

# Curso: Engenharia de Computação

Arquitetura de Computadores

Prof. Clayton J A Silva, MSc  
clayton.silva@professores.ibmec.edu.br





The background of the slide is a dense, overlapping field of 3D-rendered numbers. The numbers are in two colors: white and orange. They are arranged in a way that creates a sense of depth and movement, with some numbers appearing to be in the foreground and others receding into the background. The lighting is soft, creating subtle shadows and highlights on the surfaces of the numbers.

# Representação numérica

# Representação de grandezas numéricas

- Notação **posicional**
- Notação **polinomial**



# Representação de grandezas numéricas

- Notações posicional e polinomial

+significativo

$$1457 = 1 \times 10^3 + 4 \times 10^2 + 5 \times 10^1 + 7 \times 10^0$$

- Dígitos do conjunto  $D = \{0, 1, 2, 3, \dots, 9\}$ , base decimal

# Notação posicional

$$N_b = d_{m-1} d_{m-2} \dots d_0$$

- $N$  é a grandeza,
- $b$  é a base,
- $m$  é o número de dígitos usados na representação
- os índices  $0$  a  $m-1$  representam a posição do dígito.

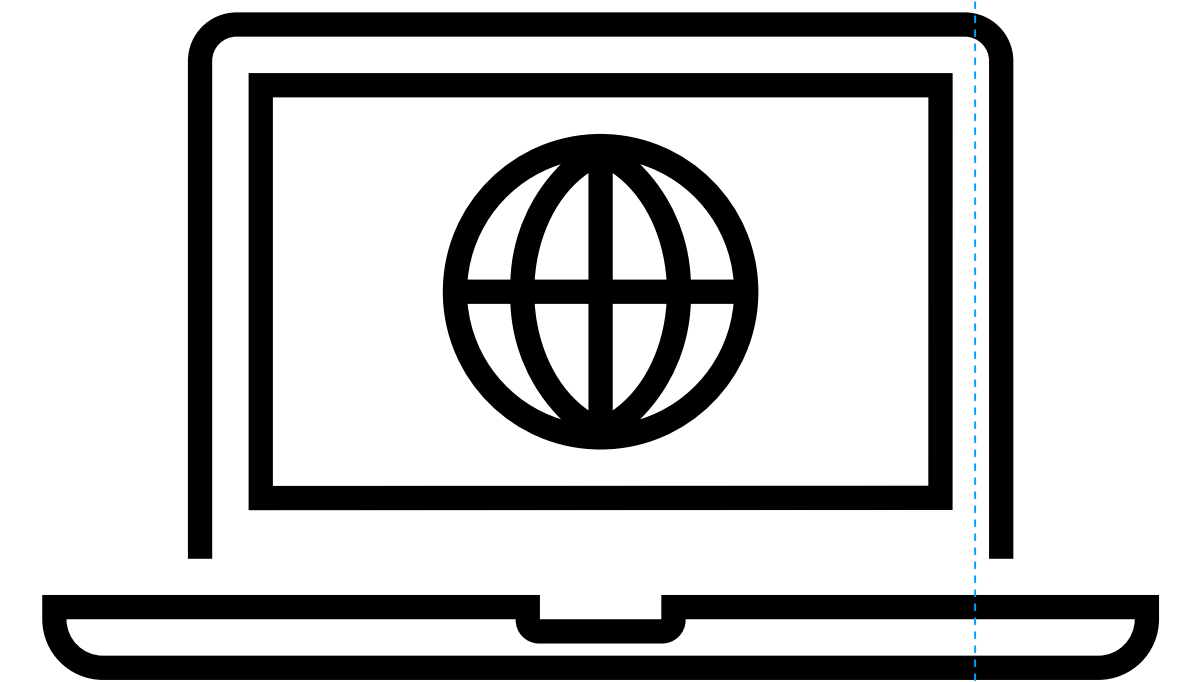
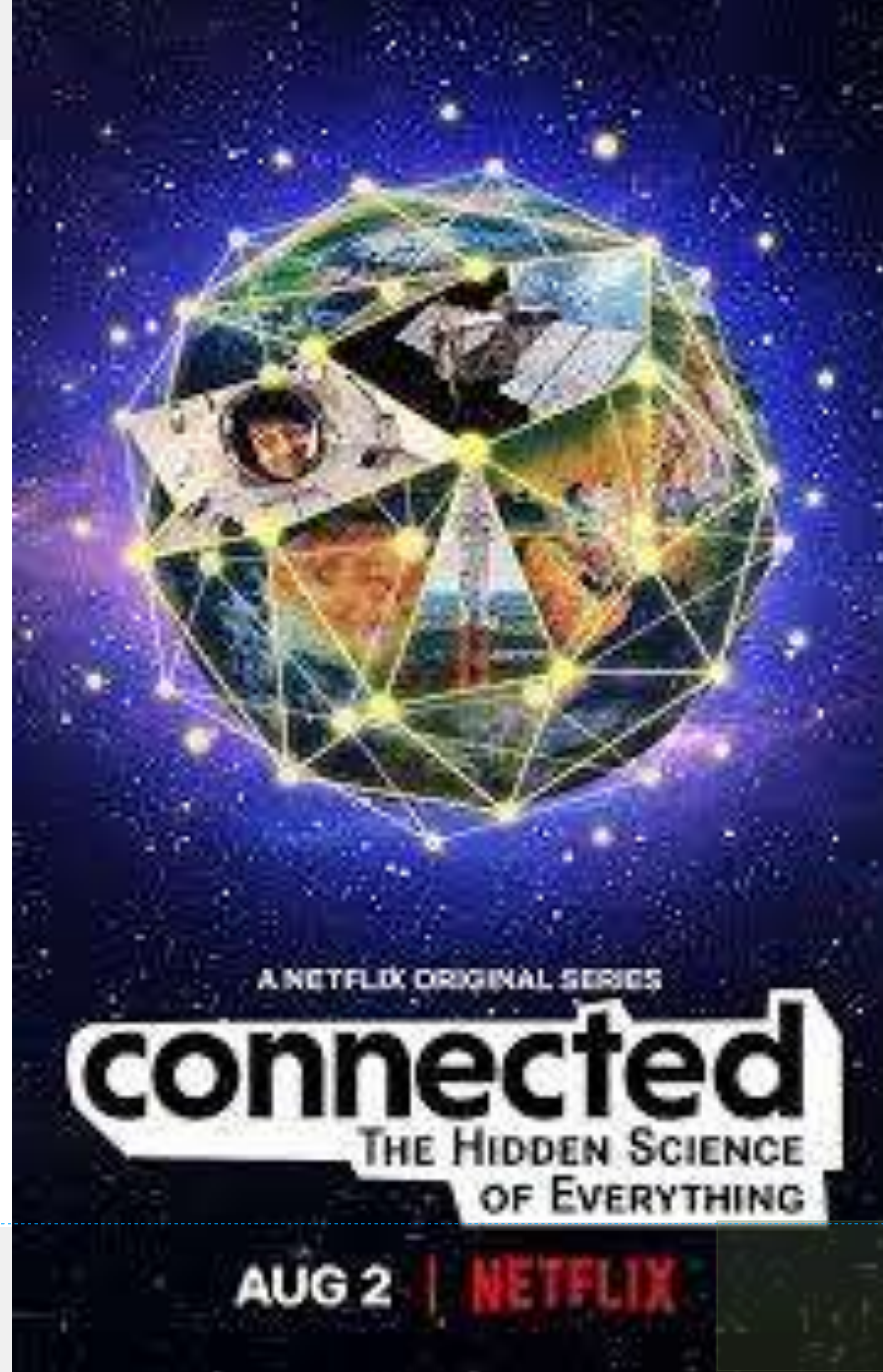
# Notação polinomial

$$N_b = d_{m-1} \times b^{m-1} + d_{m-2} \times b^{m-2} \dots + d_0 \times b^0$$

- $N$  é a grandeza,
- $b$  é a base,
- $m$  é o número de dígitos usados e
- os índices  $0$  a  $m-1$  representam a posição do dígito.



Representação  
em binário puro







# Sistema binário



# Representação binária

- bit (*binary digit*): 0 / 1
- 1 *byte* (*B*) = 8 bits

- Múltiplos:

<i>Kilo (k)</i>	$2^{10}$
<i>Mega (M)</i>	$2^{20}$
<i>Giga (G)</i>	$2^{30}$
<i>Tera (T)</i>	$2^{40}$



bases  
numéricas:  
binária  
decimal  
hexadecimal

Base 2	Base 10	Base 16
0	0	0
1	1	1
10	2	2
11	3	3
100	4	4
101	5	5
110	6	6
111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F
10000	16	10
10001	17	11

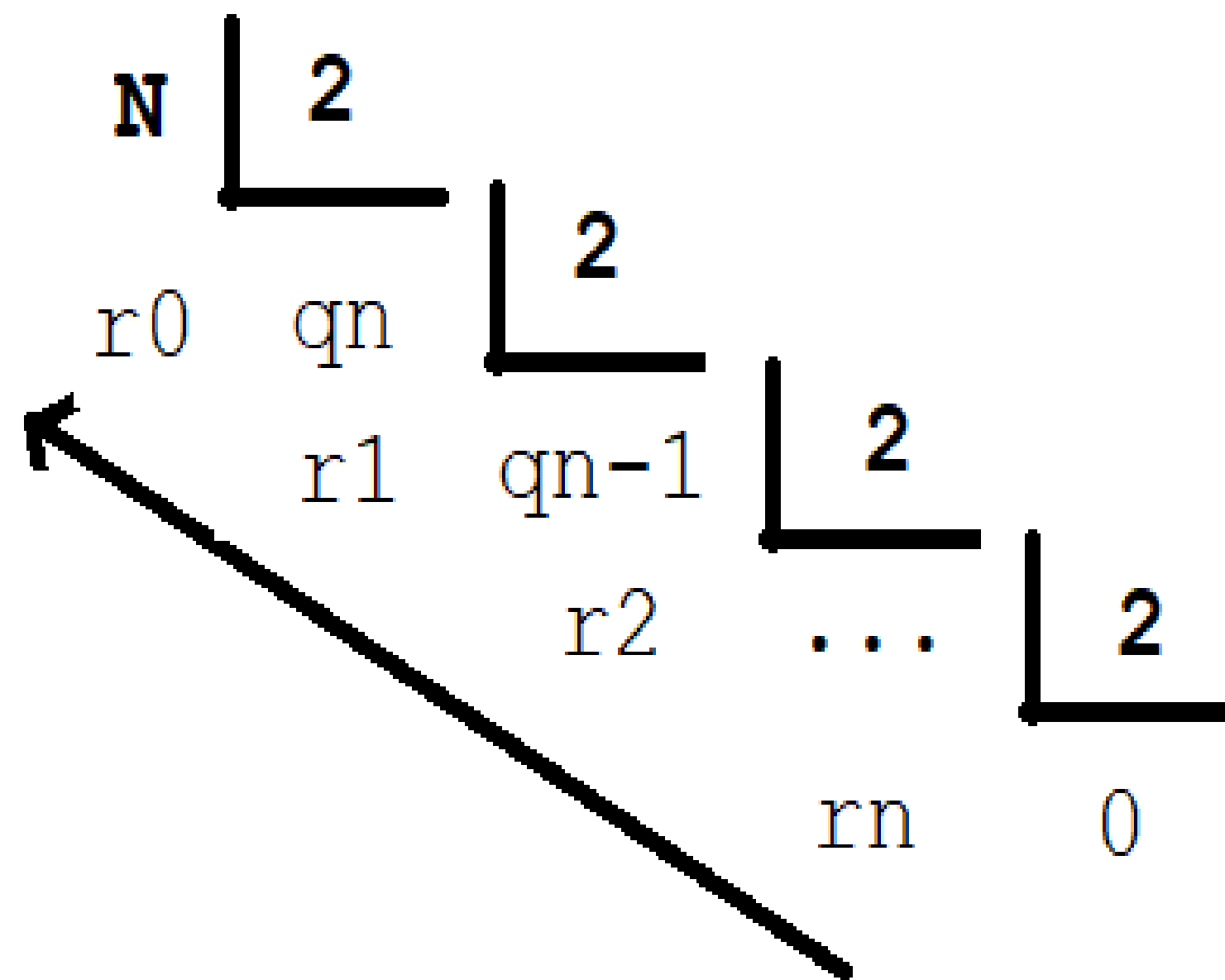


# Conversão de bases



# Conversão de bases

- Base 10 para base 2: divisões sucessivas



$$N = (r_n r_{n-1} r_{n-2} \dots r_0)_2$$



# Conversão de bases

- Base 2 para base 10: notação polinomial

$$b_n \times 2^n + b_{n-1} \times 2^{n-1} \dots + b_0 \times 2^0$$



# Conversão de bases

- Base 2 para base 16 e vice-versa

$b_n$	...	$b_6$	$b_5$	$b_4$	$b_3$	$b_2$	$b_1$	$b_0$	base 2
					$dH$				base 16

Obs. Representação das grandezas hexadecimais na forma  $Nh$ ,  $NH$  ou  $0xN$ , por exemplo endereço  $0x10A$  ou  $10Ah$  ou  $10AH$

# Aritmética binária

- Adição binária
- Subtração binária



# Aritmética binária: adição

- Sejam dois dígitos binários (bits)  $a$  e  $b$ ,

$a+b =$		$b$		
		0	1	
$a$	0	0	1	
	1	1	10	<i>carry ou vai um</i>

# Aritmética binária: adição

- Adição binária de dois números binários  $A$  e  $B$  de  $m$  bits, ou seja,  $A = a_{m-1} \dots a_1 a_0$  e  $B = b_{m-1} \dots b_1 b_0$ ,  $A+B$  é obtida por
  1. Realizar a operação bit a bit do menos ao mais significativo (da direita para a esquerda)
  2. Aplica-se a tabela anterior
  3. Se houver bit 1 de *carry* transporta-se para a soma dos bits seguintes mais significativos (à esquerda)
  4. Repete-se o processo até alcançar o bit mais significativo.



# Aritmética binária: subtração

- Sejam dois dígitos binários (bits)  $a$  e  $b$ ,

$a-b =$		$b$		
		0	1	
$a$	0	0	11	<i>carry ou vai menos um</i>
	1	1	0	

# Aritmética binária: subtração

- Subtração binária de dois números binários  $A$  e  $B$  de  $m$  bits, ou seja,  $A = a_{m-1} \dots a_1 a_0$  e  $B = b_{m-1} \dots b_1 b_0$ , onde  $A$  é o minuendo e  $B$  é o subtraendo,  $A-B$  é obtida por
  1. Operação bit a bit, do menos ao mais significativo
  2. Aplica-se a tabela anterior
  3. Se houver bit -1 de *carry* transporta-se para a subtração dos dígitos seguintes (à esquerda – mais significativos),
  4. Repete-se o processo até alcançar o bit mais significativo.
  5. Se o minuendo for menor do que o subtraendo inverter a operação e representar o **número negativo**



# Adição hexadecimal

1. Processo similar à aritmética binária. A adição se implementa dígito por dígito, do menos ao mais significativo.
  2. Para cada par de dígitos hexa, se a soma  $S$  for menor ou igual a 15 decimal ( $F$ , em hexadecimal), o resultado é o dígito hexadecimal equivalente...
  3. ...se a soma  $S$  for maior do que 15, transporta-se 1 (*carry*) ao dígito seguinte (mais significativo)...
  4. ...o resultado é o dígito equivalente a  $S-16$ ...
  5. ...com o dígito seguinte somado ao *carry*.
- P. ex.  $0x8 + 0xA = 0x12$



# Subtração hexadecimal

1. Para cada par de dígitos hexa, do menos ao mais significativo, se o minuendo  $M$  for superior ou igual ao subtraendo  $S$  realiza-se a operação normalmente...
2. ...se  $M$  for inferior a  $S$ , toma-se 1 (*borrow*) do dígito seguinte (mais significativo), soma-se 16 a  $M$  e realiza-se a subtração...
3. ...decrementa-se 1 do  $M$  do dígito seguinte para prosseguir com a operação.
4. P. ex.  $0x18 + 0x0A = 0x0E$



The background of the slide is a dense, overlapping field of 3D-rendered numbers. The numbers are in various colors, including white, orange, and grey, and are arranged in a way that creates a sense of depth and movement. Some numbers are in the foreground, while others are blurred in the background.

# Representação de números negativos



# Tipos de notação

Seja um número binário negativo com  $m$  bits, ele pode ser representado das seguintes notações:

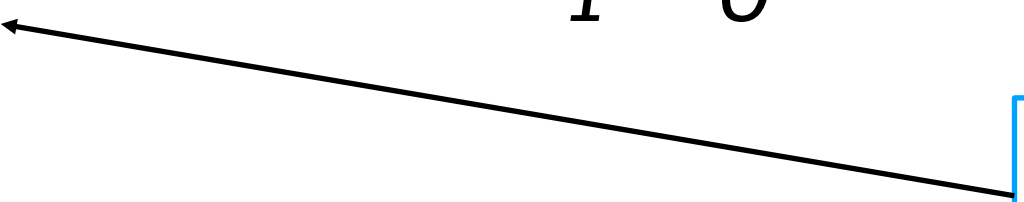
1. Representação em **bit sinal** (sinal e magnitude)
2. Representação em **complemento de 1**
3. Representação em **complemento de 2**
4. Representação em **excesso  $2^{m-1}$**

# Representação em sinal e magnitude (bit sinal)

Seja um número negativo  $B = b_{m-1} \dots b_1 b_0$

Usando a notação em bit sinal

1. Na posição mais significativa, utiliza-se o **bit sinal** para os números positivos e negativos
2. Os demais bits representam o valor representado em binário puro
3.  $B = \text{bit sinal} \dots b_1 b_0$



$b_{m-1}$



# Características

- Dos  $m$  bits, 1 bit fica reservado ao sinal
- Faixa de representação:  $-(2^{m-1}-1)$  a  $+(2^{m-1}-1)$
- Dupla representação do 0 – todos os bits iguais a 0, com bit sinal 0 ou 1

Exemplo: representação em bit sinal de palavras com 3 bits

+3	0	1	1			
+2	0	1	0			
+1	0	0	1			
0	0	0	0	1	0	0
-1	1	0	1			
-2	1	1	0			
-3	1	1	1			

# Adição e subtração em bit sinal

1. Para uma adição/subtração de número com  $m$  bits, ...
2. aplica-se a adição/subtração da representação binária, ...
3. comparando-se os números...
4. e observando o sinal.



# Complemento de números

1. O complemento de um número de  $n$  dígitos é a diferença entre o maior número de  $n$  dígitos naquela base e o número considerado.
  - Por exemplo, na base 10, o complemento de 12 é 87, pois 99 (maior número com 2 dígitos) menos 12 é igual a 87.
2. Na base 2, para obter o complemento basta inverter os bits do número binário
  - Por exemplo  $C(1011) = 1111 - 1011 = 0100$  (inversão dos bits de 1011)
3. Na base 2, o complemento é chamado **complemento de 1**

# Complemento de números

4. Na base 2, utiliza-se a soma do complemento de 1 com o bit 1, chamado de complemento de 2
- Por exemplo  $C_1(1011) = 0100$ , logo o  $C_2(1011) = 0100 + 1 = 0101$



# Representação em complemento de 1

Seja um número negativo  $B = b_{m-1} \dots b_1 b_0$

Usando a notação complemento de 1

1. Na posição mais significativa, utiliza-se o **bit sinal** para os números **positivos**
2. Os números da **faixa dos positivos** são representados pelo bit sinal seguido do binário puro
3. Os números da **faixa dos negativos** são representados pelo bit sinal seguido do complemento de 1 dos respectivos simétricos positivos

# Características

- Dos  $m$  bits, 1 fica reservado ao bit sinal
- Faixa de representação:  $-(2^{m-1}-1)$  a  $+(2^{m-1}-1)$
- Dupla representação do 0 – todos os bits iguais a 0 ou iguais a 1

Exemplo: representação em complemento de 1 de palavras com 3 bits

+3	0	1	1			
+2	0	1	0			
+1	0	0	1			
0	0	0	0	1	1	1
-1	1	1	0			
-2	1	0	1			
-3	1	0	0			



# Adição binária em complemento de 1

1. Para uma soma de número com  $m$  bits, ...
2. pode-se generalizar a operação do menos ao mais significativo – direita para a esquerda, ...
3. ‘carregando’ o bit 1 de *carry* para a soma dos dígitos seguintes...
4. até alcançar o bit mais significativo.
5. Se houver *carry* 1 no bit mais significativo, transporta-se o bit 1 e o soma ao número do resultado obtido

# Subtração binária em complemento de 1

- Soma-se o minuendo com a representação negativa do subtraendo do mesmo modo anteriormente apresentado
- Exemplo: complemento de 1 de palavras de 4 bits, seja  $A - B$ ,  $A = 0110$ ,  $B = 0011$ ,  $A - B = A + (-B) = 0110 + 1100 = 1\text{0010} = 0010 + 1 = 0011$



# Representação em complemento de 2

Seja um número negativo  $B = b_{m-1} \dots b_1 b_0$

Usando a notação complemento de 2

1. Utiliza-se o bit sinal para os números positivos
2. Os números da faixa dos positivos são representados pelo bit sinal seguido do binário puro
3. Os números da faixa dos negativos são representados pelo bit sinal seguido do complemento de 2 dos respectivos simétricos positivos
4. A faixa dos negativos possui uma palavra *bit sinal.0...0*, que não possui simétrico positivo

# Características

- Dos  $m$  bits, 1 fica reservado ao bit sinal
- Faixa de representação:  
 $-(2^{m-1}) a +(2^{m-1}-1)$
- Não possui dupla representação do 0 – todos os bits iguais a 0 ou iguais a 1

Exemplo: representação em complemento de 2 de números de 3 bits

+3	0	1	1
+2	0	1	0
+1	0	0	1
0	0	0	0
-1	1	1	1
-2	1	1	0
-3	1	0	1
-4	1	0	0



# Adição binária em complemento de 2

1. Para uma soma de número com  $m$  bits,...
2. pode-se generalizar a operação do menos ao mais significativo...
3. 'carregando' o bit 1 de *carry* para a soma dos dígitos seguintes...
4. até alcançar o bit mais significativo.
5. Se houver *carry* 1 no bit mais à esquerda, despreza-se o *carry*

# Subtração binária em complemento de 2

- Similar à soma na notação em complemento de 1...soma-se o minuendo com a representação negativa do subtraendo
- Exemplo: complemento de 2 de palavras de 4 bits, seja  $A - B$ ,  $A = 0110$ ,  $B = 0011$ ,  $A - B = A + (-B) = 0110 + 1101 = 1\text{0011} = 0011$



# Representação em excesso $2^{m-1}$ para números de $m$ bits

Seja um número negativo  $B = b_{m-1} \dots b_1 b_0$

Usando a notação em excesso  $2^{m-1}$

1. O número mais negativo é o decimal  $-2^{m-1}$ , representado por todos  $m$  bits iguais a 0
2. Incrementa-se 1 a cada número seguinte até o número mais positivo
3. Corresponde a soma de  $2^{m-1}$  a todas as grandezas a representar, desde as negativas. Por exemplo, +3 em representação excesso 8 ( $2^3$ , 4 bits) é o equivalente a 11, ou seja, 1011

# Características

- Faixa de representação:  $-(2^{m-1}) a +(2^{m-1}-1)$

Exemplo: representação em excesso 4

b2	b1	b0	
1	1	1	+3, em b10
1	1	0	+2, em b10
1	0	1	+1, em b10
1	0	0	0, em b10
0	1	1	-1, em b10
0	1	0	-2, em b10
0	0	1	-3, em b10
0	0	0	-4, em b10

# Adição binária em excesso $2^{m-1}$

1. Para uma soma de número com  $m$  bits,...
2. pode-se generalizar a operação do menos ao mais significativo...
3. ‘carregando’ o bit 1 de *carry* para a soma dos dígitos seguintes...
4. até alcançar o bit mais significativo.
- 5. Subtrair o resultado do excesso  $2^{m-1}$** 
  - Por exemplo 1:  $+3 + (-5)$ . Em excesso 8,  $A = 1011$  e  $B = 0011$ ,  $A + B = 1110 - 1000 = 0110$ . Ex 2:  $+4 + 2$ ,  $1100 + 1010 = 1110$ .



# Subtração binária em $2^m-1$

1. Subtrai-se normalmente, bit a bit, o subtraendo do minuendo, do menos ao mais significativo.
2. *Somar o resultado ao excesso  $2^{m-1}$* 
  - Por exemplo 1:  $+3 - (-2)$ . Em excesso 8,  $A = 1011$  e  $B = 0110$ ,  $A - B = 0101 + 1000 = 1101$ . Ex 2:  $+7 - 4$ ,  $1111 - 1100 = 1011$ .

Exemplo:  
Comparação  
palavras de 4 bits

base 10	bit sinal				compl 1				compl 2				excesso 8			
+7	0	1	1	1	0	1	1	1	0	1	1	1	1	1	1	1
+6	0	1	1	0	0	1	1	0	0	1	1	0	1	1	1	0
+5	0	1	0	1	0	1	0	1	0	1	0	1	1	1	0	1
+4	0	1	0	0	0	1	0	0	0	1	0	0	1	1	0	0
+3	0	0	1	1	0	0	1	1	0	0	1	1	1	0	1	1
+2	0	0	1	0	0	0	1	0	0	0	1	0	1	0	1	0
+1	0	0	0	1	0	0	0	1	0	0	0	1	1	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
-1	1	0	0	1	1	1	1	0	1	1	1	1	1	1	0	1
-2	1	0	1	0	1	1	0	1	1	1	1	0	1	0	0	1
-3	1	0	1	1	1	1	0	0	1	1	0	1	0	1	0	1
-4	1	1	0	0	1	0	1	1	1	1	0	0	0	0	0	1
-5	1	1	0	1	1	0	1	0	1	0	1	1	1	1	0	0
-6	1	1	1	0	1	0	0	1	1	0	1	0	1	0	0	0
-7	1	1	1	1	1	0	0	0	1	0	0	1	0	1	0	0
-8	não existe				não existe				1	0	0	0	0	0	0	0

# Observações

- As máquinas possuem **palavras** com tamanho definido de  $m$  bits
- Se a operação resultante ultrapassar a capacidade do sistema representar o número obtido...
- caracteriza-se ***overflow*** = '***estouro***'



# Representação em ponto flutuante

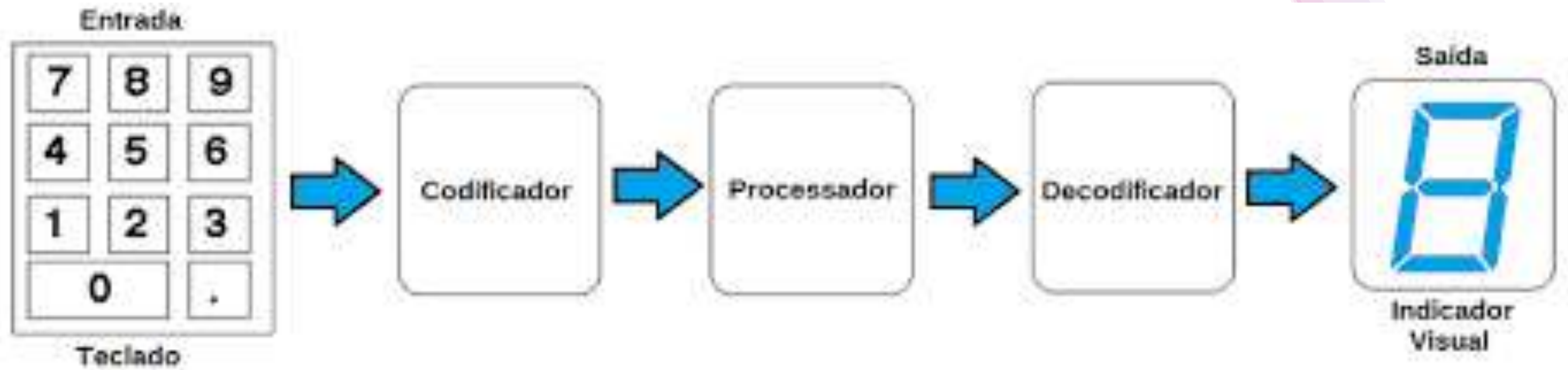
- Notação científica:  $N = f \times 10^e$

, onde  $f$  – fração ou mantissa;  $e$  – expoente

- Pela representação em ponto flutuante – equivalente computacional, quando se **convenciona o número de dígitos** para representar mantissa e expoente:
  - ☐ a faixa de representação é determinada pelo número de dígitos do expoente e
  - ☐ a precisão é determinada pelo número de dígitos da mantissa.

# Representação em ponto flutuante

- A versão de ponto flutuante nos sistemas computacionais requer a representação da mantissa e do expoente no sistema binário.



# Codificação binária



# Códigos de detecção e correção de erros

- bit de paridade
- Distância de Hamming – número de posições de bits em que duas palavras de um código são diferentes
- Em um código com distância de Hamming igual a  $d+1$  é possível detectar  $d$  erros de bits únicos
- Em um código com distância de Hamming igual a  $2d+1$  é possível corrigir  $d$  erros de bits únicos



IBMEC.BR

 /IBMEC

 IBMEC

 @IBMEC\_OFICIAL

 @IBMEC

 **ibmec**