CHƯƠNG 7: MÃ HÓA KÊNH

Đặng Lê Khoa
Email: dlkhoa@fetel.hcmus.edu.vn

Facuty of Electronics & Telecommunications, HCMUS

Nội dung trình bày:

- Mã hóa kênh (Channel coding)
- Mã hóa khối (Block codes)

  + Mã lập (Repetition Code)

  + Hamming codes

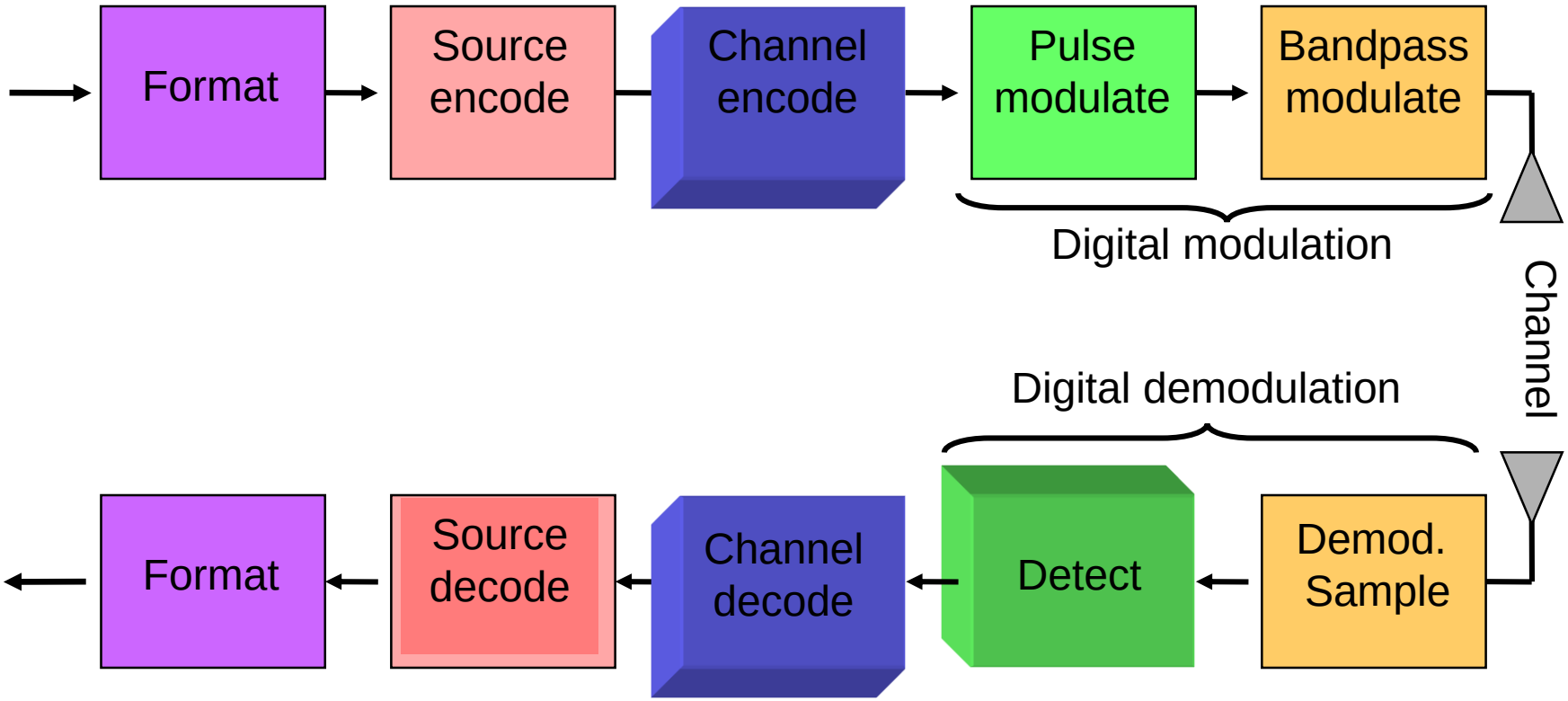
  + Cyclic codes

    * Reed-Solomon codes
- Mã hóa chập (Convolutional codes)

  + Encode

  + Decode
- Điều chế mã lưới (Trellis Coded Modulation)

Sơ đồ khối DCS



Channel coding là gì?

- Tín hiệu truyền qua kênh truyền sẽ bị ảnh hưởng bởi nhiễu, can nhiễu, fading… là tín hiệu đầu thu bị sai.
- Mã hóa kênh: dùng để bảo vệ dữ liệu không bị sai bằng cách thêm vào các bit dư thừa (redundancy).
- Ý tưởng mã hóa kênh là gởi một chuỗi bit có khả năng sửa lỗi
- Mã hóa kênh không làm giảm lỗi bit truyền mà chỉ làm giảm lỗi bit dữ liệu (bảng tin)
- Có hai loại mã hóa kênh cơ bản là: Block codes và Convolutional codes

Các loại mã hóa sửa sai

- Mã lập (Repetition Code)
- Mã khối tuyến tính (Linear Block Code), e.g. Hamming
- Mã vòng (Cyclic Code), e.g. CRC
- BCH và RS Code
- Mã chập (Convolutional Code)
  - Truyền thống, giải mã Viterbi
  - Mã Turbo
  - Mã LDPC
- Coded Modulation
  - TCM
  - BICM

Mã lập

- *For the basis* $|0\rangle$ *and* $|1\rangle$ *, we do*

$$|0\rangle \longrightarrow |0\rangle|0\rangle|0\rangle \longrightarrow |000\rangle$$
$$|1\rangle \longrightarrow |1\rangle|1\rangle|1\rangle \longrightarrow |111\rangle$$

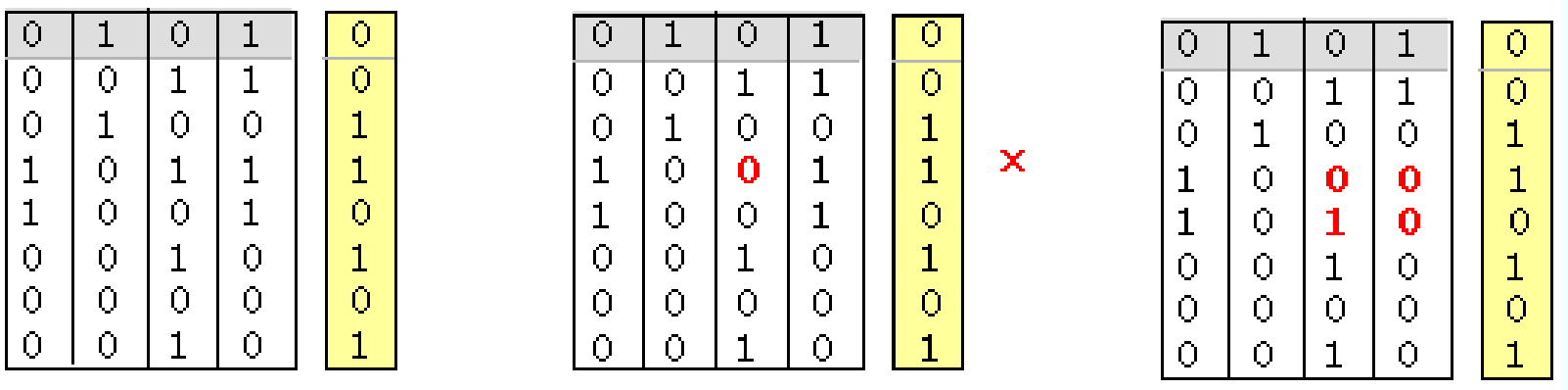
- *Recovery can be done by majority voting*

Recovered state

Initial State	$q_1 = (q_2 = q_3)?q_2 : q_1$	$q_2 = (q_1 = q_3)?q_1 : q_2$	$q_3 = (q_1 = q_2)?q_1 : q_3$
$\|000\rangle$	$\|000\rangle$	$\|000\rangle$	$\|000\rangle$
$\|001\rangle$	$\|001\rangle$	$\|00\mathbf{1}\rangle$	$\|000\rangle$
$\|010\rangle$	$\|010\rangle$	$\|000\rangle$	$\|000\rangle$
$\|011\rangle$	$\|111\rangle$	$\|1\mathbf{1}1\rangle$	$\|111\rangle$
$\|100\rangle$	$\|000\rangle$	$\|000\rangle$	$\|000\rangle$
$\|101\rangle$	$\|101\rangle$	$\|1\mathbf{1}1\rangle$	$\|111\rangle$
$\|110\rangle$	$\|110\rangle$	$\|110\rangle$	$\|111\rangle$
$\|111\rangle$	$\|111\rangle$	$\|1\mathbf{1}1\rangle$	$\|111\rangle$

Kiểm tra chẵn lẻ (Parity Check)

- Thêm 1 bit để xor các bit có kết quả là 0
  - Dữ liệu truyền, sửa lỗi, không thể sửa lỗi



0	1	0	1	0
0	0	1	1	0
0	1	0	0	1
1	0	1	1	1
1	0	0	1	0
0	0	1	0	1
0	0	0	0	0
0	0	1	0	1

0	1	0	1	0
0	0	1	1	0
0	1	0	0	1
1	0	**0**	1	1
1	0	0	1	0
0	0	1	0	1
0	0	0	0	0
0	0	1	0	1

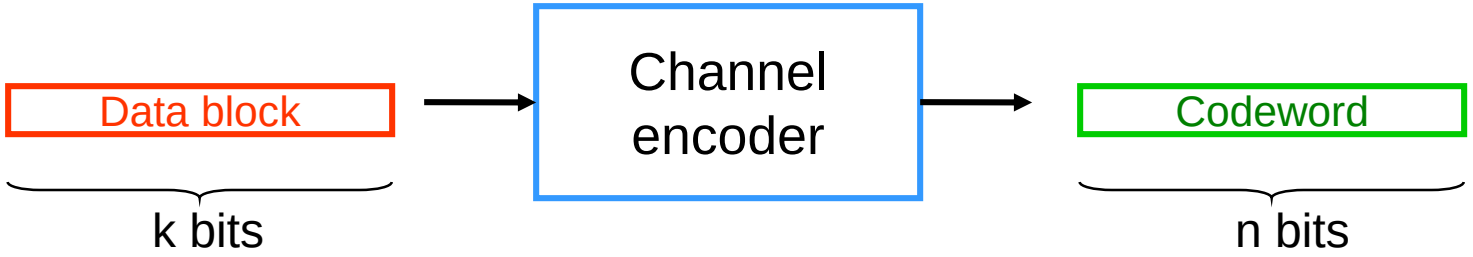
x

0	1	0	1	0
0	0	1	1	0
0	1	0	0	1
1	0	**0**	**0**	1
1	0	**1**	**0**	0
0	0	1	0	1
0	0	0	0	0
0	0	1	0	1

  - Kiểm tra hàng và cột
- Ứng dụng: ASCII, truyền dữ liệu qua cổng nối tiếp

Mã khối tuyến tính (Linear block codes)

- Chuỗi bit thông tin được chia thành từng khối k bit.
- Mỗi khối được encode thành từng khối lớn hơn có n bit.
- Các bit được mã hóa và gửi trên kênh truyền.
- Quá trình giải mã được thực hiện ở phía thu.



$n-k$ Redubi

$R_c = \frac{k}{n}$ Codate

Linear block codes – cont'd

- Khoảng cách Hamming giữa hai vector **U** và **V**, là số các phần tử khác nhau.

- Khoảng các tối thiểu của mã hóa khối là

- Ví dụ: Tính khoảng cách Hamming của $C_{1:}$ 101101 và C_2 :001100

Giải: Vì $101101 \oplus 001100 = 100001$ =>d_{12}=W(1000001)=2

- => Ta có thể giải mã để sửa sai bằng cách chọn codewords có dmin

Linear block codes – cont'd

- Khả năng phát hiện lỗi được cho bởi:

$$e = d_{\min} - 1$$

- Khả năng sửa lỗi **t** của mã hóa được định nghĩa là số lỗi tối đa có thể sửa được trên 1 từ mã (codeword)

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$$

Linear block codes – cont'd

- Encoding trong bộ mã hóa khối (n,k)

$$\mathbf{U} = \mathbf{mG}$$

$$(u_1, u_2, \ldots, u_n) = (m_1, m_2, \ldots, m_k) \cdot \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \vdots \\ \mathbf{V}_k \end{bmatrix}$$

$$(u_1, u_2, \ldots, u_n) = m_1 \cdot \mathbf{V}_1 + m_2 \cdot \mathbf{V}_2 + \ldots + m_k \cdot \mathbf{V}_k$$

  - Các hàng của G thì độc lập tuyến tính.

Linear block codes – cont'd

- Example: Block code (6,3)

$$G = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} 110100 \\ 011010 \\ 101001 \end{bmatrix}$$

Message vector	Codeword
0 0 0	0 0 0 0 0 0
1 0 0	1 1 0 1 0 0
0 1 0	0 1 1 0 1 0
1 1 0	1 0 1 1 1 0
0 0 1	1 0 1 0 0 1
1 0 1	0 1 1 1 0 1
0 1 1	1 1 0 0 1 1
1 1 1	0 0 0 1 1 1

Linear block codes – cont'd

- Mã hóa khối (n,k)
  - k phần tử đầu tiên (hoặc cuối cùng) trong từ mã là các bit thông tin.

$$G = [P \; I_k]$$

I_k = ma trận đơn vị kích thước k x k

P_k = ma trận kích thước k x (n-k)

$$U = (u_1, u_2, \ldots, u_n) = (p_1, p_2, \ldots, p_{n-k}, x_1, x_2, \ldots, x_k)$$

parity bits

Linear block codes – cont'd

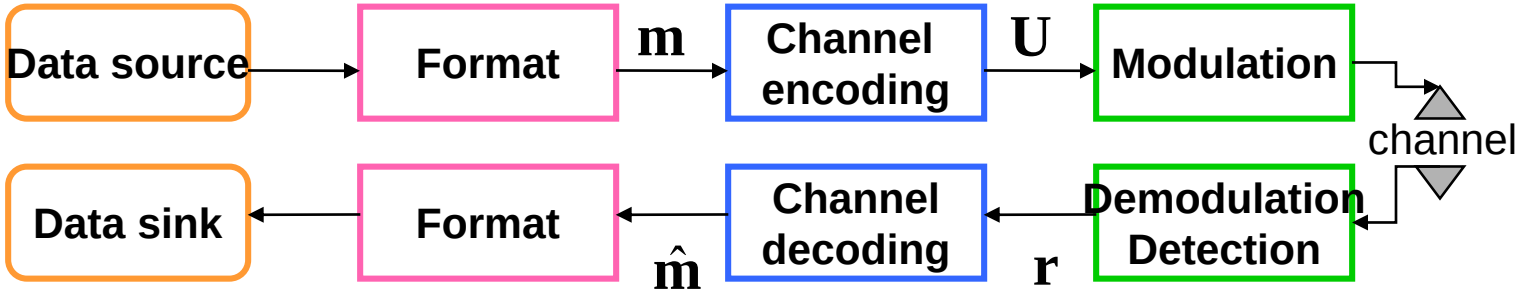
- Đối với bất kỳ mã hóa khối tuyến tính, chúng ta có một ma trận $\mathbf{H}_{(n-k)\times n}$. Các hàng của ma trận này trực giao với ma trận $\mathbf{G}$:

$$\mathbf{G}\mathbf{H}^T = \mathbf{0}$$

- **H** được gọi là ma trận kiểm tra parity và các hàng của chúng độc lập tuyến tính.
- Đối với mã hóa khối truyến tính:

$$\mathbf{H} = [\mathbf{I}_{n-k} \vdots \mathbf{P}^T]$$

Linear block codes – cont'd



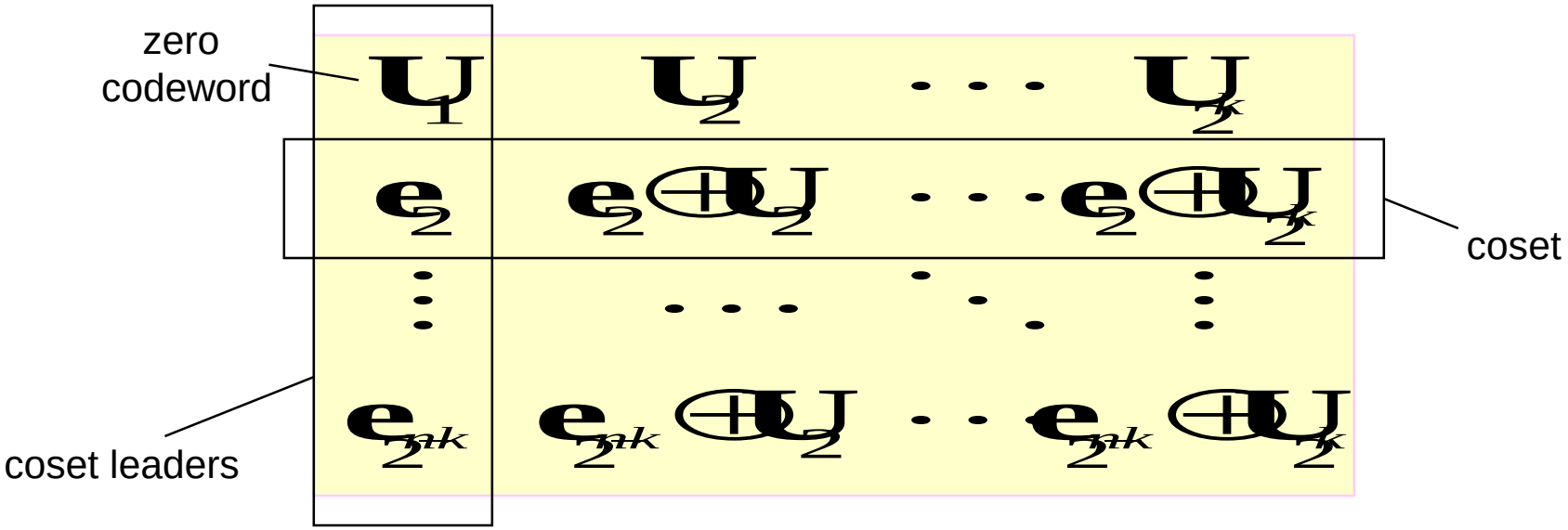
$$\mathbf{r} = \mathbf{U} + \mathbf{e}$$

- Kiểm tra đặc trưng:
  - **S** là đặc trưng của **r**, tương ứng với error pattern **e**.

$$\mathbf{S} = \mathbf{r}\mathbf{H}^T = \mathbf{e}\mathbf{H}^T$$

Linear block codes – cont'd

- Mảng tiêu chuẩn
  - Hàng $i = 2, 3 .. 2^{n-k}$ được tạo thành bằng cách cộng U với pattern $\mathbf{e}_i$



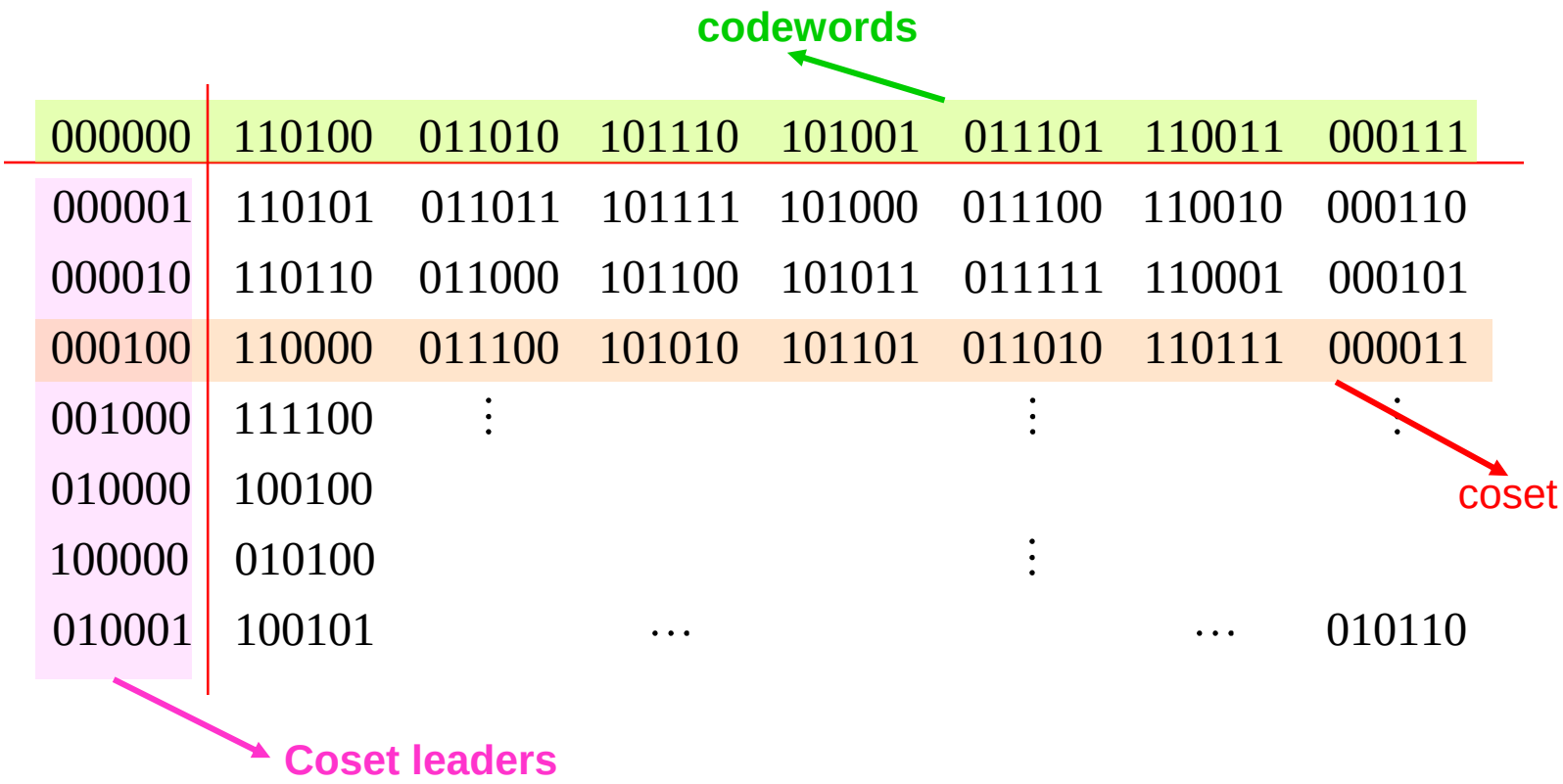
Linear block codes – cont'd

- Mảng tiêu chuẩn và đặc trưng bảng giải mã
  1. Tính $\mathbf{S} = \mathbf{r}\mathbf{H}^T$
  2. Tìm coset chính $\hat{\mathbf{e}} = \mathbf{e}_i$, tương ứng với $\mathbf{S}$.
  3. Tính $\hat{\mathbf{U}} = \mathbf{r} + \hat{\mathbf{e}}$ và tương ứng với $\hat{\mathbf{m}}$.

  - Chú ý:
    - Nếu $\hat{\mathbf{e}} = \mathbf{e}$, error được sửa.
    - Nếu $\hat{\mathbf{e}} \neq \mathbf{e}$, bộ giải mã không thể phát hiện lỗi.

Linear block codes – cont'd

- Ví dụ: Mảng chuẩn cho mã (6,3)



Linear block codes – cont'd

Error pattern	Syndrome
000000	000
000001	101
000010	011
000100	110
001000	001
010000	010
100000	100
010001	111

$\mathbf{U} = (1011100)$ was sent.

$\mathbf{r} = (0011100)$ is received.

- The syndrome of $\mathbf{r}$ is computed:

$$\mathbf{S} = \mathbf{r}\mathbf{H}^T = (0011100)\mathbf{H}^T = (100)$$

- Error pattern corresponding to this syndrome is:

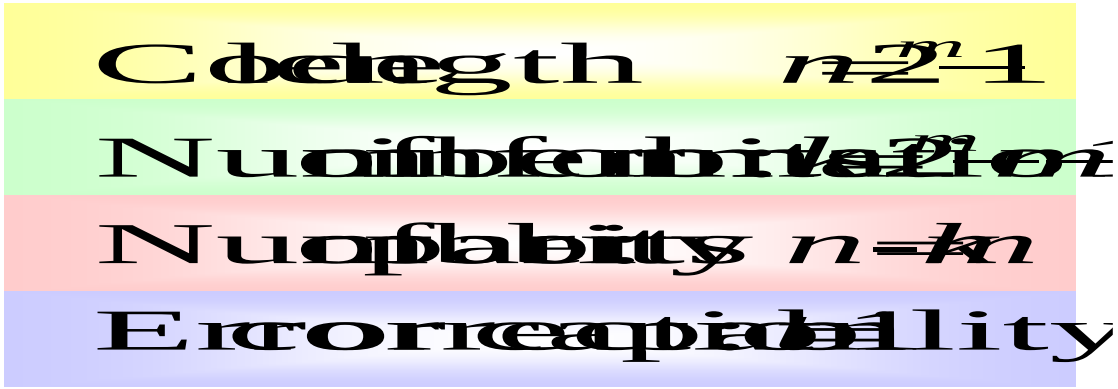
$$\hat{\mathbf{e}} = (100000)$$

- The corrected vector is estimated:

$$\hat{\mathbf{U}} = \mathbf{r} + \hat{\mathbf{e}} = (0011100) + (100000) = (1011100)$$

Hamming codes

- Hamming codes
  - Là trường hợp riêng của linear block codes
  - Diễn tả theo hàm của một số nguyên $m \geq 2$.



Hamming codes

- Example: Systematic Hamming code (7,4)

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} = [I_3 \quad P^T]$$

$$G = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} = [P \quad I_4]$$

Mã hóa Hamming

- Mã hóa: H(7,4)

  Nhiều phép kiểm tra tổng

  Message=[a b c d]

  r= (a+b+d) mod 2

  s= (a+b+c) mod 2

  t= (b+c+d) mod 2

  Code=[**r s** a **t** b c d]

  Message=[1 0 1 0]

  r=(1+0+0) mod 2 =1

  s=(1+0+1) mod 2 =0

  t=(0+1+0) mod 2 =1

  Code=[**1** **0** 1 **1** 0 1 0]

- Tốc độ mã: 4/7
  - Càng nhỏ, nhiều redundance bit, được bảo vệ tốt hơn.
  - Khác biệt giữa phát hiện và sửa lỗi

Hamming codes

- Example: Systematic Hamming code (7,4)

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} = [I_3 \quad P^T]$$

$$G = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} = [P \quad I_4]$$

Ví dụ mã hóa Hamming

H(7,4)

- Ma trận sinh G: đầu tiên là ma trân đơn vị 4x4
- Dữ liệu truyền là vector p
- Vector truyền x (G=[I/P])

$$\mathbf{p} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

$$\mathbf{G} := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix}$$

$$\mathbf{Gp} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \mathbf{x}$$

- Vector nhận r và vector lỗi e

$$\mathbf{r} = \mathbf{x} + \mathbf{e}_i$$

Sửa lỗi

- Nếu không có lỗi, vector đặc trưng (syndrome) z=zeros

$$\mathbf{Hr} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \mathbf{z}$$

- Nếu có 1 lỗi ở vị trí thứ 2

$$\mathbf{Hr} = \mathbf{H}(\mathbf{x} + \mathbf{e}_i) = \mathbf{Hx} + \mathbf{He}_i$$

- Vector đặc trưng z là $= 0 + \mathbf{He}_i = \mathbf{He}_i$

$$\mathbf{r} = \mathbf{x} + \mathbf{e}_2 = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

$$\mathbf{Hr} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \mathbf{z}$$

tương với cột thứ 2 của H. Vậy, lỗi được phát hiện ở vị trí thứ 2 và có thể sửa lại cho đúng.

Độ lợi mã hóa (Coding Gain)

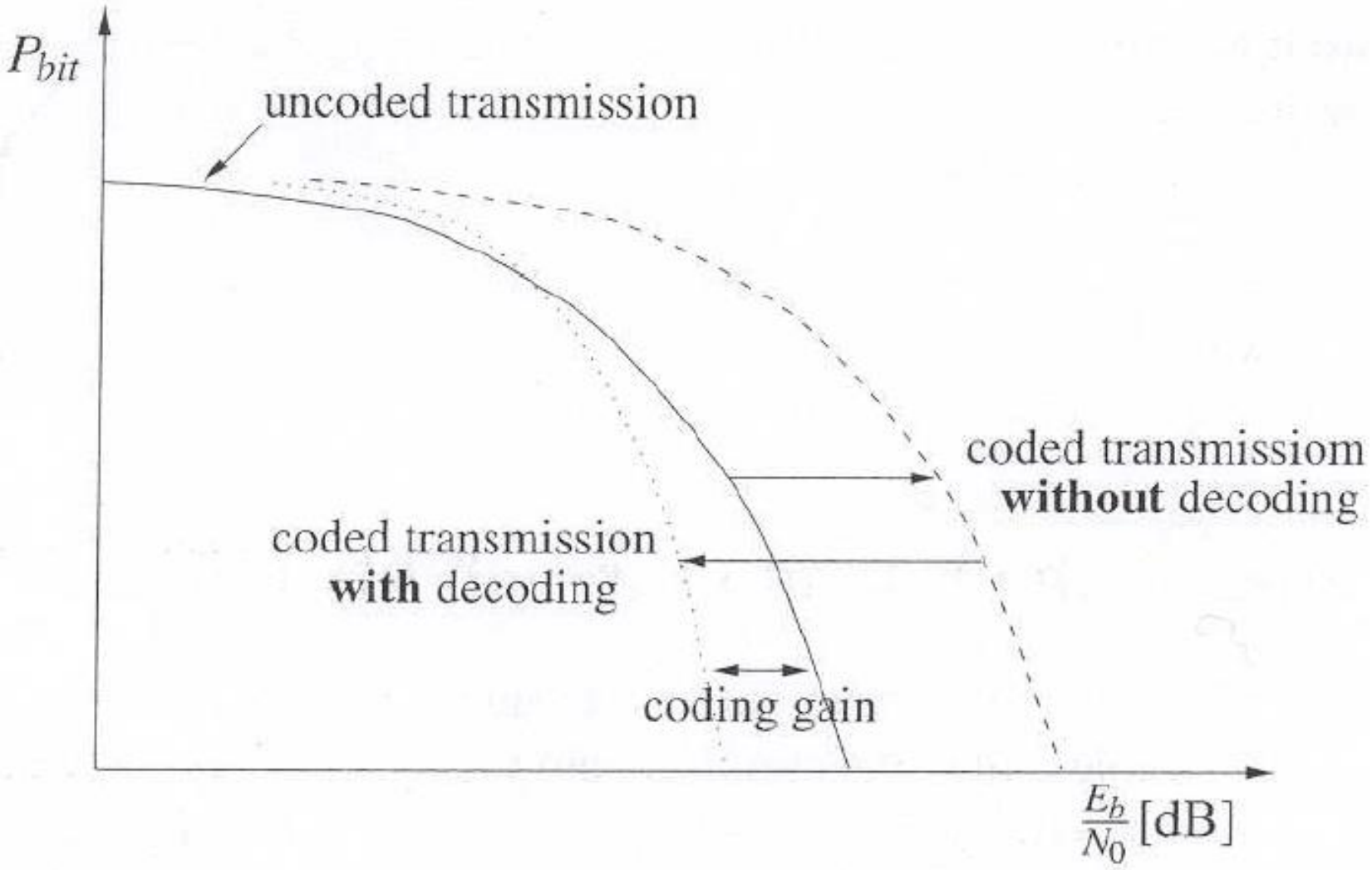
- Tốc độ mã R=k/n, k: số symbol dữ liệu, n tổng symbol
- SNR từ và SNR của bit

$$kE_b = nE_s \rightarrow E_b = \frac{E_s}{R}.$$

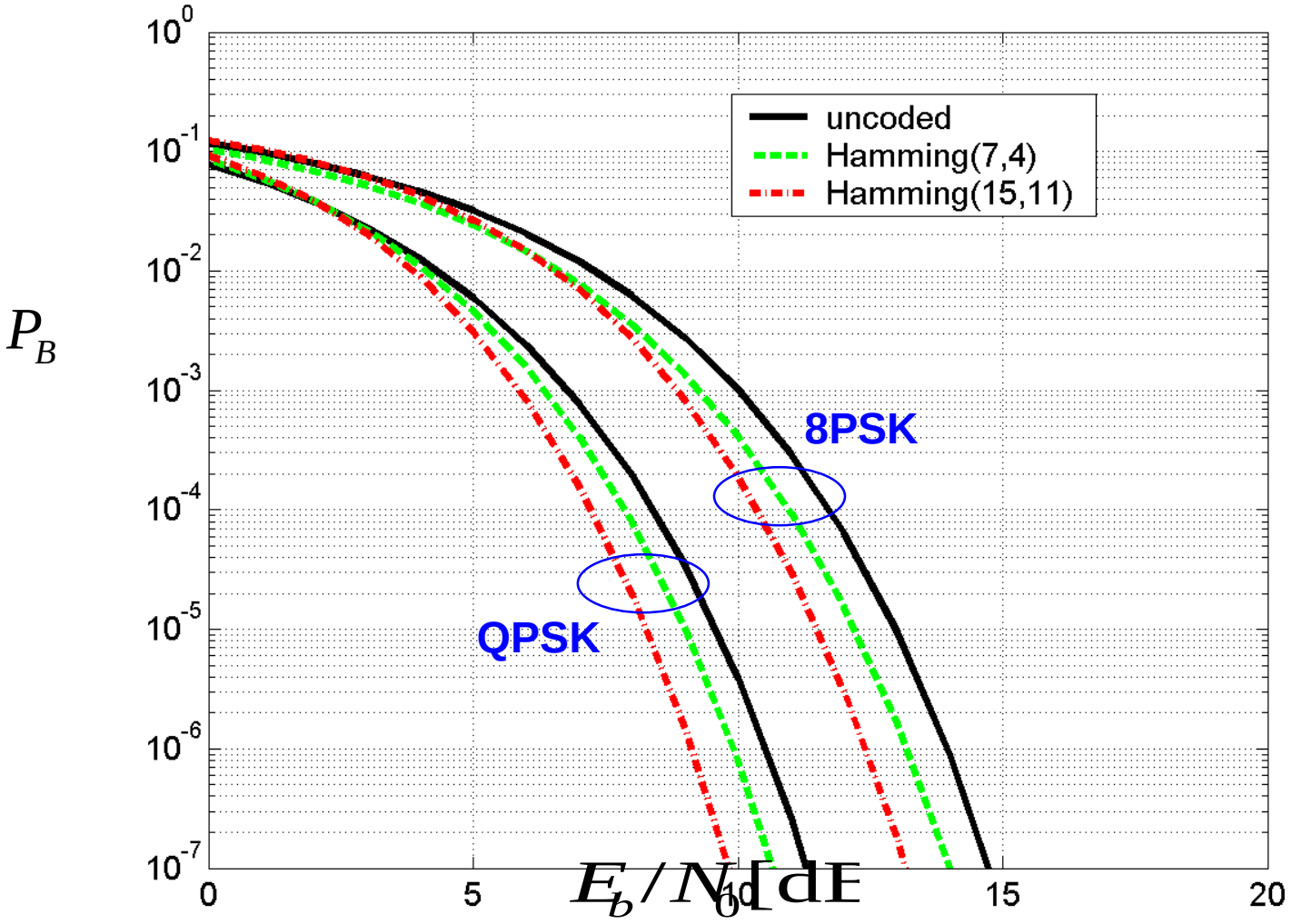
where E_b is the energy per information bit and E_s the energy per received symbol.

- Với một sơ đồ mã hóa, độ lợi mã hóa tại một sắc xuất lỗi bit được định nghĩa là sự khác biệt giữa năng lượng cần thiết cho 1 bit thông tin đã mã hóa để đạt được sắc xuất lỗi cho trước và truyền dẫn không mã hóa

Ví dụ độ lợi mã hóa



Example of the block codes

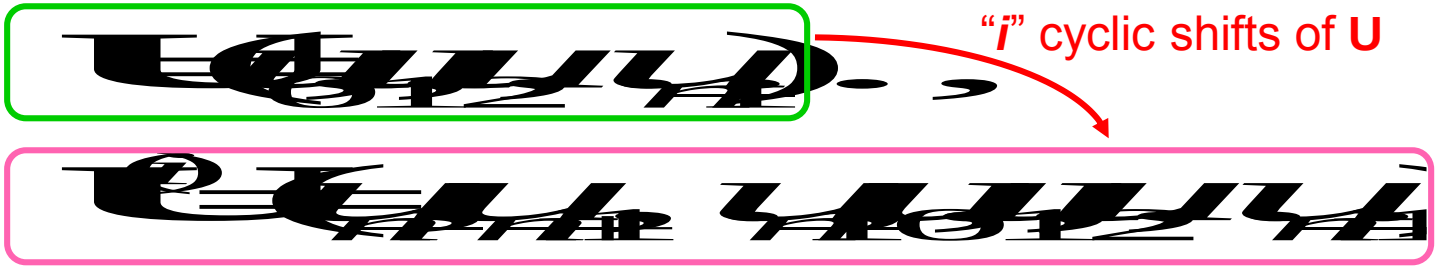


Cyclic code

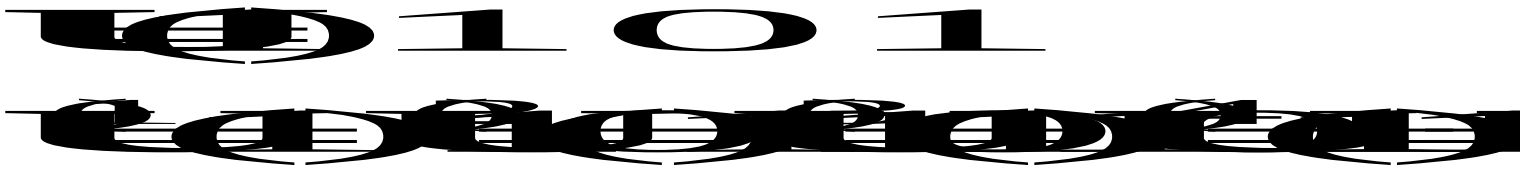
- Cyclic codes được quan tâm và quan trọng vì
  - Dựa trên cấu trúc đại số và có thể ứng dụng rộng rãi.
  - Dễ dàng thực hiện bằng thanh ghi dịch (*shift register)*
  - Được ứng dụng rộng rãi trong thực nghiệm
- Trong thực nghiệm, cyclic codes được sử dụng để phát hiện lỗi (Cyclic redundancy check, CRC)
  - Được sử dụng trong mạng chuyển mạch gói
  - Khi có 1 lỗi được phát hiện ở bộ nhận, chúng sẽ được yêu cầu truyền lại.
  - ARQ (Automatic Repeat-reQuest)

Cyclic block codes

- Một mã tuyến tính (n,k) được gọi là Cyclic code nếu khi dịch vòng 1codeword thì đó cũng là codeword.



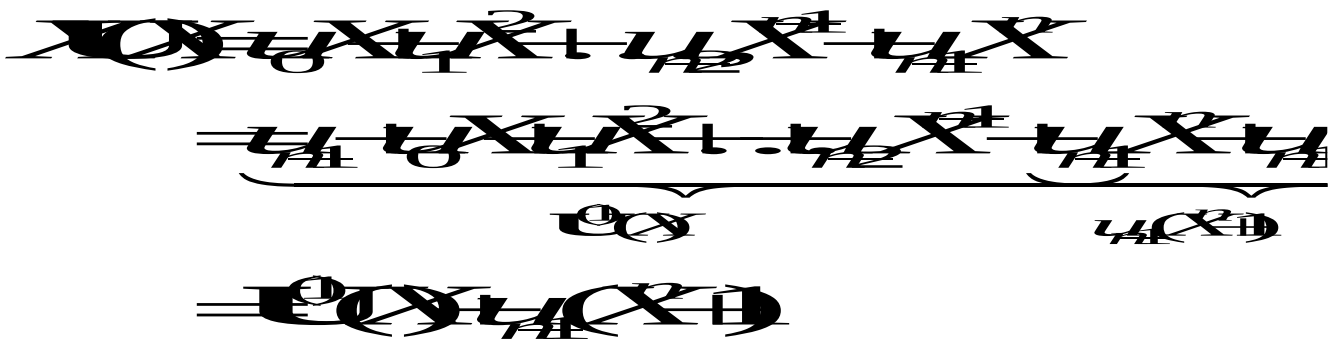
- Ví dụ:



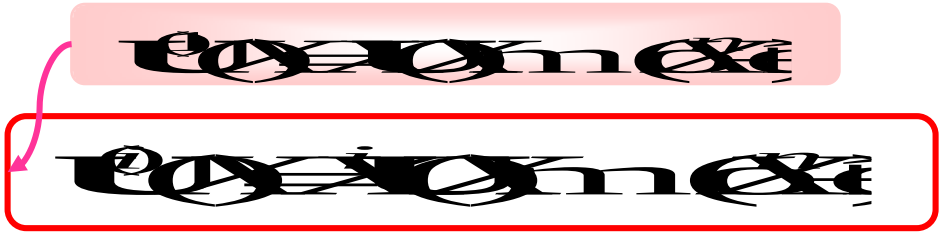
Cyclic block codes

- Cấu trúc đại số của Cyclic codes, suy ra codewords được sinh ra từ

- Mối quan hệ giữa codeword và thanh ghi dịch:



  - Vậy: By extension



Cyclic block codes

- Thuật toán mã hóa Cyclic code (n,k):
  1. Nhân thông tin với chuỗi $\mathbf{m}(X)$ bằng X^{n-k}
  2. Chia kết quả bước 1 với đa thức sinh $\mathbf{g}(X)$. Lấy $\mathbf{p}(X)$ là phần dư
  3. Thêm $\mathbf{p}(X)$ vào $X^{n-k}\mathbf{m}(X)$ để tạo thành codeword $\mathbf{U}(X)$

Cyclic block codes

- Example: For the systematic (7,4) Cyclic code with generator polynomial $g(X) = 1 + X + X^3$

1. Find the codeword for the message $\mathbf{m} = (1\ 0\ 1\ 1)$

$n = 7,\ k = 4,\ n - k = 3$

$\mathbf{m} = (1\ 0\ 1\ 1) \Rightarrow m(X) = 1 + X^2 + X^3$

$X^{n-k} m(X) = X^3 m(X) = X^3(1 + X^2 + X^3) = X^3 + X^5 + X^6$

Divide $X^{n-k} m(X)$ by $g(X)$

$$X^3 + X^5 + X^6 = \underbrace{(1 + X + X^2 + X^3)}_{\text{quotient } q(X)} \underbrace{(1 + X + X^3)}_{\text{generator } g(X)} + \underbrace{1}_{\text{remainder } p(X)}$$

Form the codeword polynomial:

$U(X) = p(X) + X^3 m(X) = 1 + X^3 + X^5 + X^6$

$\mathbf{U} = (\underbrace{1\ 0\ 0}_{\text{parity bits}}\ \underbrace{1\ 0\ 1\ 1}_{\text{message bits}})$

Cyclic block codes

- Find the generator and parity check matrices, **G** and **H**, respectively.

$$g(X) = 1 + X + X^3 \quad (n, k) = (7, 4)$$

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Not in systematic form.
We do the following:

- row(a) + row(b) → row(b)
- row(a) + row(b) → row(a)

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} = \underbrace{\mathbf{P}}_{} \; \underbrace{\mathbf{I}_{4\times 4}}_{}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} = \underbrace{\mathbf{I}_{3\times 3}}_{} \; \underbrace{\mathbf{P}^T}_{}$$

Cyclic block codes

- Giải mã Cyclic code:
  - Từ mã ở bộ thu được cho bởi

Received codeword

Error pattern

  - Đặc trưng ở phần dư có được bằng cách chia chuỗi nhận cho đa thức sinh:

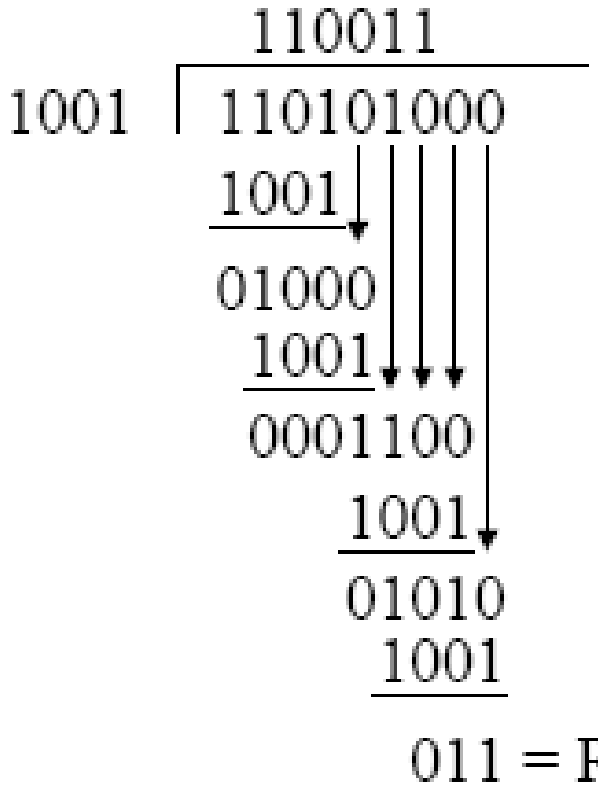
Syndrome

  - Với đặc trưng và mảng tiêu chuẩn, lỗi sẽ được ước lượng.

Ví dụ CRC

r = 3, G = 1001

M = 110101 => $M2^r$ = 110101000



Modulo 2 Division

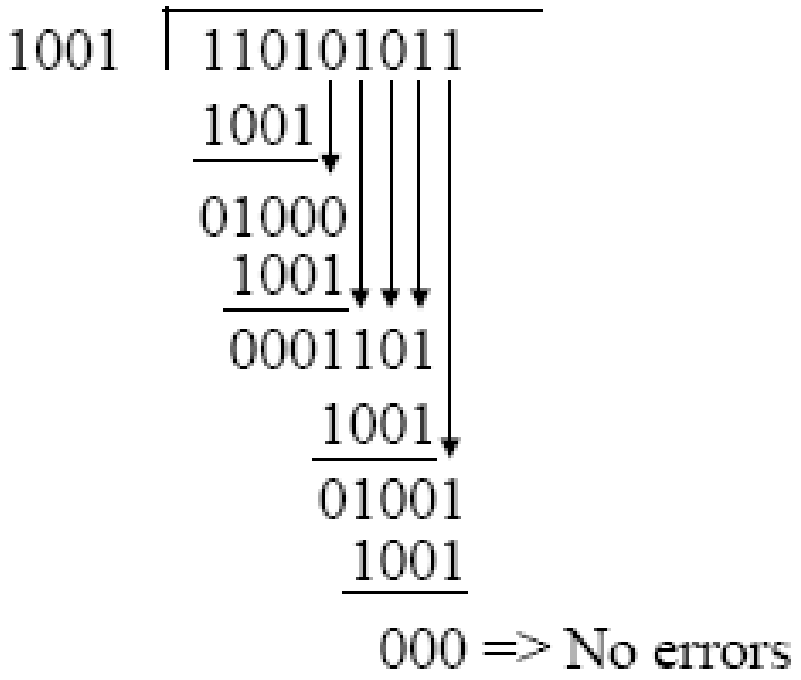
011 = R (3 bits)

Checking for errors

- Let T' be the received sequence
- Divide T' by G
  - If remainder = 0 assume no errors
  - If remainder is non zero errors must have occurred

Example:
Send T = 110101011
Receive T' = 110101011
(no errors)

No way of knowing how many errors occurred or which bits are In error



Khả năng của CRC

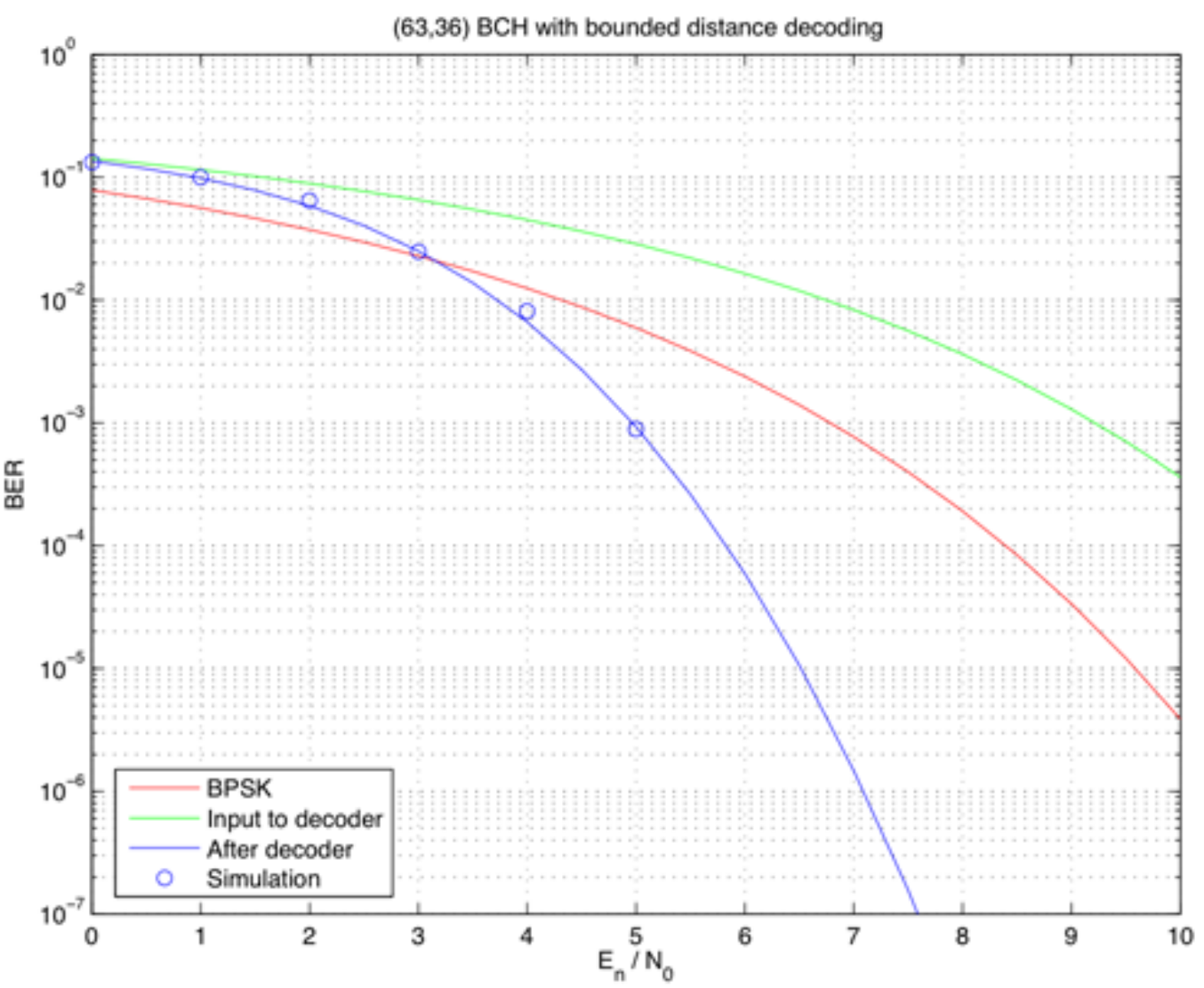
- Một lỗi E(X) không thể phát hiện khi chúng chia hết cho G(x). Ngược lại, thì có thể phát hiện lỗi.
- Có khả năng mạnh mẽ trong phát hiện lỗi

BCH Code

- Bose, Ray-Chaudhuri, Hocquenghem
  - Có khả năng sửa được nhiều lỗi
  - Dễ dàng thực hiện mã hóa và giải mã
- Các chuẩn trong công nghiệp

  - (511, 493) mã hóa BCH trong ITU-T. Chuẩn H.261- một chuẩn mã hóa video được sử dụng cho video conferencing và video phone.
  - (40, 32) mã hóa BCH trong ATM (Asynchronous Transfer Mode)

BCH Performance



Reed-Solomon Codes

- Một trường hợp riêng của non-binary BCH
- Được ứng dụng rộng rãi
  - Storage devices (tape, CD, DVD…)
  - Wireless or mobile communication
  - Satellite communication
  - Digital television/Digital Video Broadcast(DVB)
  - High-speed modems (ADSL, xDSL…)