# Representação de números inteiros em complemento a 2

# Representação complemento a 10

- Como representar números negativos no sistema decimal com 3 algarismos ?
- Divide o sistema em 2, utiliza uma metade para positivos e a outra para negativos

  |---------------|------------|-------------|-------|--|

  000 250 500 750 998 000
- Representação: 0 – 499: representam números positivos: 0 a +499
- Representação 500-999: representam números negativos que são o complemento a 10 do número positivo
- Complemento a base= $B^n$-N onde n é o número de algarismos que temos para representar um número
  - -500=$10^3$-500=1000-500=500
  - -499 = 1000 -499= 501
  - -1 = 1000 -1 = 999
- Qual a vantagem ? Só precisamos somar números e resultado é sempre representado corretamente
- Exemplo:
  - -2 +3, Complemento a 10 de -2 = 1000 - 2 =998 e de +3 = 003, somando 998+003=1001, descartando o 1 à esquerda, temos o resultado 001 que é +1 em complemento a 10.
- Para subtrair:
  - A - B = A + (-B)
  - -3 -5 = -3 + (-5)
  - -3 = 997
  - -5 = 995
  - 997+995 = 1992, retirando o 1 à esquerda temos o resultado 992 que é negativo e para sabermos o valor fazemos 1000 - x = 992, x = 8, logo resultado = -8.

# Como representar um número em complemento a 2 ?

- Se o número é positivo, o bit $d_{n-1}=0$ e a magnitude do número é
  - $d_{n-2} \times 2^{n-2} + d_{n-3} \times 2^{n-3} + ... + d_1 \times 2^1 + d_0 \times 2^0$
- Exemplo:
  - +7 utilizando-se 4 algarismos é igual a 0111
  - Representação de +27 com 8 bits = 00011011
- Se o número é negativo:
  - $C_2(-N)=2^n - N$, onde N é a representação do número positivo
  - $C_2(-N)=2^n - N=((2^{n-1}) - N)+1$
  - $2^{n-1}$ = 011111..11
  - 011111..11-N é igual a inverter os bits de N
  - Logo para achar a representação complemento a 2 de números negativos, inverte-se o número na representação dele positiva e soma-se 1
  - Exemplo:
    - -7 utilizando-se 4 algarismos 0111, inverte 1000, soma 1 = 1001
    - Representação de -27 com 8 bits = inv(00011011) + 1 = 11100100 + 1 = 11100101

# Dado um número representado em complemento a 2, o que ele está representando ?

- $N=(d_{n-1}\ d_{n-2}\ ...\ d_1 d_0)$
- O bit mais significativo indica o sinal
  - Se ele é zero o número é positivo dado por :
    - $\mathbf{N} = d_{n-2} \times 2^{n-2} + d_{n-3} \times 2^{n-3} + ... + d_1 \times 2^1 + d_0 \times 2^0$
    - Exemplo:
      - n= 4, N = 0111, $N=7_{10}$, N=0100, $N=4_{10}$
  - Se ele é 1, o número que ele representa é negativo. Como achar que número ele representa ?
  - $C_2(C_2(-N))=2^n - (2^n - N) = N$
  - Logo para descobrir a magnitude de um número negativo representado em complemento a 2, inverte-se o número na representação dele sem sinal e soma-se 1.
- Exemplos:
- O que representa 00011011 ? +27
- O que representa 11100101 ? - inv(11100101)+1= -00011011 = -27

# Dado um número representado em complemento a 2, o que ele está representando ?

- Outra maneira:
- $2^n + (-N) = d_{n-1} \times 2^{n-1} + d_{n-2} \times 2^{n-2} + \ldots + d_1 \times 2^1 + d_0 \times 2^0$
- $d_{n-1} = 1$, logo $(-N) = -2^n + 2^{n-1} + d_{n-2} \times 2^{n-2} + \ldots + d_1 \times 2^1 + d_0 \times 2^0 =$

  $2^{n-1} \times (-1) + d_{n-2} \times 2^{n-2} + \ldots + d_1 \times 2^1 + d_0 \times 2^0$
- Exemplo: 1001 = -8 +1 = -7

# Soma em Complemento a 2

- Somar +11 com +21

```
+11          00001011
+21          00010101
----------------------
+32          00100000
```

- Somar +5 com +18

```
+5           00000101
+18          00010010
----------------------
+23          00010111
```

# Subtração em Complemento a 2

- N1 - N2 = N1 + (-N2) e joga fora o vai um
- +21 -11 = +21 + (-11)

```
+21                                  00010101
-11          00001011 11110100+1=11110101
-----------------------------------------
+10                                 100001010
```

- +11 – 21 = +11 + (-21)
- +11 00001011
- -21 00010101 = 11101010 +1 = 11101011
- ----------
- -10 11110110

# Overflow ou estouro

- Ocorre quando o número não pode ser representado com o número de dígitos disponível
- Só ocorrerá overflow quando estivermos somando dois números de mesmo sinal
- Se temos 3 bits para representar números em complemento a 2, podemos representar os seguintes inteiros com sinal:
- 011 +3
- 010 +2
- 001 +1
- 000 0
- 111 -1
- 110 -2
- 101 -3
- 100 -4
- Menor número negativo = $-2^{3-1}=-4$
- Maior número positivo = $2^{3-1}-1=+3$

# Overflow ou estouro

- Ocorre quando o número não pode ser representado com o número de dígitos disponível
- Só ocorrerá overflow quando estivermos somando dois números de mesmo sinal
- Exemplo:
  - Se temos 3 bits para representar números em complemento a 2, podemos representar os seguintes inteiros com sinal:

```
011  +3
010  +2
001  +1
000   0
111  -1
110  -2
101  -3
100  -4
```

  - Menor número negativo = $-2^{3-1}=-4$
  - Maior número positivo = $2^{3-1}-1=+3$

# Overflow ou estouro

- Soma: N1+N2
- Primeiro caso:  N1 > 0,  N2 > 0:

```
          00
+2        010
+1        001
-----------------
+3        011   OK

         01
+3      0011
+1       001
-----------------
-4       100   Estouro (overflow)
```

# Overflow ou estouro

- Soma: N1+N2
- Segundo caso:  N1 > 0,  N2 < 0:

```
+1      001
-2      110
-----------------
-1      111  OK

        11
+2       010
-1       111
-----------------
+1       001 OK
```

# Overflow ou estouro

- Soma: N1+N2
- Terceiro caso: N1 > 0, N2 < 0:

```
       11
-1      111
+2      010
-----------------
+1      001  OK


        00
-2      110
+1      001
-----------------
-1      111 OK
```

# Overflow ou estouro

- Soma: N1+N2
- Quarto caso: N1 < 0, N2 < 0:

```
       11
-1      111
-2      110
-----------------
-3      101  OK

       1
-1      111
-4      100
-----------------
+3      011 Estouro (overflow)
```

# Representação de caracteres

- Um caractere deve ser representado utilizando-se um padrão de bits.
- Vários dispositivos de entrada e saída trabalham com 8 bits
- Um código padrão denominado ASCII (American Standard for Computer Information Interchange) define um padrão de seqüência diferente para cada caractere
  - Exemplo:
  - 0100 0001 é 41hex ou 6510 e representa o caractere 'A'
  - 0100 0010 é 42hex ou 6610 e representa o caractere 'B'
- Uma propriedade deste código é que se padrões de bits são comparados, então 'A' <'B', o que ajuda a ordenar elementos em ordem alfabética.
- Note que 'a‘ (61hex) é diferente de 'A' (41hex)
- '8' (38hex) é diferente do inteiro 8
- representação do inteiro 8 em complemento a 2: 0000 0000 0000 0000 0000 0000 0000 1000
- Representação ASCII do caractere '8': 0011 1000
- Representação ASCII do caractere '0' é 48 (decimal) ou 30 (hex)

# Representação de caracteres

- Como traduzir um string de caracteres para um inteiro ?
- Exemplo: '354'
  - **Lê '3'**
  - **Traduz '3' para 3 (subtrai de 48)**
  - **inteiro= 0 *10 + 3 = 3**
  - **Lê '5'**
  - **Traduz '5' para 5 (subtrai de 48)**
  - **inteiro= 3 *10 + 5 = 35**
  - **Lê '4'**
  - **Traduz '4' para 4 (subtrai de 48)**
  - **inteiro= 35 *10 + 4 = 354**

# Representação de caracteres

- Como traduzir um inteiro para string de caracteres ?
- Exemplo: 354
  - **Verifica quantos caracteres existem na base desejada (no caso 3)**
  - **Inicia com base $^{(\text{no. caracteres - 1})}$, $(10^{3-1})=100$**
  - **354 div 100, dá 3**
  - **Traduz 3 para '3' (soma 48) e imprime**
  - **354 mod 100, é 54**
  - **100/10=10**
  - **54 div 10, dá 5**
  - **Traduz 5 para '5' (soma 48) e imprime**
  - **54 mod 10, é 4**
  - **10/10=1**
  - **4 div 1, dá 4**
  - **Traduz 4 para '4' (soma 48) e imprime**
  - **4 mod 1, é 0**
  - **1/10=0, então acabou**