Circuitos Digitais

Códigos Binários

Prof. Marcelo Grandi Mandelli
mgmandelli@unb.br

Códigos Binários

- Código BCD
- Código de Gray
- Códigos k-de-n
- Códigos de paridade
- Códigos de Hamming

Códigos Binários

- Qualquer informação no computador é representada por códigos binários:
  - caracteres, números, símbolos, etc.

- Existem diversas alternativas para codificar elementos dependendo das características desejadas.

- Um código pode ser otimizado para
  - reduzir espaço de armazenamento necessário
  - representar informações de forma unívoca
  - ainda explorar redundâncias para deteção e correção de erros

Códigos BCD (Binary Coded Decimal)

- Associam os 10 algarismos decimais (0 a 9) a códigos de 4 bits (16 combinações possíveis)
- Diversas associações são utilizadas, com predominância da representação BCD natural
- Na tabela apresentada a seguir, os pesos associados a cada um dos quatro dígitos binários aparecem entre parênteses

Códigos BCD (Binary Coded Decimal)

Mais utilizado

Dígito	BCD natural (8 4 2 1)
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1

BCD Natural é igual ao código binário até o valor 9. Os valores entre 10 e 15 não são válidos.

Códigos BCD (Binary Coded Decimal)

Dígito	BCD natural (8 4 2 1)				Aiken (2 4 2 1)			
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	0
3	0	0	1	1	0	0	1	1
4	0	1	0	0	0	1	0	0
5	0	1	0	1	1	0	1	1
6	0	1	1	0	1	1	0	0
7	0	1	1	1	1	1	0	1
8	1	0	0	0	1	1	1	0
9	1	0	0	1	1	1	1	1

Aiken é igual ao
código binário até o valor 4.
Os valores 5 a 9 são formados
Pela inversão dos bits dos
valores 4 a 0.

Códigos BCD (Binary Coded Decimal)

Excesso-de-três: simplifica a aritmética BCD

Dígito	BCD natural (8 4 2 1)	Aiken (2 4 2 1)	Stibitz (8 4 2 1) +3
0	0 0 0 0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 0 1 0	0 1 0 1
3	0 0 1 1	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 0 0	0 1 1 1
5	0 1 0 1	1 0 1 1	1 0 0 0
6	0 1 1 0	1 1 0 0	1 0 0 1
7	0 1 1 1	1 1 0 1	1 0 1 0
8	1 0 0 0	1 1 1 0	1 0 1 1
9	1 0 0 1	1 1 1 1	1 1 0 0

Stibitz é igual ao BCD + 3.

Códigos BCD (Binary Coded Decimal)

Excesso-de-três: simplifica a aritmética BCD

Dígito	BCD natural (8	4	2	1)	Aiken (2	4	2	1)	Stibitz (8	4	2	1) +3	(7	4	2	1)	(6	4	2	-1)
0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	1	0	0	1	1
2	0	0	1	0	0	0	1	0	0	1	0	1	0	0	1	0	0	0	1	0
3	0	0	1	1	0	0	1	1	0	1	1	0	0	0	1	1	0	1	0	1
4	0	1	0	0	0	1	0	0	0	1	1	1	0	1	0	0	0	1	0	0
5	0	1	0	1	1	0	1	1	1	0	0	0	0	1	0	1	0	1	1	1
6	0	1	1	0	1	1	0	0	1	0	0	1	0	1	1	0	1	0	0	0
7	0	1	1	1	1	1	0	1	1	0	1	0	0	1	1	1	1	0	1	1
8	1	0	0	0	1	1	1	0	1	0	1	1	1	0	0	1	1	0	1	0
9	1	0	0	1	1	1	1	1	1	1	0	0	1	0	1	0	1	1	0	1

Decimal ←→BCD

- Exemplo Decimal → BCD

157

0001 0101 0111

- Exemplo BCD → Decimal

100010010101

895

Decimal	Binário			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Códigos BCD (Binary Coded Decimal)

- Algoritmo para soma de números BCD
  1. Efetuar a soma binária convencional dos dois números
  2. Adicionar 6 a cada nibble (grupo de 4 bits) que não seja um valor BCD válido
  3. Repetir o passo 2 até que todos os nibbles do resultado correspondam a valores BCD válidos

Códigos BCD (Binary Coded Decimal)

- Exemplo de soma de números BCD

$$
\begin{array}{rrrl}
 & 7 & 4 & \\
 & 2 & 9 & + \\ \hline
 \multicolumn{3}{c}{\underbrace{1\quad 0\quad 3}} &
\end{array}
$$

$$
\begin{array}{rcccccccccccl}
 & & 0&1&1&1&0&1&0&0 & & \\
 & & 0&0&1&0&1&0&0&1 & + & \\ \hline
 & & 1&0&0&1& \multicolumn{4}{c}{\boxed{1\;\;1\;\;0\;\;1}} & & \leftarrow \textit{Nibble}\text{ não é BCD!} \\
 & & & & & & 0&1&1&0 & + & \leftarrow \text{Some 6 ao }\textit{nibble} \\ \hline
 \textit{Nibble}\text{ não é BCD!} \rightarrow & & \multicolumn{4}{c}{\boxed{1\;\;0\;\;1\;\;0}} & 0&0&1&1 & & \\
 \text{Some 6 ao }\textit{nibble} \rightarrow & & 0&1&1&0 & & & & & + & \\ \hline
 & 1 & \multicolumn{4}{c}{\underbrace{0\;\;0\;\;0\;\;0}_{0}} & \multicolumn{4}{c}{\underbrace{0\;\;0\;\;1\;\;1}_{3}} & & \\
 & 1 & & & & & & & & & &
\end{array}
$$

Códigos BCD (Binary Coded Decimal)

- Exemplo de soma de números BCD

```
  7 4                                      0 1 1 1 0 1 0 0
  8 8 +                                    1 0 0 0 1 0 0 0 +
 ------                                    ---------------
1 6 2     Nibble não é BCD! ->             [1 1 1 1] [1 1 0 0] <- Nibble não é BCD!
          Some 6 ao nibble ->              [0 1 1 0] [0 1 1 0] + <- Some 6 ao nibble
                                           ---------------
                                        1   0 1 1 0   0 0 1 0
                                        1      6         2
```

Código de Gray

- É um código numérico binário onde dois valores sucessivos diferem em somente um bit

- Também conhecido como código binário refletido, pois o código de Gray para n bits pode ser obtido a partir da reflexão do código de Gray para (n-1) bits em torno de um eixo situado ao término do código
  - Adiciona-se "0" como bit mais significativo (MSB - Most Significant Bit) acima do eixo
  - Adiciona-se "1" como MSB abaixo do eixo.

Código de Gray

1 bit

0
1

2 bits

0	0
0	1
1	1
1	0

Refletir

3 bits

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

Refletir

4 bits

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

Refletir

Código de Gray

- Conversão Gray – binário
  - g_i = i-ésimo bit do código de Gray
    - g0 = MSB
  - b_i = i-ésimo bit do código binário
  - b_0 = MSB

$$g_0 = b_0$$

$$g_i = \begin{cases} b_i = b_{i-1} \rightarrow \; g_i = 0 \\ b_i \neq b_{i-1} \rightarrow \; g_i = 1 \end{cases}$$

Gray g_0	g_1	g_2	Binário b_0	b_1	b_2
0	0	0	0	0	0
0	0	1	0	0	1
0	1	1	0	1	0
0	1	0	0	1	1
1	1	0	1	0	0
1	1	1	1	0	1
1	0	1	1	1	0
1	0	0	1	1	1

Código de Gray

- Conversão Gray – binário
  - g_i = i-ésimo bit do código de Gray
    - g0 = MSB
  - b_i = i-ésimo bit do código binário
  - b_0 = MSB

Gray g_0	g_1	g_2	*Binário* b_0	b_1	b_2
0	0	0	0	0	0
0	0	1	0	0	1
0	1	1	0	1	0
0	1	0	0	1	1
1	1	0	1	0	0
1	1	1	1	0	1
1	0	1	1	1	0
1	0	0	1	1	1

$$g_0 = b_0$$

$$g_i = \begin{cases} b_i = b_{i-1} \rightarrow \ g_i = 0 \\ b_i \neq b_{i-1} \rightarrow \ g_i = 1 \end{cases}$$

$$g_i = b_i \; XOR \; b_{i-1}$$

Código de Gray

- Exemplo → Converter 101 para código de gray

Binário

b_0	b_1	b_2
1	0	1

Código de Gray

g_0	g_1	g_2

Código de Gray

- Exemplo → Converter 101 para código de gray

Binário

b_0	b_1	b_2
1	0	1

Código de Gray

g_0	g_1	g_2
1		

$$g_0 = b_0 = 1$$

Código de Gray

- Exemplo → Converter 101 para código de gray

Binário

b_0	b_1	b_2
1	0	1

Código de Gray

g_0	g_1	g_2
1	1	

$$g_i = \begin{cases} b_i = b_{i-1} \rightarrow \; g_i = 0 \\ b_i \neq b_{i-1} \rightarrow \; g_i = 1 \end{cases}$$

$$g_1 = \begin{cases} b_1 = b_0 \rightarrow \; g_1 = 0 \\ b_1 \neq b_0 \rightarrow \; g_1 = 1 \end{cases}$$

Código de Gray

- Exemplo → Converter 101 para código de gray

Binário

b_0	b_1	b_2
1	0	1

Código de Gray

g_0	g_1	g_2
1	1	1

$$g_i = \begin{cases} b_i = b_{i-1} \;\rightarrow\; g_i = 0 \\ b_i \neq b_{i-1} \;\rightarrow\; g_i = 1 \end{cases}$$

$$g_2 = \begin{cases} b_2 = b_1 \;\rightarrow\; g_2 = 0 \\ b_2 \neq b_1 \;\rightarrow\; g_2 = 1 \end{cases}$$

Código de Gray

- Exemplo → Converter 101 para código de gray

Binário

b_0	b_1	b_2
1	0	1

Código de Gray

g_0	g_1	g_2
1	1	1

Gray g_0	g_1	g_2	*Binário* b_0	b_1	b_2
0	0	0	0	0	0
0	0	1	0	0	1
0	1	1	0	1	0
0	1	0	0	1	1
1	1	0	1	0	0
1	1	1	1	0	1
1	0	1	1	1	0
1	0	0	1	1	1

Código de Gray

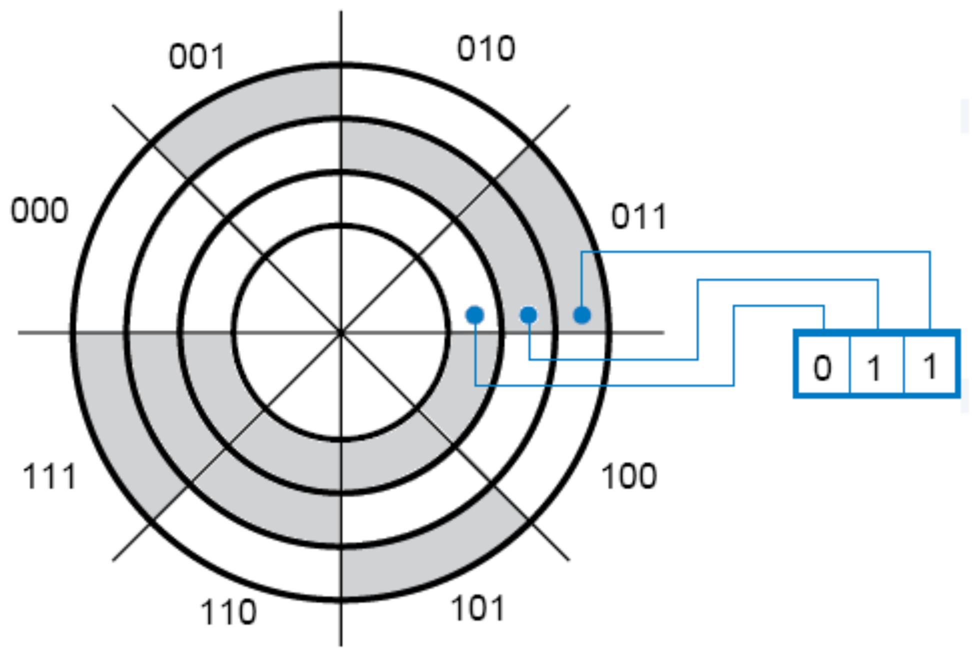
- Principais utilizações

  - Codificadores mecânicos
    - Pequenas mudanças de posição afetam apenas um único bit, diferentemente de certas situações que ocorrem com o código binário tradicional

  - Mapas de Karnaugh
    - O ordenamento das células é feito segundo o código de Gray, para possibilitar as simplificações booleanas
    - Visto nas próximas aulas

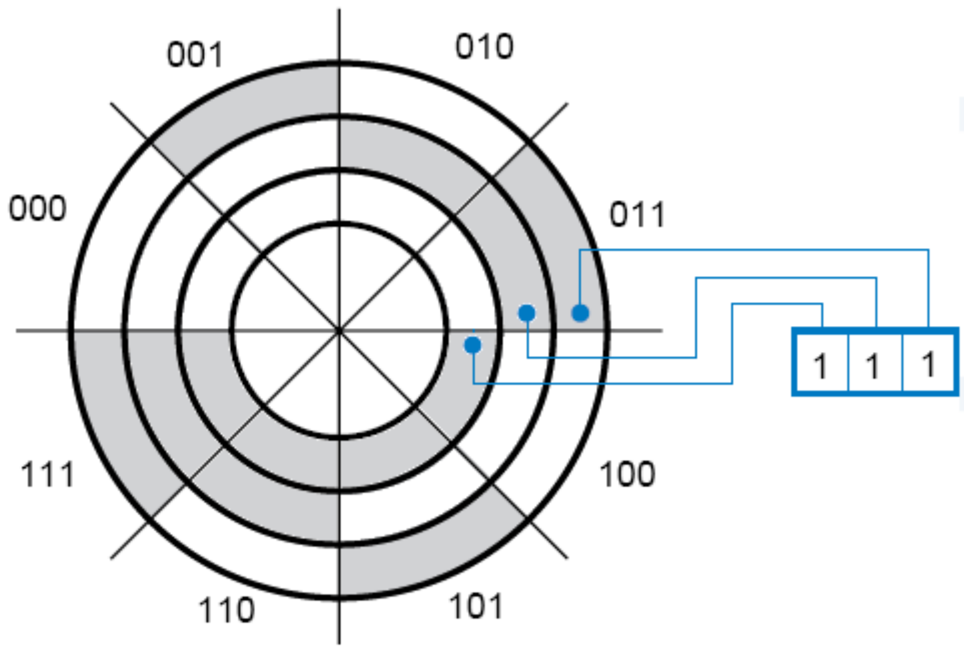
Código de Gray

- Ex: Codificador binário para um sistema mecânico rotacional
  - Codificação binária: 45°



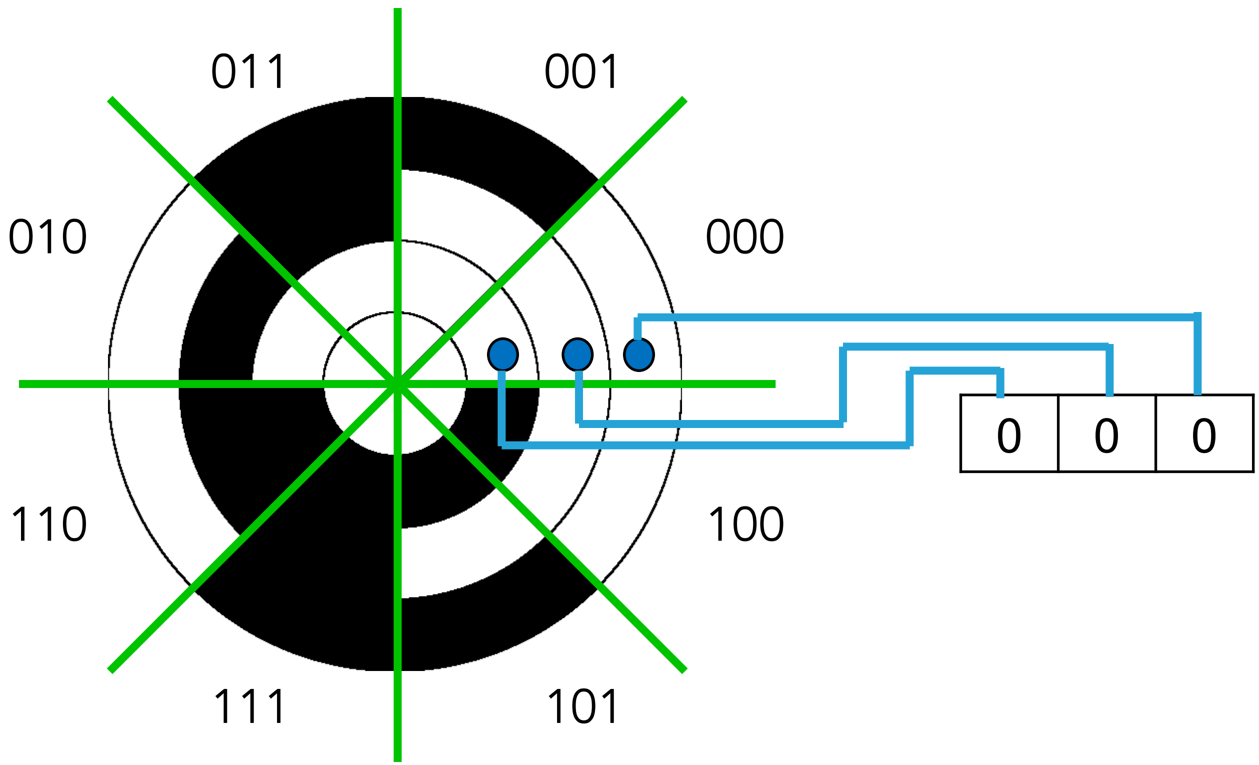
Código de Gray

- Ex: Codificador binário para um sistema mecânico rotacional
  - Caso haja desbalanceamento nas agulhas o erro produzido pode ser grande:



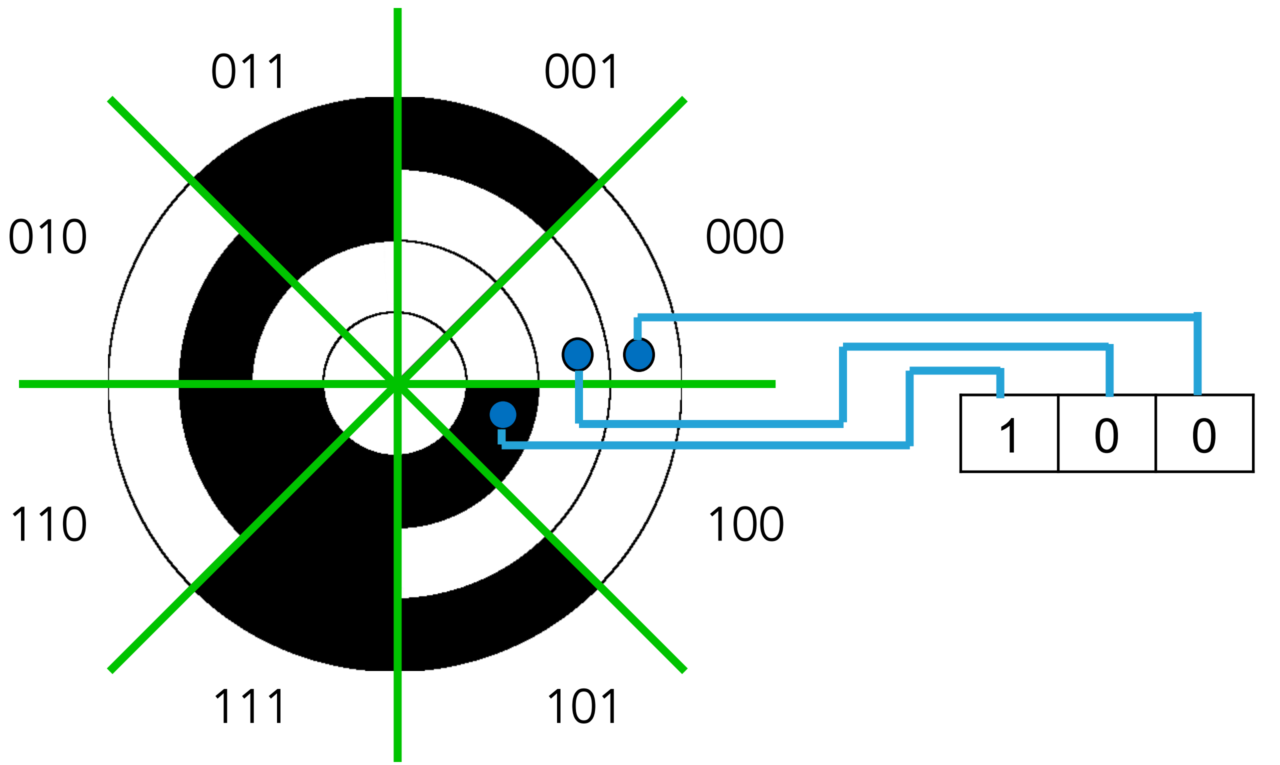
Código de Gray

- Ex: Codificador Gray para um sistema mecânico rotacional



Código de Gray

- Ex: Codificador Gray para um sistema mecânico rotacional



Código k de n

- São códigos ponderados constituídos por n bits
  - k bits são 1
  - n-k bits são 0

- Normalmente utilizados em detecção de erros transmissões de dados

- Número de combinações possíveis

$$\frac{n!}{k!(n-k)!}$$

Código k de n

- Exemplo: código 74210, ou 2 de 5
  - A detecção de erros pode ser feita simplesmente conferindo se o número de **1s** for diferente de 2

Dígito decimal	(7	4	2	1	0)
0	1	1	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	0
7	1	0	0	0	1
8	1	0	0	1	0

Caso especial

Código k de n

- Exemplo: código 2 de 7 bits ponderado
  - (50 43210), ou biquinário

Dígito decimal	(5	0	4	3	2	1	0)
0	0	1	0	0	0	0	1
1	0	1	0	0	0	1	0
2	0	1	0	0	1	0	0
3	0	1	0	1	0	0	0
4	0	1	1	0	0	0	0
5	1	0	0	0	0	0	1
6	1	0	0	0	0	1	0
7	1	0	0	0	1	0	0
8	1	0	0	1	0	0	0
9	1	0	1	0	0	0	0

Códigos de Paridade

- Em códigos de paridade simples acrescenta-se um bit à palavra de tal forma que a paridade seja par ou ímpar

- O objetivo é a detecção de erros simples na transmissão de dados

Códigos de Paridade

Paridade

Código	Par	Nº de "1"s
0 0 0	0	0
0 0 1	1	2
0 1 0	1	2
0 1 1	0	2
1 0 0	1	2
1 0 1	0	2
1 1 0	0	2
1 1 1	1	4

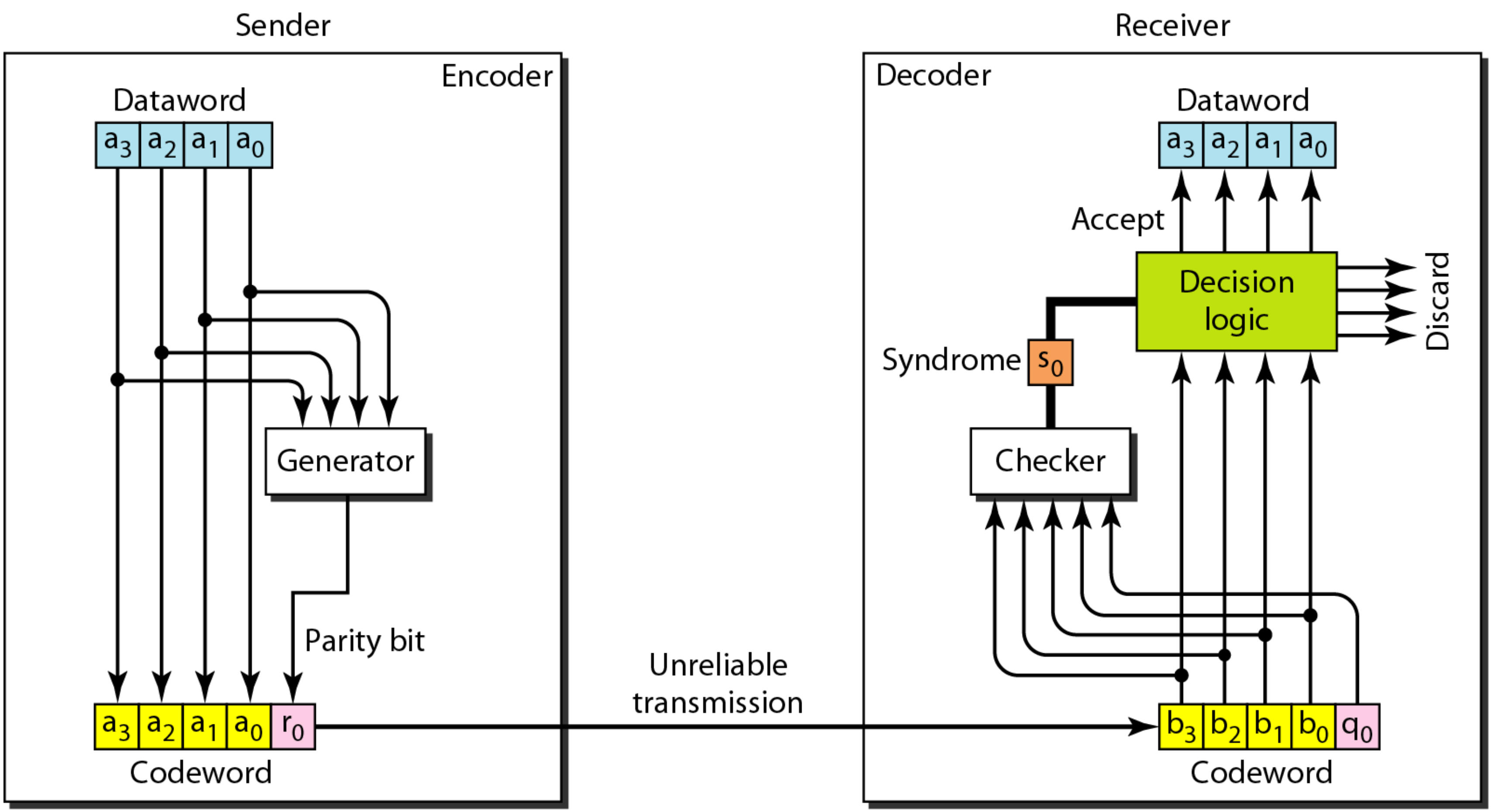
Códigos de Paridade

Paridade

Código	Ímpar	Nº de "1"s
0 0 0	1	1
0 0 1	0	1
0 1 0	0	1
0 1 1	1	3
1 0 0	0	1
1 0 1	1	3
1 1 0	1	3
1 1 1	0	3

Códigos de Paridade

- Paridade em transmissão de dados



Código de Hamming

- Códigos de Hamming utilizam vários bits de paridade para
  - a detecção e correção de erros simples (em apenas um bit da palavra)
  - a detecção (mas não a correção) de erros duplos

- Código de Hamming terá:
  - d bits de dados
  - p bits de paridade

Código de Hamming

- Com d bits de dados, precisamos que a paridade seja:

$$2^{p} \geq d + p + 1, p \geq 2$$

Exemplo

- 4 bits de dados

$$2^p \geq d + p + 1, p \geq 2$$

  - Começamos com p = 2

$$2^2 \geq 4 + 2 + 1$$
$$4 \geq 7$$ ☹

  - Agora vamos tentar p = 3

$$2^3 \geq 4 + 3 + 1$$
$$8 \geq 8$$ ☺

Código de Hamming (7,4)

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7

Definimos que a paridade será PAR

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7

Teremos 3 bits de paridade → 7 - 4 = 3

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7
	P_1	P_2		P_4			

Os bits de paridade serão colocados nos bits que representam potências de 2: 1, 2, 4

Assim teremos as paridades: $P_1P_2P_4$

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7
	P_1	P_2	1	P_4	1	0	1

Os bits de dados são colocados de forma ordenada nos bits restantes

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7
	P_1	P_2	1	P_4	1	0	1
P_1	P_1		1		1		1

Para calcular a paridade P_1 :

- Começamos a partir da posição de P_1 (1)
- Pegamos 1 bit
- Pulamos 1 bit
- Repetimos até o fim

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7
	P_1	P_2	1	P_4	1	0	1
P_1	P_1		1		1		1
P_2		P_2	1			0	1

Para calcular a paridade P_2 :

- Começamos a partir da posição de P_2 (2)
- Pegamos 2 bits
- Pulamos 2 bits
- Repetimos até o fim

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7
	P_1	P_2	1	P_4	1	0	1
P_1	P_1		1		1		1
P_2		P_2	1		1	0	
P_4				P_4	1	0	1

Para calcular a paridade P_4 :

- Começamos a partir da posição de P_4 (4)
- Pegamos 4 bits
- Pulamos 4 bits
- Repetimos até o fim

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7
	1	P_2	1	P_4	1	0	1
P_1	1		1		1		1
P_2		P_2	1			0	1
P_4				P_4	1	0	1

Definimos a paridade como PAR :

$P_1 = 1$

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7
	1	0	1	P_4	1	0	1
P_1	1		1		1		1
P_2		0	1			0	1
P_4				P_4	1	0	1

Definimos a paridade como PAR :

$$P_2 = 0$$

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7
	1	0	1	0	1	0	1
P_1	1		1		1		1
P_2		0	1			0	1
P_4				0	1	0	1

Definimos a paridade como PAR :

$P_4 = 0$

Código de Hamming

- Código de Hamming (7,4)
  - Exemplo → 4 bits de dados: 1101

Bit	1	2	3	4	5	6	7
	1	0	1	0	1	0	1
P_1	1		1		1		1
P_2		P_2	1			0	1
P_4				0	1	0	1

O valor a ser transmitido será : 1010101

Código de Hamming

- Código de Hamming (7,4) → CORREÇÃO
  - Exemplo → Valor recebido = 1010111

Bit	1	2	3	4	5	6	7
	1	0	1	0	1	1	1

Código de Hamming

- Código de Hamming (7,4) → CORREÇÃO
  - Exemplo → Valor recebido = 1010111

Bit	1	2	3	4	5	6	7
	1	0	1	0	1	1	1

BIT 6 CONTÉM ERRO!

Código de Hamming

- Código de Hamming (7,4) → CORREÇÃO
  - Exemplo → Valor recebido = 1010111

Bit	1	2	3	4	5	6	7
	1	0	1	0	1	1	1

Deveremos recalcular as paridades para detectar um erro!

Definimos a paridade como PAR

Código de Hamming

- Código de Hamming (7,4) → CORREÇÃO
  - Exemplo → Valor recebido = 1010111

Bit	1	2	3	4	5	6	7
	1	0	1	0	1	1	1
P_1	1		1		1		1

P_1 → 1 1 1 1 → Valor PAR de 1s → ☺

Código de Hamming

- Código de Hamming (7,4) → CORREÇÃO
  - Exemplo → Valor recebido = 1010111

Bit	1	2	3	4	5	6	7
	1	0	1	0	1	1	1
P_1	1		1		1		1
P_2		0	1			1	1

P_1 → 1 1 1 1 → Valor PAR de 1s → ☺

P_2 → 0 1 1 1 → Valor ÍMPAR de 1s → ☹

Código de Hamming

- Código de Hamming (7,4) → CORREÇÃO
  - Exemplo → Valor recebido = 1010111

Bit	1	2	3	4	5	6	7
	1	0	1	0	1	1	1
P_1	1		1		1		1
P_2		0	1			1	1
P_4				0	1	1	1

P_1 → 1 1 1 1 → Valor PAR de 1s → ☺

P_2 → 0 1 1 1 → Valor ÍMPAR de 1s → ☹

P_4 → 0 1 1 1 → Valor ÍMPAR de 1s → ☹

Código de Hamming

- Código de Hamming (7,4) → CORREÇÃO
  - Exemplo → Valor recebido = 1010111

Bit	1	2	3	4	5	6	7
	1	0	1	0	1	1	1

Como descobrir o bit errado?

Código de Hamming

- Código de Hamming (7,4) → CORREÇÃO
  - Exemplo → Valor recebido = 1010111

Bit	1	2	3	4	5	6	7
	1	0	1	0	1	1	1

As paridades erradas foram a 2 e a 4:

BIT ERRADO = 2 + 4 = 6

P_1 → 1 1 1 1 → Valor PAR de 1s → ☺

P_2 → 0 1 1 1 → Valor ÍMPAR de 1s → ☹

P_4 → 0 1 1 1 → Valor ÍMPAR de 1s → ☹

Código de Hamming

- Código de Hamming (7,4) → CORREÇÃO
  - Exemplo → Valor recebido = 1010111

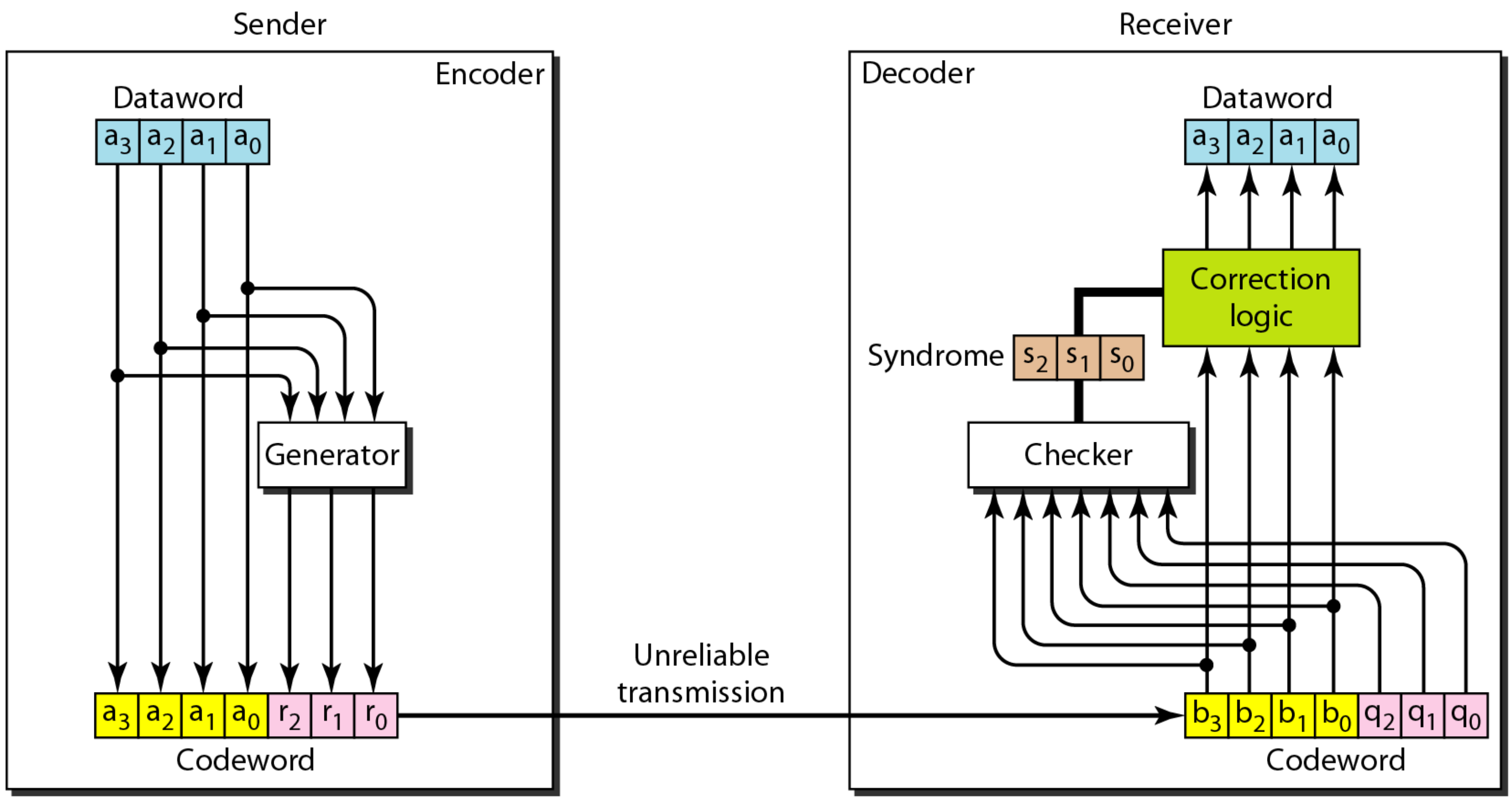
Bit	1	2	3	4	5	6	7
	1	0	1	0	1	1	1

0

O bit 6 é 1, então deveria ser 0!!

Código de Hamming

- *Codificador/decodificador para o Código de Hamming*



Código de Hamming

- *Códigos Hamming até um máximo de 255 bits*

Combinações de parâmetros do códigos de Hamming		
d	p	d + p
Bits de dados	Bits de paridade	Total da mensagem
1	2	3
4	3	7
11	4	15
26	5	31
57	6	63
120	7	127
247	8	255