}, \underline{y}\right) = \left\{ \underline{c} \in \mathbb{F}_q^n : \sum_{\ell=0}^{n-1} y_\ell c_\ell a_\ell^j = 0, \quad j = 0, \ldots, r-1 \right\}, \tag{1}$$

and is traditionally said to be t-error correcting for $t = \lfloor r/2 \rfloor$. The field elements $a_\ell \in \mathbb{F}_q$, $\ell = 0, \ldots, n-1$, are commonly called *code locators*. The elements $y_\ell \in \mathbb{F}_q$, $\ell = 0, \ldots, n-1$ are termed *column multipliers*. In digital communications, the field size q is often restricted to powers of 2, so that all elements of $\mathbb{F}_q$, $q = 2^m$, can be uniquely represented by a vector of m bits. For regular Reed–Solomon codes, which are a subclass of GRS codes, the code locators are constrained to be consecutive powers (possibly with an initial offset b) of the primitive element $\alpha \in \mathbb{F}_q$, i.e., $a_\ell = \alpha^{b+\ell}$, and the column multipliers are simply reduced to $y_\ell = 1$, $\forall \ell$.

978-1-61284-233-2/11/$26.00 ©2011 IEEE

As with all linear block codes, generalized Reed–Solomon codes can also be defined as the set of all vectors $\underline{c}$ lying in the null space of a parity-check matrix H_{GRS}, i.e.,

$$\text{GRS}\left(\mathbb{F}_q, n, k, \underline{a}, \underline{y}\right) = \left\{\underline{c} \in \mathbb{F}_q^n : H_{\text{GRS}}\underline{c}^T = \underline{0}\right\}. \quad (2)$$

The expression for the parity-check matrix H_{GRS} of a generalized Reed–Solomon code depends on $\underline{a}$ and $\underline{y}$ and follows from the combination of Equations (1) and (2),

$$H_{\text{GRS}} = \begin{pmatrix} y_0 & y_1 & \dots & y_{n-1} \\ y_0a_0 & y_1a_1 & \dots & y_{n-1}a_{n-1} \\ \vdots & \vdots & & \vdots \\ y_0a_0^{r-1} & y_1a_1^{r-1} & \dots & y_{n-1}a_{n-1}^{r-1} \end{pmatrix}$$
$$= \begin{pmatrix} 1 & 1 & \dots & 1 \\ a_0 & a_1 & \dots & a_{n-1} \\ \vdots & \vdots & & \vdots \\ a_0^{r-1} & a_1^{r-1} & \dots & a_{n-1}^{r-1} \end{pmatrix} \begin{pmatrix} y_0 & 0 & \dots & 0 \\ 0 & y_1 & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & y_{n-1} \end{pmatrix} = VY. \quad (3)$$

Note that the matrix V is a $r \times n$ *Vandermonde* matrix, where each column $\underline{v}_\ell^T$ consists of r consecutive powers of the corresponding code locator element $a_\ell \in \underline{a}$ for $\ell = 0, \dots, n-1$.

III. Efficient Recovery of Unknown Symbols

Consider the situation that a codeword $\underline{c}$ of a generalized Reed–Solomon code $\mathcal{C}_{\text{GRS}}$ of length n is at hand where only k symbols I_i, $i = 0, \dots, k-1$, are known, i.e., $r = n - k$ symbols U_j, $j = 0, \dots, r-1$, are yet unknown or have been erased. The exact indices u_j, $j = 0, \dots, r-1$, of the erased symbols within the codeword are supposed to be known, e.g.,

$$\underline{c} = (I_0, I_1, \dots, U_0, \dots, U_1, \dots, U_{r-1}, \dots, I_{k-1}) \in \mathcal{C}_{\text{GRS}}.$$
$$\begin{array}{lccccc} & & \uparrow & & \uparrow & \uparrow \\ \text{Index} = & & u_0 & & u_1 & u_{r-1} \end{array}$$

This equally represents the situation of a GRS encoder, when encoding with arbitrary parity positions u_j, as well as of a GRS erasure decoder, where the symbols at indices u_j have been erased.

In order to solve for the unknown symbols U_j given the known symbols I_i, the parity-check matrix H_{GRS} can be transformed to the following form:

$$\widetilde{H}_{\text{GRS}} = \left(\begin{array}{c|c|c|c|c} P\dots & \begin{matrix}1\\0\\\vdots\\0\end{matrix} & \dots P'\dots & \begin{matrix}0\\1\\\vdots\\0\end{matrix} & \dots P''\dots \quad \begin{matrix}0\\0\\\vdots\\1\end{matrix} \quad \dots P''' \end{array}\right). \quad (4)$$
$$\begin{array}{lccccc} & & \uparrow & & \uparrow & \uparrow \\ \text{Index} = & & u_0 & & u_1 & u_{r-1} \end{array}$$

The indices u_j of the canonical unit vectors $\underline{e}_j^T$ within $\widetilde{H}_{\text{GRS}}$ correspond to the locations of the unknown symbols U_j in the codeword $\underline{c}$. Consequently, each row of $\widetilde{H}_{\text{GRS}}$ provides an expression for one such unknown symbol U_j, $j = 0, \dots, r-1$. The particular form of Eq. (4) can be achieved by expressing $\widetilde{H}_{\text{GRS}}$ in terms of the original parity-check matrix H_{GRS} and two matrices $\widetilde{Y}$ and $\widetilde{V}$,

$$\widetilde{H}_{\text{GRS}} = \widetilde{Y}^{-1}\widetilde{V}^{-1}H_{\text{GRS}} = \widetilde{Y}^{-1}\widetilde{V}^{-1}(VY). \quad (5)$$

In order to satisfy (4) and (5), the purpose of the first matrix $\widetilde{Y}^{-1}$ of size $r \times r$ shall be to invert the column multipliers of the columns u_j in $\widetilde{H}_{\text{GRS}}$, hence

$$\widetilde{Y} = \begin{pmatrix} y_{u_0} & 0 & \dots & 0 \\ 0 & y_{u_1} & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & y_{u_{r-1}} \end{pmatrix}. \quad (6)$$

The second matrix $\widetilde{V}^{-1}$ shall be responsible for creating the canonical unit columns

$$\underline{\widetilde{h}}_{u_j}^T = \big(\underbrace{0, \dots, 0}_{j}, 1, \underbrace{0, \dots, 0}_{r-j-1}\big)^T = \underline{e}_j^T$$

in $\widetilde{H}_{\text{GRS}}$ at indices u_j and thus, $\widetilde{V}$ must be composed of the columns $\underline{v}_{u_0}^T, \dots, \underline{v}_{u_{r-1}}^T$ of the original component matrix V, i.e.,

$$\widetilde{V} = \begin{pmatrix} 1 & 1 & \dots & 1 \\ a_{u_0} & a_{u_1} & \dots & a_{u_{r-1}} \\ \vdots & \vdots & & \vdots \\ a_{u_0}^{r-1} & a_{u_1}^{r-1} & \dots & a_{u_{r-1}}^{r-1} \end{pmatrix}. \quad (7)$$

Its inverse $\widetilde{V}^{-1}$ is guaranteed to exist because $\widetilde{V}$ is a full-rank $r \times r$ Vandermonde matrix [6] made up of the particular distinct and nonzero code locators a_{u_j}, corresponding to the positions u_j of the canonical unit vectors in $\widetilde{H}_{\text{GRS}}$.

Once the matrices $\widetilde{V}$ and $\widetilde{Y}$ are generated, $\widetilde{H}_{\text{GRS}}$ could be calculated according to Eq. (5), requiring a computationally complex inversion of $\widetilde{V}$ as well as storage for the resulting $r \times k$ unknown elements in $\widetilde{H}_{\text{GRS}}$. As $\widetilde{V}$ and $\widetilde{Y}$ completely depend on the particular set of indices u_j, $j = 0, \dots, r-1$, their inverses would have to be computed again for every new set of such indices. Fortunately, by exploiting the special Vandermonde structure of H_{GRS} the computational complexity along with the storage requirements for computing $\widetilde{H}_{\text{GRS}}$ according to (5) can be significantly reduced:

First, let $\mathbb{F}_q[x]$ denote the ring of polynomials with coefficients from the field $\mathbb{F}_q$. The key idea [7] is to replace the inner product of a vector $\underline{b} = \left(b_0, b_1, b_2, \dots, b_{r-1}\right) \in \mathbb{F}_q^r$ and a Vandermonde vector $\underline{c} = \left(1, c^1, c^2, \dots, c^{r-1}\right) \in \mathbb{F}_q^r$, consisting of r consecutive powers of $c \in \mathbb{F}_q$, by the evaluation of a polynomial $B(x) \in \mathbb{F}_q[x]$ with coefficient vector $\underline{b}$ and degree $r-1$ at the element c, i.e.,

$$\langle \underline{b}, \underline{c} \rangle = b_0 + b_1c^1 + b_2c^2 + \dots + b_{r-1}c^{r-1} = B(c). \quad (8)$$

Now revisiting (5), each entry $\widetilde{h}_{j,\ell}$ of $\widetilde{H}_{\text{GRS}}$ results from the inner product of the j-th row $\underline{\widetilde{v}}^{(j)}$ of $\widetilde{V}^{-1}$ and the ℓ-th column $\underline{v}_\ell^T$ of V, multiplied by the term y_ℓ/y_{u_j} resulting from the product of row j of $\widetilde{Y}^{-1}$ and column ℓ of Y, respectively:

$$\widetilde{h}_{j,\ell} = \frac{y_\ell}{y_{u_j}}\langle \underline{\widetilde{v}}^{(j)}, \underline{v}_\ell \rangle = \frac{y_\ell}{y_{u_j}}\left(\widetilde{v}_0^{(j)} + \widetilde{v}_1^{(j)}a_\ell + \dots + \widetilde{v}_{r-1}^{(j)}a_\ell^{r-1}\right)$$
$$= \frac{y_\ell}{y_{u_j}}\Lambda^{(j)}(a_\ell). \quad (9)$$

By replacing all entries of $\widetilde{H}_{\text{GRS}}$ with the expression derived in Eq. (9):

$$\begin{aligned}\widetilde{H}_{\text{GRS}} &= \widetilde{Y}^{-1}\widetilde{V}^{-1}VY \\ &= \widetilde{Y}^{-1}\begin{pmatrix} \widetilde{v}_0^{(0)} & \widetilde{v}_1^{(0)} \dots & \widetilde{v}_{r-1}^{(0)} \\ \widetilde{v}_0^{(1)} & \widetilde{v}_1^{(1)} \dots & \widetilde{v}_{r-1}^{(1)} \\ \vdots & \vdots & \vdots \\ \widetilde{v}_0^{(r-1)} & \widetilde{v}_1^{(r-1)} \dots & \widetilde{v}_{r-1}^{(r-1)} \end{pmatrix}\begin{pmatrix} 1 & 1 & \dots & 1 \\ a_0 & a_1 & \dots & a_{n-1} \\ \vdots & \vdots & & \vdots \\ a_0^{r-1} & a_1^{r-1} & \dots & a_{n-1}^{r-1} \end{pmatrix} Y \\ &= \widetilde{Y}^{-1}\begin{pmatrix} \Lambda^{(0)}(a_0) & \Lambda^{(0)}(a_1) & \dots & \Lambda^{(0)}(a_{n-1}) \\ \Lambda^{(1)}(a_0) & \Lambda^{(1)}(a_1) & \dots & \Lambda^{(1)}(a_{n-1}) \\ \vdots & \vdots & & \vdots \\ \Lambda^{(r-1)}(a_0) & \Lambda^{(r-1)}(a_1) & \dots & \Lambda^{(r-1)}(a_{n-1}) \end{pmatrix} Y. \quad (10)\end{aligned}$$

In order to determine the coefficients of the polynomials $\Lambda^{(j)}(x) \in \mathbb{F}_q[x]$, $j = 0, \dots, r-1$, we require that all columns of $\widetilde{H}_{\text{GRS}}$ at indices u_j, $j = 0, \dots, r-1$, shall be of canonical unit form:

$$\underline{\widetilde{h}}_{u_j}^T = \big(\underbrace{0, \dots, 0}_{j}, 1, \underbrace{0, \dots, 0}_{r-j-1}\big)^T = \underline{e}_j^T.$$

Following from these constraints, each polynomial $\Lambda^{(j)}(x) \in \mathbb{F}_q[x]$, $j = 0, \dots, r-1$, becomes the j-th *Lagrange polynomial*, that is, the unique monic polynomial of degree $r-1$ from $\mathbb{F}_q[x]$ satisfying

$$\Lambda^{(j)}(a_{u_m}) = \begin{cases} 1 & \text{for } m = j, \\ 0 & \text{for } m \neq j, \end{cases} \quad (11)$$

for $m = 0, \dots, r-1$. We now proceed with similar steps as in [8] by first defining a monic *(erasure) locator polynomial*

$$\begin{aligned}\Lambda(x) &= \prod_{j=0}^{r-1} \big(x - a_{u_j}\big) \\ &= \lambda_0 + \lambda_1 x + \ldots + \lambda_{r-1}x^{r-1} + x^r\end{aligned} \quad (12)$$

of degree r from $\mathbb{F}_q[x]$ with r distinct roots at the *(erasure) locators* a_{u_j}, $j = 0, \dots, r-1$, as well as its formal derivative $\Lambda'(x)$ of degree $r-1$ from $\mathbb{F}_q[x]$. With these definitions we now formulate the j-th Lagrange polynomial of Eq. (11) as

$$\Lambda^{(j)}(x) = \frac{1}{\Lambda'(a_{u_j})} \frac{\Lambda(x)}{(x - a_{u_j})}. \quad (13)$$

Last but not least, each unknown symbol U_j can be determined from one row of $\widetilde{H}_{\text{GRS}}$. By combining the definition of generalized Reed–Solomon codes in Eq. (2) with the modified parity-check matrix $\widetilde{H}_{\text{GRS}}$ from Eq. (10), the following set of equations for $j = 0, \dots, r-1$ can be obtained:

$$U_j = \sum_{\substack{\ell=0 \\ \ell \notin \{u_j\}}}^{n-1} \frac{y_\ell}{y_{u_j}} c_\ell \, \Lambda^{(j)}(a_\ell) = \sum_{\ell=0}^{n-1} \frac{y_\ell}{y_{u_j}} \widetilde{I}_\ell \, \Lambda^{(j)}(a_\ell). \quad (14)$$

Here, the vector $\underline{\widetilde{I}}$ contains only the known part of the codeword $\underline{c}$, i.e., the symbols I_i, $i = 0, \dots, k-1$, as well as zeros at the unknown indices u_j, for example:

$$\begin{aligned}\underline{\widetilde{I}} &= (I_0, I_1, 0, 0, I_3, I_4, \dots, I_{k-2}, 0, I_{k-1}) \,. \\ &\quad\ \uparrow\ \uparrow \qquad\qquad\ \uparrow \\ \text{Index} = &\quad\ u_0\ u_1 \qquad\quad u_{r-1}\end{aligned}$$

By substituting (13) in (14) we arrive at the core equation of the proposed method,

$$U_j = \frac{1}{y_{u_j}\Lambda'(a_{u_j})} \sum_{\ell=0}^{n-1} \widetilde{I}_\ell \, y_\ell \, \frac{\Lambda(a_\ell)}{a_\ell} \frac{1}{1 - a_{u_j}/a_\ell}, \quad (15)$$

which is schematically depicted in detail in Fig. 1 above.

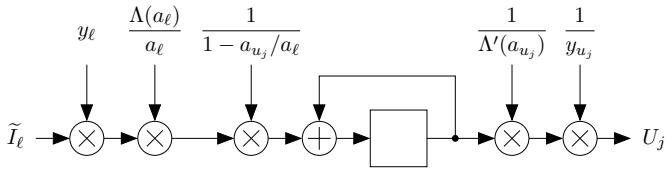

Fig. 1. Circuitry representing Eq. (15) for a single erased symbol U_j.

IV. Algorithmic Description of the Proposed Method

Utilizing Eq. (15), the proposed method can recover unknown or erased symbols at arbitrary positions in GRS codewords with the algorithm shown in Fig. 2:

Input: Padded vector $\underline{\widetilde{I}}$ with known symbols I_i, $i = 0 \dots k-1$

Required: The code locators a_ℓ of $\mathcal{C}_{\text{GRS}}$ for $\ell = 0 \dots n-1$
Required: The column multipliers y_ℓ and y_ℓ^{-1}, $\ell = 0 \dots n-1$
Required: The term $(1-x)^{-1}$ evaluated at $x \in \mathbb{F}_q \backslash \{0, 1\}$
Required: The term $\Lambda'(x)^{-1}$ evaluated at a_{u_j}, $j = 0 \dots r-1$
Required: The term $\Lambda(x)x^{-1}$ evaluated at a_ℓ, $\ell = 0 \dots n-1$

1: initialize: $\underline{c} \leftarrow \underline{\widetilde{I}}$
2: **for** $j = 0$ to $r-1$ **do**
3: initialize: $U_j \leftarrow 0$
4: **for** $\ell = 0$ to $n-1$ **do**
5: $U_j \leftarrow U_j + \widetilde{I}_\ell \cdot y_\ell \cdot \big[\Lambda(a_\ell)/a_\ell\big] \cdot \big[1 - a_{u_j}/a_\ell\big]^{-1}$
6: **end for**
7: normalize: $U_j \leftarrow U_j \cdot \big[y_{u_j} \cdot \Lambda'(a_{u_j})\big]^{-1}$
8: store U_j at position u_j of $\underline{c}$
9: **end for**

Output: Recovered codeword $\underline{c}$

Fig. 2. Proposed algorithm to recover the unknown symbols U_j.

The code locators a_ℓ and the column multipliers y_ℓ only depend on the specific GRS code and are thus fixed and known ahead of time. Also, the values of the term $(1-x)^{-1}$, for $x \in \mathbb{F}_q \backslash \{0, 1\}$, can be computed once and stored in memory. As the values of $y_{u_j}^{-1}$ and a_{u_j}/a_ℓ depend on the index j of the particular symbol U_j to be computed, the content of the corresponding shift registers must be accessible at different indices simultaneously.

For binary extension fields $\mathbb{F}_{2^m}$ which are considered here and assuming r to be even (for the sake of simplicity), the formal derivative $\Lambda'(x)$ takes the form

$$\Lambda'(x) = \sum_{i=1}^{r} i\lambda_i x^{i-1} = \lambda_1 + \lambda_3 x^2 + \ldots + \lambda_{r-1} x^{r-2}. \quad (16)$$

Hence, $\Lambda'(x)$ can be evaluated as a byproduct of the evaluation of $\Lambda(x)x^{-1}$ in the same circuit, as both terms share monomials of the form $\lambda_{2i+1}x^{2i}$ for $i = 0, \ldots, t-1$. However, this combined evaluation is not straightforward to implement with an efficient Chien search circuit, as the code locators a_ℓ of GRS codes may not necessarily be consecutive powers of the primitive element $\alpha \in \mathbb{F}_q$. One inefficient solution may be to calculate all required powers a_ℓ^j, $j = 0, \ldots, r-1$ of the current erasure locator element on the fly or beforehand, but this demands additional computational or memory resources.

V. Efficient Implementation using Chien Search

In order to use a Chien search circuit for the simultaneous evaluation of the polynomials $\Lambda(x)x^{-1}$ and $\Lambda'(x)$, we need to overcome the problem of unordered code locators a_ℓ in $\mathcal{C}_{\text{GRS}}$. By a simple reordering of the known vector $\underline{\widetilde{I}}$ prior to the summation in (14), the Vandermonde matrix V in the expression for $\widetilde{H}_{\text{GRS}}$ in (5) can be transformed into a standard Reed–Solomon Vandermonde parity-check matrix H_{RS}, that is, with column base elements consisting of consecutive powers of the primitive element $\alpha \in \mathbb{F}_q$:

$$H_{\text{RS}(\mathbb{F}_q,\, q-1,\, k)} = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ 1 & \alpha^1 & \ldots & \alpha^{q-2} \\ \vdots & \vdots & & \vdots \\ 1 & \alpha^{r-1} & \ldots & \alpha^{(q-2)(r-1)} \end{pmatrix}. \quad (17)$$

Recalling that $\underline{a} = \{a_0, a_1, \ldots, a_{n-1}\} \in \mathbb{F}_q$ is a subset of $n < q$ distinct and nonzero elements of a finite field $\mathbb{F}_q$, we can rephrase

$$V = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ a_0 & a_1 & \ldots & a_{n-1} \\ \vdots & \vdots & & \vdots \\ a_0^{r-1} & a_1^{r-1} & \ldots & a_{n-1}^{r-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ 1 & \alpha^1 & \ldots & \alpha^{q-2} \\ \vdots & \vdots & & \vdots \\ 1 & \alpha^{r-1} & \ldots & \alpha^{(q-2)(r-1)} \end{pmatrix} \begin{pmatrix} (q-1)\times n \\ \text{Column} \\ \text{Permutation} \\ \text{Matrix } \Pi \end{pmatrix}, \quad (18)$$

such that the rows $\underline{\pi}_i$ of Π are of the form

$$\underline{\pi}_i = \big(\underbrace{0, \ldots, 0}_{\ell}, 1, \underbrace{0, \ldots, 0}_{n-1-\ell}\big) \quad \text{if } \alpha^i = a_\ell$$

and

$$\underline{\pi}_i = \underline{0} \quad \text{if } \alpha^i \notin \underline{a},$$

for $i = 0, \ldots, q-2$ and $\ell = 0, \ldots, n-1$. Thus, the i-th row of Π indicates that the ℓ-th column in V is identical to the i-th column of $H_{\text{RS}(\mathbb{F}_q,\, q-1,\, k)}$.

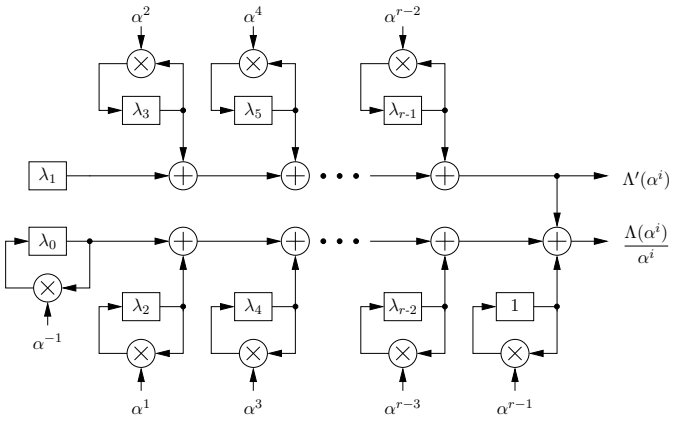

Fig. 3. Chien search circuit simultaneously evaluating $\Lambda'(x)$ and $\frac{\Lambda(x)}{x}$.

At this point, the circuit design can be based solely on $H_{\text{RS}(\mathbb{F}_q,\, q-1,\, k)}$, its code locators $\underline{\alpha} = \{1, \alpha^1, \alpha^2, \ldots, \alpha^{q-2}\}$ and a reordered known vector $\underline{\widehat{I}} = \underline{\widetilde{I}}\,\Pi$, resulting in the following mapping for $i = 0, \ldots, q-2$:

$$\widehat{I}_i = \begin{cases} \widetilde{I}_\ell & \text{if } \alpha^i = a_\ell, \\ 0 & \text{if } \alpha^i \notin \underline{a}. \end{cases} \quad (19)$$

The permutation matrix Π must likewise be applied to the sequence of column multipliers $\underline{\widehat{y}} = \underline{y}\,\Pi$, which can be prepared offline and stored in memory. Also the summation in Eq. (5) must be extended, such that it evaluates the polynomials $\Lambda(x)x^{-1}$ and $\Lambda'(x)^{-1}$ at $x = \alpha^i$, $i = 0, \ldots, q-2$, and consequently allowing the use of an efficient Chien search circuit as depicted in Fig. 3. In case the proposed method is used for erasure decoding of GRS codes in an concatenated and interleaved coding scheme, the erasure locations u_j, $j = 0, \ldots, r-1$ may be identical for a large number of codewords due to loss of complete packets. Hence, the polynomials $\Lambda(x)x^{-1}$ and $\Lambda'(x)^{-1}$ do not change and only need to be evaluated once for the first codeword, further reducing computational complexity.

Input: Padded vector $\underline{\widetilde{I}}$ with known symbols I_i, $i = 0 \ldots k-1$
Required: The column permutation matrix Π
Required: The permuted column multipliers $\widehat{y}_i$, $i = 0 \ldots q-2$
Required: Values $(1-\alpha^{u_j - i})^{-1}$ for all $(u_j - i) = 0, \ldots, q-2$
Required: The term $\Lambda'(x)^{-1}$ evaluated at α^{u_j}, $j = 0 \ldots r-1$
Required: The term $\Lambda(x)x^{-1}$ evaluated at α^i, $i = 0 \ldots q-2$

1: initialize: $\underline{c} \leftarrow \underline{\widetilde{I}}$
2: rearrange: $\underline{\widehat{I}} \leftarrow \underline{\widetilde{I}} \cdot \Pi$
3: **for** $j = 0$ to $r-1$ **do**
4: initialize: $U_j \leftarrow 0$
5: **for** $i = 0$ to $q-2$ **do**
6: $U_j \leftarrow U_j + \widehat{I}_i \cdot \widehat{y}_i \cdot \big[\Lambda(\alpha^i)/\alpha^i\big] \cdot \big[1 - \alpha^{u_j - i}\big]^{-1}$
7: **end for**
8: normalize: $U_j \leftarrow U_j \cdot \big[\widehat{y}_{u_j} \cdot \Lambda'(\alpha^{u_j})\big]^{-1}$
9: store U_j at position u_j of $\underline{c}$
10: **end for**

Output: Recovered codeword $\underline{c}$

Fig. 4. Proposed algorithm to recover U_j using a Chien search.

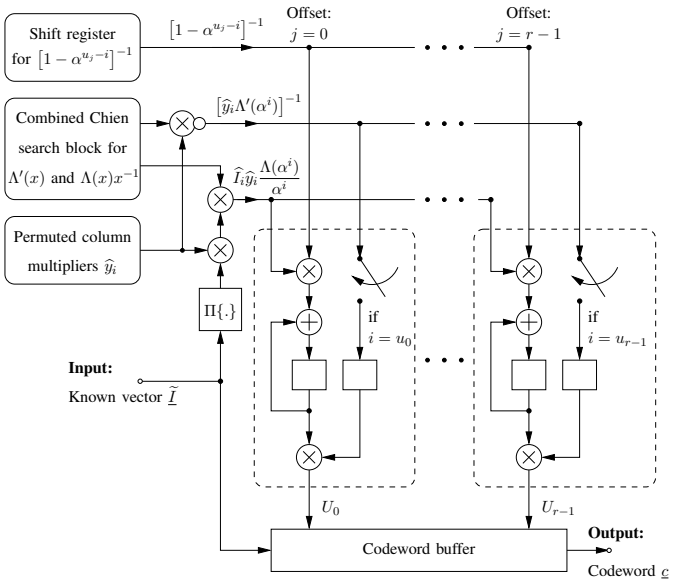

Fig. 5. Circuitry implementing the algorithm from Fig. 4.

The proposed method using a Chien search circuit is described in Fig. 4 along with a circuitry in Fig. 5. Note that this algorithm always needs the full number of $q-1$ clock cycles to recover the missing symbols U_j. Thus, for GRS codes with $n \ll q-1$, this method introduces a delay of $q-1-n$ clock cycles. For use with regular RS codes, the permutation Π is not necessary and thus no additional delay is introduced, as well as the column multipliers reduce to $y_\ell = 1,\ \ell = 0, \ldots, n-1$.

VI. Summary and Further Work

The proposed method recovers erased or unknown symbols for codes based on Vandermonde parity-check matrices, by exploiting this explicit structure of the parity-check matrix of such codes to perform a series of polynomial evaluations. For the case of GRS codes with arbitrarily ordered code locators, an efficient circuitry using Chien search can be designed by rearranging its parity-check matrix. In further work, the exact computational complexity of the proposed method will be compared with other available erasure decoding algorithms.

References

[1] I. S. Reed and G. Solomon, "Polynomial Codes Over Certain Finite Fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, Jun. 1960.

[2] ETSI, *Digital Video Broadcasting (DVB); DVB specification for data broadcasting (ETSI European Standard EN 301 192 V1.5.1)*, European Telecommunications Standards Institute Std., Nov. 2009.

[3] ATSC, *ATSC-Mobile DTV Standard, Part 2 - RF/Transmission System Characteristics (A/153, Part 2:2009)*, Advanced Television Systems Committee Std., Oct. 2009.

[4] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson, "RAID: High-Performance, Reliable Secondary Storage," Berkeley, CA, USA, Tech. Rep., 1993.

[5] A. Wiebalck, P. T. Breuer, V. Lindenstruth, and T. M. Stinbeck, "Fault-Tolerant Distributed Mass Storage for LHC Computing," in *Proc. of 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (IEEE CCGrid 2003)*, May 2003, pp. 266 – 273.

[6] R. Horn and C. Johnson, *Matrix Analysis*. Cambridge Univ. Press, 1985.

[7] F. D. Parker, "Inverses of Vandermonde Matrices," *The American Mathematical Monthly*, vol. 71, pp. 410–411, Apr. 1964.

[8] J. Brauchle and R. Koetter, "A Systematic Reed–Solomon Encoder with Arbitrary Parity Positions," in *Proc. of 52nd IEEE Global Telecommunications Conference (IEEE GLOBECOM 2009)*, Dec. 2009.