Generalized Integrated Interleaved Codes

Yingquan Wu

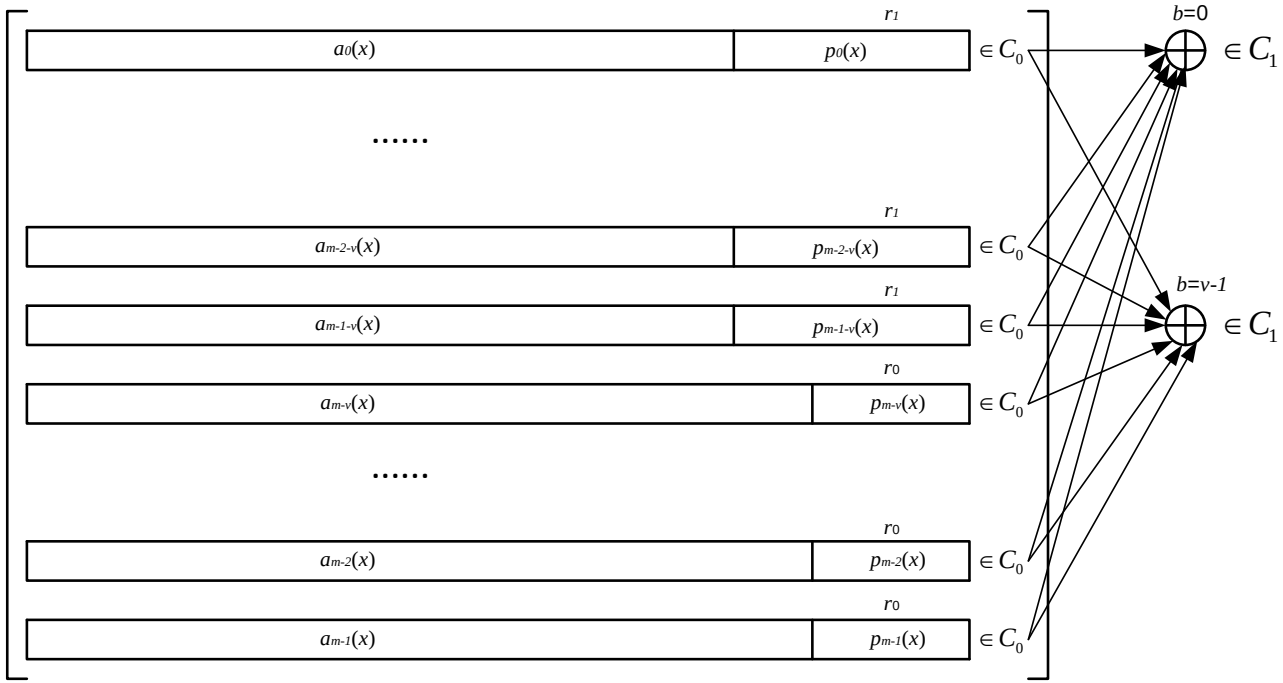
March 30, 2016

Micron Technology

Outline

- Generalized Integrated Interleaved Reed-Solomon (GII-RS) codes
  - Code characterization
  - Hard decoding algorithm
  - Performance analysis
  - Systematic encoding algorithm
- Generalized Integrated Interleaved BCH (GII-BCH) codes
  - Code characterization
  - Hard decoding and performance analysis
  - Systematic encoding

II-RS Codes

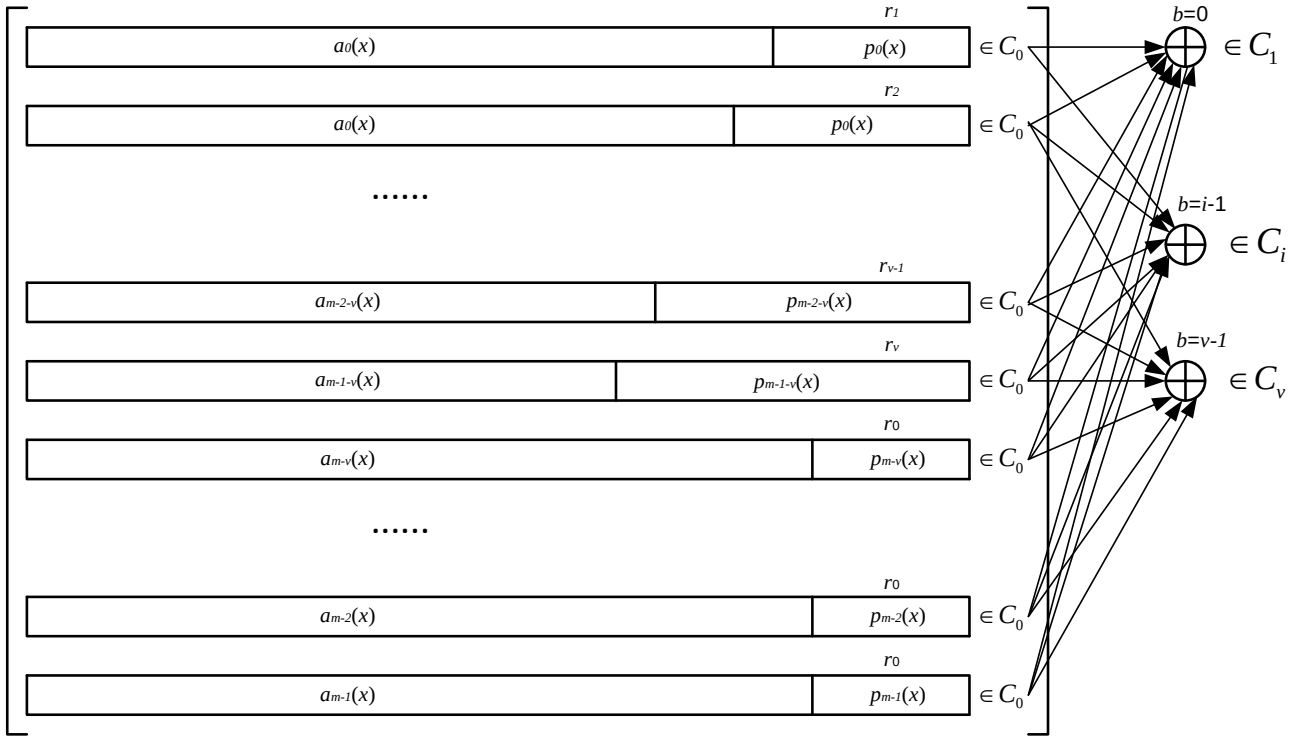


- An II-RS code is defined as

$$\mathcal{C} \stackrel{\triangle}{=} \left\{ \mathbf{c} = [\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{m-1}] : \ \mathbf{c}_i \in \mathcal{C}_0, \ \sum_{i=0}^{m-1} \alpha^{bi} \mathbf{c}_i \in \mathcal{C}_1, \ 0 \le b < v \right\},$$

where $v < m < q$, and $\{\mathcal{C}_i(n, k_i, d_i)\}_{i=0}^{1}$ are RS codes satisfying $\mathcal{C}_1 \subset \mathcal{C}_0$.

GII-RS Codes



- A GII-RS code is defined as

$$\mathcal{C} \stackrel{\triangle}{=} \left\{ \mathbf{c} = [\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{m-1}] : \ \mathbf{c}_i \in \mathcal{C}_0, \ \sum_{i=0}^{m-1} \alpha^{bi} \mathbf{c}_i \in \mathcal{C}_{b+1}, \ 0 \le b < v \right\},$$

where $v < m < q$, and $\{\mathcal{C}_i(n, k_i, d_i)\}_{i=0}^{v}$ are RS codes satisfying

$$\mathcal{C}_v \subset \mathcal{C}_{v-1} \subset \mathcal{C}_{v-2} \subset \ldots \subset \mathcal{C}_1 \subset \mathcal{C}_0$$

GII-RS Codes

- GII-RS Characterization

  Let $\mathcal{C}_0 \supset \mathcal{C}_1 \supset \mathcal{C}_2 \supset \mathcal{C}_{v-1} \supset \mathcal{C}_v$. A GII-RS code $\mathcal{C}(N, K, d_{\min})$ is a linear block code of length $N = mn$, dimension $K = \sum_{i=1}^{v} k_i + (m-v)k_0$, and minimum distance

$$d_{\min} = \min\{(v+1)d_0, vd_1, \ldots, 2d_{v-1}, d_v\}.$$

- Proof for dimension:

  For a codeword $\mathbf{c} = [\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_m] \in \mathcal{C}$, we define a mapping word $\mathscr{V}(\mathbf{c}) = [\mathbf{c}'_0, \ldots, \mathbf{c}'_{v-1}, \mathbf{c}_v, \ldots, \mathbf{c}_{m-1}]$ such that

$$\begin{bmatrix} \mathbf{c}'_0 \\ \mathbf{c}'_1 \\ \vdots \\ \mathbf{c}'_{v-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ \alpha & \alpha^2 & \ldots & \alpha^{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{v-1} & \alpha^{2(v-1)} & \ldots & \alpha^{(m-1)(v-1)} \end{bmatrix} \begin{bmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_{m-1} \end{bmatrix}$$

  Since the connection matrix is a Vandemonde matrix, we have $\mathscr{V}(\mathbf{c}) \neq \mathscr{V}(\mathbf{c}')$ iff $\mathbf{c} \neq \mathbf{c}'$. By definition,

$$\mathbf{c}'_i \in \mathcal{C}_{i+1}, \quad i = 0, 1, 2, \ldots, v-1.$$

  Therefore, we obtain the size of the code

$$|\mathcal{C}| = |\mathscr{V}(\mathcal{C})| = q^{\sum_{i=1}^{v} k_i + (m-v)k_0}.$$

- Proof for minimum distance:

If a codeword $\mathbf{c}$ has at least $v+1$ nonzero interleaves, then its Hamming weight is at least $(v+1)d_0$. On the other hand, let

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ \alpha & \alpha^2 & \dots & \alpha^v \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{v-1} & \alpha^{2(v-1)} & \dots & \alpha^{v(v-1)} \end{bmatrix} \begin{bmatrix} 1 \\ \gamma_1^{(0)} \\ \vdots \\ \gamma_v^{(0)} \end{bmatrix} = \mathbf{0}$$

and $\mathbf{c}_0 \in \mathcal{C}_0$ be with minimum Hamming weight d_0, then the codeword in $\mathcal{C}$

$$\mathbf{c} = [\mathbf{c}_0, \gamma_1^{(0)}\mathbf{c}_0, \dots, \gamma_v^{(0)}\mathbf{c}_0, \mathbf{0}, \dots, \mathbf{0}]$$

has exactly $v+1$ nonzero interleaves and weight $(v+1)d_0$.

When a codeword $\mathbf{c}$ has exactly v nonzero interleaves, it can be shown that all its nonzero component words must lie in $\mathcal{C}_1$. let $\gamma_1^{(1)}, \gamma_2^{(1)} \dots, \gamma_{v-1}^{(1)}$, satisfy

$$\begin{bmatrix} \alpha & \alpha^2 & \dots & \alpha^v \\ \alpha^2 & \alpha^4 & \dots & \alpha^{2v} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{v-1} & \alpha^{2(v-1)} & \dots & \alpha^{(v-1)v} \end{bmatrix} \begin{bmatrix} 1 \\ \gamma_1^{(1)} \\ \vdots \\ \gamma_{v-1}^{(1)} \end{bmatrix} = \mathbf{0}$$

and $\mathbf{c}_1 \in \mathcal{C}_1$ be with minimum Hamming weight d_1. It is easy to verify that

$$\mathbf{c} = [\mathbf{c}_1, \gamma_1^{(1)}\mathbf{c}_1, \dots, \gamma_{v-1}^{(1)}\mathbf{c}_1, \mathbf{0}, \dots, \mathbf{0}]$$

is a valid codeword in $\mathcal{C}$ with minimum Hamming weight vd_1.

Proof is concluded by induction.

GII-RS Codes for Advanced RAID System

- An example of RAID6

D	D	D	D	D	D	D	D	D	D	D	D	P	P

- An example of advanced RAID system using GII-RS scheme.

D	D	D	D	D	D	D	D	D	D	D	D	P	P
D	D	D	D	D	D	D	D	D	D	D	D	P	P
D	D	D	D	D	D	D	D	D	D	D	P	P	P
D	D	D	D	D	D	D	D	D	D	D	D	D	P
D	D	D	D	D	D	D	D	D	D	D	D	D	P
D	D	D	D	D	D	D	D	D	D	D	D	D	P
D	D	D	D	D	D	D	D	D	D	D	D	D	P
D	D	D	D	D	D	D	D	D	D	D	D	D	P

- GII-RS scheme provides stronger protection with smaller disk overhead than the conventional RAID6.

RS Decoding Basics

- Syndrome Computation and resulting polynomial

$$S_i \stackrel{\triangle}{=} y(\alpha^{i+1}) - c(\alpha^{i+1}) = y(\alpha^{i+1}), \quad i = 0, 1, 2, \ldots, 2t-1.$$

$$S(x) \stackrel{\triangle}{=} S_0 + S_1 x + S_2 x^2 + \ldots + S_{2t-1} x^{2t-1}.$$

- Error locator polynomial

$$\Lambda(x) \stackrel{\triangle}{=} \prod_{i=1}^{e} (1 - X_i x) = 1 + \Lambda_1 x + \Lambda_2 x^2 + \ldots + \Lambda_e x^e.$$

- Error evaluator polynomial

$$\Omega(x) = \sum_{i=1}^{e} Y_i X_i \prod_{j=1, j\neq i}^{e} (1 - X_j x) = \Omega_0 + \Omega_1 x + \Omega_2 x^2 + \ldots + \Omega_{e-1} x^{e-1}.$$

- Key equation

$$\Omega(x) = \Lambda(x) S(x) \qquad (\text{mod } x^{2t}).$$

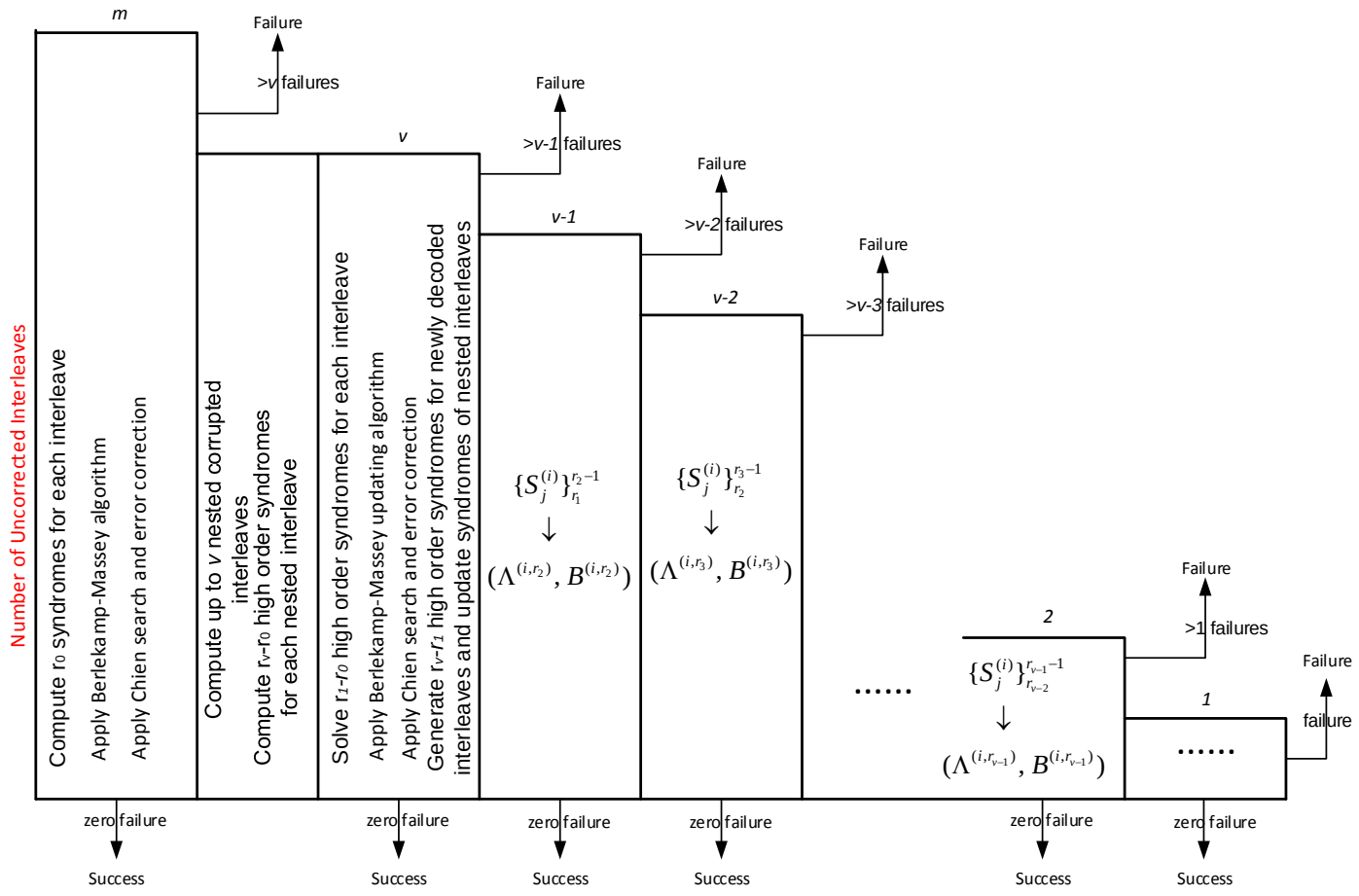
- High-order syndrome computation

$$S_i = -\Lambda_1 S_{i-1} - \Lambda_2 S_{i-2} - \ldots - \Lambda_e S_{i-e}, \quad i \geq 2t.$$

Berlekamp-Massey Updating Algorithm

- Input: $[S_0, \ldots, S_{2t-1}, \overbrace{S_{2t}, \ldots, S_{2t'-1}}^{new}]$, $\Lambda^{(2t)}(x)$, $B^{(2t)}(x)$, $L_\Lambda^{(2t)}$, $L_B^{(2t)}$
- For $r = 2t, 2t+1, 2t+2, \ldots, 2t'-1$, do:
  - Compute $\Delta^{(r)} = \sum_{i=0}^{L_\Lambda} \Lambda_i^{(r)} \cdot S_{r-i}$
  - Compute $\Lambda^{(r+1)}(x) = \Lambda^{(r)}(x) - \Delta^{(r)} \cdot xB^{(r)}(x)$
  - If $\Delta^{(r)} \neq 0$ and $L_\Lambda^{(r)} \leq L_B^{(r)}$, then
    - Set $B^{(r+1)}(x) \leftarrow (\Delta^{(r)})^{-1} \cdot \Lambda^{(r)}(x)$
    - Set $L_\Lambda^{(r+1)} \leftarrow L_B^{(r)} + 1$, $L_B^{(r+1)} \leftarrow L_\Lambda^{(r)}$
  - Else
    - Set $B^{(r+1)}(x) \leftarrow xB^{(r)}(x)$
    - Set $L_\Lambda^{(r+1)} \leftarrow L_\Lambda^{(r)}$, $L_B^{(r+1)} \leftarrow L_B^{(r)} + 1$
- Outputs: $\Lambda^{(2t')}(x)$, $B^{(2t')}(x)$, $L_\Lambda^{(2t')}$, $L_B^{(2t')}$

GII-RS Decoder Block Diagram



GII-RS Decoder Characterization

- Decoding capability

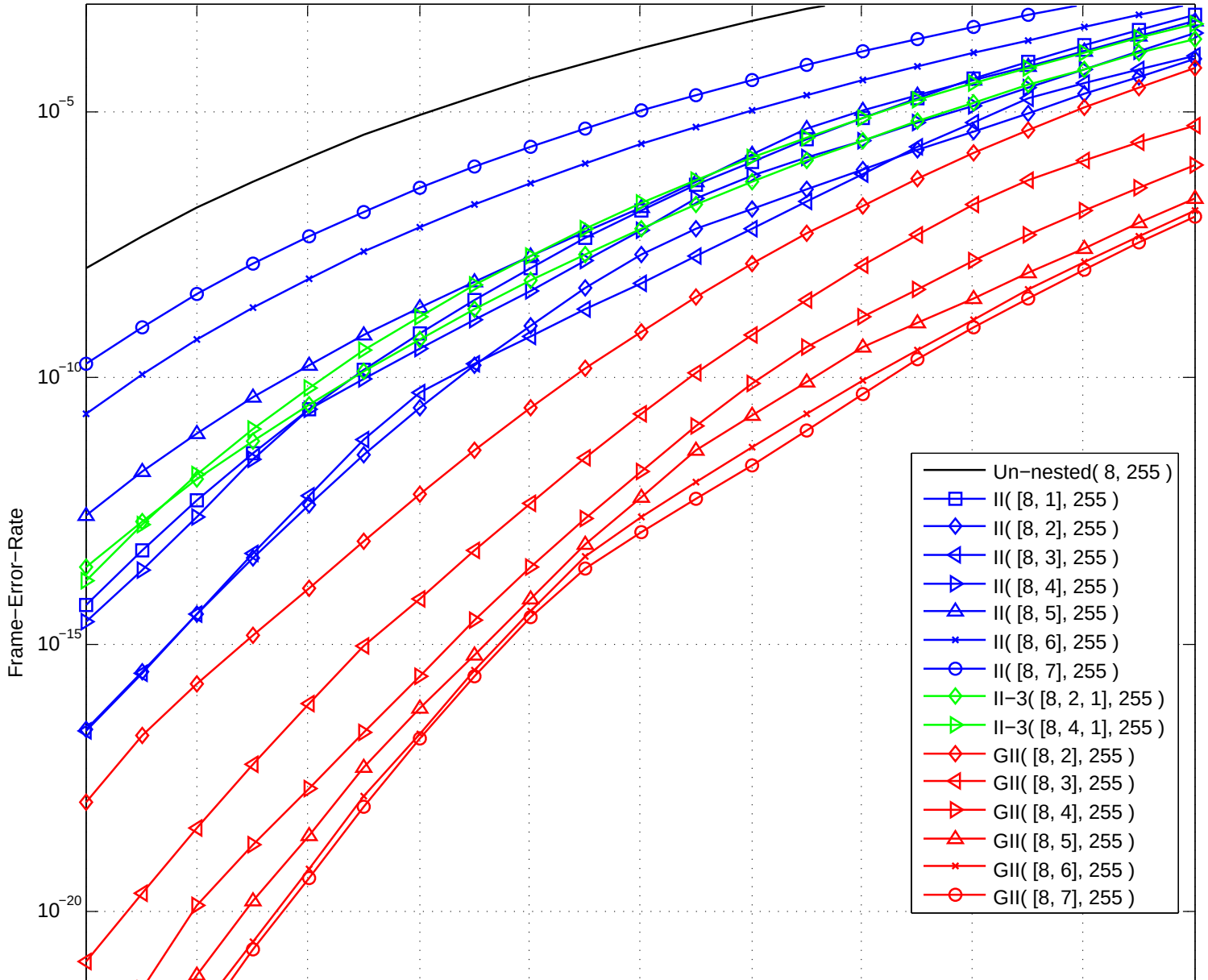
  By neglecting miscorrection of interleave self-decoding, let e_0, e_1, ..., e_{m-1} denote the number of errors over received interleaves $\mathbf{y}_0$, $\mathbf{y}_1$, ..., $\mathbf{y}_{m-1}$, respectively, and be reordered in ascending order $e_{l_1} \leq e_{l_2} \leq \ldots \leq e_{l_m}$, then the decoding is successful iff

$$e_{l_{m-j}} \leq t_{v-j} \triangleq \lfloor \frac{d_{v-j}-1}{2} \rfloor, \quad j = 0, 1, 2, \ldots, v.$$

- Decoding failure probability

$$P_e = \sum_{b=1}^{m} \binom{m}{b} \left(\sum_{w=t_v+1}^{n} \phi_w^n(p_s) \right)^b \left(\sum_{w=0}^{t_v} \phi_w^n(p_s) \right)^{m-b}$$
$$+ \sum_{i=0}^{v-1} \sum_{b=v-i+1}^{m} \binom{m}{b} \left(\sum_{w=t_i+1}^{t_{i+1}} \phi_w^n(p_s) \right)^b \left(\sum_{w=0}^{t_i} \phi_w^n(p_s) \right)^{m-b}$$

  where $\phi_w^n(p_s)$ is the probability of having w errors among n symbols, and p_s the symbol error rate.


Frame−Error−Rate
10^{-5}
10^{-10}
10^{-15}
10^{-20}
Un−nested(8, 255)
II([8, 1], 255)
II([8, 2], 255)
II([8, 3], 255)
II([8, 4], 255)
II([8, 5], 255)
II([8, 6], 255)
II([8, 7], 255)
II−3([8, 2, 1], 255)
II−3([8, 4, 1], 255)
GII([8, 2], 255)
GII([8, 3], 255)
GII([8, 4], 255)
GII([8, 5], 255)
GII([8, 6], 255)
GII([8, 7], 255)

GII-RS Decoder Characterization

Performance comparisons of various nested RS codes optimized under the symbol-error-rate of 0.02.

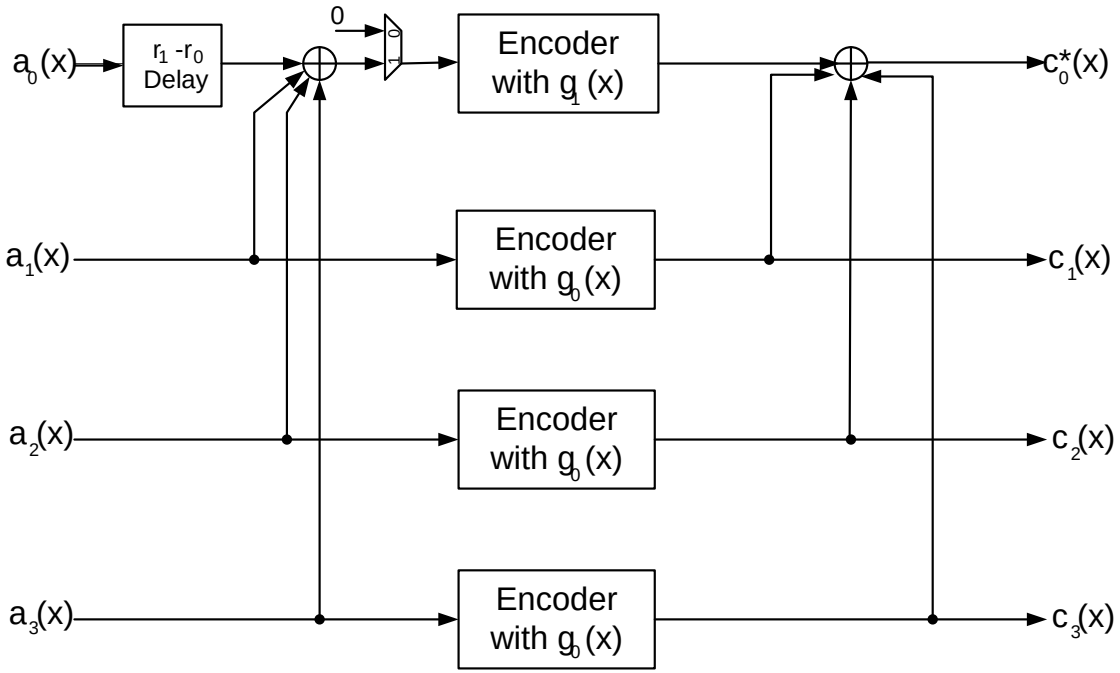
Nesting Scheme	$d_{\min}$	$P_e(0.02)$	$P_v(0.02)$
Un-nested(8, 255, 33)	33	1.55e-4	
II([8, 1], 255, [31, 47])	47	1.37e-7	5.51e-4
II([8, 2], 255, [27, 51])	51	2.14e-8	5.78e-3
II([8, 3], 255, [25, 47])	47	6.09e-9	1.69e-2
II([8, 4], 255, [21, 45])	45	6.01e-8	0.111
II([8, 5], 255, [19, 43])	43	1.58e-7	0.242
GII([8, 2], 255, [29, 37, 53])	53	7.48e-10	1.84e-3
GII([8, 3], 255, [27, 33, 39, 57])	57	2.09e-11	5.78e-3
GII([8, 4], 255, [25, 29, 35, 41, 59])	59	1.78e-12	1.69e-2
GII([8, 5], 255, [23, 27, 31, 35, 43, 59])	59	5.97e-13	4.54e-2
GII([8, 6], 255, [23, 25, 27, 31, 35, 41, 59])	59	2.53e-13	4.54e-2
GII([8, 7], 255, [21, 23, 25, 27, 31, 35, 43, 59])	59	1.29e-13	0.111

GII-RS Decoder Characterization

Performance comparisons of various nested RS codes optimized under the symbol-error-rate of 0.01.

Nesting Scheme	$d_{\min}$	$P_e(0.01)$	$P_i(0.01)$
Un-nested(8, 255, 33)	33	1.12e-8	
II([8, 1], 255, [31, 47])	47	5.38e-15	7.96e-8
II([8, 2], 255, [27, 51])	51	2.55e-17	3.31e-6
II([8, 3], 255, [23, 51])	51	2.35e-17	1.04e-4
II([8, 4], 255, [19, 47])	47	2.73e-15	2.34e-3
II([8, 5], 255, [19, 43])	43	2.64e-13	2.34e-3
GII([8, 2], 255, [29, 35, 55])	55	1.13e-18	5.31e-7
GII([8, 3], 255, [27, 31, 39, 59])	59	1.17e-21	3.31e-6
GII([8, 4], 255, [25, 27, 33, 41, 63])	63	9.51e-23	1.93e-5
GII([8, 5], 255, [23, 25, 29, 33, 43, 65])	65	5.06e-25	1.04e-4
GII([8, 6], 255, [21, 23, 25, 29, 35, 43, 67])	67	5.69e-26	5.15e-4
GII([8, 7], 255, [19, 21, 23, 25, 29, 35, 45, 67])	67	2.68e-26	2.34e-3

GII-RS Encoder ($v = 1$)



$$c_0^*(x) = \mathscr{E}\left(a_0(x) + \sum_{i=1}^{m-1} \mathscr{U}_{r_1-r_0} a_i(x), \quad g_1(x)\right) = \left[a_0(x) + \sum_{i=1}^{m-1} \mathscr{U}_{r_1-r_0} a_i(x), \quad p_0^*(x)\right].$$

which yields the desired codeword, $c_0(x)$, to encode $a_0(x)$,

$$c_0(x) \stackrel{\triangle}{=} c_0^*(x) - \sum_{i=1}^{m-1} c_i(x) = \left[a_0(x), \; p_0^*(x) - \sum_{i=1}^{m-1} \mathscr{L}_{r_1} c_i(x)\right].$$

GII-RS Encoder

- Encoding linear equation system

$$\begin{cases} c_0(x) & +c_1(x) & +\ldots & +c_{v-1}(x) & \equiv -\sum_{i=v}^{m-1} c_i(x) & (\text{mod } g_1(x)) \\ c_0(x) & +\alpha c_1(x) & +\ldots & +\alpha^{v-1} c_{v-1}(x) & \equiv -\sum_{i=v}^{m-1} \alpha^i c_i(x) & (\text{mod } g_2(x)) \\ c_0(x) & +\alpha^2 c_1(x) & +\ldots & +\alpha^{2(v-1)} c_{v-1}(x) & \equiv -\sum_{i=v}^{m-1} \alpha^{2i} c_i(x) & (\text{mod } g_3(x)) \\ \vdots & \vdots & \ddots & \vdots & \vdots & \\ c_0(x) & +\alpha^{v-1} c_1(x) & +\ldots & +\alpha^{(v-1)(v-1)} c_{v-1}(x) & \equiv -\sum_{i=v}^{m-1} \alpha^{i(v-1)} c_i(x) & (\text{mod } g_v(x)) \end{cases}$$

- To determine $c_0(x)$, it suffices to relax mod$g_i(x)$, $i = 1, 2, \ldots, v$, to mod $g_1(x)$, i.e.,

$$\begin{cases} c_0(x) & +c_1(x) & +\ldots & +c_{v-1}(x) & \equiv -\sum_{i=v}^{m-1} c_i(x) & (\text{mod } g_1(x)) \\ c_0(x) & +\alpha c_1(x) & +\ldots & +\alpha^{v-1} c_{v-1}(x) & \equiv -\sum_{i=v}^{m-1} \alpha^i c_i(x) & (\text{mod } g_1(x)) \\ c_0(x) & +\alpha^2 c_1(x) & +\ldots & +\alpha^{2(v-1)} c_{v-1}(x) & \equiv -\sum_{i=v}^{m-1} \alpha^{2i} c_i(x) & (\text{mod } g_1(x)) \\ \vdots & \vdots & \ddots & \vdots & \vdots & \\ c_0(x) & +\alpha^{v-1} c_1(x) & +\ldots & +\alpha^{(v-1)(v-1)} c_{v-1}(x) & \equiv -\sum_{i=v}^{m-1} \alpha^{i(v-1)} c_i(x) & (\text{mod } g_1(x)) \end{cases}$$

GII-RS Encoder

- Denote by θ_i the determinant

$$\theta_i \triangleq \begin{vmatrix} \alpha^{ii} & \alpha^{i(i+1)} & \dots & \alpha^{i(v-1)} \\ \alpha^{i(i+1)} & \alpha^{(i+1)(i+1)} & \dots & \alpha^{(i+1)(v-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{i(v-1)} & \alpha^{(i+1)(v-1)} & \dots & \alpha^{(v-1)(v-1)} \end{vmatrix}, \quad 0 \le i < v$$

and by $\theta_{i,j}$ the determinant

$$\theta_{i,j} \triangleq - \begin{vmatrix} \alpha^{ji} & \alpha^{i(i+1)} & \alpha^{i(i+2)} & \dots & \alpha^{i(v-1)} \\ \alpha^{j(i+1)} & \alpha^{(i+1)(i+1)} & \alpha^{(i+1)(i+2)} & \dots & \alpha^{(i+1)(v-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha^{j(v-1)} & \alpha^{(v-1)(i+1)} & \alpha^{(v-1)(i+2)} & \dots & \alpha^{(v-1)(v-1)} \end{vmatrix}, \quad j \ge v \text{ or } j < i.$$

- Solution of $c_0(x)$

$$c_0(x) \equiv \sum_{j=v}^{m-1} \mu_{0,j} c_j(x) \pmod{g_1(x)}$$

where $\mu_{i,j} \triangleq \theta_i^{-1}\theta_{i,j}$ are pre-computed.

GII-RS Encoding Algorithm

1. Apply LFSR encoding to $a_v(x)$, $a_{v+1}(x)$, $\ldots$, $a_{m-1}(x)$, respectively,

$$c_i(x) = \mathscr{E}\left(a_i(x),\ \ g_0(x)\right) = [a_i(x),\ \ p_i(x)], \quad i = v, v+1, \ldots, m-1,$$

where $\deg(p_i(x)) < r_0$, $v \le i < m$.

2. For $i = 0, 1, 2, \ldots, v-1$, do:

(a) Compute message $a_i^*(x)$

$$a_i^*(x) = a_i(x) + \sum_{0 \le j < i, v \le j < m} \mu_{i,j} \cdot \mathscr{U}_{r_{i+1}-r_0} a_j(x)$$

where $\mathscr{U}_r a(x) \stackrel{\triangle}{=} (a(x) - \mathscr{L}_r a(x))/x^r$ or $\mathscr{U}_r[a_{k-1}, a_{k-2}, \ldots, a_0] \stackrel{\triangle}{=} [a_{k-1}, a_{k-2}, \ldots, a_r]$.

(b) Apply LFSR encoding to $a_i'(x)$ with respect to $g_{i+1}(x)$

$$\mathscr{E}\left(a_i^*(x),\ \ g_{i+1}(x)\right) = [a_i^*(x),\ \ p_i^*(x)]$$

(c) Determine the parity polynomial $p_i(x)$ associated with $a_i(x)$

$$p_i(x) = p_i^*(x) - \sum_{0 \le j < i, v \le j < m} \mu_{i,j} \cdot \mathscr{L}_{r_{i+1}} c_j(x)$$

where $\mathscr{L}_r a(x) \stackrel{\triangle}{=} a(x) \bmod x^r$ or $\mathscr{L}_r[a_{k-1}, a_{k-2}, \ldots, a_0] \stackrel{\triangle}{=} [a_{r-1}, a_{r-2}, \ldots, a_0]$.

GII-BCH Codes

- $\gamma \triangleq [\gamma_{q-1}, \gamma_{q-2}, \ldots, \gamma_1, \gamma_0] = \gamma_{q-1}\alpha^{q-1} + \gamma_{q-2}\alpha^{q-2} + \ldots + \gamma_1\alpha + \gamma_0$, where $\gamma_i \in \mathrm{GF}(2)$, $0 \leq i < q$.
- $\gamma(x) \triangleq \gamma_{q-1}x^{q-1} + \gamma_{q-2}x^{q-2} + \ldots + \gamma_1 x + \gamma_0$, which yields

$$\begin{aligned}
\alpha(x^j) &= x^j \bmod \psi(x) \quad (\psi(x) \text{ denotes the primitive polynomial}) \\
\alpha(\gamma) &= \gamma, \quad \forall \gamma \in \mathrm{GF}(q), \\
\alpha^i(x) &= \alpha(x^i), \quad 0 \leq i < q, \\
\alpha^{-1}(x) &= \alpha(x)^{-1} \pmod{\psi(x)}.
\end{aligned}$$

- A GII-BCH code is defined as

$$\mathcal{C} \triangleq \left\{ \mathbf{c} = [\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{m-1}] : \ \mathbf{c}_i(x) \in \mathcal{C}_0, \ \sum_{i=0}^{m-1} \alpha(x^{bi})\mathbf{c}_i(x) \in \mathcal{C}_{b+1}, \ 0 \leq b < v \right\}$$

where $\{\mathcal{C}_i(n, k_i, d_i)\}_{i=0}^{v}$ are binary BCH codes over $\mathrm{GF}(2^q)$ such that

$$\mathcal{C}_0 \supset \mathcal{C}_1 \supset \mathcal{C}_2 \supset \ldots \supset \mathcal{C}_{v-1} \supset \mathcal{C}_v.$$

Key Insight

- Let $c_i(x)$, $i = 0, 1, \ldots, v-1$, be BCH codewords in $\mathcal{C}_0$. If

$$\begin{bmatrix} 1 & 1 & \ldots & 1 \\ \alpha(x) & \alpha(x^2) & \ldots & \alpha(x^{v-1}) \\ \vdots & \vdots & \ddots & \vdots \\ \alpha(x^{v-1}) & \alpha(x^{2(v-1)}) & \ldots & \alpha(x^{(v-1)(v-1)}) \end{bmatrix} \begin{bmatrix} c_0(x) \\ c_1(x) \\ \vdots \\ c_{v-1}(x) \end{bmatrix} = \begin{bmatrix} c'_0(x) \\ c'_1(x) \\ \vdots \\ c'_{v-1}(x) \end{bmatrix}$$

  lies in $\mathcal{C}_1$, where $\mathcal{C}_1 \subset \mathcal{C}_0$. Then, $c_i(x)$, $i = 0, 1, \ldots, v-1$, must lie in $\mathcal{C}_1$.

- *Proof:* It suffices to show that each $c_i(x)$, $i = 0, 1, \ldots, v-1$, contains all roots of $\frac{g_1(x)}{g_0(x)}$. Let γ be a root of $\frac{g_1(x)}{g_0(x)}$. Evaluating the above equations with $x = \gamma$, we obtain

$$\begin{bmatrix} 1 & 1 & \ldots & 1 \\ \gamma & \gamma^2 & \ldots & \gamma^{v-1} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma^{v-1} & \gamma^{2(v-1)} & \ldots & \gamma^{(v-1)(v-1)} \end{bmatrix} \begin{bmatrix} c_0(\gamma) \\ c_1(\gamma) \\ \vdots \\ c_{v-1}(\gamma) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

  Since the connection matrix is non-singular, the above equality indicates $c_i(\gamma) = 0, \quad i = 0, 1, \ldots, v-1$. Therefore, $c_i(x)$, $i = 0, 1, \ldots, v-1$, all divide $g_1(x)$.

GII-BCH for Flash Memory

- Flash page size (current): 16KB + 8–12% overhead
- GII-BCH configuration

2KB data $a_0(x)$	parity $p_0(x)$
2KB data $a_1(x)$	parity $p_1(x)$
2KB data $a_2(x)$	parity $p_2(x)$
2KB data $a_3(x)$	parity $p_3(x)$
2KB data $a_4(x)$	parity $p_4(x)$
2KB data $a_5(x)$	parity $p_5(x)$
2KB data $a_6(x)$	parity $p_6(x)$
2KB data $a_7(x)$	parity $p_7(x)$

GII-BCH for Flash Memory

- BCH code with 2KB data operates in GF(2^{15}).
  –15-bit full multiplier uses 224 XOR's
  –14-bit full multiplier uses 273 XOR's (for 1KB data size)
  –16-bit full multiplier uses >512 XOR's (for 4KB data size).
- Equal data size must be enforced to accommodate 4KB random read. This results in unequal word length and uneven read location.
- For random read, superior error-correcting performance is only achieved at twice long latency (using an extra read) over the un-nested scheme.