Introdução à Ciência da Computação
Introdução à Engenharia de Computação

Codificação Binária

Profa. Ana Marilza Pernas Fleishmann
Profa. Lisane Brisolara de Brisolara
Prof. Giovani Parente Farias
Prof. Rafael Iankowski Soares

Introdução

- Computadores trabalham com números binários
- No dia-a-dia, usamos o sistema decimal
- Obriga a conversão de números de decimal para binário e vice-versa

Introdução

- Computadores trabalham com números binários
- Qualquer informação seja numérica ou alfabética deve ser representada **por bits**
- **Codificação** é o mapeamento de símbolos para uma representação binária (em bits)
  - Ex. de codificação para números: complemento de 1 , complemento de 2, sinal/magnitude
- Porém nem tudo são números...

Introdução

- Precisamos manipular caracteres alfanuméricos, símbolos especiais (pontuação, letras gregas, etc)

- Com *n* bits pode-se representar 2^n símbolos distintos → 2^n combinações

- O código pode não usar todas as combinações possíveis

- Cada sistema de codificação pode definir suas próprias regras de formação de códigos

Alguns Códigos Binários

- Ponderados (BCD, Excesso-de-3, ...)
- Gray
- Hamming
- Códigos Alfanuméricos:
  - EBCDIC
  - ASCII
  - Unicode...

BCD

(*BINARY CODED DECIMAL*)

Sistema BCD

- **BCD**: *Binary Coded Decimal* ou Decimal Codificado em Binário
  - **Cada algarismo decimal será representado por 4 bits** (em binário)
    - Com **4 bits** pode-se representar **16** combinações diferentes (0000-1111)
    - Mas, são somente 10 dígitos binários (0-9), então algumas combinações não serão usadas
  - Código ponderado: baseado em pesos

Sistema BCD

- **Exemplo**

$832_{\text{(base 10)}} = 1000\ 0011\ 0010_{\text{BCD}}$

8	3	2	
1000	0011	0010	
8 4 2 1	8 4 2 1	8 4 2 1	← pesos

Ilustração do método de codificação: cada algarismo do número decimal é convertido para um número binário de 4 bits, usando pesos 1, 2, 4, 8 (da direita para a esquerda)

BCD ou Normal BCD

Também chamado BCD 8-4-2-1

8	3	2
1000	0011	0010
8 4 2 1	8 4 2 1	8 4 2 1

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

Tabela com códigos BCD para algarismos de 0 a 9

códigos **inválidos** no BCD: 1010, 1011, 1100, 1101, 1110, 1111

Como ficam em BCD?

- 983
- 372
- 1633

Como ficam em BCD?

- 983 → 1001 1000 0011
- 372 → 0011 0111 0010
- 1633 → 0001 0110 0011 0011

Excesso de 3

- **Ou Stibitz**
- **Ponderado:** Usa mesmos pesos do BCD 8-4-2-1, mas **subtrai 3** unidades
- Nenhum código utiliza a combinação “0000”, “0001” e nem “0010”
- Se temos o código “0111”, qual o correspondente em decimal?

0111 = 0*8 + 1*4 + 1*2 + 1*1 **- 3** = 4 base 10

Excesso de 3

Decimal	BCD	Excesso de 3	
0	0000	0011	1*2+1*1**-3**=0
1	0001	0100	
2	0010	0101	1*4+1*1**-3**=2
3	0011	0110	
4	0100	0111	1*4+1*2+1*1**-3**=4
5	0101	1000	
6	0110	1001	
7	0111	1010	
8	1000	1011	
9	1001	1100	

Tabela com códigos de 4 bits, BCD e Excesso de 3, correspondentes aos algarismos de 0 a 9

Códigos de 4 bits

Tabela de códigos ponderados: Variações do BCD

Decimal	**NBCD (8421)**	**Stibitz (8421-3) Execesso de 3**	**Aiken (2421)**	**Cod. 7421 (7421)**
0	0000	0011	0000	0000
1	0001	0100	0001	0001
2	0010	0101	0010	0010
3	0011	0110	0011	0011
4	0100	0111	0100	0100
5	0101	1000	1011	0101
6	0110	1001	1100	0110
7	0111	1010	1101	1000
8	1000	1011	1110	1001
9	1001	1100	1111	1010

Códigos de 4 bits

Tabela de códigos ponderados: Variações do BCD

Decimal	**NBCD (8421)**	**Stibitz (8421-3) Execesso de 3**	**Aiken (2421)**	**Cod. 7421 (7421)**
0	0000	0011	0000	0000
1	0001	0100	0001	[illegible]
2	0010	0101	0010	0010
3	0011	0110	0011	0011
4	0100	0111	0100	0100
5	0101	1000	1011	0101
6	0110	1001	1100	0110
7	0111	1010	1101	1000
8	1000	1011	1110	1001
9	1001	1100	1111	1010

Excesso de 3

0111: 0***8** + 1***4** + 1***2** + 1***1** = 7 **- 3** = 4

Códigos de 4 bits

Aiken: pesos diferenciados, códigos são diferentes a partir do 5 comparado ao NBCD

Tabela de c

Decimal	**NBCD (8421)**	**Stibitz (8421-3) Execesso de 3**	**Aiken (2421)**	**Cod. 7421 (7421)**
0	0000	0011	0000	0000
1	0001	0100	0001	0001
2	0010	0101	0010	0010
3	0011	0110	0011	0011
4	0100	0111	0100	0100
5	0101	1000	1011	0101
6	0110	1001	1100	0110
7	0111	1010	1101	1000
8	1000	1011	1110	1001
9	1001	1100	1111	1010

$1011 = 1*\mathbf{2} + 0*\mathbf{4} + 1*\mathbf{2} + 1*\mathbf{1} = 5$

Códigos de 4 bits

7421: pesos diferenciados, códigos são diferentes a partir do 7 (comparado ao NBCD)

Tabela de

Decimal	NBCD (8421)	Stibitz (8421-3) Execesso de 3	Aiken (2421)	Cod. 7421 (7421)
0	0000	0011	0000	0000
1	0001	0100	0001	0001
2	0010	0101	0010	0010
3	0011	0110	0011	0011
4	0100	0111	0100	0100
5	0101	1000	1011	0101
6	0110	1001	1100	0110
7	0111	1010	1101	1000
8	1000	1011	1110	1001
9	1001	1100	1111	1010

1010 = 1***7** + 0***4** + 1***2** + 0***1** = 9

CÓDIGOS GRAY

Códigos Gray (ou cíclicos)

- **NÃO** muito usado para cálculos aritméticos

- São usados para indicar a variação de medidas analógicas (temperatura, ângulo, pressão) em forma digital
  - Estas grandezas variam seus valores de forma contínua, não brusca

- **Quando avançamos de um número para o seu adjacente mudamos apenas 1 bit**!
  - Códigos adjacentes se diferenciam por uma posição binária (variação de um único bit)

Código Gray 4 bits

- Todas as 16 combinações possíveis são usadas
- Não existe combinações que indiquem erro
- **Código cíclico** do 15 para o 0 também muda apenas um bit

Decimal	Cód. Gray
0	0 0 0 0
1	0 0 0 1
2	0 0 1 1
3	0 0 1 0
4	0 1 1 0
5	0 1 1 1
6	0 1 0 1
7	0 1 0 0
8	1 1 0 0
9	1 1 0 1
10	1 1 1 1
11	1 1 1 0
12	1 0 1 0
13	1 0 1 1
14	1 0 0 1
15	1 0 0 0

Variação do 3 (0011) para o 4 (0100):

Em binário mudaria 3 bits!

Em Gray apenas 1!

Binário para Código Gray

- Conversão a partir de decimal ou binário
- Decimal -> **Binário** **-> Gray (4 bits)**

5_{10} = 0 1 0 1 Código Binário

Gray

Binário para Código Gray

- Conversão a partir de decimal ou binário
- Decimal -> **Binário -> Gray**

$$5_{10} = 0 \xrightarrow{+} 1 \xrightarrow{+} 0 \xrightarrow{+} 1 \quad \text{Código Binário}$$

$$0 \quad 1 \quad 1 \quad 1 \quad \text{Código Gray}$$

1) Repete o bit de mais alta ordem (MSB) do binário

2) Cada bit é somado em módulo 2 (sem vai-um) com o bit da direita (ou bits adjacentes iguais converte para 0 e diferentes para 1), começando da esquerda para direita.

Ilustração da conversão para gray de 4 bits

Binário para Código Gray

- Trabalhando com mais bits

- **Calcule 45_{10} em código gray de 6 bits**

Binário para Código Gray

- **Calcule 45_{10} em código gray 6 bits**

$45_{10} = 1\ \ 0\ \ 1\ \ 1\ \ 0\ \ 1$ Código Binário

Binário para Código Gray

- **Calcule 45_{10} em código gray?**

$$45_{10} = \boxed{\begin{matrix}1 \\ 1\end{matrix}} \begin{matrix}0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1\end{matrix}$$

Código Binário

Código Gray

1) Repete o bit de mais alta ordem (MSB)

2) Cada bit do código binário é somado em módulo 2 (sem vai-um) com o bit da direita, começando pelo bit MSB e isso gera os próximos bits

Código Gray para Binário

- **Gray → Binário**

1 1 0 1 1

\+ \+ \+ \+

1 0 0 1 0

Código Gray

Código Binário

1) Repete o bit de mais alta ordem (MSB)

2) Some cada bit do código binário gerado ao bit do código Gray na posição adjacente, gerando o próximo bit do binário.

Exercícios

- Quantos números diferentes podem ser representados em uma palavra binária de 6 bits?

- Qual é a representação BCD para 1049_{base10}?

- O seguinte conjunto de bits 110010100011 representa um número usando codificação excesso de 3. Qual é o valor do número decimal representado neste código?

- Calcule o código Gray para o número 23_{base10}

Exercícios

- Quantos números diferentes podem ser representados em uma palavra binária de 6 bits?

$$2^n = 2^6 = 64$$

- Qual é a representação BCD para 1049_{base10}?

- O seguinte conjunto de bits 110010100011 representa um número usando codificação excesso de 3. Qual é o valor do número decimal representado neste código?

- Calcule o código Gray para o número 23_{base10}

Exercícios

- Quantos números diferentes podem ser representados em uma palavra binária de 6 bits?

$$2^n = 2^6 = 64$$

- Qual é a representação BCD para 1049 $_{base10}$?

0001 0000 0100 1001

- O seguinte conjunto de bits 110010100011 representa um número usando codificação excesso de 3. Qual é o valor do número decimal representado neste código?

- Calcule o código Gray para o número 23 $_{base10}$

Exercícios

- Quantos números diferentes podem ser representados em uma palavra binária de 6 bits?

$$2^n = 2^6 = 64$$

- Qual é a representação BCD para 1049 $_{base10}$?

0001 0000 0100 1001

- O seguinte conjunto de bits 110010100011 representa um número usando codificação excesso de 3. Qual é o valor do número decimal representado neste código?

1100 1010 0011 = 970

- Calcule o código Gray para o número 23 $_{base10}$

Exercícios

- Quantos números diferentes podem ser representados em uma palavra binária de 6 bits?

$$2^n = 2^6 = 64$$

- Qual é a representação BCD para 1049 $_{base10}$?

0001 0000 0100 1001

- O seguinte conjunto de bits 110010100011 representa um número usando codificação excesso de 3. Qual é o valor do número decimal representado neste código?

1100 1010 0011 = 970

- Calcule o código Gray para o número 23 $_{base10}$

23 → 10111

11100

Códigos de Detecção de Erros

Códigos de detecção de erros

- Quando a informação é transmitida não é incomum a ocorrência de erros (ruídos, mau funcionamento de um componente)
- Se a codificação usa todas as combinações
  - Ou seja, se todas as combinações são válidas
  - **Não pode detectar erros**, nem corrigi-los
- Para detectar erros é preciso **redundância**
  - **Bits extras**
  - Combinações inválidas

Códigos de detecção de erros

- Técnicas de detecção de erros
  - Códigos m-de-n
  - Códigos de Paridade
  - Códigos de Hamming

CÓDIGO M DE N

Códigos de 7 bits (Código m de n)

- Dois grupos de bits:
  - um grupo com **2 bits** e um outro com **5 bits**
- Somente 2 bits em "um", os demais em "zero"
  - Sendo 1 desses bits está na esquerda (grupo de 2) e outro na direita (grupo dos 5)
  - Ex: **01 00001**
- Códigos diferentes disso significam um erro

Códigos de 7 bits (Código m de n)

- O grupo à esquerda indica se o número é

  menor ou igual a 4

ou

  maior ou igual a 5

Dec.	50	43210
0	01	00001
1	01	00010
2	01	00100
3	01	01000
4	01	10000
5	10	00001
6	10	00010
7	10	00100
8	10	01000
9	10	10000

pesos

Tabela com decimal e código correspondente com 7 bits

Códigos de 7 bits (Código m de n)

10 00111 : é válido?

01 00100 : é válido?

00 00110 : é válido?

Códigos de 7 bits (Código m de n)

10 00111 : é válido? NÃO

01 00100 : é válido? SIM

00 00110 : é válido? NÃO

CÓDIGOS DE PARIDADE

Códigos de Paridade

- Detecção de erros baseado na paridade
- Adição de 1 bit à palavra codificada para representar a paridade (0-par, 1-impar)
  - Indicando se o número de "uns" é par ou ímpar

- Se o código for de paridade par, todas as palavras com número ímpar de "uns" são rejeitadas

- Detecta erro simples (inversão de 1 bit)
- Não consegue detectar dois ou mais erros
- Nem consegue corrigir o bit errôneo

Códigos de Paridade

Ex: 8 bits (1 bit de paridade 0: par; 1: impar)

- 0 0000011: código válido
- 1 0000011: código inválido

No exemplo, o código inválido tem 2 "uns" → Número par de "uns", mas bit de paridade indica que código tem número impar de uns!

Códigos de Paridade

Ex: 8 bits (1 bit de paridade 0: par; 1: impar)

- Se for enviado o código “10000001”
- Mas for recebido o código “10001001”
  - Detecta?

ERRO DETECTADO!

Códigos de Paridade

Ex: 8 bits (1 bit de paridade 0: par; 1: impar)

- Se for enviado o código "10000001"
- E se for recebido o código "10001000"
  - Detecta?

ERRO NÃO DETECTADO!

CÓDIGO DE HAMMING

Código de Hamming

- Introduzem vários bits de paridade
- Permitem tanto detecção quanto correção

- **Distância de Hamming:** o número de bits que são alterados entre dois códigos adjacentes

Exemplos:

0001 e 0100 – distância é 2

0000 e 1111 – distância é 4

Código de Hamming

- Se a distância for de 1:
  - não é possível nem a detecção e nem a correção de erros, pois cada erro é transformado em um código também válido.
    - Exemplo: 000, 100, 101, 001, 011, 111, 110 e 010

- Se a distância for 2, como por exemplo 000, 011, 110 e 101:
  - é possível detectar um erro, mas não corrigi-lo
    - Exemplo: 001 poderia ter sido originado de 000, de 011 ou de 101

Código de Hamming

- Se a distância for de 3, como por exemplo em 011 e 100:
  - **erro simples** (erro de apenas um bit) pode ser **detectado e corrigido**!
    - Exemplo: 000 só poderia ter sido originado de 100
  - a correção é feita alterando a palavra para o código válido mais próximo (de menor distância).

Código de Hamming

Usando 3 bits de paridade

- Bit A: paridade das posições 1, 3, 5 e 7
- Bit B: paridade das posições 2, 3, 6 e 7
- Bit C: paridade das posições 4, 5, 6 e 7

Hamming

Tabela com código Hamming com distância mínima de 3

Posição	1	2	3	4	5	6	7
Código	A	B	8	C	4	2	1
0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	1
2	0	1	0	1	0	1	0
3	1	0	0	0	0	1	1
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	1
6	1	1	0	0	1	1	0
7	0	0	0	1	1	1	1
8	1	1	1	0	0	0	0
9	0	0	1	1	0	0	1
10	1	0	1	1	0	1	0
11	0	1	1	0	0	1	1
12	0	1	1	1	1	0	0
13	1	0	1	0	1	0	1
14	0	0	1	0	1	1	0
15	1	1	1	1	1	1	1

Hamming

Tabela com código Hamming com distância mínima de 3

Bits de Paridade: posições 1,2 e 4 (da esquerda para direita)

Posição	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Código	A	B	8	C	4	2	1
0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	1
2	0	1	0	1	0	1	0
3	1	0	0	0	0	1	1
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	1
6	1	1	0	0	1	1	0
7	0	0	0	1	1	1	1
8	1	1	1	0	0	0	0
9	0	0	1	1	0	0	1
10	1	0	1	1	0	1	0
11	0	1	1	0	0	1	1
12	0	1	1	1	1	0	0
13	1	0	1	0	1	0	1
14	0	0	1	0	1	1	0
15	1	1	1	1	1	1	1

Hamming

Tabela com código Hamming com distância mínima de 3

Bits de Paridade: posições 1,2 e 4 (da esquerda para direita)

Pesos: posições 3, 5, 6 e 7

Posição	1	2	3	4	5	6	7
Código	A	B	8	C	4	2	1
0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	1
2	0	1	0	1	0	1	0
3	1	0	0	0	0	1	1
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	1
6	1	1	0	0	1	1	0
7	0	0	0	1	1	1	1
8	1	1	1	0	0	0	0
9	0	0	1	1	0	0	1
10	1	0	1	1	0	1	0
11	0	1	1	0	0	1	1
12	0	1	1	1	1	0	0
13	1	0	1	0	1	0	1
14	0	0	1	0	1	1	0
15	1	1	1	1	1	1	1

Hamming

Tabela com código Hamming com distância mínima de 3

Bits de Paridade: posições 1,2 e 4 (da esquerda para direita)

Pesos: posições 3, 5, 6 e 7

Posição	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Código	A	B	8	C	4	2	1
0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	1
2	0	1	0	1	0	1	0
3	1	0	0	0	0	1	1
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	1
6	1	1	0	0	1	1	0
7	0	0	0	1	1	1	1
8	1	1	1	0	0	0	0
9	0	0	1	1	0	0	1
10	1	0	1	1	0	1	0
11	0	1	1	0	0	1	1
12	0	1	1	1	1	0	0
13	1	0	1	0	1	0	1
14	0	0	1	0	1	1	0
15	1	1	1	1	1	1	1

Hamming

Tabela com código Hamming com distância mínima de 3

Ex: Do 8 para o 9 : distância de 4 bits

Representando o 8:
1*8+4*0+2*0+0*1=8

Posição	1	2	3	4	5	6	7
Código	A	B	8	C	4	2	1
0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	1
2	0	1	0	1	0	1	0
3	1	0	0	0	0	1	1
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	1
6	1	1	0	0	1	1	0
7	0	0	0	1	1	1	1
8	1	1	1	0	0	0	0
9	0	0	1	1	0	0	1
10	1	0	1	1	0	1	0
11	0	1	1	0	0	1	1
12	0	1	1	1	1	0	0
13	1	0	1	0	1	0	1
14	0	0	1	0	1	1	0
15	1	1	1	1	1	1	1

Hamming: Detecção de Erros

3 bits de paridade, controlando posições diferentes (posições da esquerda para direita)

- Bit C (posição4): paridade das posições 4, 5, 6 e 7
- Bit B (posição 2): paridade das posições 2, 3, 6 e 7
- Bit A (posição 1): paridade das posições 1, 3, 5 e 7

- Avalia-se o valor dos bits de paridade e se todos forem 0, número recebido está correto, caso contrário há erro!

$$1\ 1\ 0\ 0\ 1\ 1\ 0 = 6_{10}$$

Hamming: Detecção de Erros

Exemplo 1100110 (6_{base10}):

- 1 1 0 0 $\boxed{1}$ 1 0 <u>recebido errado</u> (1 1 0 0 $\boxed{0}$ 1 0)
  - Verificando posições 4, 5, 6 e 7: paridade ímpar: C = 1
  - Verificando posições 2, 3, 6 e 7: paridade par: B = 0
  - Verificando posições 1, 3, 5 e 7: paridade ímpar: A = 1
  - CBA = 101 = 5_{10}

- Posição 5 está errada e bit deve ser invertido
- Se CBA = 000 → código recebido corretamente

Próximos slides: animação/ilustração da aplicação do método de detecção

Hamming: Detecção de Erros

Exemplo:

- 1 1 0 0 1 1 0 recebido errado (1 1 0 0 0 1 0)
  - Verificando posições 4, 5, 6 e 7: paridade ímpar: C = 1
  - Verificando posições 2, 3, 6 e 7: paridade par: B = 0
  - Verificando posições 1, 3, 5 e 7: paridade ímpar: A = 1
  - CBA = 101 = 5_{10}

- Posição 5 está errada e bit deve ser invertido
- Se CBA=000 -> código recebido corretamente

Hamming: Detecção de Erros

Exemplo:

- 1 1 0 0 1 1 0 <u>recebido errado</u> (1 1 0 0 0 1 0)
  - Verificando posições 4, 5, 6 e 7: paridade ímpar: C = 1
  - Verificando posições 2, 3, 6 e 7: paridade par: B = 0
  - Verificando posições 1, 3, 5 e 7: paridade ímpar: A = 1
  - CBA = 101 = 5_{10}

- Posição 5 está errada e bit deve ser invertido
- Se CBA=000 -> código recebido corretamente

Hamming: Detecção de Erros

Exemplo:

- 1 1 0 0 1 1 0 recebido errado ($\boxed{1}$ 1 $\boxed{0}$ 0 $\boxed{0}$ 1 $\boxed{0}$)
  - Verificando posições 4, 5, 6 e 7: paridade ímpar: C = 1
  - Verificando posições 2, 3, 6 e 7: paridade par: B = 0
  - Verificando posições 1, 3, 5 e 7: paridade ímpar: A = 1
  - CBA = 101 = 5_{10}

- Posição 5 está errada e bit deve ser invertido
- Se CBA=000 -> código recebido corretamente

Hamming: Detecção de Erros

Exemplo:

- 1 1 0 0 1 1 0 recebido errado (1 1 0 0 0 1 0)
  - Verificando posições 4, 5, 6 e 7: paridade ímpar: C = 1
  - Verificando posições 2, 3, 6 e 7: paridade par: B = 0
  - Verificando posições 1, 3, 5 e 7: paridade ímpar: A = 1
  - CBA = 101 = 5_{10}
- Posição 5 está errada e bit deve ser invertido
- Se CBA=000 -> código recebido corretamente

Hamming: Detecção de Erros

Exemplo:

- 1 1 0 0 1 1 0 <u>recebido certo</u> (1 1 0 0 1 1 0)
  - Verificando posições 4, 5, 6 e 7: paridade par: C = 0
  - Verificando posições 2, 3, 6 e 7: paridade par: B = 0
  - Verificando posições 1, 3, 5 e 7: paridade par: A = 0

  - CBA = 000 → código recebido corretamente

Próximos slides: animação/ilustração da aplicação do método de detecção

Hamming: Detecção de Erros

Exemplo:

- 1 1 0 0 1 1 0 recebido certo   (1 1 0 0 1 1 0)

  - Verificando posições 4, 5, 6 e 7: paridade par: C = 0
  - Verificando posições 2, 3, 6 e 7: paridade par: B = 0
  - Verificando posições 1, 3, 5 e 7: paridade par: A = 0

  - CBA = 000 → código recebido corretamente

Hamming: Detecção de Erros

Exemplo:

- 1 1 0 0 1 1 0 <u>recebido certo</u> (1 $\boxed{1\ 0}$ 0 1 $\boxed{1\ 0}$)
  - Verificando posições 4, 5, 6 e 7: paridade par: C = 0
  - Verificando posições 2, 3, 6 e 7: paridade par: B = 0
  - Verificando posições 1, 3, 5 e 7: paridade par: A = 0

  - CBA = 000 → código recebido corretamente

Hamming: Detecção de Erros

Exemplo:

- 1 1 0 0 1 1 0 recebido certo (1 1 0 0 1 1 0)

  - Verificando posições 4, 5, 6 e 7: paridade par: C = 0
  - Verificando posições 2, 3, 6 e 7: paridade par: B = 0
  - Verificando posições 1, 3, 5 e 7: paridade par: A = 0

  - CBA = 000 → código recebido corretamente

Hamming: Detecção de Erros

Exemplo:

- 1 1 0 0 1 1 0 recebido certo (1 1 0 0 1 1 0)

  - Verificando posições 4, 5, 6 e 7: paridade par: C = 0
  - Verificando posições 2, 3, 6 e 7: paridade par: B = 0
  - Verificando posições 1, 3, 5 e 7: paridade par: A = 0

  - CBA = 000 -> código recebido corretamente

Códigos Alfanuméricos

Códigos Alfanuméricos

- Os computadores não manipulam apenas informações numéricas - precisam representar também caracteres alfabéticos, pontuação, etc.

- ASCII
- EBCDIC

$\left.\begin{matrix} \\ \\ \end{matrix}\right\}$ Amplamente usado

- Unicode
  - UTF-8 (8-bit *Unicode Transformation Format)*

Código ASCII

- ASCII - *American Standard Code for Information Interchange*
- Código mais usado em todas as plataformas
- 7 bits
- 8 bits (versão estendida – ASCII extended)

Código ASCII 7 bits

- 7 bits (128 combinações)
- Códigos de 0 a 31 reservados para caracteres de controle (tabulação, retorno, ejeção de página, etc)
- Caracteres visíveis vão do 32 (espaço) ao 126 (til)
- Diferença de 32 entre letras maiúsculas e minúsculas
  - A: 100 0001 (65_{10})
  - a: 110 0001 (97_{10})
  - 97-65=32

Tabela ASCII – 7 bits

Table 1–2
ASCII Code

Character	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Character	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Character	A_6	A_5	A_4	A_3	A_2	A_1	A_0
space	0	1	0	0	0	0	0	@	1	0	0	0	0	0	0	`	1	1	0	0	0	0	0
!	0	1	0	0	0	0	1	A	1	0	0	0	0	0	1	a	1	1	0	0	0	0	1
"	0	1	0	0	0	1	0	B	1	0	0	0	0	1	0	b	1	1	0	0	0	1	0
#	0	1	0	0	0	1	1	C	1	0	0	0	0	1	1	c	1	1	0	0	0	1	1
$	0	1	0	0	1	0	0	D	1	0	0	0	1	0	0	d	1	1	0	0	1	0	0
%	0	1	0	0	1	0	1	E	1	0	0	0	1	0	1	e	1	1	0	0	1	0	1
&	0	1	0	0	1	1	0	F	1	0	0	0	1	1	0	f	1	1	0	0	1	1	0
'	0	1	0	0	1	1	1	G	1	0	0	0	1	1	1	g	1	1	0	0	1	1	1
(	0	1	0	1	0	0	0	H	1	0	0	1	0	0	0	h	1	1	0	1	0	0	0
)	0	1	0	1	0	0	1	I	1	0	0	1	0	0	1	i	1	1	0	1	0	0	1
*	0	1	0	1	0	1	0	J	1	0	0	1	0	1	0	j	1	1	0	1	0	1	0
+	0	1	0	1	0	1	1	K	1	0	0	1	0	1	1	k	1	1	0	1	0	1	1
,	0	1	0	1	1	0	0	L	1	0	0	1	1	0	0	l	1	1	0	1	1	0	0
–	0	1	0	1	1	0	1	M	1	0	0	1	1	0	1	m	1	1	0	1	1	0	1
.	0	1	0	1	1	1	0	N	1	0	0	1	1	1	0	n	1	1	0	1	1	1	0
/	0	1	0	1	1	1	1	O	1	0	0	1	1	1	1	o	1	1	0	1	1	1	1
0	0	1	1	0	0	0	0	P	1	0	1	0	0	0	0	p	1	1	1	0	0	0	0
1	0	1	1	0	0	0	1	Q	1	0	1	0	0	0	1	q	1	1	1	0	0	0	1
2	0	1	1	0	0	1	0	R	1	0	1	0	0	1	0	r	1	1	1	0	0	1	0
3	0	1	1	0	0	1	1	S	1	0	1	0	0	1	1	s	1	1	1	0	0	1	1
4	0	1	1	0	1	0	0	T	1	0	1	0	1	0	0	t	1	1	1	0	1	0	0
5	0	1	1	0	1	0	1	U	1	0	1	0	1	0	1	u	1	1	1	0	1	0	1
6	0	1	1	0	1	1	0	V	1	0	1	0	1	1	0	v	1	1	1	0	1	1	0
7	0	1	1	0	1	1	1	W	1	0	1	0	1	1	1	w	1	1	1	0	1	1	1
8	0	1	1	1	0	0	0	X	1	0	1	1	0	0	0	x	1	1	1	1	0	0	0
9	0	1	1	1	0	0	1	Y	1	0	1	1	0	0	1	y	1	1	1	1	0	0	1
:	0	1	1	1	0	1	0	Z	1	0	1	1	0	1	0	z	1	1	1	1	0	1	0
;	0	1	1	1	0	1	1	[	1	0	1	1	0	1	1	{	1	1	1	1	0	1	1
<	0	1	1	1	1	0	0	\	1	0	1	1	1	0	0	|	1	1	1	1	1	0	0
=	0	1	1	1	1	0	1	]	1	0	1	1	1	0	1	}	1	1	1	1	1	0	1
>	0	1	1	1	1	1	0	^	1	0	1	1	1	1	0	~	1	1	1	1	1	1	0
?	0	1	1	1	1	1	1	_	1	0	1	1	1	1	1	delete	1	1	1	1	1	1	1

Tabela ASCII 7 bits

Bits inferiores	Bits superiores (mais significativos)							
	000	001	010	011	100	101	110	111
0000	null	dle		0	@	P	`	p
0001	soh	dc1	!	1	A	Q	a	q
0010	stx	dc2	"	2	B	R	b	r
0011	etx	dc3	#	3	C	S	c	s
0100	eot	dc4	$	4	D	T	d	t
0101	enq	nak	%	5	E	U	e	u
0110	ack	syn	&	6	F	V	f	v
0111	bell	etb	'	7	G	W	g	w
1000	bsp	can	(	8	H	X	h	x
1001	ht	em	)	9	I	Y	i	y
1010	lf	sub	*	:	J	Z	j	z
1011	vt	esc	+	;	K	[	k	{
1100	ff	fs	,	<	L	\	l	|
1101	cr	gs	-	=	M	]	m	}
1110	so	rs	.	>	N	^	n	~
1111	si	us	/	?	O	_	o	del

Tabela 8.12 - Código ASCII de 7 bits

ASCII de 8 bits (extended)

- 8 bits (256 combinações)
  - Permite representar caracteres acentuados (â,ã,á,à...) e símbolos específico de diversas línguas
  - Não existe definição única, cada fabricante definiu a sua

- Atualmente 8 bits já é considerado insuficiente, portanto existem propostas de uso de 16 bits – 65532 símbolos distintos (Unicode)

EBCDIC

- EBCDIC (*Extended Binary Coded Decimal Interchange Code*)

- Usado em plataformas de grande porte da IBM

- Padrão 8 bits
  - possibilidade de codificar 256 estados diferentes.

Unicode

- Unicode é um padrão que permite aos computadores representar e manipular, de forma consistente, texto de qualquer sistema de escrita existente.

- UTF-8 (8 bits)
  - Compatível com o ASCII
  - Utilizada para os sistemas latinos

- UTF-16 (16 bits)
  - Representação de até 65536 símbolos

Onde aprender mais ?

[1] WEBER, Raul F. **Fundamentos de Arquiteturas de Computadores**. Porto Alegre: Sagra-Luzzato, 2000.

[2] MONTEIRO, M. A. **Introdução à Organização de Computadores**. Rio de Janeiro: Livros Técnicos e Científicos, 1996.

[3] UYEMURA. **Sistemas Digitais**. São Paulo: Pioneira Thomson Learning, 2002.