

# PSC

## Aula2

### Representação de números inteiros

- Complemento de 2

0000 0001 0000 0000 b (256)

1111 1110 1111 1111 b

+ 1 b

1111 1111 0000 0000 b (-256)

- Inverter os bits e somar 1

- 16 bits -> -32 768 a 32767

- Se o bit mais significativo for 1, o número é negativo

## Representação de números inteiros

- Exemplo:

- 1111 1111 1111 1111b
- Representa o valor 65535 como UNSIGNED
- SIGNED
- 1111 1111 1111 1111b (-1)
- 0000 0000 0000 0000b (complemento)
- + 1b = 1

## Adição

- Operação sobre inteiros com sinal (SIGNED)

- 3 + 1 = -2
- 3 = 0011b => 1101b (complemento de 2)
- 1 = +0001b
- 1110b => 0001 + 1 = 0010b

- Operação sobre inteiros sem sinal (UNSIGNED)

- 1101 (13 em decimal)
- +0001
- 1110 (14 em decimal)

## Flags - Carry

- Carry = (vai um)
  - Atribuído a 1 se após um operação aritmética o bit de mais alta ordem (MSB) for 1

```
0xF + 0x1 = 1111b
      +1b
      ---
1 0000b
  ↙
(carry bit)
```

## Flags - Overflow

- Se dois números positivos somados produzem um resultado negativo ou se dois números negativos somados produzem um número positivo o *flag* OF= 1

- Exemplo

```
0x7E => 0111 1110b
0x02 => 0000 0010b
      ---
      1000 0000b
```

*CF= 0, OF= 1*

## Como usar CF e OF ?

- No exemplo anterior se 0x7E representa um número sinalizado

```
0111 1110b  126
0000 0010b  + 2
              128
```

Porém o resultado 0x80

```
1000 0000b
```

```
0111 1111b
```

```
+ _____ 1b
```

```
100000000b  0x80 é -128 e não 128
```

## Como usar CF e OF ?

- Se 0x7E e 0x02 são números não sinalizados

```
0111 1110
```

```
+0000 0010
```

```
1000 0000 = 128 em decimal ou seja o
resultado é correto
```

## Como usar o CF e OF?

- Considere a seguinte soma de 0xFF com 0x01

```
  1111 1111b
+0000 0001b
-----
  0000 0000b
```

**CF= 1, OF= 0**

- Se 0xFF e 0x01 forem sinalizados

0xFF = -1

0x01 = 1

0 o valor esta correto!

- Se 0xFF e 0x01 não forem sinalizados

0xFF = 255

0x01 = 1

256 o resultado esta incorreto!

## Como usar o CF e OF?

- Para números sinalizados, o *flag overflow* indica que o resultado é um resultado inválido, e o *flag carry* pode ser desconsiderado
- Para números não sinalizados, o *flag carry* indica que o resultado é um resultado inválido, e o *flag overflow* pode ser desconsiderado

## FLAGS

- Outros *flags* afetados pelas operações aritméticas
  - *ZERO*
  - *SIGN*
  - *PARITY*
  - *AF* = *auxiliary flag* (indica se houve um carry-out entre o quarto e quinto bit, utilizado em operações BCD)

## Flags Register



- Flags de status:
  - CF Bit 0 vai-um (*carry*)
  - PF Bit 2 paridade (*parity*) – oito bits menos significativo (#'s de 1 for par, PF= 1)
  - AF Bit 4 vai-um auxiliar (*auxiliary carry*)
  - ZF Bit 6 zero
  - SF Bit 7 sinal (*sign*)
  - OF Bit 11 overflow
- Flags de controle:
  - TF Bit 8 trap
  - IF Bit 9 interrupt enable
  - DF Bit 10 direção (*direction*)
- Flags específicos do 80386:
  - IOPL Bits 12 e 13 nível de privilégio de E/S
  - NT Bit 14 tarefa aninhada
  - RF Bit 16 flag resume
  - VM Bit 17 modo virtual

## Exercícios

- Considerando dados de 8 bits, faça a soma dos números abaixo, indicando o estado dos *flags* CARRY e OVERFLOW
  - $0xFF + 0x01 = 0x0$  (Flag: 0x0357)
  - $0x7E + 0x7D = 0xFB$ (Flag: 0x0b92)
  - $0x01 + 0xEE = 0xEF$  (Flag: 0x0382)
  - $0x12 + 0x12 = 0x24$  (Flag: 0x0306)

## Subtração

- SUB src, dest => dest = dest -src

$$0xD - 0x1 = 1101b$$

$$\begin{array}{r} 1101b \\ -0001b \\ \hline \end{array}$$

$$1100b$$

$$\text{Unsigned } 13 - 1 = 12$$

$$\text{Signed } -3 - 1 = -4$$

$$(1100 \rightarrow 0011+1=0100b)$$



## Como usar o CF e OF?

- Para números sinalizados, o *flag overflow* indica que o resultado é um resultado inválido, e o *flag carry* pode ser desconsiderado
- Para números não sinalizados, o *flag carry* indica que o resultado é um resultado inválido, e o *flag carry* pode ser desconsiderado

## Exercícios

- Considerando a representação de 16 bits, faça a soma dos números abaixo, indicando o estado dos *flags* CARRY e OVERFLOW
  - $0xFF - 0x01 = 0xFE$  (0x0382)
  - $0x7E - 0x7D = 0x01$  (0x0302)
  - $0x01 - 0xEE = 0x13$  (0x0313)
  - $0x12 - 0x12 = 0x00$  (0x0346)