Codificação de Canal: Correção de erro por codificação em bloco.

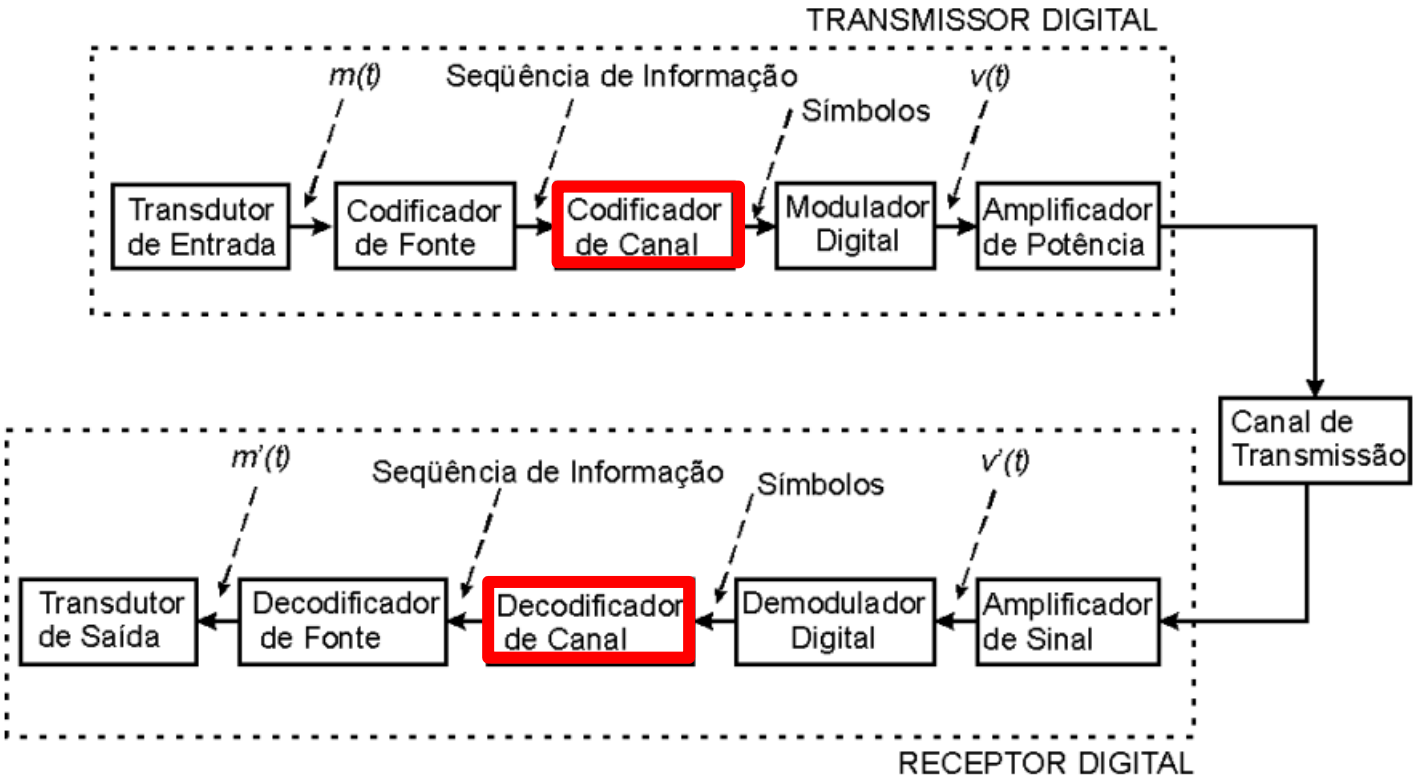
Departamento de Eletrônica e Computação

Centro de Tecnologia

ELC1120 – TELECOMUNICAÇÕES II

Profa. Candice Müller Prof. Fernando DeCastro

Codificador de Canal



- **Codificador de Canal**: A Codificação de Canal é o processo responsável em um sistema digital por manter a taxa de erro dentro de um limite máximo aceitável pelo usuário.

Codificador de Canal

- Quando informação digital é enviada através de um canal de transmissão, ruído e interferência inerentes a qualquer canal prático degradam o sinal de forma que os dados recebidos contêm erros.

- O usuário do sistema de transmissão digital geralmente estabelece uma taxa de erro máxima aceitável – uma mensagem errada em $1x10^6$ mensagens recebidas, por exemplo (i.e., uma taxa de erro de 1×10^{-6}) – acima da qual os dados recebidos não são considerados utilizáveis pelo usuário. Esta taxa de erro máxima aceitável depende da informação que transita pelo canal.

- A título de comparação, a taxa máxima de erro permitida para transmissão de voz através de telefonia celular é muito maior do que a taxa exigida para transmissão de dados, por exemplo (porque, na pior das hipóteses, mesmo sob uma alta taxa de erro e consequente distorção, o sistema auditivo humano é capaz de compreender o significado das frases pelo contexto da conversa, o que já não acontece quando dois computadores trocam dados).

- O Codificador de Canal é o responsável em um sistema digital por manter a taxa de erro dentro de um limite máximo aceitável pelo usuário.

- A possibilidade do uso de codificação para controlar com eficiência a taxa de erro de um sistema de comunicação digital foi demonstrada por Shannon em 1948, através do Teorema Fundamental de Shannon, já discutido no Cap II das notas de aula:

Teorema Fundamental de Shannon:

Se a taxa (= velocidade) de transmissão R [bits/s] da informação a ser enviada pelo canal é menor que uma quantidade C [bits/s] denominada de Capacidade do Canal, então a comunicação através do canal pode ser estabelecida com probabilidade de erro tão baixa quanto se deseje, através do uso de um código adequado para correção de erro.

- Vimos que o Teorema Fundamental de Shannon estabelece a existência de um código corretor de erro tal que a informação pode ser transmitida através do canal de comunicação com uma taxa de erro arbitrariamente baixa, caso a taxa de transmissão *R* [*bits/s*] seja menor ou igual à capacidade do canal *C* [*bits/s*].

- Estudaremos os membros mais importantes de duas grandes classes de códigos para correção de erro:

os códigos de bloco e os códigos convolucionais.

- Um código de bloco pode ser considerado como um operador $\boldsymbol{\theta}\{\cdot\}$, tal que $\boldsymbol{C} = \boldsymbol{\theta}\{\boldsymbol{X}\}$, onde:

  $\boldsymbol{X} = \{\underline{x}_i\} = \{\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{M-1}\}$ é o conjunto de M possíveis **mensagens** $\underline{x}_i$ a serem codificadas e

  $\boldsymbol{C} = = \{\underline{c}_i\} = \{\underline{c}_0, \underline{c}_1, \ldots, \underline{c}_{M-1}\}$ é o conjunto de M possíveis **palavras-código** $\underline{c}_i$ resultantes da codificação, com $i = 0,1, \ldots, M-1$.

- O operador $\boldsymbol{\theta}\{\cdot\}$ efetua um mapeamento unívoco entre cada mensagem $\underline{x}_i$ e a respectiva palavra-código $\underline{c}_i$.

- O **conjunto de caracteres do código** ou **alfabeto do código** é o conjunto $\mathbf{A} = \{a_0, a_1, \ldots, a_{D-1}\}$ composto por D elementos, de cuja composição são formadas cada mensagem e sua respectiva palavra-código (para códigos binários $\mathbf{A} = \{0,1\}$).

- Cada mensagem $\underline{x_i} \in \boldsymbol{X}$ é considerada como um vetor $\underline{x_i} = [x_{i(k-1)} x_{i(k-2)} \dots x_{i1} x_{i0}]$ de k componentes, $x_{ij} \in \boldsymbol{A}, j = k-1, k-2, \dots 1,0.$

- Visto que os k componentes da i-ésima mensagem $\underline{x_i}$ pertencem ao alfabeto $\boldsymbol{A}$, é válida a relação de pertinência $\underline{x_i} \in \boldsymbol{A}^k$.

- Da mesma forma, cada palavra-código $\underline{c_i} \in \boldsymbol{C}$ é considerada como um vetor $\underline{c_i} = [c_{i(n-1)} c_{i(n-2)} \dots c_{i1} c_{i0}]$ de n componentes $c_{ij} \in \boldsymbol{A}, j = n-1, n-2, \dots 1,0.$

- Visto que os n componentes da i-ésima palavra-código $\underline{c_i}$ pertencem ao alfabeto $\boldsymbol{A}$, é válida a relação de pertinência $\underline{c_i} \in \boldsymbol{A}^n$.

Por exemplo: a palavra-código binária 0101, de $n = 4$ bits, é representada pelo vetor $\underline{c}$ = [0 1 0 1],

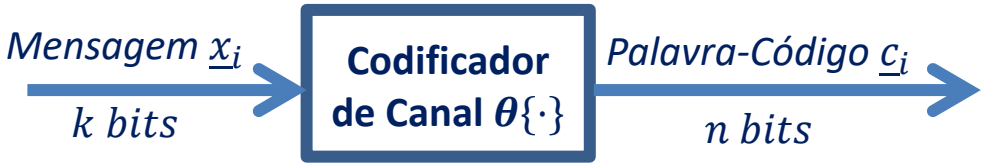
$$\underline{c} \in \boldsymbol{A}^4$$

$$\mathbf{A} = \{0,1\}$$

- Um código de bloco binário $\boldsymbol{\theta}\{\cdot\}$ mapeia um conjunto $\boldsymbol{X} = \{\underline{x}_i\} = \{\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{M-1}\}$ de $M = 2^k$ mensagens binárias, cada uma delas com k bits, em um conjunto $\boldsymbol{C} = \{\underline{c}_i\} = \{\underline{c}_0, \underline{c}_1, \ldots, \underline{c}_{M-1}\}$ palavras-código binárias, cada uma delas com n bits, onde $n > k$.

- Um código de bloco $\boldsymbol{\theta}\{\cdot\}$ binário cujas mensagens a serem codificadas apresentam k bits e são mapeadas em palavras-código de n bits é representado pelo operador $\boldsymbol{\theta}(n,k)\{\cdot\}$ ou simplesmente $\boldsymbol{\theta}(n,k)$.

- Um código $\boldsymbol{\theta}(n,k)$ é **sistemático** quando cada palavra-código de n bits é formada pelos k bits da respectiva mensagem associada, acrescidos (por justaposição) de r bits adicionais destinados ao controle e correção de erros, denominados de **bits de paridade**.



- Portanto, em um código sistemático cada mensagem contendo k bits de informação é expandida em uma palavra-código de $n = k + r$ bits onde **r é o número de bits representativos da informação redundante adicionada visando o controle e correção de erro.**

- Um código $\boldsymbol{\theta}(n, k)$ é **não-sistemático** quando nas palavras-códigos de n bits não aparecem explicitamente representados os k bits de informação da respectiva mensagem associada.

- É possível converter um código não-sistemático em um código sistemático. Em função disto, nossa atenção será dada aos códigos sistemáticos.

- Tanto o código não sistemático, quanto o código convertido em um código sistemático, possuem a mesma capacidade de correção e de detecção, por isso são ditos códigos equivalentes.

- **Exemplo 1 :** Por exemplo, o código $\boldsymbol{\theta}(4,3)$ do *codebook* abaixo é sistemático, porque cada palavra-código $\underline{c_i}$ de $n = 4$ bits é formada pela justaposição de 1 bit de paridade aos $k = 3$ bits de informação da mensagem $\underline{x_i}$ associada.

- Observe que, como $n > k$, no conjunto de todas as 2^npossíveis palavras-códigos de n bits existem 2^n-2^k elementos que não são associados a qualquer elemento do conjunto $\boldsymbol{X} = \{\underline{x_i}\} = \{\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_{M-1}\}$ de $M = 2^k$ mensagens binárias de k bits.

- Por exemplo, para o código binário $\boldsymbol{\theta}(4,3)$ ao lado, existem $2^n - 2^k = 2^4 - 2^3 = 8$ elementos no conjunto de todas as $2^n = 2^4 = 16$ possíveis palavras-códigos de 4 bits sem associação com qualquer mensagem do conjunto

$$\boldsymbol{X} = \{000,001,010,011,100,101,110,111\}.$$

Mensagem x_i	Palavra-código c_i
000	0000
001	0011
010	0101
011	0110
100	1001
101	1010
110	1100
111	1111

Razão de codificação

- O tempo $n\tau_s$ de duração de uma palavra-código deve ser igual ao tempo de duração $k\tau_x$ de uma mensagem, onde τ_s representa a largura (duração) dos bits em uma palavra-código e τ_x representa a largura dos bits em uma mensagem. Se esta condição não é obedecida o espectro da informação codificada será alterado, conforme já visto no Cap I das notas de aula.
- Assim, se $n\tau_s = k\tau_x$, então a razão de codificação R_c de um código de bloco é $R_c = k/n = \tau_s/\tau_x\,, (n > k)$.

Peso de uma Palavra-Código

- O **peso** de uma palavra-código é definido como o número de dígitos "1" nela presentes.
- O conjunto de todos os pesos de um código constitui a **distribuição de pesos** do código.
- Quando todas as M palavras-código têm pesos iguais, o código é denominado de **código de peso constante**.
- Por exemplo, o peso da palavra-código $\underline{c} = [0 \quad 1 \quad 0 \quad 1]$ é 2.

Códigos de Bloco – códigos polinomiais

- O processo de codificação/decodificação de um código de bloco baseia-se na propriedade algébrica de que o conjunto de **palavras-código** $\boldsymbol{C} = \{\underline{c}_i\} = \{\underline{c}_0, \underline{c}_1, \dots, \underline{c}_{M-1}\}$ pode ser mapeado em um **conjunto de polinômios** $\{C_i(p)\} = \{C_0(p), C_1(p), \dots C_{M-1}(p)\}$.

- Os componentes do **vetor** $\underline{c_i} = [c_{i(n-1)}\ \ c_{i(n-2)}\ \ \dots\ \ c_{i1}\ \ c_{i0}]$ que representa a i-ésima **palavra-código** correspondem aos coeficientes do polinômio $C_i(p) = c_{i(n-1)}p^{n-1} + c_{i(n-2)}p^{n-2} + \dots + c_{i1}p + c_{i0}$ associado à palavra-código.

- A mesma propriedade algébrica pode ser aplicada sobre o conjunto de **mensagens** $\boldsymbol{X} = \{\underline{x}_i\} = \{\underline{x}_0, \underline{x}_1, \dots, \underline{x}_{M-1}\}$ de modo que este também pode ser mapeado em um conjunto de polinômios $\{X_i(p)\} = \{X_0(p), X_1(p) \cdots X_{M-1}(p)\}$

- Os componentes do **vetor** $\underline{x_i} = [x_{i(n-1)}\ \ x_{i(n-2)}\ \ \dots\ \ x_{i1}\ \ x_{i0}]$ que representa a i-ésima **mensagem** correspondem aos coeficientes do polinômio $X_i(p) = x_{i(n-1)}p^{n-1} + x_{i(n-2)}p^{n-2} + \dots + x_{i1}p + x_{i0}$ associado à mensagem.

- Por este motivo os códigos de bloco são também denominados de <u>códigos polinomiais</u>.

- Por exemplo, a representação polinomial do código do Exemplo 1 é mostrada na Tabela 2.

Tabela 2 – Representação polinomial do código do Exemplo 1

Mensagem x_i	Polinômio $X_i(p)$	Palavra-código c_i	Polinômio $C_i(p)$
000	0	0000	0
001	1	0011	$p+1$
010	p	0101	p^2+1
011	$p+1$	0110	p^2+p
100	p^2	1001	p^3+1
101	p^2+1	1010	p^3+p
110	p^2+p	1100	p^3+p^2
111	p^2+p+1	1111	p^3+p^2+p+1

- O processo de codificação/decodificação envolve operações aritméticas de adição e multiplicação realizadas sobre o conjunto de polinômios $\{C_i(p)\} = \{C_0(p), C_1(p) \cdots C_{M-1}(p)\}$ que representam as palavras-código, conforme veremos.

- Um código corretor de erro deve ser tal que o conjunto $\{C_i(p)\}$ e as operações aritméticas sobre ele definidas obedeçam a determinadas restrições, caso contrário a unicidade e o custo computacional do processo de codificação/decodificação resultarão prejudicados.

- Especificamente, os coeficientes dos polinômios em $\{C_i(p)\}$ devem pertencer a um tipo especial de conjunto denominado de **campo algébrico**.

- Um campo algébrico é uma entidade matemática estudada em Álgebra Linear.

Um campo **F** é um conjunto de elementos que permite duas operações sobre seus elementos – adição e multiplicação – e que satisfaz aos seguintes propriedades:

Adição

1- O conjunto **F** é fechado sob adição, i.e., se $a,b \in \mathbf{F}$ então $a + b \in \mathbf{F}$.

2- A adição em **F** é associativa, i.e., se $a,b,c \in \mathbf{F}$ então $a + (b + c) = (a + b) + c$.

3- A adição em **F** é comutativa, i.e., se $a,b \in \mathbf{F}$ então $a + b = b + a$.

4- O conjunto **F** contém um único elemento denominado **zero**, representado por "0", que satisfaz a condição $a + 0 = a$, $\forall\ a \in \mathbf{F}$.

5- Cada elemento em **F** tem o seu elemento negativo (simétrico). Se $b \in \mathbf{F}$ então seu simétrico é denotado por $-b$ tal que $b + (-b) = 0$. Se $a \in \mathbf{F}$, então a subtração $a - b$ entre os elementos a e b é definida como $a + (-b)$.

Multiplicação

1- O conjunto **F** é fechado sob multiplicação, i.e., se $a,b \in \mathbf{F}$ então $ab \in \mathbf{F}$.

2- A multiplicação em **F** é associativa, i.e., se **F** ∈ *c b a* , , então $a.(bc)=(ab)$c

3- A multiplicação em **F** é comutativa, i.e., se $a,b \in \mathbf{F}$ então $ab = ba$.

4- A multiplicação em **F** é distributiva sobre a adição, i.e., se $a,b,c \in \mathbf{F}$ então $a(b + c) = ab + ac$.

5- O conjunto **F** contém um único elemento denominado **identidade**, representado por "1", que satisfaz a condição $1a = a$, $\forall\ a \in \mathbf{F}$.

6- Cada elemento de **F**, exceto o elemento 0 , possui um elemento **inverso**. Assim, se $b \in \mathbf{F}$ e $b \neq 0$ então o inverso de b é definido como b^{-1} tal que $bb^{-1}= 1$. Se $a \in \mathbf{F}$, então a divisão a / b entre os elementos a e b é definida como ab^{-1}.

O conjunto $\Re$ dos números reais é um campo algébrico com infinitos elementos, assim como também o é o conjunto dos números complexos $\mathbb{C}$. Estes dois conjuntos obedecem as propriedades dos campos algébricos descritas anteriormente.

- Um campo algébrico finito com D elementos é denominado de Campo de Galois (*Galois Field*) e é designado por **GF**(D).
- Nem para todos os valores de D é possível formar um campo.
- Em geral, quando D é primo (ou uma potência inteira de um número primo) é possível construir o campo finito **GF**(D) consistindo dos elementos $\{0,1,\cdots,D-1\}$, desde que as operações de adição e multiplicação sobre **GF**(D) sejam operações **módulo *D*** .

Nota: Uma operação **op é módulo *D*** quando pode ser representada por $(a\ \boldsymbol{op}\ b) mod\ D$, onde $x\ mod\ y$ é o operador que resulta no resto da divisão x/y .

Por exemplo, a operação de **soma módulo 5** entre os números **4** e **3**, $(4\ \boldsymbol{op}\ 3) mod\ 5$, resulta em **2** visto que o resto da divisão 7/5 é 2, portanto $(4\ +\ 3)\ mod\ \ 5\ =\ 2$.

- No nosso caso, utilizaremos um Campo de Galois 2 - **GF**(2).
- O **GF**(2) é formado pelo conjunto {0,1} e pelas operações módulo 2 de soma e multiplicação dadas pelas Tabelas 3 e 4.

Tabela 3: Soma sobre **GF**(2)		
+	0	1
0	0	1
1	1	0

Tabela 4: Multiplicação sobre **GF**(2)		
*	0	1
0	0	0
1	0	1

Note nas Tabelas 3 e 4 que:

- A soma entre dois elementos *a* e *b* pertencentes a **GF**(2) é implementada pela operação lógica $a \oplus b$ (ou *a* XOR *b*) e que
- A multiplicação entre dois elementos *a* e *b* pertencentes a **GF**(2) é implementada pela operação lógica *a.b* (ou *a* AND *b*).

- Dada a facilidade de implementação com portas lógicas AND e XOR, é usual os códigos corretores serem construídos em **GF**(2).

- Assim, um código corretor de erro binário é tal que os coeficientes dos polinômios em $\{C_i(p)\}$ pertencem a GF(2);

- $\mathrm{A} = \{0,1\}$ e as operações aritméticas realizadas sobre o conjunto de polinômios $\{C_i(p)\} = \{C_0(p), C_1(p) \cdots C_{M-1}(p)\}$ (ou, equivalentemente, sobre o conjunto de palavras-código $C = \{\underline{c}_i\} = \{\underline{c}_0, \underline{c}_1, \ldots, \underline{c}_{M-1}\}$) durante o processo de codificação/decodificação obedecem às Tabelas 3 e 4.

- Suponhamos que $\underline{c_i}$ e $\underline{c_j}$ sejam duas palavras-código quaisquer do código $\boldsymbol{\theta}(n, k)$.
- Uma medida da diferença (distância) entre duas palavras-código é o número de bits em posições correspondentes que diferem entre si.
- Esta medida é denominada de **Distância de Hamming** e é denotada por d_{ij}.
- Por exemplo, sejam $\underline{c_i}$ = [0 1 0 1] e $\underline{c_j}$ = [1 0 0 0]. Então d_{ij} = 3.
- Observe que d_{ij} sempre satisfaz a condição $0 < d_{ij} \leq n, i \neq j$, para duas palavras-código $\underline{c_i}$ e $\underline{c_j}$, ambas de n bits (por definição, em um código $\boldsymbol{\theta}(n, k)$, $\underline{c_i} \neq \underline{c_j}$, $\forall i$ e $\forall j$ com $i \neq j$).
- O menor valor no conjunto $\{d_{ij}\}$, $i, j = 0, 1, \ldots, M-1, \ i \neq j, M = 2^k$ é denominado **distância mínima** do código e é denotado como d_{min} .

- Por exemplo, $d_{min} = 2$ para o código do Exemplo 1, {0000, 0011, 0101, 0110, 1001, 1010, 1100, 1111}.
- A Distância de Hamming d_{ij} é uma medida do grau de separação entre duas palavras-código c_i e c_j .
- Portanto, d_{min} está associado à capacidade do código $\boldsymbol{\theta}(n, k)$ em identificar palavras-código demoduladas no receptor quando estas são recebidas com erro, como consequência do ruído e interferência presentes no canal.
- Em outras palavras, quanto maior d_{min} maior a capacidade de um código $\boldsymbol{\theta}(n, k)$ detectar e corrigir erros.

Demonstra-se que:

- Seja $\boldsymbol{\theta}(n, k)$ um código corretor binário;
- seja d o número máximo de erros que $\boldsymbol{\theta}(n, k)$ é capaz de **detetar;**
- seja t o número máximo de erros que $\boldsymbol{\theta}(n, k)$ é capaz de **corrigir;**
- seja d_{min} a distância mínima de $\boldsymbol{\theta}(n, k)$; Então:

$\boldsymbol{\theta}(n, k)$ detecta d erros: $$d = d_{min} - 1$$

$\boldsymbol{\theta}(n, k)$ corrige t erros: $$t = \left\lfloor \frac{d_{min}-1}{2} \right\rfloor$$

sendo $\lfloor . \rfloor$ o operador que resulta no inteiro mais próximo e menor que o argumento.

Por exemplo, $d_{min} = 2$ para o código $\boldsymbol{\theta}(4,3)$ da Tabela 1.

Temos que

$$d = d_{min} - 1 = 2 - 1 = 1$$

e

$$t = \left\lfloor \frac{d_{min}-1}{2} \right\rfloor = \left\lfloor \frac{2-1}{2} \right\rfloor = 0$$

Portanto o código $\boldsymbol{\theta}(4,3)$ da Tabela 1 detecta no máximo 1 erro por palavra-código, mas não tem capacidade de correção.

De fato, este código é um simples código *parity-check.*

A Matriz Geradora de um Código $\theta(n,k)$

- Seja a i-ésima mensagem de um código binário $\boldsymbol{\theta}(n,k)$ representada pelo vetor $\underline{x_i} = [x_{i0}\ x_{i1}\ \ldots x_{i(k-1)}]$ e seja a i -ésima palavra-código de $\boldsymbol{\theta}(n,k)$ representada pelo vetor $\underline{c_i} = [c_{i0}\ c_{i1}\ \ldots c_{i(n-1)}]$, onde $i = 0, 1, \ldots, M - 1\ , M\ = 2^k$.

- O processo de codificação da mensagem $\underline{x_i} = [x_{i0}\ x_{i1}\ \ldots x_{i(k-1)}]$ na respectiva palavra-código $\underline{c_i} = [c_{i0}\ c_{i1}\ \ldots c_{i(n-1)}]$ efetuado por um código binário $\boldsymbol{\theta}(n,k)$ pode ser representado em forma matricial por

$$\underline{c_i} = \underline{x_i}\boldsymbol{G}$$

onde a matriz $\boldsymbol{G}_{k\times n}$ é denominada de **matriz geradora** do código $\boldsymbol{\theta}(n,k)$ e é dada por:

$$\mathbf{G} = \begin{bmatrix} g_{00} & g_{01} & \cdots & g_{0(n-1)} \\ g_{10} & g_{11} & \cdots & g_{1(n-1)} \\ \vdots & \vdots & & \vdots \\ g_{(k-1)0} & g_{(k-1)1} & \cdots & g_{(k-1)(n-1)} \end{bmatrix}$$

A Matriz Geradora de um Código $\theta(n, k)$

• Podemos interpretar a matriz **G** como um conjunto de k vetores-linha $\underline{g_j}$, $j = 0, 1, \ldots, k-1$, tal que

$$\mathbf{G} = \begin{bmatrix} g_{00} & g_{01} & \cdots & g_{0(n-1)} \\ g_{10} & g_{11} & \cdots & g_{1(n-1)} \\ \vdots & \vdots & & \vdots \\ g_{(k-1)0} & g_{(k-1)1} & \cdots & g_{(k-1)(n-1)} \end{bmatrix} = \begin{bmatrix} \leftarrow & \underline{g}_0 & \rightarrow \\ \leftarrow & \underline{g}_1 & \rightarrow \\ & \vdots & \\ \leftarrow & \underline{g}_{(k-1)} & \rightarrow \end{bmatrix}$$

• Desta maneira, de $\underline{c_i} = \underline{x_i}\boldsymbol{G}$, cada palavra-código $\underline{c_i} = [c_{i0}\, c_{i1} \ldots c_{i(n-1)}]$ é simplesmente uma combinação linear dos vetores $\underline{g_j}$ com coeficientes da combinação determinados pela mensagem associada $\underline{x_i} = [x_{i0}\, x_{i1} \ldots x_{i(k-1)}]$, isto é:

$$\underline{c_i} = x_{i0}\, \underline{g_0} + x_{i1}\, \underline{g_1} + \ldots + x_{i(k-1)}\, \underline{g_{(k-1)}}$$

Exemplo 2: Verifique se a matriz **G** é a matriz geradora do código $\boldsymbol{\theta}(4,3)$ da Tabela 1.

Solução: Cada palavra-código $\underline{c_i} = [c_{i0}\ c_{i1} \ldots c_{i(n-1)}]$ de $n = 4$ bits é gerada através de $\underline{c_i} = \underline{x_i}\boldsymbol{G}$ a partir da respectiva mensagem $\underline{x_i} = [x_{i0}\ x_{i1} \ldots x_{i(k-1)}]$ de $k = 3$ bits. No total, existem $2^k = 8$ palavras-código em $\boldsymbol{\theta}(4,3)$. Assim,

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

$\underline{x}_i$	$\underline{x}_i\mathbf{G} = \underline{c}_i$		$\underline{x}_i$	$\underline{x}_i\mathbf{G} = \underline{c}_i$
$[0\ 0\ 0]$	$[0\ 0\ 0]\begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&1\end{bmatrix} = [0\ 0\ 0\ 0]$		$[1\ 0\ 0]$	$[1\ 0\ 0]\begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&1\end{bmatrix} = [1\ 0\ 0\ 1]$
$[0\ 0\ 1]$	$[0\ 0\ 1]\begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&1\end{bmatrix} = [0\ 0\ 1\ 1]$		$[1\ 0\ 1]$	$[1\ 0\ 1]\begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&1\end{bmatrix} = [1\ 0\ 1\ 0]$
$[0\ 1\ 0]$	$[0\ 1\ 0]\begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&1\end{bmatrix} = [0\ 1\ 0\ 1]$		$[1\ 1\ 0]$	$[1\ 1\ 0]\begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&1\end{bmatrix} = [1\ 1\ 0\ 0]$
$[0\ 1\ 1]$	$[0\ 1\ 1]\begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&1\end{bmatrix} = [0\ 1\ 1\ 0]$		$[1\ 1\ 1]$	$[1\ 1\ 1]\begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&1\end{bmatrix} = [1\ 1\ 1\ 1]$

Portanto **G** é geradora de $\boldsymbol{\theta}(4,3)$.

A Matriz Geradora de um Código $\theta(n,k)$

- Qualquer matriz geradora **G** de um código $\boldsymbol{\theta}(n,k)$ pode, através de operações elementares em suas linhas e permutações em suas colunas, ser reduzida à **forma sistemática** quando, então, o código gerado é sistemático.
- Uma matriz geradora **G** encontra-se na forma sistemática quando

$$\mathbf{G} = \left[\mathbf{I}_k \;\vdots\; \mathbf{P}\right] = \left[\begin{array}{ccccc|ccc} 1 & 0 & 0 & \cdots & 0 & p_{00} & \cdots & p_{0(n-k-1)} \\ 0 & 1 & 0 & \cdots & 0 & p_{10} & \cdots & p_{1(n-k-1)} \\ \vdots & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 & p_{(k-1)0} & \cdots & p_{(k-1)(n-k-1)} \end{array}\right]$$

onde $\mathbf{I}_k$ é a matriz identidade $k \times k$ e **P** é uma matriz $k \times (n-k)$ que determina os $n-k$ bits de paridade na palavra-código c_i de n bits, a partir dos k bits da mensagem x_i.

$$\mathbf{G} = \left[\mathbf{I}_k \;\vdots\; \mathbf{P}\right]$$

A Matriz Geradora de um Código $\theta(n, k)$

A matriz geradora do Exemplo 2 é dada por:

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

Está na forma sistemática e o código $\boldsymbol{\theta}(4,3)$ gerado é um código sistemático, i.e., cada palavra-código de n bits é formada pelos k bits da respectiva mensagem associada, acrescidos (por justaposição) de $n - k$ bits de paridade.

Mensagem x_i	Palavra-código c_i
000	0000
001	0011
010	0101
011	0110
100	1001
101	1010
110	1100
111	1111

A Matriz Geradora de um Código $\theta(n, k)$

- No contexto de comunicação digital, as palavras-código passam por um processo de modulação no transmissor e são enviadas através de um canal com ruído/interferência.
- Dois códigos que diferem somente na ordem (arranjo) de suas palavras-código, apresentam a mesma probabilidade de erro de decodificação no receptor, porque as distâncias de Hamming entre as palavras-código são as mesmas [Peterson]. Tais códigos são denominados **equivalentes**.
- Especificamente, o código $\boldsymbol{\theta_e}(n, k)$ é equivalente ao código $\boldsymbol{\theta}(n, k)$ se a matriz geradora $\mathbf{G_e}$ de $\boldsymbol{\theta_e}(n, k)$ puder ser obtida através da permutação de **colunas** da matriz **G** geradora de $\boldsymbol{\theta}(n, k)$ ou através de operações elementares realizadas entre as **linhas** de **G**.
- **Uma operação elementar em GF(2) entre duas linhas de uma matriz consiste em permutar as linhas ou em substituir uma linha pela soma dela com outra linha.**
- Assim sempre podemos transformar uma matriz **G** qualquer para a forma sistemática **G***, mantendo a equivalência entre os respectivos códigos gerados.

Exemplo 3: Dada a matriz geradora **G**, colocá-la na forma sistemática **G*** .

- Verifique se **G*** gera um código equivalente ao gerado por **G**.

Solução: Visto que a matriz geradora é uma matriz $\mathbf{G}_{3\times4}$, então o código gerado será um código $\boldsymbol{\theta}(4,3)$.

G* pode ser obtida pelo seguinte conjunto de operações elementares feito sobre as linhas de **G**:

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

Operação Elementar	Matriz G alterada
$L_0 \leftrightarrow L_2$	$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$
$L_0 \leftarrow (L_0 + L_1)$	$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$
$L_0 \leftarrow (L_0 + L_2)$	$\mathbf{G}^* = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$

A Matriz Geradora de um Código $\theta(n,k)$

O código gerado por **G** é:

$\underline{x}_i$	$\underline{x}_i\mathbf{G}=\underline{c}_i$	$\underline{x}_i$	$\underline{x}_i\mathbf{G}=\underline{c}_i$
$[0\ 0\ 0]$	$[0\ 0\ 0]\begin{bmatrix}0&0&1&1\\0&1&0&1\\1&1&1&1\end{bmatrix}=[0\ 0\ 0\ 0]$	$[1\ 0\ 0]$	$[1\ 0\ 0]\begin{bmatrix}0&0&1&1\\0&1&0&1\\1&1&1&1\end{bmatrix}=[0\ 0\ 1\ 1]$
$[0\ 0\ 1]$	$[0\ 0\ 1]\begin{bmatrix}0&0&1&1\\0&1&0&1\\1&1&1&1\end{bmatrix}=[1\ 1\ 1\ 1]$	$[1\ 0\ 1]$	$[1\ 0\ 1]\begin{bmatrix}0&0&1&1\\0&1&0&1\\1&1&1&1\end{bmatrix}=[1\ 1\ 0\ 0]$
$[0\ 1\ 0]$	$[0\ 1\ 0]\begin{bmatrix}0&0&1&1\\0&1&0&1\\1&1&1&1\end{bmatrix}=[0\ 1\ 0\ 1]$	$[1\ 1\ 0]$	$[1\ 1\ 0]\begin{bmatrix}0&0&1&1\\0&1&0&1\\1&1&1&1\end{bmatrix}=[0\ 1\ 1\ 0]$
$[0\ 1\ 1]$	$[0\ 1\ 1]\begin{bmatrix}0&0&1&1\\0&1&0&1\\1&1&1&1\end{bmatrix}=[1\ 0\ 1\ 0]$	$[1\ 1\ 1]$	$[1\ 1\ 1]\begin{bmatrix}0&0&1&1\\0&1&0&1\\1&1&1&1\end{bmatrix}=[1\ 0\ 0\ 1]$

O código gerado por **G*** possui a mesma distância de Hamming do código gerado no Exemplo 2.

Os códigos gerados por **G*** e **G** são equivalentes, porque diferem apenas no arranjo de suas palavras-código.

A Matriz de Paridade de um Código $\theta(n, k)$

- Seja um código $\boldsymbol{\theta}(n, k)$ com matriz geradora **G** dada na forma sistemática,

$$\mathbf{G} = \left[\mathbf{I}_k \;\vdots\; \mathbf{P}\right]$$

- A i-ésima palavra-código $\underline{c_i} = [c_{i0}\, c_{i1} \ldots c_{i(n-1)}]$ relaciona-se com a respectiva mensagem $\underline{x_i} = [x_{i0}\, x_{i1} \ldots x_{i(k-1)}]$ através de $\underline{c_i} = \underline{x_i}\mathbf{G}$.

- Já que **G** encontra-se na forma sistemática, a palavra-código $\underline{c_i}$ pode ser decomposta em $\underline{c_i} = [\underline{x_i}\ \underline{a_i}]$ onde $\underline{a_i} = \underline{x_i}\mathbf{P}$ é um vetor-linha que contém os $n - k$ bits de paridade de $\underline{c_i}$.

- Visto que $\underline{a_i} = \underline{x_i}\mathbf{P}$, e considerando que a soma em **GF**(2) é uma operação módulo 2, então

$$\underline{x_i}\mathbf{P} + \underline{a_i} = \underline{0}$$

que pode ser escrita matricialmente como $\left[\underline{x_i}\ \underline{a_i}\right]\begin{bmatrix}\mathbf{P}\\ \mathbf{I}_{n-k}\end{bmatrix} = \underline{0}$

Matriz de paridade transposta $\mathbf{H}^T$

A Matriz de Paridade de um Código $\theta(n,k)$

Definindo

$$\mathbf{H}^T = \begin{bmatrix} \mathbf{P} \\ \mathbf{I}_{n-k} \end{bmatrix} \quad \Rightarrow \quad \mathbf{H} = [\mathbf{P}^T \quad \mathbf{I}_{n-k}]$$

Sendo

$$\mathbf{H} = (\mathbf{H}^T)^T = \begin{bmatrix} \mathbf{P} \\ \mathbf{I}_{n-k} \end{bmatrix}^T = [\mathbf{P}^T \quad (\mathbf{I}_{n-k})^T] = [\mathbf{P}^T \quad \mathbf{I}_{n-k}]$$

Temos que

$$\overbrace{[\underline{x_i} \ \underline{a_i}]}^{\underline{c_i}} \overbrace{\begin{bmatrix} \mathbf{P} \\ \mathbf{I}_{n-k} \end{bmatrix}}^{\mathbf{H}^T} = \underline{0} \quad \rightarrow \quad \underline{c_i}\mathbf{H}^T = \underline{0}$$

- Portanto, de $\underline{c_i}\mathbf{H}^T = \underline{0}$, infere-se que cada palavra-código do código $\boldsymbol{\theta}(n,k)$ é ortogonal a cada linha da matriz **H** (se $u \cdot \underline{v^T} = 0$ então os vetores $\underline{u}$ e $\underline{v}$ são ortogonais).

A Matriz de Paridade de um Código $\theta(n, k)$

$$\underline{c_i}\mathbf{H}^T = \underline{0}$$

• Deste modo, observa-se que a matriz $\mathbf{H}$ pode ser usada no receptor digital para detectar se ocorreu erro como consequência da degradação imposta pelo canal de transmissão.

• Seja $\underline{c_i}$ a palavra-código transmitida e $\underline{y_i}$ a palavra-código recebida,

Se $\underline{y_i}\mathbf{H}^T \neq 0$ então $\underline{y_i} \neq \underline{c_i}$ e, logo $\underline{y_i}$ apresenta erros.

Se $\underline{y_i}\mathbf{H}^T = 0$ então $\underline{y_i} = \underline{c_i}$ e, logo $\underline{y_i}$ foi recebida sem erros.

• Por este motivo, $\mathbf{H}_{(n-k)\times n}$ é denominada de matriz de paridade.

Exemplo 4:

(a) Determine a matriz de paridade **H** do código $\boldsymbol{\theta}(4,3)$ do Exemplo 3.

(b) Verifique se $\mathbf{G}\mathbf{H}^T = \underline{0}$.

(c) Verifique se $\underline{c_i}\mathbf{H}^T = \underline{0}$

Solução:

(a) A matriz geradora de $\boldsymbol{\theta}(4,3)$ na forma sistemática é

$$\mathbf{G} = \left[\mathbf{I}_3 \;\vdots\; \mathbf{P}\right] = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \Rightarrow \quad \mathbf{H} = \left[\mathbf{P}^T \;\vdots\; \mathbf{I}_{n-k}\right] =$$

(b) Verificando se $\mathbf{G}\mathbf{H}^T = \underline{0}$:

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

Exemplo 4:

(a) Determine a matriz de paridade **H** do código $\boldsymbol{\theta}(4,3)$ do Exemplo 3.

(b) Verifique se $\mathbf{GH}^T = \underline{0}$.

(c) Verifique se $\underline{c_i}\mathbf{H}^T = \underline{0}$

Solução:

(a) A matriz geradora de $\boldsymbol{\theta}(4,3)$ na forma sistemática é

$$\mathbf{G} = \left[\mathbf{I}_3 \vdots \mathbf{P}\right] = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \Rightarrow \mathbf{H} = \left[\mathbf{P}^T \vdots \mathbf{I}_{n-k}\right] = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}$$

(b) Verificando se $\mathbf{GH}^T = \underline{0}$:

$$\mathbf{GH}^T = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

A Matriz de Paridade de um Código $\theta(n,k)$

(c) Verificando se $\underline{c_i}\mathrm{H}^T=\underline{0}$

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

A Matriz de Paridade de um Código $\theta(n,k)$

(c) Verificando se $\underline{c_i}\mathrm{H}^T=\underline{0}$

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

Decodificação pela Mínima Distância (Decodificacão ML - *Maximum-Likelihood Decoding*)

- No receptor digital, os n bits provenientes do demodulador, correspondentes à i-ésima palavra-código recebida são entregues ao decodificador do código $\boldsymbol{\theta}(n, k)$.
- Quando utilizada a decodificação pela mínima distância, o decodificador compara $\underline{y_i}$ com as $M = 2^k$ possíveis palavras-código $\underline{c_j}$ de $\boldsymbol{\theta}(n, k)$, $j = 0, 1, \ldots, M-1$, e decide em favor daquela palavra-código (portanto, em favor da mensagem associada) que é mais próxima da palavra-código recebida em termos da Distância de Hamming.
- Matematicamente esta operação pode ser expressa por

$$\theta^{-1}\left\{\underline{y_i}\right\} = argmin_{\underline{c_j}} \left|\underline{y_i} - \underline{c_j}\right|_H \text{ onde } \underline{c_j} \in \mathbf{C},$$

$$\mathbf{C} = \left\{\underline{c_i}\right\} = \{\underline{c_0}, \underline{c_1}, \ldots, \underline{c_{M-1}}\}$$

e $\left|\underline{y_i} - \underline{c_j}\right|_H$ denota a Distância de Hamming entre a palavra código recebida $\underline{y_i}$ e a palavra código $\underline{c_j}$ pertencente ao conjunto $\mathbf{C}$.

- Embora a **decodificação ML** possa ser realizada através de

$$\theta^{-1}\{\underline{y_i}\} = argmin_{\underline{c_j}} \left|\underline{y_i} - \underline{c_j}\right|_H,$$

existe uma maneira mais eficiente de implementar um decodificador ML, aproveitando as propriedades da matriz de paridade $\mathbf{H}_{(n-k)\times n}$ de um código $\boldsymbol{\theta}(n,k)$, denominada de Decodificação por Arranjo Padrão (*Standard Array Decoding*).

- A desvantagem da decodificação ML é a necessidade de calcular $M = 2^k$ Distâncias de Hamming para decodificar a palavra-código recebida.

- Veremos a seguir como reduzir este número de distâncias calculadas para 2^{n-k} utilizando o conceito de Arranjo Padrão, já que, na prática, usualmente $n - k < k$.

Arranjo Padrão

- Seja $\underline{c_i}$ a palavra-código transmitida pelo transmissor digital através do canal de transmissão e seja $\underline{y_i}$ a palavra-código recebida resultante na saída do demodulador do receptor digital.
- Devido à degradação do sinal no canal, em consequência de ruído/interferência, a palavra-código $\underline{y_i}$ recebida pode conter erros, de modo que $\underline{y_i}$ pode ser expressa por $\underline{y_i} = \underline{c_i} + \underline{e_i}$

onde $\underline{e_i}$ é o vetor-linha de n bits que representa o **padrão de erro** (i.e., os bits errados em $\underline{y_i}$) resultante da degradação do sinal no canal.

Palavra-código transmitida: $\underline{c_i} = [0\ 1\ 0\ 1]$

Palavra-código recebida: $\underline{y_i} = [0\ 1\ 0\ 0]$

Padrão de erro: $\underline{e_i} = [0\ 0\ 0\ 1]$

- Note que o peso do padrão de erro é a Distância de Hamming entre e $\underline{y_i}$ e $\underline{c_i}$.

Peso do padrão de erro: $\underline{e_i} = 1$

- Pós-multiplicando $\underline{y_i} = \underline{c_i} + \underline{e_i}$ por $\mathbf{H}^T$ obtemos

$$\underline{y_i}\mathbf{H}^T = \left(\underline{c_i} + \underline{e_i}\right)\mathbf{H}^T = \underline{c_i}\mathbf{H}^T + \underline{e_i}\mathbf{H}^T = \underline{e_i}\mathbf{H}^T \Rightarrow \boxed{\underline{y_i}\mathbf{H}^T = \underline{e_i}\mathbf{H}^T}$$

Nota: Lembre que $\underline{c_i}\mathbf{H}^T = \underline{\mathbf{0}}$, ou seja, as palavras-código de um código são ortogonais à sua matriz de paridade.

- Define-se o vetor $n-k$ dimensional $\underline{s}$, denominado **síndrome do padrão de erro**, ou simplesmente **síndrome**, como

$$\boxed{\underline{s_i} = \underline{e_i}\mathbf{H}^T}$$

Dimensão de $\underline{e_i} = n$; dimensão de $H = (n-k) \times n$; dimensão de $\underline{s_i} = (n-k)$.

- É importante enfatizar que o conjunto de síndromes $\{\underline{S}\}$ é determinado pelo conjunto de padrões de erro $\{\underline{e_i}\}$, mas **não** pelo conjunto **C** de palavras-código transmitidas, como podemos inferir de $\underline{s_i} = \underline{y_i}\mathbf{H}^T = \underline{e_i}\mathbf{H}^T$.

Observe que:

- $\underline{e_i}$ é um vetor de n bits (i.e., $\underline{e_i}$ é um vetor n dimensional em **GF**(2)) → existem 2^n possíveis padrões de erro no conjunto $\{\underline{e_i}\}$;

- $\underline{s}$ é um vetor de $n-k$ bits → existem 2^{n-k} possíveis síndromes no conjunto $\{\underline{S}\}$.

- Em consequência, $\underline{s_i} = \underline{e_i}\mathbf{H}^T$ mapeia diferentes padrões de erro $\underline{e_i}$ na mesma síndrome $\underline{s}$.

Arranjo Padrão

- O mapeamento do padrão de erro $\underline{e_i}$ em uma síndrome $\underline{s_i}$, através da matriz de paridade $\mathbf{H}^T$ resulta na Tabela de Síndromes.

Síndrome **Padrão de Erro**

Síndrome $\underline{s_i}$	Padrão de Erro $\underline{e_i}$
[0 0 0]	[0 0 0 0 0]
[0 0 1]	[0 0 0 0 1]
[0 1 0]	[0 0 0 1 0]
[0 1 1]	[0 1 0 0 0]
[1 0 0]	[0 0 1 0 0]
[1 0 1]	[1 0 0 0 0]
[1 1 0]	[1 1 0 0 0]
[1 1 1]	[1 0 0 1 0]

• O processo de decodificação pode ser definido nas seguintes etapas

1) Cálculo da síndrome através da multiplicação da palavra-código recebida $\underline{y_i}$ pela matriz de paridade transposta $\mathbf{H}^T$

$$\underline{s_i} = \underline{y_i}\mathbf{H}^T$$

2) Identificação do erro padrão $\underline{e_i}$ associado a síndrome $\underline{s_i}$, através de consulta a tabela de síndromes.

3) Cálculo da palavra-código decodificada $\underline{c_{dec}}$ através da soma da palavra-código recebida $\underline{y_i}$ com o erro padrão $\underline{e_i}$.

$$\underline{c_{dec}} = \underline{y_i} + \underline{e_i}$$

4) Recuperação da mensagem transmitida $\underline{x_{dec}}$. Para códigos sistemáticos, a mensagem corresponde aos primeiros bits da palavra-código. Deste modo, para obter $\underline{x_{dec}}$ basta descartar os $n - k$ bits de paridade de $\underline{c_{dec}}$.

- O AP também é uma tabela que possui 2^{n-k} **linhas**, cada uma delas associada a uma das 2^{n-k} possíveis **síndromes**.
- O nº de **colunas** do AP é 2^k , correspondendo ao nº de **palavras-código** do código $\theta(n,k)$.
- Quando implementado, a linha superior do AP recebe a designação L0 e a coluna mais à esquerda recebe a designação C0.
- O AP é formado de $2^{n-k} \times 2^k = 2^n$ células (i.e. 2^n possíveis padrões de erro).

Tabela 5 – Forma geral do arranjo padrão

	C0	C1	C2	…	$C(2^k-1)$
L0	$\underline{e}_0 = \underline{c}_0 = \underline{0}$	$\underline{c}_1$	$\underline{c}_2$	…	$\underline{c}_{(2^k-1)}$
L1	$\underline{e}_1$	$\underline{c}_1 + \underline{e}_1$	$\underline{c}_2 + \underline{e}_1$	…	$\underline{c}_{(2^k-1)} + \underline{e}_1$
L2	$\underline{e}_2$	$\underline{c}_1 + \underline{e}_2$	$\underline{c}_2 + \underline{e}_2$	…	$\underline{c}_{(2^k-1)} + \underline{e}_2$
⋮	⋮	⋮	⋮		⋮
$L(2^{n-k}-1)$	$\underline{e}_{(2^{n-k}-1)}$	$\underline{c}_1 + \underline{e}_{(2^{n-k}-1)}$	$\underline{c}_2 + \underline{e}_{(2^{n-k}-1)}$	…	$\underline{c}_{(2^k-1)} + \underline{e}_{(2^{n-k}-1)}$

- Na linha $\mathrm{L0}$ do AP são listadas, da esquerda para a direita, as 2^k palavras-código de $\boldsymbol{\theta}(n,k)$, cada uma delas representada por um vetor n dimensional em **GF**(2).

- A palavra-código $\underline{c}_0$ pertencente à célula identificada pela intersecção da coluna $\mathrm{C0}$ com a linha $\mathrm{L0}$ (célula $\mathrm{L0} \times \mathrm{C0}$) obrigatoriamente deve ser aquela representada pelo vetor $\underline{0}$.

- Na coluna $\mathrm{C0}$, abaixo da palavra-código $\underline{0}$, são listados, de alto a baixo, os $2^{n-k} - 1$ padrões de erro relativos à palavra-código $\underline{c}_0 = \underline{0}$.

- Primeiramente são listados todos os n padrões de erro de peso 1, isto é, todos os padrões de erro que resultam de uma Distância de Hamming unitária entre a palavra-código $\underline{y}$ recebida e $c_0 = \underline{0}$.

- Se $2^{n-k} > n$, então lista-se a seguir em $\mathrm{C0}$ todos os possíveis padrões de erro de peso 2.

Em seguida lista-se em C0 todos os possíveis padrões de erro de peso 3, e assim sucessivamente até que todas as 2^{n-k} células de C0 estejam preenchidas.

Neste contexto, $\underline{e}_0 = \underline{c}_0 = 0$ representa o padrão de erro de peso 0, isto é, representa a não-ocorrência de erro.

	C0	C1	C2	…	$C(2^k-1)$
L0	$\underline{e}_0 = \underline{c}_0 = \underline{0}$	$\underline{c}_1$	$\underline{c}_2$	…	$\underline{c}_{(2^k-1)}$
L1	$\underline{e}_1$	$\underline{c}_1 + \underline{e}_1$	$\underline{c}_2 + \underline{e}_1$	…	$\underline{c}_{(2^k-1)} + \underline{e}_1$
L2	$\underline{e}_2$	$\underline{c}_1 + \underline{e}_2$	$\underline{c}_2 + \underline{e}_2$	…	$\underline{c}_{(2^k-1)} + \underline{e}_2$
⋮	⋮	⋮	⋮		⋮
$L(2^{n-k}-1)$	$\underline{e}_{(2^{n-k}-1)}$	$\underline{c}_1 + \underline{e}_{(2^{n-k}-1)}$	$\underline{c}_2 + \underline{e}_{(2^{n-k}-1)}$	…	$\underline{c}_{(2^k-1)} + \underline{e}_{(2^{n-k}-1)}$

Nota: Visto que cada linha do AP necessita corresponder a uma única síndrome dentre as 2^{n-k} possíveis síndromes, devemos ter o cuidado de, na construção de C0, assegurar que distintos padrões de erro de peso maior que 1 em C0 correspondam a síndromes que são distintas entre si e que são simultaneamente distintas daquelas que correspondem a padrões de erro de peso 1.

- Dando prosseguimento à construção do AP, adicionamos o padrão de erro contido na i-ésima célula de $\mathrm{C0}$ à palavra-código na célula $\mathrm{L0} \times \mathrm{C1}$ e colocamos o resultado na i-ésima célula em C1.
- Em seguida, adicionamos o padrão de erro contido na i —ésima célula de $\mathrm{C0}$ à palavra-código na célula $\mathrm{L0} \times \mathrm{C2}$ e colocamos o resultado na i-ésima célula em $\mathrm{C2}$, e assim sucessivamente até completar a última coluna $C\left(2^k - 1\right)$, mais à direita do AP, sendo $i = 0,1,2 \cdots, 2^{n=k} - 1$.

	C0	C1	C2	…	$\mathrm{C}(2^k-1)$
L0	$\underline{e}_0 = \underline{c}_0 = \underline{0}$	$\underline{c}_1$	$\underline{c}_2$	…	$\underline{c}_{(2^k-1)}$
L1	$\underline{e}_1$	$\underline{c}_1 + \underline{e}_1$	$\underline{c}_2 + \underline{e}_1$	…	$\underline{c}_{(2^k-1)} + \underline{e}_1$
L2	$\underline{e}_2$	$\underline{c}_1 + \underline{e}_2$	$\underline{c}_2 + \underline{e}_2$	…	$\underline{c}_{(2^k-1)} + \underline{e}_2$
⋮	⋮	⋮	⋮		⋮
$\mathrm{L}(2^{n-k}-1)$	$\underline{e}_{(2^{n-k}-1)}$	$\underline{c}_1 + \underline{e}_{(2^{n-k}-1)}$	$\underline{c}_2 + \underline{e}_{(2^{n-k}-1)}$	…	$\underline{c}_{(2^k-1)} + \underline{e}_{(2^{n-k}-1)}$

Exemplo 6: Seja o codificador de canal no transmissor de um sistema de comunicação digital que utiliza o código de bloco gerado por:

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

a) Determine um possível AP para este código e a Tabela de Síndromes associada, visando o projeto do decodificador no receptor.

b) Suponha que o transmissor digital envie a palavra-código $\underline{c}$= [1 0 1 0 1] através do canal. O canal degrada o sinal de forma que o demodulador no receptor envia para o decodificador a palavra código $\underline{y}$ = [1 1 1 0 1] (erro no segundo bit). Verifique a capacidade do decodificador em detectar e corrigir este erro.

c) Suponha que o ruído/interferência no canal seja alto de forma que o demodulador no receptor envia para o decodificador a palavra-código $\underline{y}$ = [1 1 1 1 1] (erro no segundo e quarto bits). Verifique a capacidade do decodificador em detectar e corrigir este erro duplo.

Solução:

a) A matriz geradora não necessita ser transformada por permutação de colunas ou por operações elementares em linhas visto que já encontra-se na forma sistemática, isto é,

$$\mathbf{G} = \left[\mathbf{I}_k \;\vdots\; \mathbf{P}\right] \qquad \mathbf{G} = \left[\begin{array}{cc:ccc} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{array}\right] = \left[\mathbf{I}_2 \;\vdots\; \mathbf{P}\right]$$

Visto que $G_{k\times n} = G_{2\times 5}$, o código em questão é $\boldsymbol{\theta}(5,2)$.

As $2^k = 2^2 = 4$ palavras-código de $\boldsymbol{\theta}(5,2)$ gerado por **G** são obtidas de $\underline{c_i} = \underline{x_i}\boldsymbol{G}$.

$$\underline{c}_0 = \begin{bmatrix}0 & 0\end{bmatrix}\mathbf{G} = \begin{bmatrix}0 & 0 & 0 & 0 & 0\end{bmatrix}$$

$$\underline{c}_1 = \begin{bmatrix}0 & 1\end{bmatrix}\mathbf{G} = \begin{bmatrix}0 & 1 & 0 & 1 & 1\end{bmatrix}$$

$$\underline{c}_2 = \begin{bmatrix}1 & 0\end{bmatrix}\mathbf{G} = \begin{bmatrix}1 & 0 & 1 & 0 & 1\end{bmatrix}$$

$$\underline{c}_3 = \begin{bmatrix}1 & 1\end{bmatrix}\mathbf{G} = \begin{bmatrix}1 & 1 & 1 & 1 & 0\end{bmatrix}$$

A partir de **G**, podemos definir **H**:

$$\mathbf{H}_{(n-k)\times n} = \left[\mathbf{P}^T \;\vdots\; \mathbf{I}_{n-k}\right] = \left[\begin{array}{cc:ccc} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

Para determinar os padrões de erro da coluna C0 do AP precisamos verificar quais as síndromes resultantes dos $n = 5$ padrões de erro de peso 1 para que não ocorra igualdade com as síndromes resultantes dos padrões de erro de peso maior que 1.

Os padrões de erro de peso 1 são:
$[0\ 0\ 0\ 0\ 1]$, $[0\ 0\ 0\ 1\ 0]$, $[0\ 0\ 1\ 0\ 0]$, $[0\ 1\ 0\ 0\ 0]$ e $[1\ 0\ 0\ 0\ 0]$.

Verificando as síndromes resultantes dos padrões de erro de peso 1:
$[0\ 0\ 1]$, $[0\ 1\ 0]$, $[1\ 0\ 0]$, $[0\ 1\ 1]$, $[1\ 0\ 1]$

Obviamente a síndrome resultante do padrão de erro de peso 0:

inexistência de erro) é
$[0\ 0\ 0\ 0\ 0]\ \mathbf{H}^T = [0\ 0\ 0]$.

$\underline{e}_i$	$\underline{e}_i\mathbf{H}^T = \underline{s}_i$
$[0\ 0\ 0\ 0\ 1]$	$[0\ 0\ 0\ 0\ 1]\mathbf{H}^T = [0\ 0\ 1]$
$[0\ 0\ 0\ 1\ 0]$	$[0\ 0\ 0\ 1\ 0]\mathbf{H}^T = [0\ 1\ 0]$
$[0\ 0\ 1\ 0\ 0]$	$[0\ 0\ 1\ 0\ 0]\mathbf{H}^T = [1\ 0\ 0]$
$[0\ 1\ 0\ 0\ 0]$	$[0\ 1\ 0\ 0\ 0]\mathbf{H}^T = [0\ 1\ 1]$
$[1\ 0\ 0\ 0\ 0]$	$[1\ 0\ 0\ 0\ 0]\mathbf{H}^T = [1\ 0\ 1]$

- O AP a ser construído possui $2^{n-k} = 2^{5-2} = 8$ linhas (correspondentes às 2^{n-k} síndromes).
- Já determinamos $n + 1 = 6$ síndromes (padrões de erro de peso 0 e peso 1).
- Ainda faltam determinar $2^{n-k} - (n + 1) = 8 - (5 + 1) = 2$ síndromes.
- Estas 2 síndromes faltantes devem **obrigatoriamente** ser distintas entre si e distintas das $n + 1 = 6$ síndromes já determinadas.
- Tendo esta condição em mente, verifica-se na tabela de síndromes que as síndromes faltantes são $[1\ 1\ 0]$ *e* $[1\ 1\ 1]$.
- Os padrões de erro que resultam nestas 2 síndromes devem ser padrões de erro de peso 2, visto que já esgotamos os possíveis padrões de erro de peso 0 e de peso 1.

$\underline{e}_i$	$\underline{e}_i\mathbf{H}^T = \underline{s}_i$
$[0\ 0\ 0\ 0\ 1]$	$[0\ 0\ 0\ 0\ 1]\mathbf{H}^T = [0\ 0\ 1]$
$[0\ 0\ 0\ 1\ 0]$	$[0\ 0\ 0\ 1\ 0]\mathbf{H}^T = [0\ 1\ 0]$
$[0\ 0\ 1\ 0\ 0]$	$[0\ 0\ 1\ 0\ 0]\mathbf{H}^T = [1\ 0\ 0]$
$[0\ 1\ 0\ 0\ 0]$	$[0\ 1\ 0\ 0\ 0]\mathbf{H}^T = [0\ 1\ 1]$
$[1\ 0\ 0\ 0\ 0]$	$[1\ 0\ 0\ 0\ 0]\mathbf{H}^T = [1\ 0\ 1]$

- Se expressarmos o padrão de erro por $\underline{e_i} = [b_4\ b_3\ b_2\ b_1 b_0]$, onde b_i representa a ordem do bit, e considerando que $\underline{s_i} = \underline{e_i}\mathbf{H}^T$, temos que para a síndrome $[1\ 1\ 0]$:

$$[1 \quad 1 \quad 0] = [b_4 \quad b_3 \quad b_2 \quad b_1 \quad b_0] \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

o que resulta no seguinte sistema de equações em $\mathbf{GF}(2)$:

$$b_4 + b_2 = 1 \rightarrow b_4 = b_2 + 1 \rightarrow b_4 = \overline{b_2}$$

$$b_3 + b_1 = 1 \rightarrow b_3 = b_1 + 1 \rightarrow b_3 = \overline{b_1}$$

$$b_4 + b_3 + b_0 = 0 \rightarrow b_2 + 1 + b_1 + 1 + b_0 = 0 \rightarrow b_2 + b_1 + b_0 = 0$$

onde $\overline{b}$ representa a negação do valor lógico do bit b.

- Um possível padrão de erro de peso 2 que obedece às equações acima é $\underline{e_i} = [1\ 1\ 0\ 0\ 0]$.
- Portanto este será o padrão de erro que associaremos à síndrome $[1\ 1\ 0]$.

- Para a síndrome $[1\ 1\ 1]$ temos que:

$$\underline{e_i} = [b_4\ b_3\ b_2\ b_1 b_0]$$
$$\underline{s_i} = \underline{e_i}\mathbf{H}^T$$

$$[1\ \ 1\ \ 1] = [b_4\ \ b_3\ \ b_2\ \ b_1\ \ b_0]\begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 1\\ 1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}$$

o que resulta no seguinte sistema de equações em $\mathbf{GF}(2)$:

$$b_4 + b_2 = 1 \rightarrow b_4 = b_2 + 1 \rightarrow b_4 = \overline{b_2}$$
$$b_3 + b_1 = 1 \rightarrow b_3 = b_1 + 1 \rightarrow b_3 = \overline{b_1}$$
$$b_4 + b_3 + b_0 = 1 \rightarrow b_2 + 1 + b_1 + 1 + b_0 = 1 \rightarrow b_2 + b_1 + b_0 = 1$$

- Um possível padrão de erro de peso 2, distinto do anterior, que obedece às equações acima é $\underline{e_i} = [1\ 0\ 0\ 1\ 0]$.
- Portanto este será o padrão de erro que associaremos à síndrome $[1\ 1\ 1]$.

De posse destes resultados, o AP é construído como:

Arranjo Padrão:				
	C0	C1	C2	C3
L0	[0 0 0 0 0]	[0 1 0 1 1]	[1 0 1 0 1]	[1 1 1 1 0]
L1	[0 0 0 0 1]	[0 1 0 1 0]	[1 0 1 0 0]	[1 1 1 1 1]
L2	[0 0 0 1 0]	[0 1 0 0 1]	[1 0 1 1 1]	[1 1 1 0 0]
L3	[0 0 1 0 0]	[0 1 1 1 1]	[1 0 0 0 1]	[1 1 0 1 0]
L4	[0 1 0 0 0]	[0 0 0 1 1]	[1 1 1 0 1]	[1 0 1 1 0]
L5	[1 0 0 0 0]	[1 1 0 1 1]	[0 0 1 0 1]	[0 1 1 1 0]
L6	[1 1 0 0 0]	[1 0 0 1 1]	[0 1 1 0 1]	[0 0 1 1 0]
L7	[1 0 0 1 0]	[1 1 0 0 1]	[0 0 1 1 1]	[0 1 1 0 0]

E a Tabela de Síndromes para implementação do decodificador é:

Tabela de Síndromes (implementada em ROM):	
Síndrome $\underline{s}_i$	Padrão de Erro $\underline{e}_i$
[0 0 0]	[0 0 0 0 0]
[0 0 1]	[0 0 0 0 1]
[0 1 0]	[0 0 0 1 0]
[0 1 1]	[0 1 0 0 0]
[1 0 0]	[0 0 1 0 0]
[1 0 1]	[1 0 0 0 0]
[1 1 0]	[1 1 0 0 0]
[1 1 1]	[1 0 0 1 0]

b) Sabemos que $\underline{y_i}\mathbf{H}^T = \underline{e_i}\mathbf{H}^T = \underline{s_i}$.

Dado $\underline{y_i} = [1\ 1\ 1\ 0\ 1]$, então

$$\underline{s_i} = \underline{y_i}\mathbf{H}^T = [0\ 1\ 1]$$

Consultando a Tabela de Síndromes verifica-se que o padrão de erro correspondente é $\underline{e_i} = [0\ 1\ 0\ 0\ 0]$.

Para encontrar a palavra-código decodificada $\underline{c_{dec}}$:

$$\underline{y_i} = \underline{c_i} + \underline{e_i},$$

$$\underline{y_i} + \underline{e_i} = \underline{c_i} + \underline{e_i} + \underline{e_i} = \underline{c_{dec}}$$

$$\underline{c_{dec}} = \underline{y_i} + \underline{e_i} = [1\ 0\ 1\ 0\ 1] \quad \rightarrow \quad \underline{x_{dec}} = [1\ 0]$$

Portanto, para este caso, o decodificador detectou e corrigiu o erro.

c) Partindo de $\underline{y_i}\mathbf{H}^T = \underline{e_i}\mathbf{H}^T = \underline{s_i}$.

Dado $y = [1\ 1\ 1\ 1\ 1]$, então

$$\underline{s_i} = \underline{y_i}\mathbf{H}^T = [0\ 0\ 1]$$

Consultando a Tabela de Síndromes verifica-se que o padrão de erro correspondente é $\underline{e_i} = [0\ 0\ 0\ 0\ 1]$.

$$\underline{c_{dec}} = \underline{y_i} + \underline{e_i} = [1\ 1\ 1\ 1\ 0] \rightarrow \underline{x_{dec}} = [1\ 1]$$

Portanto, para este caso, o decodificador detectou o erro mas **não** corrigiu o erro duplo.

- A impossibilidade deste código corrigir todos os padrões de erro com peso maior que 1 pode ser também verificada bastando consultar a coluna $C0$ do AP.

- Por inspeção da coluna $C0$ conclui-se que este código corrige todos os 5 padrões de erro de peso 1 possíveis e somente 2 padrões de erro de peso 2, quais sejam, $\underline{e_i}$= [1 1 0 0 0] e $\underline{e_i}$ = [1 0 0 1 0].

- Em geral o projetista do código escolhe os padrões de erro de peso w que corrigem w erros com base em alguma peculiaridade do sistema digital.

- Por exemplo, no Exemplo 6 o número total de padrões de erro de peso 2 é dado pela combinação de $n = 5$ bits tomados $m = 2$ a m, isto é, $Comb(n, m) = Comb(5,2) = 10$, onde $Comb(n, m) = n!/[m!\,(n - m)!]$.

- No entanto, na construção do AP foi possível utilizar apenas 2 deles:

$$\underline{e_i} = [1\ 1\ 0\ 0\ 0] \text{ e } \underline{e_i} = [1\ 0\ 0\ 1\ 0].$$

- $\boldsymbol{\theta}(n,k) = \boldsymbol{\theta}(2^m, m+1)$, caracterizados por $d_{\min} = m+1$, onde $m \geq 1$ é um número inteiro.

- Em geral, os Códigos de Hadamard apresentam baixa razão de codificação $R_C = k/n = \tau_s/\tau_x = (m+1)/2^m$, onde τ_s representa a largura (duração no tempo) dos bits em uma palavra-código e τ_x representa a largura dos bits na respectiva mensagem.

- Portanto, como τ_s/τ_x é pequeno, o uso de um Código de Hadamard implica em um considerável aumento na banda-passante do sistema, e, por isso, não é muito utilizado.

- $\boldsymbol{\theta}(23,12)$, caracterizado por $d_{\min} = 7$, o que significa:
  - uma capacidade de correção de até $t = \left\lfloor \frac{d_{min}-1}{2} \right\rfloor = \left\lfloor \frac{7-1}{2} \right\rfloor = 3$ erros simultâneos e
  - uma capacidade de detecção de até $d = d_{min} - 1 = 7 - 1 = 6$ erros simultâneos.
- Este código é peculiar porque ele é o único código conhecido de 23 bits capaz de corrigir até 3 erros simultâneos.

- $\boldsymbol{\theta}(2^m - 1, 2^m - 1 - m)$, bastante populares por serem caracterizados pela extrema facilidade de construção, aliada a uma distância mínima $d_{\min} = 3$ (detecta até 2 erros simultâneos e corrige até 1 erro), sendo $m = n - k$ um inteiro positivo. Por exemplo, se $m = 3$, obtemos um Código de Hamming $\boldsymbol{\theta}(7,4)$.

- Em geral, a construção de um código de bloco $\boldsymbol{\theta}(n, k)$ consiste em:
  - definirmos a sua matriz de paridade $\mathbf{H}_{(n-k)\times n}$ e, a partir da definição de **H**,
  - obtermos a sua matriz geradora $\mathbf{G}_{k\times n}$.

- Lembrando que: $\mathbf{G} = \left[\mathbf{I}_k \vdots \mathbf{P}\right]$ e $\mathbf{H}_{(n-k)\times n} = \left[\mathbf{P}^T \vdots \mathbf{I}_{n-k}\right]$

- A matriz $\mathbf{H}$ de um Código de Hamming $\boldsymbol{\theta}(2^m - 1, 2^m - 1 - m)$, caracteriza-se pelas suas $n = 2^m - 1$ colunas serem formadas por todos os vetores distintos m dimensionais em $\mathbf{GF}(2)$, exceto o vetor $\underline{0}$.

- Por exemplo, um código $\boldsymbol{\theta}(3,1)$ é um Código de Hamming com $m = 2$ em que a matriz $\mathbf{H}$ é formada pelos $n = 3$ vetores colunas $[0\ 1]^T$, $[1\ 0]^T$, $[1\ 1]^T$.

$$\mathbf{H}_{(n-k)\times n} = \mathbf{H}_{2\times 3} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

Principais Códigos de Blocos - Códigos Reed-Solomon

- Os Códigos Reed-Solomon constituem uma sub-classe de uma ampla classe de códigos cíclicos denominada de Códigos BCH (Bose – Chaudhuri – Hocquenghem).
- Os Códigos Reed-Solomon (RS) encontram-se entre os códigos com alta capacidade de correção de erro, sendo largamente utilizados em muitos sistemas digitais como:
  - Dispositivos de armazenamento (Fita Magnética,CDs, DVD, códigos de barra, etc.).
  - Comunicações Móveis e *wireless* (Telefonia celular, *links* de microondas, etc.).
  - Comunicações via Satélite.
  - Televisão Digital.

- Vimos anteriormente que um código de bloco binário $\theta(n, k)$ codifica mensagens de k **bits** em palavras-código de n **bits**, podendo corrigir até $t = \left\lfloor \frac{d_{\min}-1}{2} \right\rfloor$ **bits** errados.

- Um Código Reed-Solomon $\theta(n, k)$, representado por $\mathbf{RS}(n, k)$, codifica mensagens de k **símbolos** em palavras-código de n **símbolos**, sendo capaz de corrigir até $t = \left\lfloor \frac{n-k}{2} \right\rfloor$ **símbolos** errados.

- Cada **símbolo** em uma palavra-código (ou em uma mensagem) de um código $\mathbf{RS}(n, k)$ é um bloco de m bits.

- Daí, portanto, o poder de correção de erro de um código $\mathbf{RS}(n, k)$: Mesmo que **todos** os m bits de cada um dos t símbolos recebidos estejam errados, o código $\mathbf{RS}(n, k)$ efetua a correção não importando a localização dos símbolos na palavra-código.

- Ainda, não importando o número e a posição dos bits errados em cada símbolo, o código $\boldsymbol{RS}(n, k)$ corrigirá até t símbolos e, caso o número de símbolos errados ultrapassar t, o código $\boldsymbol{RS}(n, k)$ detectará esta situação.

- No contexto do codificador de canal de um sistema de comunicações digitais esta característica é extremamente vantajosa porque permite a correção de um surto de $m \times t$ bits sequenciais recebidos em erro (*error burst correction*).
- Se o número de erros ultrapassar t, então o código $\boldsymbol{RS}(n, k)$ avisa o sistema de que não foi capaz de corrigir todos os erros.

Principais Códigos de Blocos - Códigos Reed-Solomon

- É de especial interesse o caso em que $m = 8$, quando cada símbolo representa 1 *byte*.
- Por exemplo, consideremos um código $\mathbf{RS}(20,16)$ com $m = 8$.
- Suponhamos que queiramos codificar a mensagem de $k = 16$ bytes:
- O código $\mathbf{RS}(20,16)$ adiciona $n - k = 4$ bytes de paridade e codifica a mensagem acima na palavra-código em forma sistemática abaixo:

255	100	012	098	120	003	233	111	077	163	000	001	088	200	101	007

- Observe que nenhum símbolo é maior do que 255, valor máximo decimal para 1 byte.

255	100	012	098	120	003	233	111	077	163	000	001	088	200	101	007	208	107	221	076

Capaz de corrigir até $t = \left\lfloor \frac{n-k}{2} \right\rfloor = \left\lfloor \frac{20-16}{2} \right\rfloor = 2\ símbolos \rightarrow 16bits$

- Observe também que as operações entre polinômios são todas executadas em $\boldsymbol{GF}(2^m) = \boldsymbol{GF}(2^8) = \boldsymbol{GF}(256)$.
- Foge ao escopo deste texto o estudo da álgebra de polinômios em $\boldsymbol{GF}(2^m)$ e, portanto, não nos aprofundaremos na teoria dos Códigos Reed-Solomon.

Códigos LDPC – Low-Density Parity-Check

- Os códigos *Low-Density Parity-Check* (LDPC) são uma subcategoria dos códigos de bloco lineares e foram, originalmente, introduzidos por Gallager nos anos 1960
- Códigos LDPC são códigos de bloco com matriz de paridade $\boldsymbol{H}$ com muitos 0s e poucos 1s.
- Códigos LDPC longos, quando decodificados com o algoritmo Soma-Produto (SPA), são capazes de atingir um desempenho muito próximo ao limite de Shannon.
- Além do notável desempenho, o processo de codificação e decodificação adotado pelos códigos LDPC é menos complexo, quando comparado à outra classe de códigos cujo desempenho aproxima-se do Limite de Shannon, os códigos Turbo.

- Outro fator importante a ser observado é a presença de estruturas de código altamente paralelas nos códigos LDPC, as quais são extremamente adequadas para desenvolvimento em FPGA.
- A decodificação dos códigos LDPC é realizada através de um processo iterativo do tipo *soft-decision*.
- O algoritmo utilizado para a decodificação dos códigos LDPC é um algoritmo de passagem de mensagem, onde as mensagens são passadas entre os dois conjuntos de nodos de validação CN e os nodos de bit BN, representados através de um grafo de Tanner.