# CAPÍTULO 2
# SISTEMAS DE NUMERAÇÃO E CÓDIGOS

- Código BCD;
- Comparação entre BCD e Binário;
- Circuitos Digitais para BCD;
- Código Gray;
- Código ASCII;
- Detecção de erros pelo método de Paridade

# O que é um Código?

- Quando números, letras ou palavras são representados por um grupo especial de símbolos, dizemos que eles estão **codificados**.
- O grupo de símbolos utilizado é denominado **código**. Ex. código morse.
- Um número pode ser representado pelo número binário equivalente, resultado da conversão direta.
- Se o número for representado pelo seu binário equivalente, dizemos que é uma **codificação em binário puro**.
- Há codificações que não são em binário puro para simplificar as conversões ou outra vantagem. Por exemplo: o BCD, o código Gray.

# Decimal Codificado em Binário

- Se cada dígito de um número decimal for representado pelo seu equivalente em binário puro, o resultado será um código **denominado decimal codificado em binário** (também conhecido por BCD – Binary-Coded-Decimal)
- Como um dígito decimal pode ter no máximo o valor 9, são necessários 4 bits para codificar cada dígito decimal
- Portanto as codificações para cada dígito são:

| Decimal | BCD | Decimal | BCD |
|---|---|---|---|
| 0 | 0000 | 5 | 0101 |
| 1 | 0001 | 6 | 0110 |
| 2 | 0010 | 7 | 0111 |
| 3 | 0011 | 8 | 1000 |
| 4 | 0100 | 9 | 1001 |

# Código BCD

- Exemplos

| 8 | 7 | 4 |
|---|---|---|
| ↓ | ↓ | ↓ |
| 1000 | 0111 | 0100 |

| 9 | 4 | 3 |
|---|---|---|
| ↓ | ↓ | ↓ |
| 1001 | 0100 | 0011 |

# Código BCD

- Cada conjunto de 4 bits representa um dígito decimal
- Portanto o código BCD não usa as sequências: 1010, 1011, 1100, 1101, 1110 e 1111.
- São usados apenas 10 dos possíveis 16 grupos de 4 bits.
- Se qualquer um desses números de 4 bits não utilizados (ou “proibidos”) aparecer em um circuito que usa o código BCD, deverá ocorrer uma indicação de erro.

# Exemplos Binário para BCD

1. Converta 0110100000111001 (BCD) para seu equivalente decimal

$$\underbrace{0110}_{6}\ \underbrace{1000}_{8}\ \underbrace{0011}_{3}\ \underbrace{1001}_{9}$$

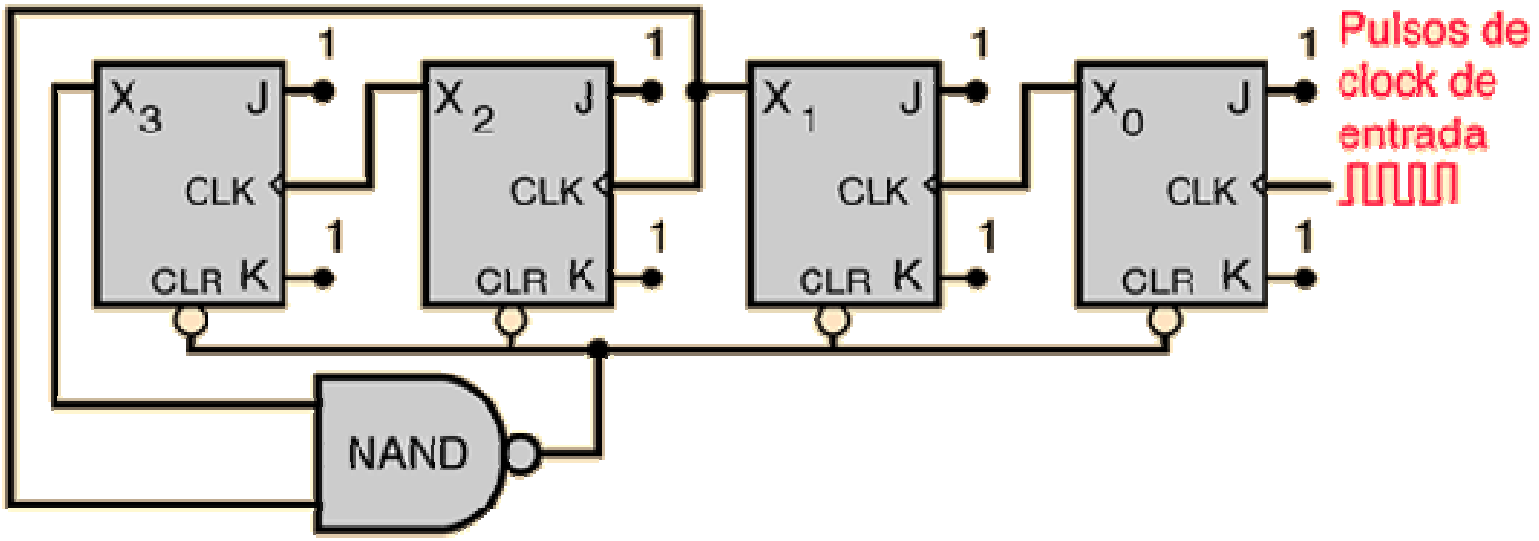
2. Converta 011111000001 (BCD) para seu equivalente decimal

$$\underbrace{0111}_{7}\ \underbrace{1100}_{}\ \underbrace{0001}_{1}$$

**Erro no código BCD: grupo de 4 bits proibido**

# Um contador BCD

- Como veremos mais a frente, uma simples porta lógica pode transformar um contador comum em contador BCD



# Comparação entre BCD e binário

- O BCD não é outro sistema de numeração como o binário, decimal, octal e hexadecimal. É um sistema binário, porém cada dígito é codificado separadamente.
- BCD não é o mesmo que o binário puro.
- Por exemplo, veja a comparação do número 137 em binário puro e BCD:

$$137_{10} = 10001001_2 \qquad \text{(binário)}$$

$$137_{10} = 0001\ \ 0011\ \ 0111 \qquad \text{(BCD)}$$

- BCD requer mais bits que o binário puro para representar os números decimais maiores que um dígito
- Isto acontece porque o BCD não usa todos os 16 grupos possíveis de 4 bits

# Questões sobre BCD

1. Represente o valor decimal 178 em BCD e binário puro

2. Quantos bits são necessários para representar, em BCD, um número decimal de 8 dígitos? E se o mesmo número fosse representado em binário puro, quantos bits seriam necessários?

3. Qual é a vantagem da codificação BCD de um número decimal sobre a codificação em binário puro? E qual é a desvantagem?

# Relações entre as Representações Numéricas

| Decimal | Binário | Octal | Hexadecimal | BCD |
|---|---|---|---|---|
| 0 | 0000 | 0 | 0 | 0000 |
| 1 | 0001 | 1 | 1 | 0001 |
| 2 | 0010 | 2 | 2 | 0010 |
| 3 | 0011 | 3 | 3 | 0011 |
| 4 | 0100 | 4 | 4 | 0100 |
| 5 | 0101 | 5 | 5 | 0101 |
| 6 | 0110 | 6 | 6 | 0110 |
| 7 | 0111 | 7 | 7 | 0111 |
| 8 | 1000 | 10 | 8 | 1000 |
| 9 | 1001 | 11 | 9 | 1001 |
| 10 | 1010 | 12 | A | 0001 0000 |
| 11 | 1011 | 13 | B | 0001 0001 |
| 12 | 1100 | 14 | C | 0001 0010 |
| 13 | 1101 | 15 | D | 0001 0011 |
| 14 | 1110 | 16 | E | 0001 0100 |
| 15 | 1111 | 17 | F | 0001 0101 |

# Códigos Alfanuméricos

- Um computador precisa ser capaz de manipular informações não numérica
- Código ASCII (*American Standard Code for Information Interchange*)
  - Possui 7 bits (portanto tem $2^7 = 128$ representações codificadas)
  - Também representa códigos de controle como o <RETURN> ou CR (Carriage Return) (`'\n'` em C) e o <LINEFEED> ou LF (`'\r'` em C)
  - Códigos ASCII representam texto puro (o chamado *plain text*) em computadores e equipamentos de comunicação
- Outros exemplos: ISO 8859 e  Unicode (UTF-8, UTF-16), EBCDIC (Mainframes IBM)

# Código ASCII

| Bits $b_4$ $b_3$ $b_2$ $b_1$ | Row ↓ / Column → | $b_7$ $b_6$ $b_5$ = 0 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1 0 0 | 1 0 1 | 1 1 0 | 1 1 1 |
|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 0 0 0 | 0 | NUL | DLE | SP | 0 | @ | P | ` | p |
| 0 0 0 1 | 1 | SOH | DC1 | ! | 1 | A | Q | a | q |
| 0 0 1 0 | 2 | STX | DC2 | " | 2 | B | R | b | r |
| 0 0 1 1 | 3 | ETX | DC3 | # | 3 | C | S | c | s |
| 0 1 0 0 | 4 | EOT | DC4 | $ | 4 | D | T | d | t |
| 0 1 0 1 | 5 | ENQ | NAK | % | 5 | E | U | e | u |
| 0 1 1 0 | 6 | ACK | SYN | & | 6 | F | V | f | v |
| 0 1 1 1 | 7 | BEL | ETB | ' | 7 | G | W | g | w |
| 1 0 0 0 | 8 | BS | CAN | ( | 8 | H | X | h | x |
| 1 0 0 1 | 9 | HT | EM | ) | 9 | I | Y | i | y |
| 1 0 1 0 | 10 | LF | SUB | * | : | J | Z | j | z |
| 1 0 1 1 | 11 | VT | ESC | + | ; | K | [ | k | { |
| 1 1 0 0 | 12 | FF | FC | , | < | L | \ | l | \| |
| 1 1 0 1 | 13 | CR | GS | - | = | M | ] | m | } |
| 1 1 1 0 | 14 | SO | RS | . | > | N | ^ | n | ~ |
| 1 1 1 1 | 15 | SI | US | / | ? | O | _ | o | DEL |

# Código ASCII

| Dec | Hx | Oct | Char | Dec | Hx | Oct | Char | Dec | Hx | Oct | Char | Dec | Hx | Oct | Char |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 000 | NUL (null) | 32 | 20 | 040 | SPACE | 64 | 40 | 100 | @ | 96 | 60 | 140 | ` |
| 1 | 1 | 001 | SOH (start of heading) | 33 | 21 | 041 | ! | 65 | 41 | 101 | A | 97 | 61 | 141 | a |
| 2 | 2 | 002 | STX (start of text) | 34 | 22 | 042 | " | 66 | 42 | 102 | B | 98 | 62 | 142 | b |
| 3 | 3 | 003 | ETX (end of text) | 35 | 23 | 043 | # | 67 | 43 | 103 | C | 99 | 63 | 143 | c |
| 4 | 4 | 004 | EOT (end of transmission) | 36 | 24 | 044 | $ | 68 | 44 | 104 | D | 100 | 64 | 144 | d |
| 5 | 5 | 005 | ENQ (enquiry) | 37 | 25 | 045 | % | 69 | 45 | 105 | E | 101 | 65 | 145 | e |
| 6 | 6 | 006 | ACK (acknowledge) | 38 | 26 | 046 | & | 70 | 46 | 106 | F | 102 | 66 | 146 | f |
| 7 | 7 | 007 | BEL (bell) | 39 | 27 | 047 | ' | 71 | 47 | 107 | G | 103 | 67 | 147 | g |
| 8 | 8 | 010 | BS (backspace) | 40 | 28 | 050 | ( | 72 | 48 | 110 | H | 104 | 68 | 150 | h |
| 9 | 9 | 011 | TAB (horizontal tab) | 41 | 29 | 051 | ) | 73 | 49 | 111 | I | 105 | 69 | 151 | i |
| 10 | A | 012 | LF (NL line feed, new line) | 42 | 2A | 052 | * | 74 | 4A | 112 | J | 106 | 6A | 152 | j |
| 11 | B | 013 | VT (vertical tab) | 43 | 2B | 053 | + | 75 | 4B | 113 | K | 107 | 6B | 153 | k |
| 12 | C | 014 | FF (NP form feed, new page) | 44 | 2C | 054 | , | 76 | 4C | 114 | L | 108 | 6C | 154 | l |
| 13 | D | 015 | CR (carriage return) | 45 | 2D | 055 | - | 77 | 4D | 115 | M | 109 | 6D | 155 | m |
| 14 | E | 016 | SO (shift out) | 46 | 2E | 056 | . | 78 | 4E | 116 | N | 110 | 6E | 156 | n |
| 15 | F | 017 | SI (shift in) | 47 | 2F | 057 | / | 79 | 4F | 117 | O | 111 | 6F | 157 | o |
| 16 | 10 | 020 | DLE (data link escape) | 48 | 30 | 060 | 0 | 80 | 50 | 120 | P | 112 | 70 | 160 | p |
| 17 | 11 | 021 | DC1 (device control 1) | 49 | 31 | 061 | 1 | 81 | 51 | 121 | Q | 113 | 71 | 161 | q |
| 18 | 12 | 022 | DC2 (device control 2) | 50 | 32 | 062 | 2 | 82 | 52 | 122 | R | 114 | 72 | 162 | r |
| 19 | 13 | 023 | DC3 (device control 3) | 51 | 33 | 063 | 3 | 83 | 53 | 123 | S | 115 | 73 | 163 | s |
| 20 | 14 | 024 | DC4 (device control 4) | 52 | 34 | 064 | 4 | 84 | 54 | 124 | T | 116 | 74 | 164 | t |
| 21 | 15 | 025 | NAK (negative acknowledge) | 53 | 35 | 065 | 5 | 85 | 55 | 125 | U | 117 | 75 | 165 | u |
| 22 | 16 | 026 | SYN (synchronous idle) | 54 | 36 | 066 | 6 | 86 | 56 | 126 | V | 118 | 76 | 166 | v |
| 23 | 17 | 027 | ETB (end of trans. block) | 55 | 37 | 067 | 7 | 87 | 57 | 127 | W | 119 | 77 | 167 | w |
| 24 | 18 | 030 | CAN (cancel) | 56 | 38 | 070 | 8 | 88 | 58 | 130 | X | 120 | 78 | 170 | x |
| 25 | 19 | 031 | EM (end of medium) | 57 | 39 | 071 | 9 | 89 | 59 | 131 | Y | 121 | 79 | 171 | y |
| 26 | 1A | 032 | SUB (substitute) | 58 | 3A | 072 | : | 90 | 5A | 132 | Z | 122 | 7A | 172 | z |
| 27 | 1B | 033 | ESC (escape) | 59 | 3B | 073 | ; | 91 | 5B | 133 | [ | 123 | 7B | 173 | { |
| 28 | 1C | 034 | FS (file separator) | 60 | 3C | 074 | < | 92 | 5C | 134 | \ | 124 | 7C | 174 | \| |
| 29 | 1D | 035 | GS (group separator) | 61 | 3D | 075 | = | 93 | 5D | 135 | ] | 125 | 7D | 175 | } |
| 30 | 1E | 036 | RS (record separator) | 62 | 3E | 076 | > | 94 | 5E | 136 | ^ | 126 | 7E | 176 | ~ |
| 31 | 1F | 037 | US (unit separator) | 63 | 3F | 077 | ? | 95 | 5F | 137 | _ | 127 | 7F | 177 | DEL |

# Exemplo

- A seguinte sequência de bits é uma mensagem codificada em ASCII. Que mensagem é essa?

  1001000 1000101 1001100 1010000

- Solução:

Converta cada código de 7 bits em seu equivalente Hexa:

48 45 4C 50

Localize na tabela:

H E L P

# Exemplo

- Um operador está digitando um programa em Basic em um determinado computador. Ele converte cada tecla no código ASCII equivalente e armazena o código como um byte na memória. Determine a cadeia de caracteres binária que deve ser armazenada na memória quando o operador digita a seguinte instrução em Basic:

GOTO 25

Solução:

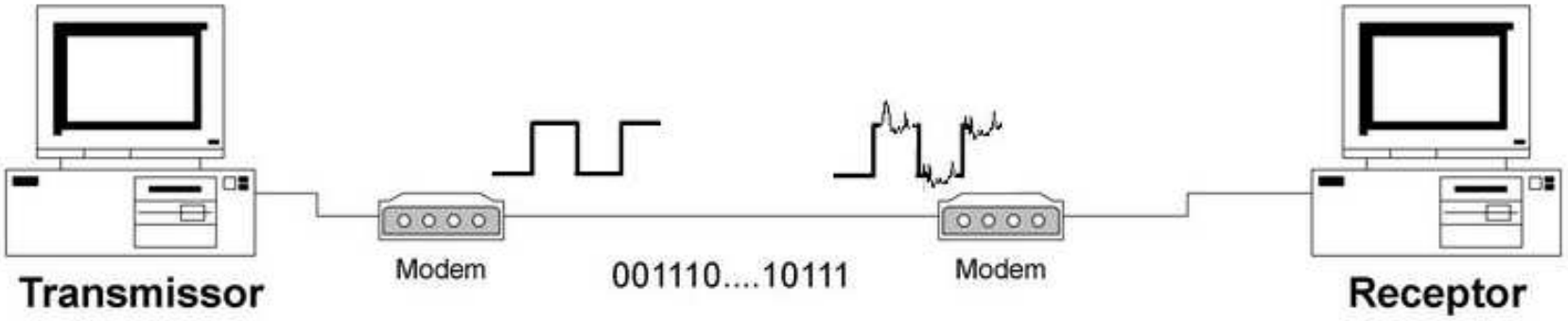
| | |
|---|---|
| G | 01000111 |
| O | 01001111 |
| T | 01010100 |
| O | 01001111 |
| (espaço) | 00100000 |
| 2 | 00110010 |
| 5 | 00110101 |

# Detecção de Erros pelo Método da Paridade

- A movimentação de dados e códigos binários de um local para outro é a operação mais frequente em circuitos digitais. Exemplos:
  - Transmissão de voz digitalizada por um enlace (link) de microondas
  - Armazenamento e recuperação de dados em disco
  - Transmissão de dados por meio de linha telefônica (modem)
- Há a possibilidade de ocorrência de erros
- Causa principal: ruído elétrico (flutuações aleatórias da tensão ou corrente que estão presentes em todos os sistemas eletrônicos em intensidades diversas)

# Bit de Paridade

- Ruído em uma transmissão de dados:



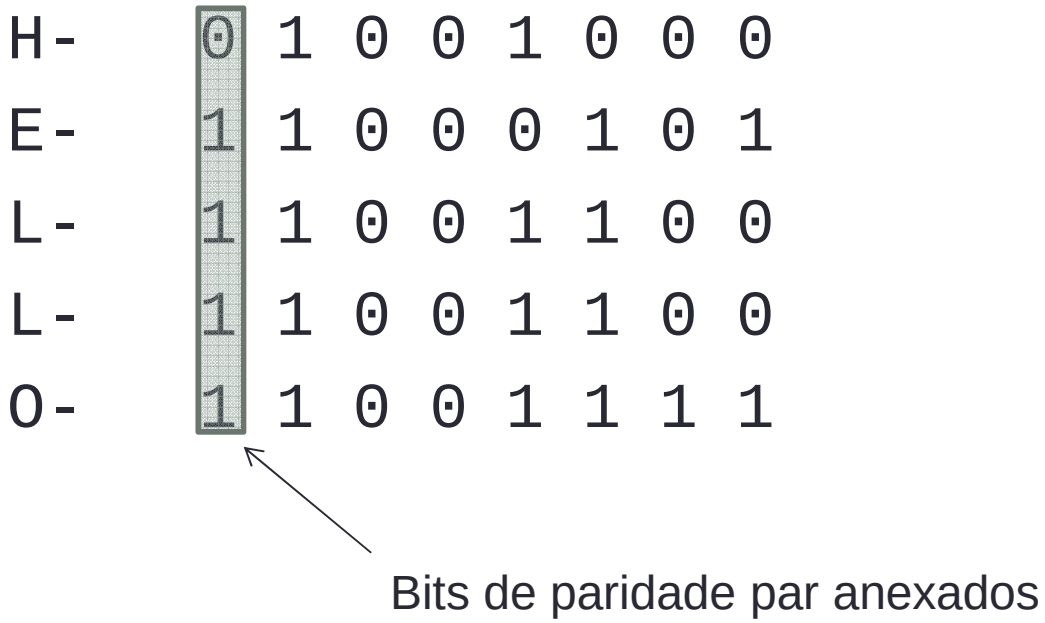

- No momento de uma flutuação, o modem receptor pode interpretar 0 como 1 e vice-versa.
- Os sistemas devem usar algum método de detecção de erro. Uma das técnicas mais simples é o método do **Bit de Paridade**

# Bit de Paridade

- Consiste em adicionar um bit extra ao conjunto de bits do código, podendo ser 0 ou 1. Existem dois métodos:
- **Paridade par**
  - O total de bits 1 do código, incluindo o bit de paridade, deve ser par.
  - Suponha que o conjunto de bits seja 1000011 (caractere C do código ASCII). Neste caso o bit de paridade será 1. A sequência transmitida será **1**1000011
  - Se o número de bits já for par, então adiciona-se o bit 0 de paridade. Ex.: Se o código for 1000001 a sequência transmitida será **0**1000001
- **Paridade ímpar**
  - Idêntico, porém o total de bits 1 do código, incluindo o bit de paridade, deve ser ímpar.

# Exemplo

- Para se transmitir a sequência de caracteres “HELLO” usando-se a tabela ASCII e bits de paridade par, qual será a sequência de bits a ser transmitida?

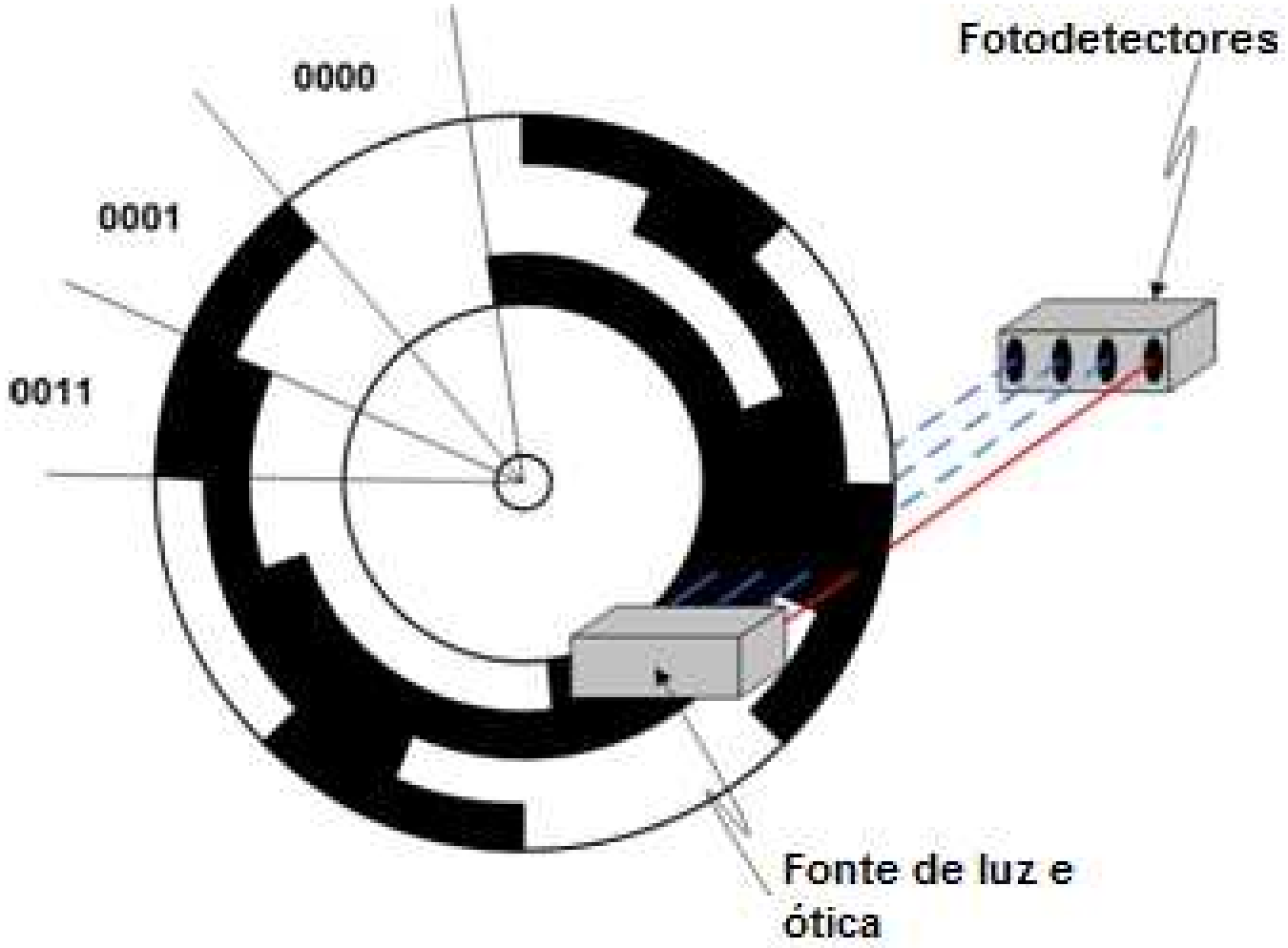


# Código Gray

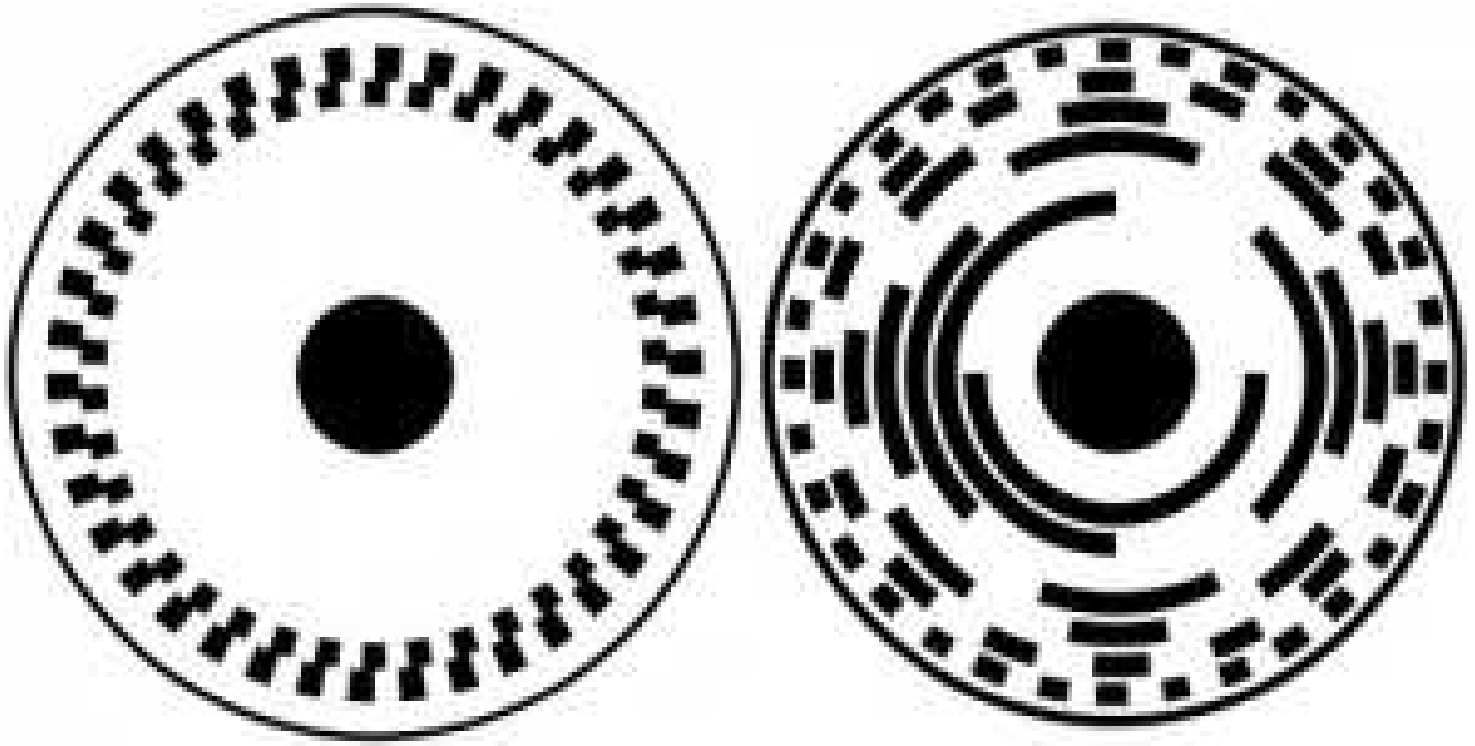
- O código Gray é um sistema de código binário inventado por Frank Gray.
- De um número para outro apenas um bit varia

| Decimal | Binário | Gray | Decimal | Binário | Gray |
|---|---|---|---|---|---|
| 0 | 0000 | 0000 | 8 | 1000 | 1100 |
| 1 | 0001 | 0001 | 9 | 1001 | 1101 |
| 2 | 0010 | 0011 | 10 | 1010 | 1111 |
| 3 | 0011 | 0010 | 11 | 1011 | 1110 |
| 4 | 0100 | 0110 | 12 | 1100 | 1010 |
| 5 | 0101 | 0111 | 13 | 1101 | 1011 |
| 6 | 0110 | 0101 | 14 | 1110 | 1001 |
| 7 | 0111 | 0100 | 15 | 1111 | 1000 |

# Aplicação Código Gray: encoder absoluto



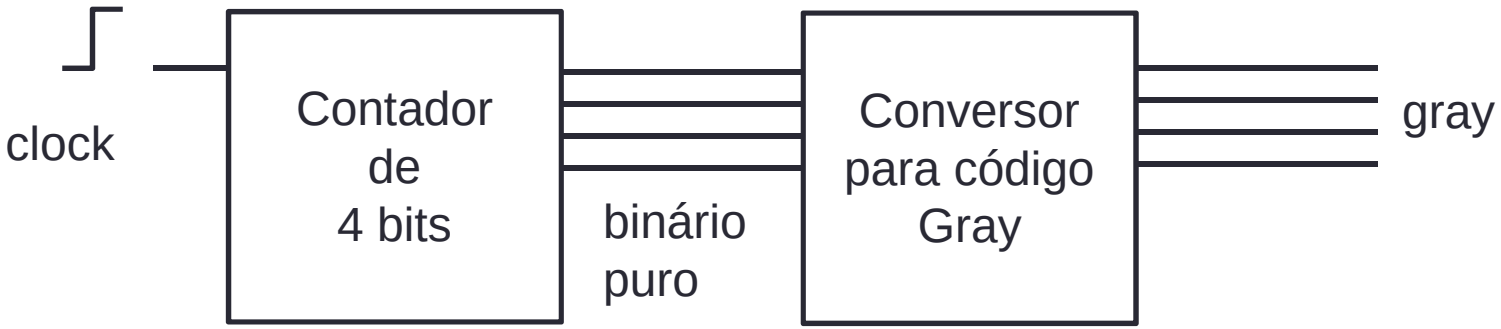
# Encoder absoluto x incremental



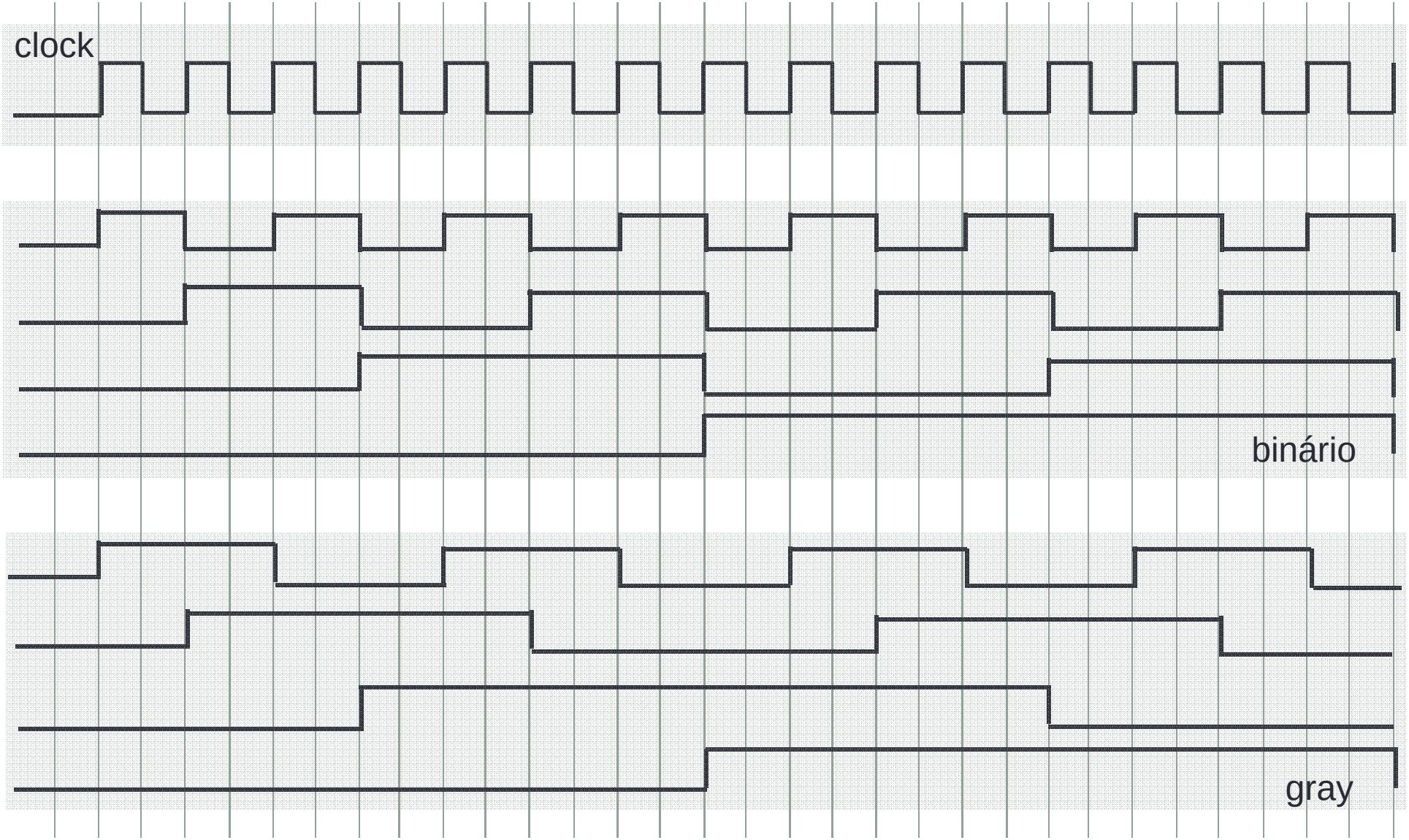
# Encoder absoluto

# Exercício

- Um contador de 4 bits está ligado a um circuito digital que converte a entrada binária em código Gray. Desenhe o diagrama de tempo: (a) para a saída binária do contador e (b) para a saída em código gray do conversor.



# Solução:



# Exercício

- Em um determinado sistema digital os números decimais de 000 a 999 são codificados em BCD. Um bit de paridade ímpar foi anexado ao final de cada sequência de bits. Determine quais das sequências estão com erro.

- A) 1001010110000
- B) 0100011101100
- C) 0111110000011
- D) 1000011000101