ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Instruções de multiplicação

- **Instruções de multiplicação**

  MUL fonte

  IMUL fonte

  – **MUL (*multiply*) -> usada com números em representação não-sinalizada;**
  – **IMUL (*integer multiply*) -> usada com números sinalizados.**

- **Multiplicação com números em formato *byte*:**
  – **registradores de 8 bits e variáveis de tipo DB;**
  – **segundo operando é assumido em AL;**
  – **resultado (destino) pode atingir 16 bits e se encontra em AX.**

- **Multiplicação com números em formato *word*:**
  – **registradores de 16 bits e variáveis de tipo DW**
  – **segundo operando é assumido em AX**
  – **resultado pode atingir 32 bits (tamanho *doubleword*) e se encontra em:**
    - **DX -> 16 bits mais significativos (*high word*)**
    - **AX -> 16 bits menos significativos (*low word*)**

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Instruções de multiplicação

Observação:

- **Para números positivos (MSB = 0), MUL e IMUL dão o mesmo resultado**
- **Flags afetados:**
  – **SF, ZF, AF, PF -> indefinidos;**
  – **após MUL, CF/OF (ambos) = 0 , se a metade superior do resultado é 0;**
  = 1 , caso contrário;
  – **após IMUL, CF/OF (ambos) = 0 , se a metade superior do resultado for extensão do sinal da metade inferior;**
  = 1 , caso contrário

- **Exemplos de casos de multiplição:**

1) Suponha que AX contenha 0FFF h:
antes: AX = 0FFF h = 0000 1111 1111 1111 h = 4095 ou + 4095

Instrução	Resultado decimal	Resultado hexadecimal	DX	AX	CF/OF
MUL AX	16769025	00FFE001h	00FFh	E001h	1
IMUL AX	16769025	00FFE001h	00FFh	E001h	1

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Instruções de multiplicação

- **Exemplos de casos de multiplição:**

2) Suponha que AX contenha 0001h BX contenha FFFFh:
antes: AX = 0001 h = 0000 0000 0000 0001b = 1 ou +1
BX = FFFF h = 1111 1111 1111 1111b = 65535 ou -1

Instrução	Resultado decimal	Resultado hexadecimal	DX	AX	CF/OF
MUL BX	65535	0000FFFFh	0000h	FFFFh	0
IMUL BX	-1	FFFFFFFFh	FFFFh	FFFFh	0

3) Suponha que AL contenha 80h BL contenha FFh:
antes: AL = 80 h = 1000 0000 b = 128 ou -128
BL = FF h = 1111 1111 b = 255 ou -1

Instrução	Resultado decimal	Resultado hexadecimal	AH	AL	CF/OF
MUL BL	326407	7F80h	7Fh	80h	1
IMUL BL	128	0080h	00h	80h	1

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Instruções de divisão

- **Instruções de divisão**

DIV fonte
IDIV fonte

  - **DIV (*divide*) -> usada com números em representação não-sinalizada**
  - **IDIV (*integer* divide) -> usada com números sinalizados**
    - **fonte deve ser considerado como divisor (não pode ser uma constante)**

- **Divisão com números em formato *byte*:**
  - **o divisor é assumido ser de 8 bits (1 byte)**
  - **o dividendo é assumido estar em AX (16 bits)**
  - **após a execução: o quociente de 8 bits estará em AL o resto de 8 bits estará em AH**

- **Divisão com números em formato *word*:**
  - **o divisor é assumido ser de 16 bits (1 word)**
  - **o dividendo é assumido ser de 32 bits:**
    - **DX -> 16 bits mais significativos do dividendo (*high word*)**
    - **AX -> 16 bits menos significativos do dividendo (*low word*)**
    - **após a execução: o quociente de 16 bits estará em AX o resto de 16 bits estará em DX**

- **Para números positivos (MSB = 0), DIV e IDIV fornecem o mesmo resultado.**

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Instruções de divisão

- **Flags afetados: todos ficam indefinidos**
- **Em divisão de números em representação sinalizada, o resto possui o mesmo sinal do dividendo.**
- **Exemplos de casos de divisão:**

1) Suponha que DX e AX contenham 0000h e 0005h, e BX contenha FFFEh:

antes: DX : AX = 0000 0005 h = 5 ou + 5
BX = FFFE h = 65534 ou - 2

Instrução	Quociente decimal	Resto decimal	AX	DX
DIV BX	0	5	0000h	0005h
IDIV BX	-2	1	FFFEh	0001h

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Instruções de divisão

- **Exemplos de casos de divisão:**

2) Suponha que AX contenha 0005h e BL contenha FFh:
antes: AX = 0005 h = 5 ou + 5
BL = FF h = 256 ou - 1

Instrução	Quociente decimal	Resto decimal	AL	AH
DIV BL	0	5	00h	05h
IDIV BL	- 5	0	FBh	00 h

3) Suponha que AX contenha 00FB h e BL contenha FF h:
antes: AX = 00FB h = 251 ou + 251
BL = FF h = 256 ou - 1

Instrução	Quociente decimal	Resto decimal	AL	AH
DIV BL	0	251	00h	FBh
IDIV BL	-251 *	-	-	-

- **(*) como -251 não cabe em AL (8 bits) -> ocorre *DIVIDE OVERFLOW* - situação de erro catastrófico que faz com que o programa termine.**

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Extensão do sinal do dividendo

- **Em operações em formato word: Caso o dividendo de uma divisão (composto de DX : AX) ocupe apenas AX, DX deve ser preparado, pois é sempre considerado.**
  - **Em DIV -> DX deve ser zerado**
  - **EM IDIV -> DX deve ter a extensão de sinal de AX**

CWD

- **Instrução sem operandos (zero operandos) que converte *word* para *doubleword* e estende o sinal de AX para DX. Deve ser usada com IDIV.**

- **Em operações em formato byte: Caso o dividendo de uma divisão (composto por AX) ocupe apenas AL, AH deve ser preparado, pois é sempre considerado.**
  - **Em DIV -> AH deve ser zerado**
  - **EM IDIV -> AH deve ter a extensão de sinal de AL**

CBW

- **Instrução sem operandos (zero operandos) que converte *byte* para *word* e estende o sinal de AL para AH. Deve ser usada com IDIV**

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Exemplos de divisões com ajuste de extensão:

1) Crie um trecho de programa que divida -1250 por 7.

```
...
MOV  AX, -1250    ;AX recebe o dividendo
CWD               ;estende o sinal de AX para DX
MOV  BX,7                 ;BX recebe o divisor
IDIV  BX          ;executa a divisão
                  ;após a execução, AX recebe o
                  ;quociente e DX recebe o resto
...
```

2) Crie um trecho de programa que divida a variável sinalizada XBYTE por -7.

```
...
MOV  AL,XBYTE     ;AL recebe o dividendo
CBW               ;estende o sinal (eventual) de
                  ;AL para AH
MOV BL, -7        ;BL recebe o divisor
IDIV BL           ;executa a divisão
                  ;após a execução, AL recebe o
                  ;quociente e AH recebe o resto
...
```

Obs: Não há efeito de CBW e CWD sobre os FLAGS.

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Exemplos de aplicação

- **Entrada de números decimais:**
  - ***string* de caracteres números de 0 a 9, fornecidos pelo teclado;**
  - **CR é o marcador de fim de *string*;**
  - **AX é assumido como registrador de armazenamento;**
  - **valores decimais permitidos na faixa de - 32768 a + 32767;**
  - **sinal negativo deve ser apresentado.**
- **Algoritmo básico em linguagem de alto nível:**

```
total = 0
negativo = FALSO
ler um caracter
CASE caracter OF
    ' - '   :  negativo = VERDADEIRO  e  ler um caracter
    ' + '   :  ler um caracter
END_CASE
REPEAT
    converter caracter em valor binário
    total = 10 x total + valor binário
    ler um caracter
UNTIL caracter é um carriage return (CR)
IF negativo = VERDADEIRO
    THEN total = - (total)
END_IF
```

Obs: O *loop* do tipo CASE pode ser entendido como um IF múltiplo, que testa simultaneamente os vários "casos": se algum deles for verdadeiro, executa as instruções relacionadas; se todos os "casos" forem falsos, não executa nada e vai para o fim do CASE.

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Exemplos de aplicação

- **Saída de números decimais:**
  - **AX é assumido como registrador de armazenamento;**
  - **valores decimais na faixa de - 32768 a + 32767;**
  - **exibe sinal negativo, se o conteúdo de AX for negativo;**
  - ***string* de caracteres números de 0 a 9, exibidos no monitor de vídeo.**
- **Algoritmo básico em linguagem de alto nível:**

```
IF AX < 0
    THEN    exibe um sinal de menos substitui-se AX pelo seu complemento de 2
END_IF
contador = 0
REPEAT
    dividir quociente por 10
    colocar o resto na pilha
    contador = contador + 1
UNTIL quociente = 0
FOR contador vezes DO
    retirar um resto (número) da pilha
    converter para caracter ASCII
    exibir o caracter no monitor
END_FOR
```

Idéia básica da técnica de decomposição decimal do número em AX:

		Pilha
24618 dividido por 10 = 2461	com resto 8	0008h
2461 dividido por 10 = 246	com resto 1	0001h
246 dividido por 10 = 24	com resto 6	0006h
24 dividido por 10 = 2	com resto 4	0004h
2 dividido por 10 = 0	com resto 2	0002h ← Topo