Aula 7

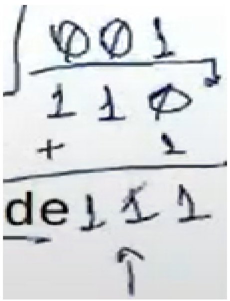
Aritmética Digital

O complemento de dois é o estilo aritmético mais usado pelos computadores, ele tem apenas uma representação para o zero e é fácil obter os valores e realizar operações.

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

Para representar um número negativo utilizando o complemento e dois deve
Inverter todos os bits do valor e soma +1.

exemplo para o -1:



Se o resultado for maior que um número composto por 3 bits, representa um estouro de representação.

Representação sem sinal

Se a instrução estiver acompanhada com a letra “u” no final, significa que é unsigned (sem sinal) e deve ser interpretado como representação de binário natural.
Ex:

addiu…

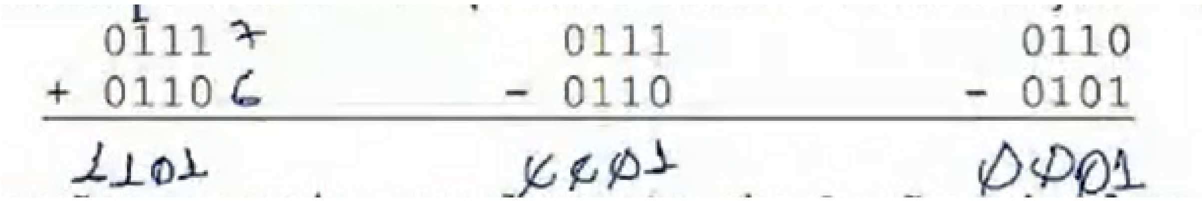
Se o número de N bits tiver mais de N bits, o número mais significativo (de sinal) é copiado nos outros bits.

```
0010  -> 0000 0010
1010  -> 1111 1010

4 bits -> 8 bits
```

Adição, subtração e multiplicação

É feita com aritmética em binário natural (vai um)

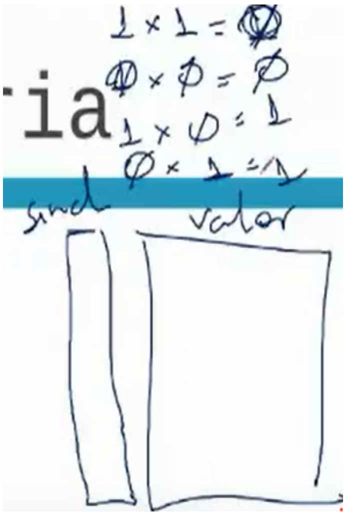


A subtração é feita a partir da adição de número negativo (em complemento de dois)

Na multiplicação, deve-se utilizar o método de produtos parciais, o resultado possui o dobro de bits e funciona apenas com operandos inteiros positivos.

```
     1011
    x1101
 --------
     1011
    0000
   1011
  1011
 --------
 10001111
```

Se for em complemento de dois, deve dividir a multiplicação em duas partes, uma será com apenas bits de sinal e na outra, com os bits dos valores.



Algoritmo para facilitar a multiplicação

Inicia o resultado com zeros, e se o bit do multiplicador for um, repete o multiplicando e em seguida, desloca (shift) o multiplicando para a esquerda .

```
   10000  (multiplicando)
  x 1001  (multiplicador)
 --------
00000000
+   1000
 --------
00001000 (produto parcial)
```

Caso o bit seja zero, não irá alterar o produto parcial, apenas irá deslocar o multiplicando para a esquerda

```
  100000 (multiplicando)
  x 1001  (multiplicador)
 --------
00000000
+   1000
 --------
00001000 (produto parcial)
```

Após passar por todos os bits do multiplicador, deve somar todas as parcelas para gerar o produto final.

```
 1000000 (multiplicando)
   x1001 (multiplicador)
 --------
00000000
+   1000
 --------
00001000
+ 1000000
 --------
01001000 (produto final)
```

Versão 2 do algoritmo de multiplicação

Se o Bit Multiplicador = 1 → adicionar multiplicando ao produto parcial e deslocar o produto parcial à direita (soma e desloca para a direita)

```
    1000 (multiplicando)
   x1001 (multiplicador)
00000000
+1000
10000000 (produto parcial)
01000000 (novo produto parcial)
```

Se o Bit Multiplicador = 0 → desloca produto parcial à direita apenas

```
    1000  (multiplicando)
   x1001  (multiplicador)
01000000  (produto parcial)
00100000 (novo produto parcial)
```

Após passar por todos os bits do multiplicador, deve somar todas as parcelas para gerar o produto final.

```
    1000 (multiplicando)
   x1001 (multiplicador)
00010000
+1000
10010000 (novo produto parcial)
01001000 (produto final)
```

Overflow

Toda vez que uma operação aritmética resultar num número que não caiba em 32 bits (comprimento de uma palavra) gera um overflow.

- Deve se verificar o bit mais significativo nas operações, se caso o resultado gerar um valor maior/menor que [-8…+7] é dito que houve um overflow
- Quando se somar numeros positivos, deve resultar em um numero positivo, caso contrário, é dito que houve overflow
- Quando se soma dois negativos e resulta em um positivo, é overflow
- Quando se subtrai um negativo de um positivo, e gera um negativo é overflow
- Quando se subtrai um positivo de um negativo, é overflow

Operation	Operand A	Operand B	Result indicating overflow
$A + B$	≥ 0	≥ 0	< 0
$A + B$	< 0	< 0	≥ 0
$A - B$	≥ 0	< 0	< 0
$A - B$	< 0	≥ 0	≥ 0