

# Forward Error Correction (FEC) techniques for optical communications

IEEE 802.3 High-Speed Study Group  
Plenary meeting, Montreal  
July 1999

Kamran Azadet, Mark Yu  
Lucent Technologies  
Phone: 732-949-7280  
Email: ka@lucent.com  
or myu@lucent.com

# Outline

- Why FEC ?
- Cyclic codes - Overview
- Algorithms and implementation
  - Encoder
  - Decoder
- Performance analysis
- Latency issues
- Summary

# Why FEC ?

# Why FEC ?

- ❑ FEC lowers BER by a great deal for low overhead
- ❑ E.g. for RS(255,239) overhead is 6%: For input BER= $10^{-4}$ , output BER= $10^{-14}$  !
- ❑ Very low overhead codes exist (less than 0.1%) and have been widely used in other optical communication applications
  - ❑ Possible trade-off between performance and overhead (illustrated in this talk)
  
- ❑ Overhead
  - ❑ increased rate -> slightly reduces the overall coding gain
  - ❑ electronics cost (MUX/DEMUX if line rate is increased)
  - ❑ adds FEC circuitry (encoder is simpler than decoder)

# How to incorporate FEC in 10GbE ?



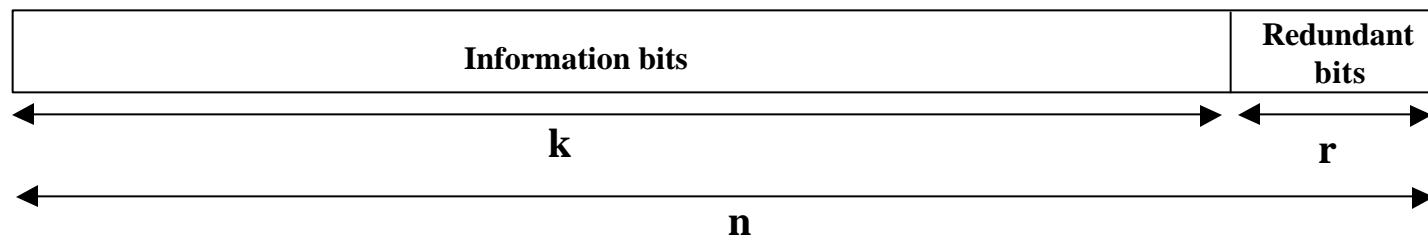
- ❑ WAN applications incorporate already FEC, solutions are currently proprietary. There is however an on-going effort to adopt an “in-band” FEC standard in 10G SONET (most likely based on a BCH-3 code)
- ❑ There may be ways to exploit already existing overhead in the Ethernet LAN (IPG?)
  - ❑ However available overhead may be limited
  - ❑ Ethernet packets are variable size -> decoding would be more complicated !
- ❑ For increased performance, line rate could be slightly increased
  - ❑ fixed length blocks are preferable
- ❑ FEC can be combined with constrained codes such as 8b/10b (or 16b/18b), or with zero-overhead scrambled codes.

# Cyclic codes - overview

# Some FEC definitions

## □ Systematic block code

- An encoder accept  $k$  information symbols and appends separately a set of  $r$  redundant symbols (parity bits) derived from the information symbols. Note the information word is not disturbed in any way in the encoder. This code is called  $(n, k)$  code, where  $n=k+r$ .
- Since code space is enlarged and codewords are a constrained subset, error detection and correction is possible based on maximum likelihood decoding.



- Hamming distance of two codewords: number of non-zero elements in the codeword formed by their difference
- Hamming distance of a code =  $d$  (is the minimum distance between two codewords)
- Singleton bound:  $d \leq n - k + 1$

# Some FEC definitions - cont.

## ❑ Cyclic code

- ❑ An  $(n, k)$  code is cyclic if a cyclic shift of a codeword is also a codeword.
- ❑ Associated with a generating polynomial (usually implemented as LFSR)
- ❑ Two representations of a codeword  $w = (w_0, w_1, \dots, w_{n-2}, w_{n-1}) \leftrightarrow$   
$$W(x) = w_0 + w_1x + \dots + w_{n-2}x^{n-2} + w_{n-1}x^{n-1}$$

## ❑ Examples of popular systematic cyclic block codes: binary BCH, Reed-Solomon

- ❑ Reed-Solomon is famous for its ability to correct “bursty” errors.

# Galois fields

- ❑ Theory developed by french mathematician E. Galois
- ❑ Galois Fields: for a prime  $p$ ,  $GF(p)$  denote the integer ring (over  $[+,x]$ ) modulo  $p$
- ❑ Example:  $GF(2)$ : addition is modulo-2 addition / XOR-gate  
multiplication is modulo-2 multiplication / AND-gate

Simple codes such as Hamming codes or binary BCH codes are built over  $GF(2)$ .

- ❑ Galois Fields definition can be generalized to  $GF(p^m)$ , for instance  $GF(2^8)$ , where words are **Bytes** and not bits (this is used in Reed-Solomon codes)

# Hamming code

- ❑ The well-known  $(2^n-1, 2^n-n-1)$  code with Hamming distance of 3 is capable of correcting 1 error
- ❑ Error syndrome gives the error location
  - ❑ Easy to decode, but only correct one error
- ❑ Are there codes able to correct more than one error and easy to decode?

# RS code

- ❑ RS(n,k) consists of all polynomials of degree at most n that are multiples of the generator polynomial  
 $g(x) = (x-1)(x-\alpha^1)\dots(x-\alpha^{2t-1})$ , where  $\alpha$  is a primitive element of  $GF(2^m)$
- ❑ Minimum distance between codewords is  $d=n-k=2t+1$ , thus RS(n,k) can correct t symbols
- ❑ 2t Erasures can be corrected (an Erasure is an error with known position)

# BCH codes

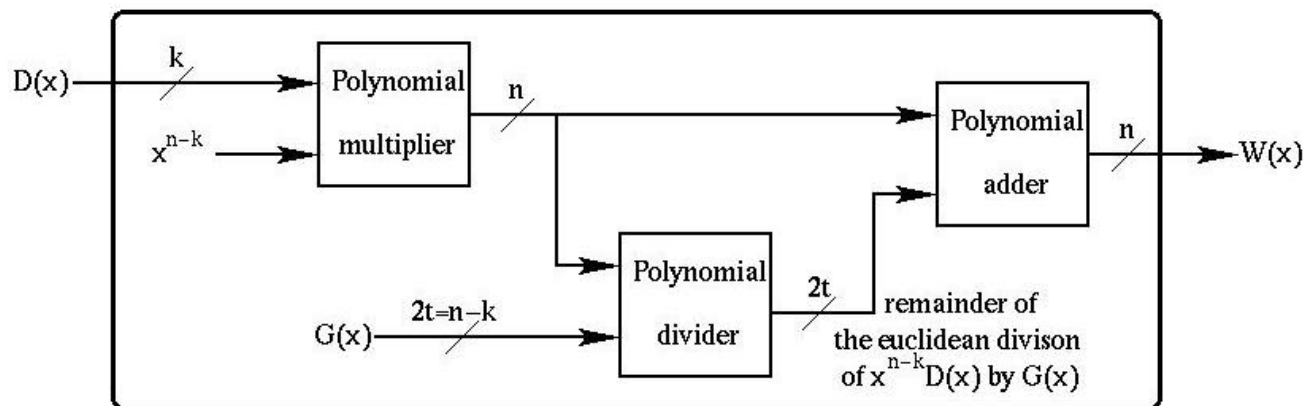
- ❑ The cyclic code of length  $n$  over  $GF(q)$ ,  $n$  coprime with  $q$ , whose generator polynomial is  $g(x)$ . RS code is a subclass of BCH code
- ❑ Binary BCH code: a BCH code on  $GF(2)$ , i.e.,  $q=2$
- ❑ Binary BCH codes are most popular
- ❑ BCH codes usually correct few bits in a block (1-error correcting or Hamming, 2-error correcting, 3-error correcting, etc.)
- ❑ Encoding and decoding procedures are similar to those of RS code, except there is no need to compute error magnitude for binary BCH code

# Algorithms and implementation

## - Encoder -

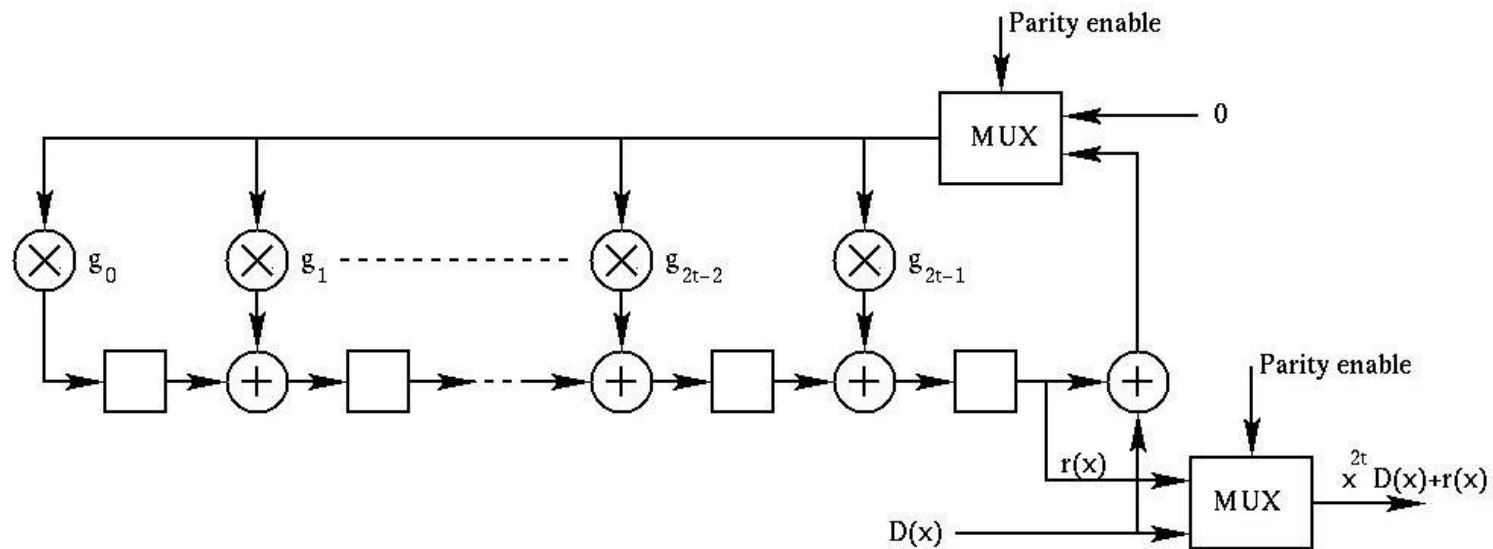
# RS encoding

- Send  $W(x) = x^{n-k}D(x) - r(x)$ , where  $D(x)$  is the information symbols, and  $r(x)$  is the remainder of  $x^{n-k}D(x)$  divided by the generating polynomial  $G(x)$ 
  - $W(x)$  is a codeword, because  $G(x)$  divides  $W(x)$
  - $W(x)$  is systematic, i.e., redundant symbols are added after information symbols



# RS encoding (2)

- RS is a cyclic code and often implemented as a LFSR with  $GF(2^m)$  operators



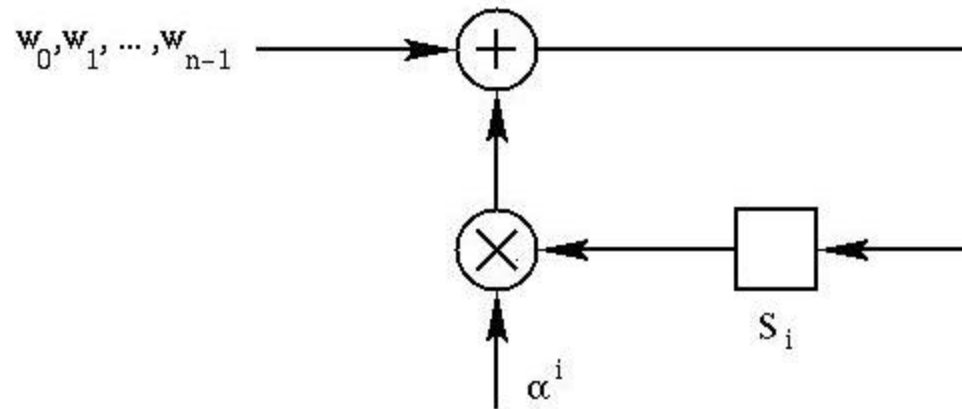
# Algorithms and implementation

## - Decoder -

# RS decoding

First some more definitions:

- ❑ Error word polynomial  $e(x) = e_0 + e_1x + \dots + e_{n-1}x^{n-1}$
- ❑ The received word polynomial is given by  $w(x) = c(x) + e(x)$
- ❑ Syndromes
$$S_j = w(\alpha^j) = \sum_{i=0}^{n-1} e_i \alpha^{ij} \text{ for } j = 0, \dots, 2t-1$$
- ❑ Syndromes can be computed using the Horner algorithm:



# RS decoding - cont.

- Suppose that there are  $r$  errors,  $r \leq t$ , occurred
  - at locations  $i_1, \dots, i_r$  (let  $X_i = \alpha^{i_1}$ )
  - with values  $e_{i_1}, \dots, e_{i_r}$  (let  $Y_i = e_{i_1} \alpha^{i_1} = e_{i_1} X_i$ )

- Reformulate  $S_j = \sum_{i=1}^r e_i a^{i_j}$   $\implies$

$$\begin{bmatrix} S_0 \\ S_1 \\ \dots \\ S_{2t-1} \end{bmatrix} = \begin{bmatrix} X_1^0 & X_2^0 & \dots & X_r^0 \\ X_1^1 & X_2^1 & \dots & X_r^1 \\ \dots & \dots & \dots & \dots \\ X_1^{2t-1} & X_2^{2t-1} & \dots & X_r^{2t-1} \end{bmatrix} \cdot \begin{bmatrix} Y_0 \\ Y_1 \\ \dots \\ Y_r \end{bmatrix}$$

2t equations for 2r unknowns,  $r \leq t$

# RS decoding - cont.

- Define error locator polynomial

$$s(x) = (1-xX_1)(1-xX_2)\dots(1-xX_r) = s_r x^r + \dots + s_1 x + 1$$

- Inverse of zeros of  $s(x)$  gives error locations

- Multiplying both sides with  $Y_i X_i^{j+r}$  and letting  $x=X_i^{-1} \Rightarrow s(X_i^{-1})=0$ , for all  $i,j$ , we get

$$\begin{pmatrix} S_1 & S_2 & \dots & S_r \\ S_2 & S_3 & \dots & S_{r+1} \\ \dots & \dots & \dots & \dots \\ S_r & S_{r+1} & \dots & S_{2r-1} \end{pmatrix} \begin{pmatrix} s_r \\ s_{r-1} \\ \dots \\ s_1 \end{pmatrix} = \begin{pmatrix} -S_{r+1} \\ -S_{r+2} \\ \dots \\ -S_{2r} \end{pmatrix}$$

$$M \equiv$$

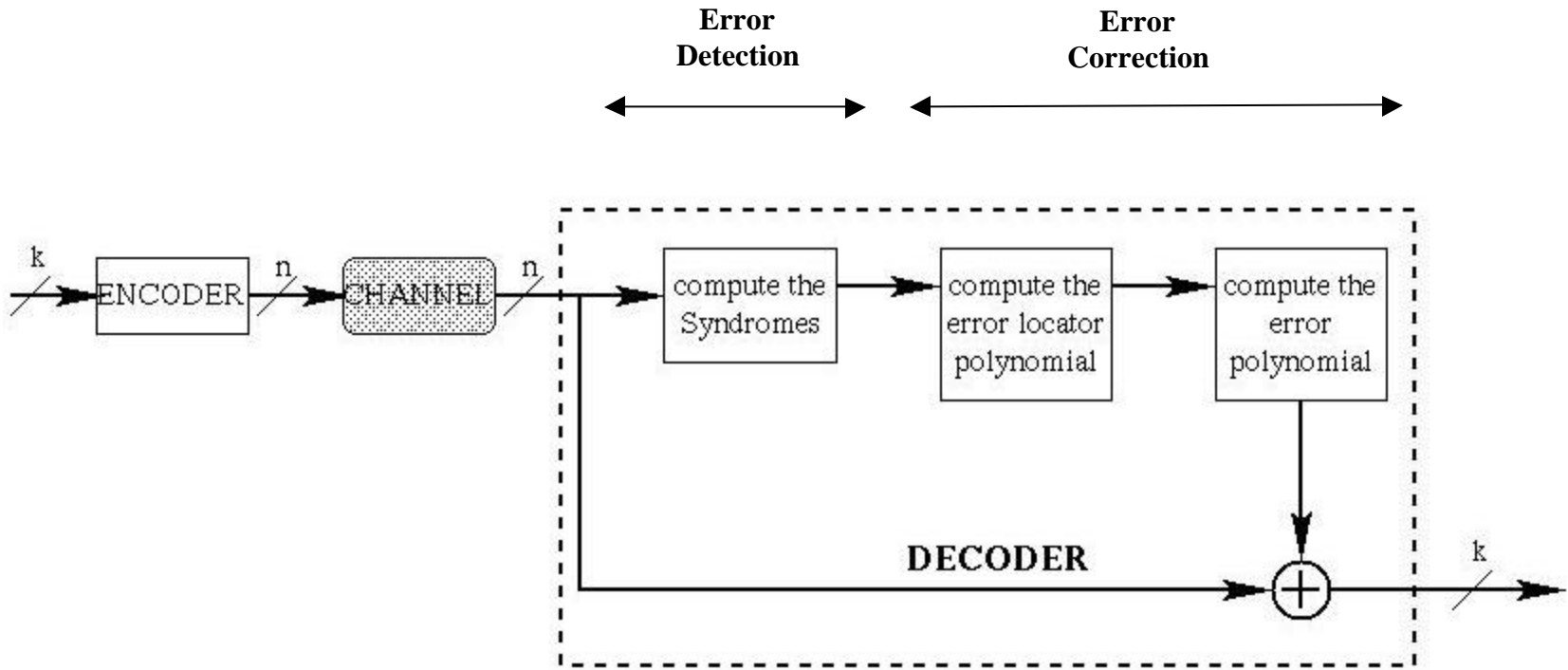
- $M$  is singular if  $m > r$  !

- So we can determine how many errors have occurred

# RS decoding steps

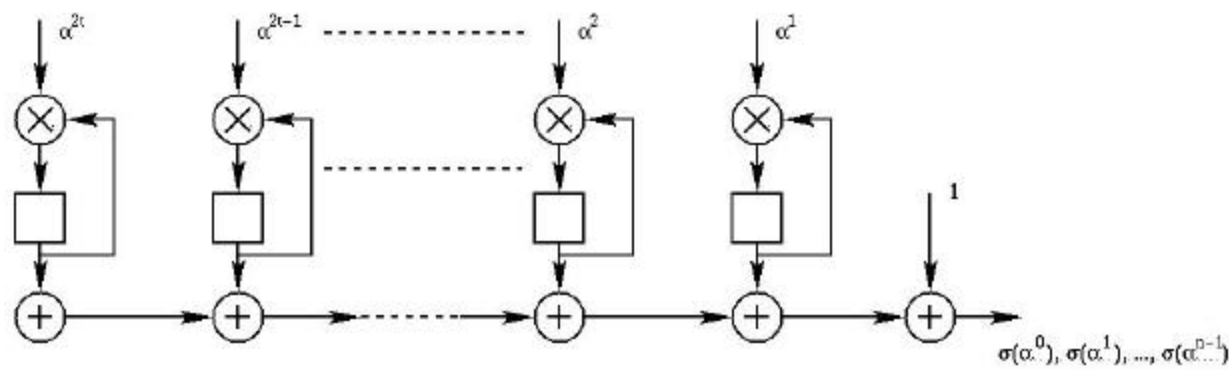
1. Compute Syndromes  $S_i = w(\alpha^i)$ , for  $i=1, \dots, 2t$
2. Determine the maximum number  $r$  so that  $M$  is nonsingular.  
 $r$  is then the number of errors occurred.
3. Find the coefficients of the error-locator polynomial (solving the key equation) by computing
$$M^{-1} \times \begin{bmatrix} S_{r+1} \\ S_{r+2} \\ \dots \\ S_{2r} \end{bmatrix}$$
4. Solve  $s(x)=0$
5. Find error locations
6. Compute error magnitude at error locations and correct the errors

# RS decoder architecture



# More on RS decoding algorithms

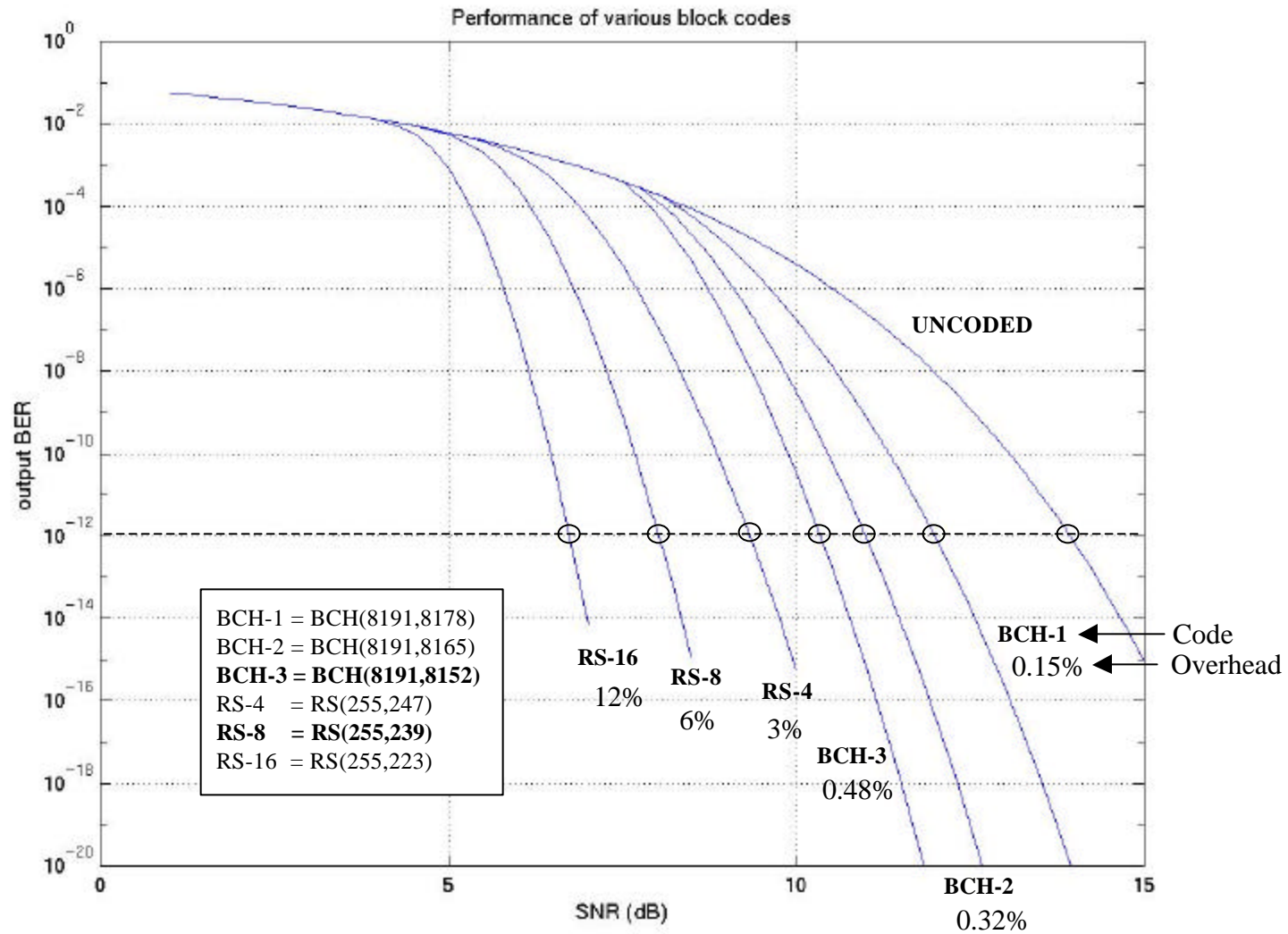
- ❑ Clearly finding  $|M|$  and  $M^{-1}$  is not easy for large  $t$
- ❑ Fortunately, RS has been researched and widely used in many applications for a long time, therefore there are more efficient algorithms for each task.
- ❑ E.g.,
  - ❑ to solve the key equation ----> Euclidean algorithm, Berlekamp algorithm, Berlekamp-Massey algorithm
  - ❑ to find roots of  $s(x)$  ----> Chien Search algorithm



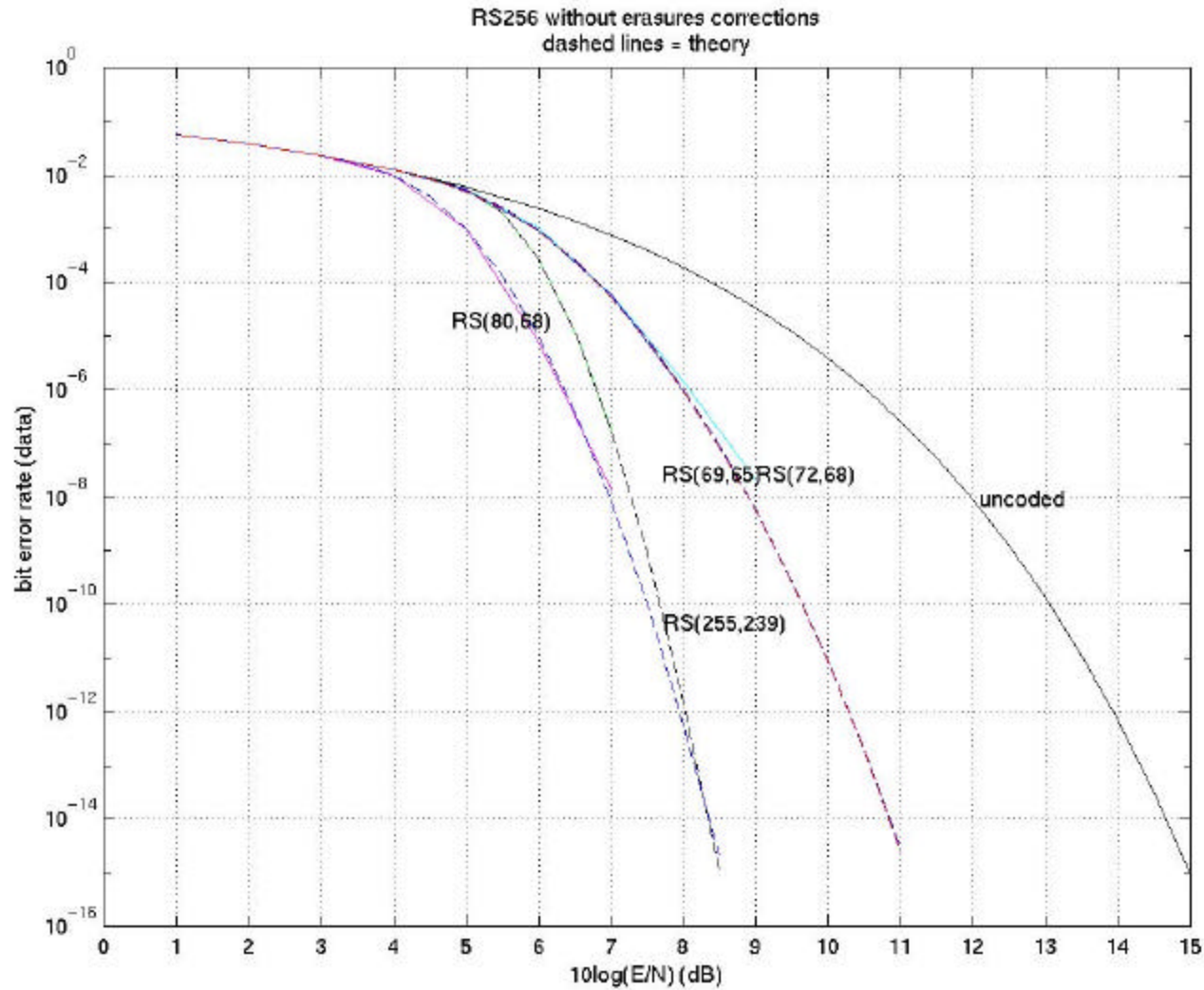
- ❑ to compute error magnitude -----> Forney algorithm

# Performance analysis

# Performance analysis



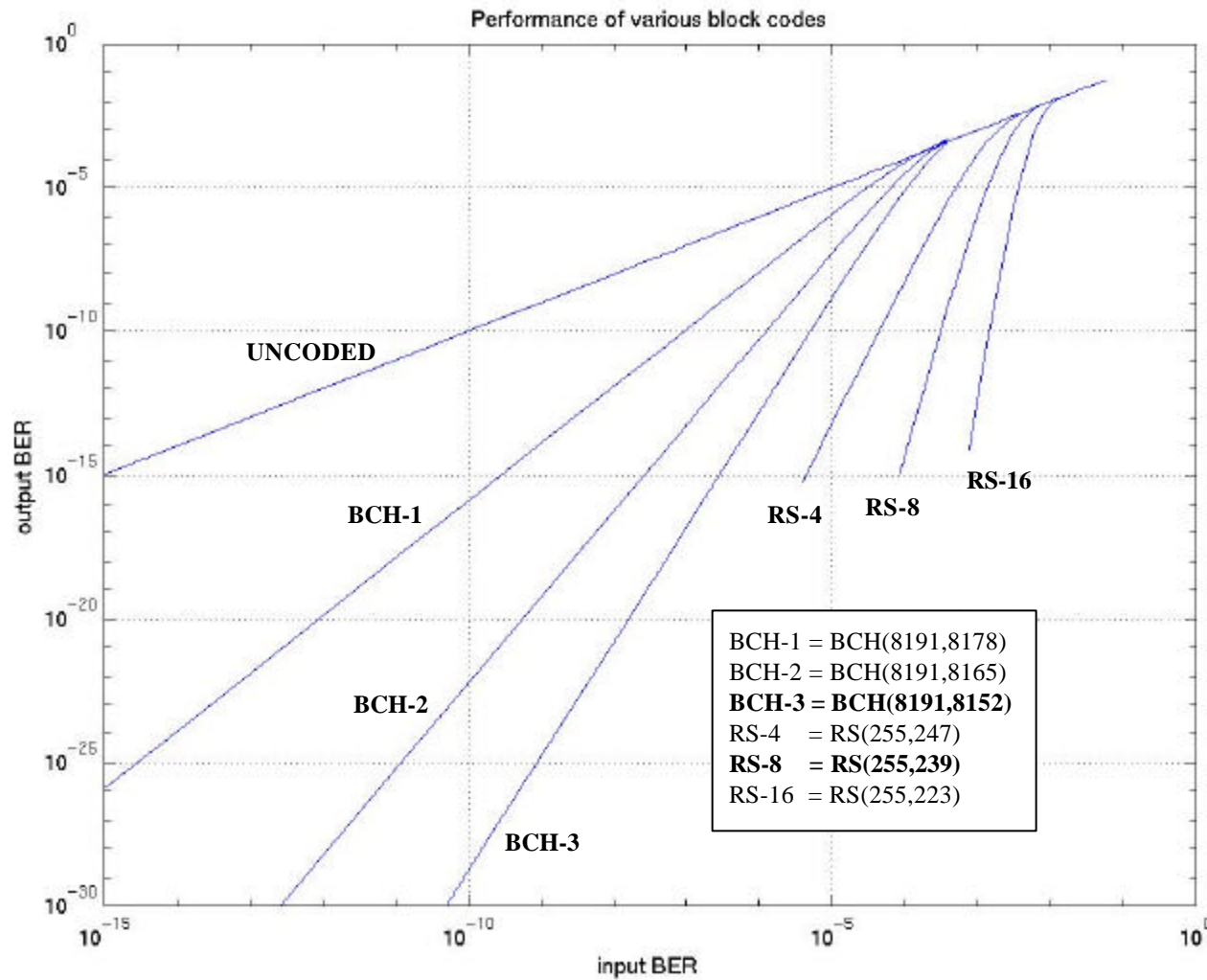
# RS codes: simulated results



# An alternative representation

- ❑ The above coding gains (in dB) do not map directly in optical gain (in dBm)
- ❑ Optical gain depends on channel and receiver response
- ❑ Usually optical power is proportional to detector current, and not to electrical power !
- ❑ Input BER versus output BER is a **unique representation** of the performance of a given code. It does not depend on optical response. It is very useful for comparing the performance of different codes with variable size / overhead.

# Input BER versus Output BER



# Latency issues

# Latency issues

- ❑ Latency is obviously implementation dependant
- ❑ In many cases, total latency of encoding + decoding can be on the order of  $2n-3n$  ( $n$  block size)
- ❑ For instance for RS(255,239) total latency could potentially be  $\sim 3 \times 255 \times 8 \times 100 \text{ps} < 1 \mu\text{s}$
- ❑ Some existing FEC chips used in WAN have  $30 \mu\text{s}$  total latency (longer blocks, lower overhead)
- ❑ Propagation time in fiber:
  - ❑ 300m       $\rightarrow 1 \mu\text{s}$
  - ❑ 3km         $\rightarrow 10 \mu\text{s}$
  - ❑ 30km       $\rightarrow 100 \mu\text{s}$
- ❑ In LAN applications, smaller block sizes are preferable (to minimize buffer size).

# Summary

# Summary

- ❑ Conventional FEC schemes can achieve 6dB coding gain or more without need for “soft decisions”
- ❑ Error correction on the decode side is up to implementers
- ❑ Parity check could be used to perform bit alignment (in a serial approach) or de-skewing (in a WDM approach)
- ❑ Parity check could be used to indicate link quality
- ❑ Clear trade-off between FEC performance on one hand and [complexity,latency,overhead] on the other hand
- ❑ Simple FEC codes exist (Hamming, binary BCH)
- ❑ BCH & RS codes can be (are) implemented in generic CMOS technology