LCC
Sistemas de **C**omputação

Práticas Laboratoriais

Aula 2

Fev 2020

Bruno Medeiros
António Pina
António Esteves

Operações aritméticas em complemento para 2

- Uma vantagem do complemento para 2 é que podemos somar e subtrair dois números sem ter que separar os bits de sinal

- Se o resultado da operação aritmética for demasiado grande (positivo ou negativo) para caber no conjunto de bits utilizado então ocorre **overflow**

Somar valores com sinais diferentes

- Nesta situação nunca ocorre **overflow**
- Usar a regra da soma e descartar o bit de transporte que resulta da soma no último bit (***carry out***)
- **Exemplo**: -8 + 3, utilizando 8 bits

```
    (−8)   1111 1000
+   (+3)   0000 0011
-----------------
    (−5)   1111 1011
```

Somar valores com o mesmo sinal

- **Exemplo**: -2 + (-5)

```
 +1+1+1+1+1 +1+1
      1 1 1 1  1 1 1 0  (-2)
 +    1 1 1 1  1 0 1 1  (-5)
---------------------------
   ~~1~~ 1 1 1 1  1 0 0 1  (-7)
```

→ Deita-se fora o bit de *carry out*

Somar valores com o mesmo sinal: *overflow*

- Se dois números em complemento para 2 forem somados e ambos tiverem o mesmo sinal (ambos positivos ou ambos negativos), ocorre ***overflow*** <u>apenas</u> quando o resultado tiver um sinal aposto

- Regra do ***overflow*** na soma em complemento para 2 (considerando que A, B, C são positivos):

$(+A) + (+B) = -C \Longrightarrow$ ***overflow***

$(-A) + (-B) = +C \Longrightarrow$ ***overflow***

Somar valores com o mesmo sinal: *overflow*

- **Exemplo**: Somar dois valores em complemento para 2 com 4 bits. A gama de valores é [-8, +7].

```
    1 0 0 1   (-7)
  + 1 0 1 0   (-6)
---------------------
  1 0 0 1 1  ⟹ Overflow (-A-B=+C)
```

$\cancel{1}\ 0\ 0\ 1\ 1 \Longrightarrow$ Overflow (−A−B=+C)

O resultado deveria ser **−13** → não cabe na gama permitida.

Subtração em complemento para 2

- Na subtração **a-b**
  - **a** é o **minuendo**
  - **b** é o **subtraendo**

Subtração em complemento para 2

- Normalmente, negamos o subtraendo e efetuamos uma soma
- Qualquer *carry out* é descartado
- **Exemplo** : $+8-(+5)$ usando 8 bits

5_{10} = 0000 0101_2

-5_{10} = 1111 1011_2 (em complemento para 2)

+8 – (+5) = +8 + (-5)

```
+1+1+1+1+1
    0  0  0  0  1  0  0  0   (+8)
+   1  1  1  1  1  0  1  1   (−5)
-------------------------
 1  0  0  0  0  0  0  1  1   (+3)
```

$\cancel{1}$ 0 0 0 0 0 0 1 1 (+3) ⟹ descartar o *carry-out*

Overflow na subtração

- Se dois números em complemento para 2 são subtraídos e os seus **sinais são opostos,** então ocorre ***overflow*** <u>apenas</u> se o resultado tem o mesmo **sinal do subtraendo**

- **Ocorre *overflow* se:**
  - **(+A) − (−B) = −C**
  - **(−A) − (+B) = +C**

Overflow na subtração

- **Exemplo**: **+7−(−6)** com 4 bits em complemento para 2. A gama de valores é [-8,+7].

$-6_{10} = \mathbf{1}010$

$+7 - (-6) = +7 + (+6)$

```
 +1+1
   0 1 1 1   (+7)
 + 0 1 1 0   (+6)
 ---------
   1 1 0 1  ⟹  overflow
```

Sumário: Aritmética complemento p/ 2

- **Soma**
  - Somar os valores e descartar o bit *carry out*
- **Subtração**
  - Negar o subtraendo, somar e descartar o *carry out*
- **Overflow**:
  - Somar 2 **valores positivos** e o resultado é **negativo**
  - Somar 2 **valores negativos** e o resultado é **positivo**
  - Somar 2 valores de sinais diferentes não causa overflow