Interface Hardware/Software

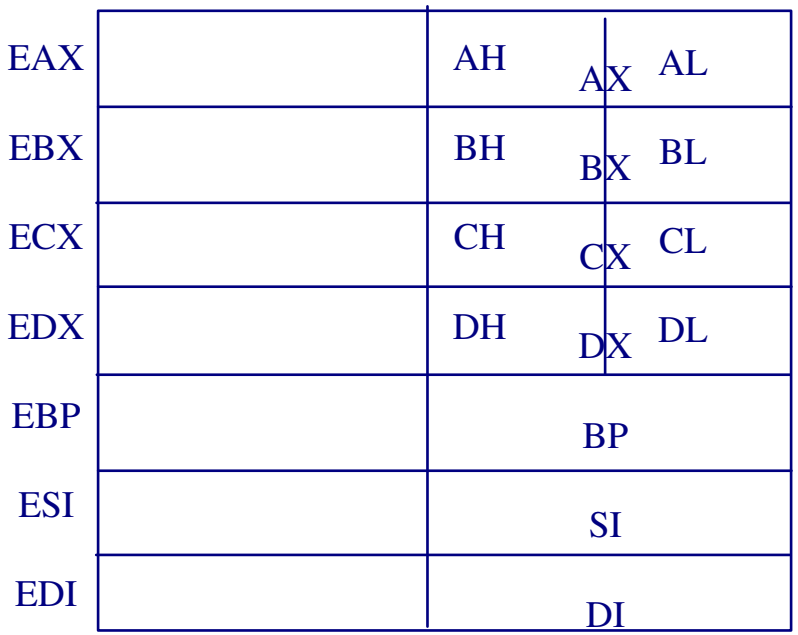
Centro de Informática - UFPE
Paulo Maciel
prmm@cin.ufpe.br

Iniciação Cientifica

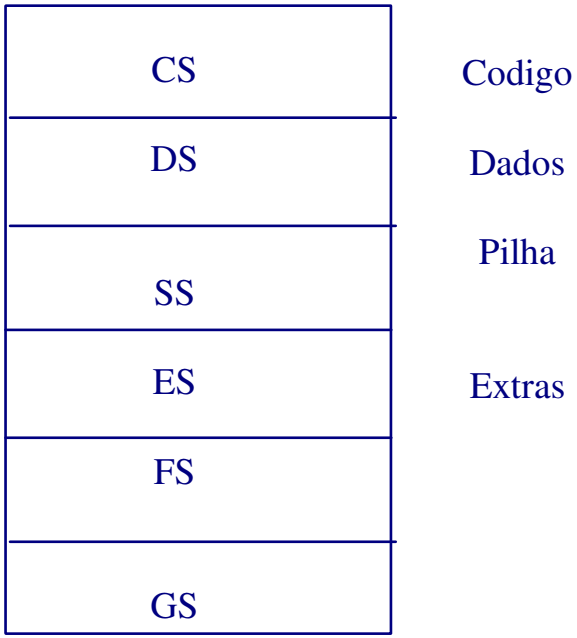
- Desenvolvimento de uma ferramenta para modelagem e análise.
  - UML/SysML
  - MARTE
  - RdP
  - INA

Modelo de Software

Registradores de Uso Geral



Registradores de Segmento



Modelo de Software

- Registradores de Dado:
  - **EAX (Acumulador)** - Utilizado como acumulador em operações aritméticas e lógicas.
  - **EBX (Base)** - Usado como registrador de BASE para referenciar posições de memória.
  - **ECX (Contador)** - Utilizado em operações iterativas e repetitivas para contar .
  - **EDX (Dados)** – Utilizado em operações de multiplicação para armazenar parte de um produto de 32 bits, ou em operações de divisão, para armazenar o resto.

Modelo de Software

- Registradores apontador de pilha e de índice:
  - **ESP (apontador de pilha)** - É utilizado em conjunto com SS, para acessar a área de pilha na memória; aponta para o topo da pilha.
  - **EBP (apontador de base)** - É o ponteiro que, em conjunto com SS, permite acesso de dados dentro do segmento de pilha.
  - **ESI (índice fonte)** - Usado como registrador índice em alguns modos de endereçamento indireto, em conjunto com DS.
  - **EDI (índice destino)** - Similar ao ESI, atuando em conjunto com ES.

Modelo de Software

- Registradores de Segmento:
  - **CS** - Endereça o segmento de código.
  - **DS** - Endereça o segmento de dados.
  - **SS** - Endereça o segmento de pilha.
  - **ES, FS e GS** - Endereça o segmento de dados extra.

Modelo de Software

Registrador de Flags

CF - Vai-um
PF - Paridade
AF - Vai-um auxiliar
ZF - Zero
SF - Sinal
OF - Estouro

IF – Interrupção
DF - Direção
TF - Passo Simples
IOPL - Nível de Prioridade da Tarefa
NT - Tarefa Aninhada

Modelo de Software

- Flags:
  - **CF (Carry Flag)** – Indica um vai-um do 7° para o 8° bit.
    - Ex.: Mov Bh, FFH ;255
      Mov Ah, 7
      Add Ah,Bh
      Ah = 257 (255-FF)
  - **ZF (Zero Flag)** – É setado (Z= 1) se o resultado de uma operação (que afeta o flag ZF, operações aritméticas e lógicas) é zero.

Modelo de Software

- Registradores:
  - **SF (Sign Flag)** - É setado (SF=1) quando o operando destino é negativo e é zerado (SF=0) quando o operando destino é positivo.
  - **OF (Overflow Flag)** – Indica que houve um estouro no cálculo entre números com sinal.
    - 8 bits – Varia de -128 a +128
    - 16 bits – Varia de -32768 a + 32768

Modelo de Software

- Registradores:
  - **AF (Auxiliary Carry Flag)** – Indica um vai-um do 3° para o 4° bit. Usado por operações que convertem números para BCD.

Modos de Endereçamento de Dados

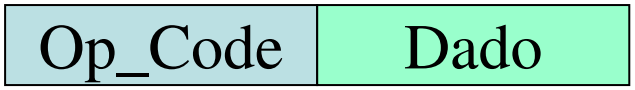
- Imediato
- Registrador Indireto
- Direto
- Indexado
- Base
- Base Indexado
- Base Indexado com Deslocamento
- Porta

Modos de Endereçamento de Dados

- Vantagem:
  - Flexibilidade.
- Desvantagem:
  - Complexidade.

Modos de Endereçamento de Dados

- **Imediato** - O operando é um dado contido na própria instrução.



Ex.: mov cx,1024H

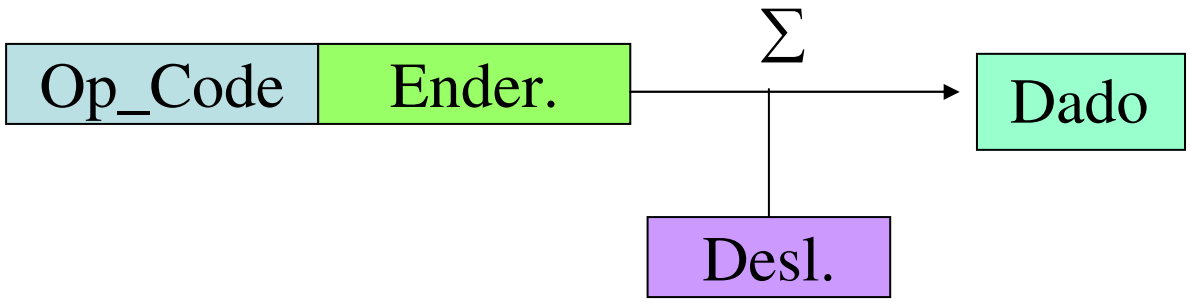
Modos de Endereçamento de Dados

- **Registrador Indireto** - O valor do operando está num registrador.

  - Ex.:mov bx,10
    mov cx,bx

Modos de Endereçamento de Dados

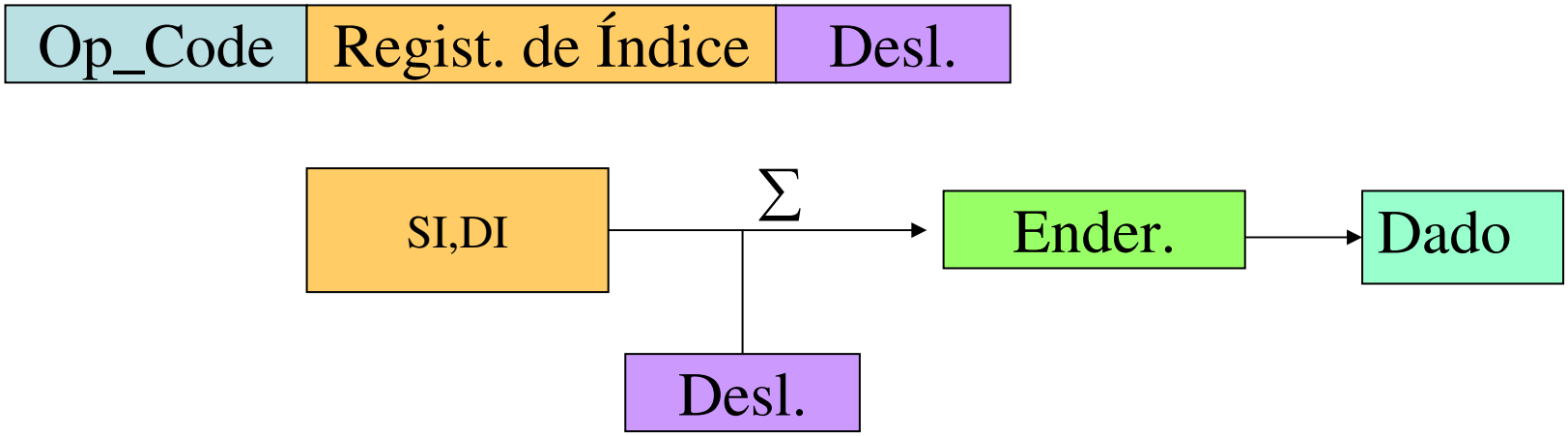
- **Direto** - O endereço efetivo de memória aponta para o valor.



  - Ex.: mov al,[0300h]

Modos de Endereçamento de Dados

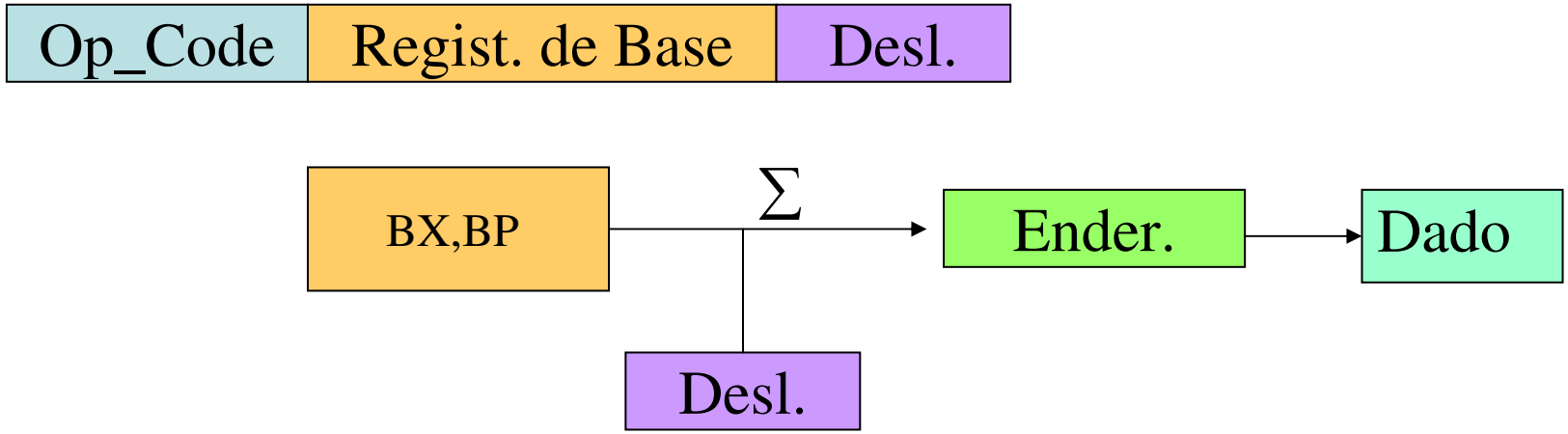
- **Indexado** - Endereço efetivo é a soma de um registrador de índice, e o deslocamento. Este endereço aponta para o valor a ser utilizado.



  - Ex.: mov si,200
    mov cx,[si+02]

Modos de Endereçamento de Dados

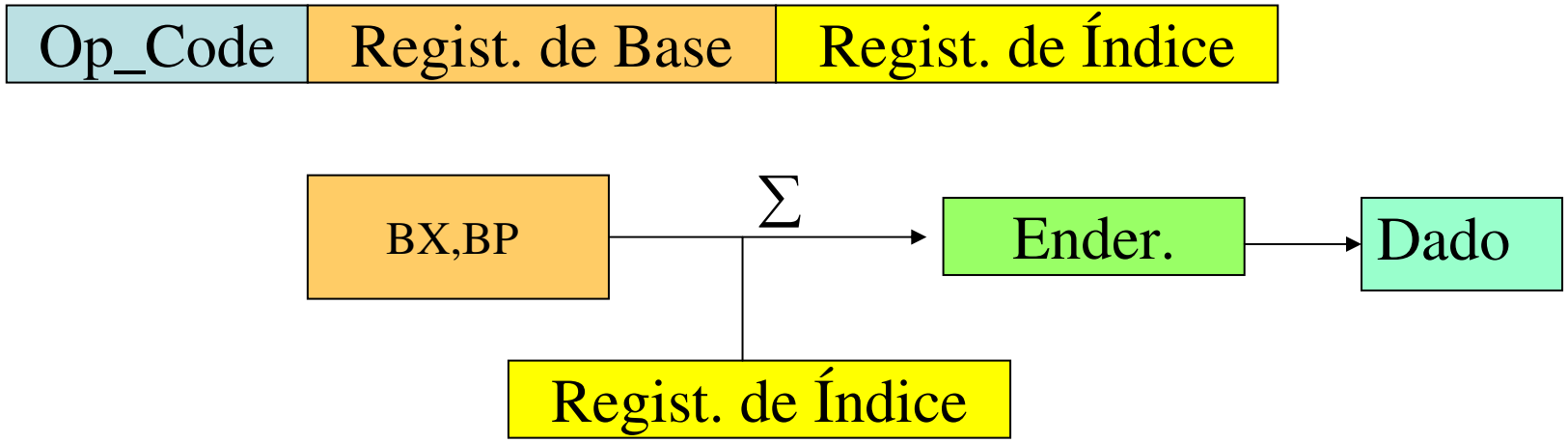
- **Base** - Endereço efetivo é a soma de um registrador de base e o deslocamento. Este endereço aponta para o valor a ser utilizado.



  - Ex.: mov bx,100
    mov cx,[bx+02]

Modos de Endereçamento de Dados

- **Base Indexado** - Endereço efetivo é a soma do registrador de base com o registrador de índice. Este endereço aponta para o valor a ser utilizado.

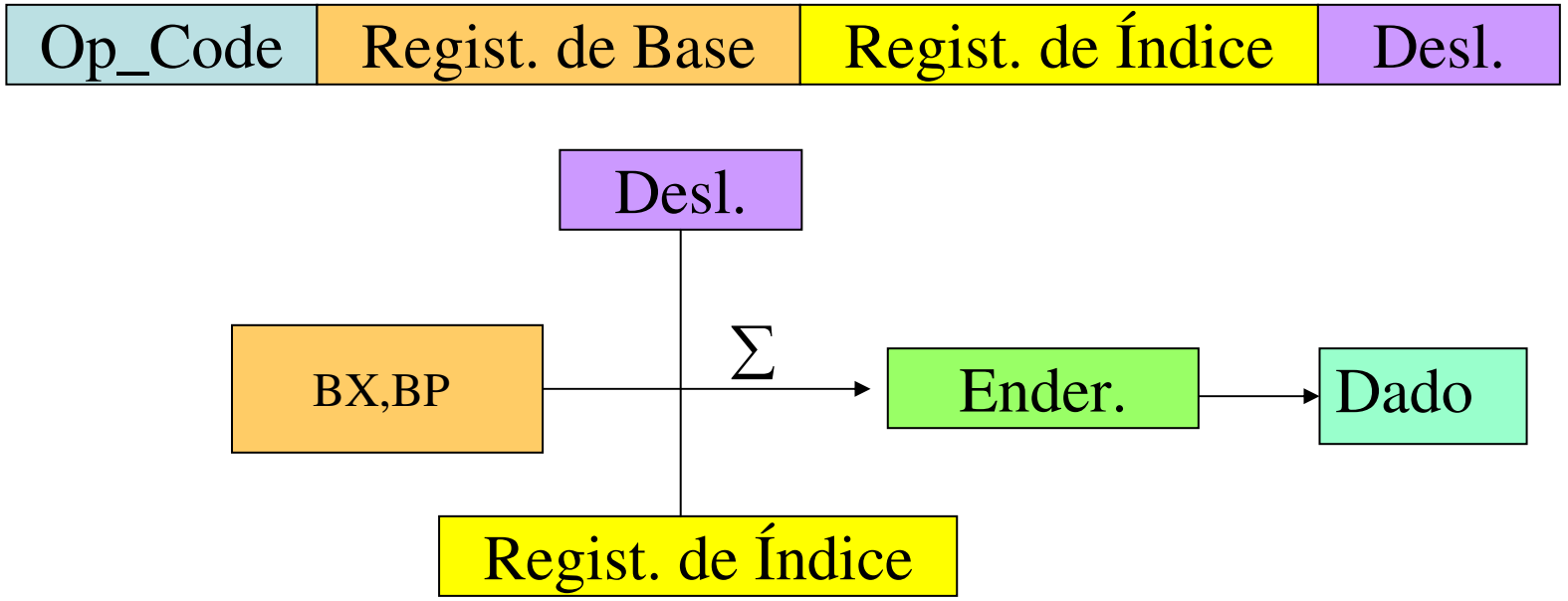


Modos de Endereçamento de Dados

- **Base Indexado**
  - Ex.: Mov bx,100
    Mov si,100
    Mov cx,[bx+si]

Modos de Endereçamento de Dados

- **Base Indexado com Deslocamento.**



  - Ex.: mov ch,[bx+si+03]

Modos de Endereçamento de Dados

- **Porta**
  - Comunicação entre o processador e o endereços de I/O. Ao se executar as instruções de IO, ao invés de MRQ se tornar ativo, IORQ é ativado. Portanto, os endereços de IO são distintos dos endereços de memória.

  - Ex.:
```
in al,80h  ;espaço de  end. = 256
out 90h,al
in al,dx ;espaço de end. = 65536
out dx,al
```

Tipo de Instruções

- Transferência de dados
- Aritmética
- Manipulação de bit
- Laços e saltos
- Subrotinas e interrupções
- Controle
- String

Tipo de Instruções

- Instruções Monádicas
  - inst por_dest
    - opr dest - operando destino.
    - Ex.: INC AX
- Instruções Diádicas
  - inst por_dest, por_fonte
    - opr dest - operando destino.
    - opr fonte - operando fonte.
    - EX.: ADD AX,BX

Transferência de Dados

- São usadas para mover dados entre:
  - Registradores
  - Memória
  - Mundo Externo (portas)

- Também são usadas para manipular a pilha e as *flags*.

Transferência de Dados

Instruções

MOV	Move		
MOVSX	Ext. Zero	**IN**	Input
MOVZX	Ext. C/ sinal	**OUT**	Output
PUSH	Empilhamento	**XCHG**	Permuta
PUSHW/PUSHD	Imedit./ 32 bits	**XLAT**	Translate-table
PUSHA/PUSHAD	Todos Reg./ 32 bits	**LEA**	Carga de End. efetivo
PUSHF/PUSHFD	16 bits Flags. / 32bits	**LDS e similares**	Carga de Ponteiro
POP	Desempilhamento	**BSWAP**	Troca
POPA/POPAD	Todos Reg./ 32 bits	**LAHF**	Carg. os flags em AH
POPF/POPFD	16 bits Flags/ 32 bits	**SAHF**	Arm. de AH nos flags

Transferência de Dados

- MOV – Transfere um dado do operando fonte para o operando destino.

  – Sintaxe:

MOV OP_DEST,OP_FONTE

EX.: MOV AL,01
MOV AX,[BX+2]
MOV AL,[0400H]

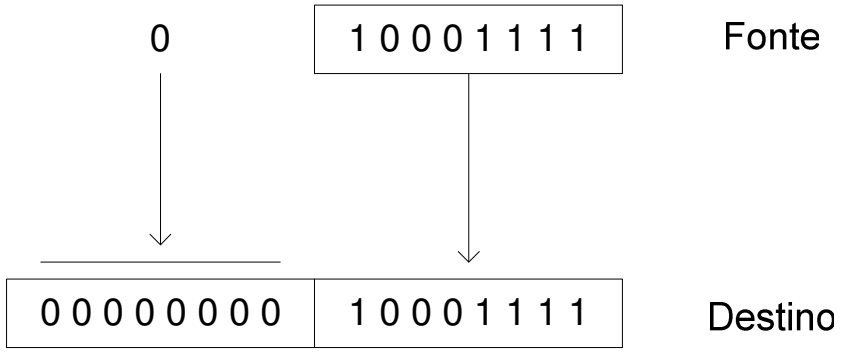
Transferência de Dados

- MOVSX - Quando se copia um valor menor para um destino maior, a instrução MOVZX estende a parte mais significativa do destino com zeros.

  – Sintaxe:

MOVSX OP_DEST,OP_FONTE

EX.: MOVSX AX,AL



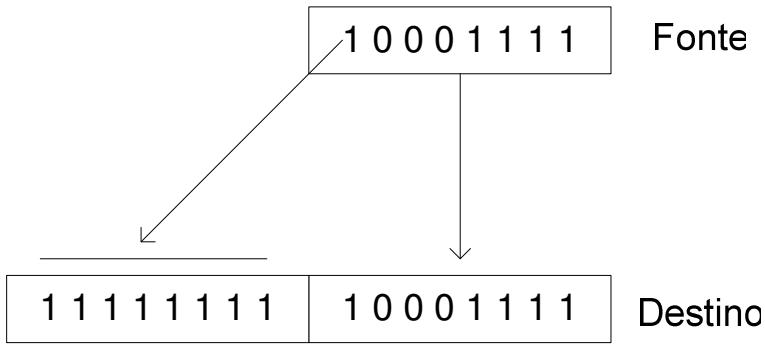
Transferência de Dados

- MOVZX - A instrução MOVZX preenche a parte mais significativa do destino com uma cópia do bit de sinal do operando fonte.

  – Sintaxe:

MOVZX OP_DEST,OP_FONTE

EX.: MOVZX AX,AL



Transferência de Dados

- PUSH – Decrementa o SP (Stack Point) e transfere o dado do operador fonte para o topo da pilha.

  – Sintaxe:

  PUSH OP_FONTE

  EX.: PUSH BX

Transferência de Dados

- PUSH
  - Ex.: Mov bx, 100H
    Mov ax, 200H
    PUSH BX
    PUSH AX
    POP BX

Transferência de Dados

- PUSHW – Transfere um dado imediato do operador fonte (16 bits) para o topo da pilha.

  – Sintaxe:

  PUSHW OP_FONTE

  EX.: PUSHW 34ADh

Transferência de Dados

- PUSHD – Transfere um dado (32 bits) do operador fonte para o topo da pilha.

  – Sintaxe:

  PUSHD OP_FONTE

  EX.: PUSHD EAX

Transferência de Dados

- PUSHA - Transfere todos os registradores para a pilha (16 bits).
  - Sintaxe:

    PUSHA

    EX.: PUSHA

Transferência de Dados

- PUSHAD - Transfere todos os registradores para a pilha (32 bits).

  – Sintaxe:

  PUSHAD

  EX.: PUSHAD

Transferência de Dados

- PUSHF - Transfere o estado de cada flag para a pilha (16 bits flags).
  - Sintaxe:

PUSHF

EX.: PUSHF

Transferência de Dados

- PUSHFD - Transfere o estado de cada flag para a pilha (32 bits flags).
  - Sintaxe:

PUSHFD

EX.: PUSHFD

Transferência de Dados

- POP – Recupera um dado localizado na posição corrente da pilha do operador destino e decrementa o SP (stack point).
  - Sintaxe:
  - POP OP_DEST

EX.: POP BX

Transferência de Dados

- POPA - Recupera todos os registradores na pilha (16 bits).

  – Sintaxe:

  POPA

  EX.: POPA

Transferência de Dados

- POPAD - Recupera todos os registradores na pilha (32 bits).

  – Sintaxe:

  POPAD

  EX.: POPAD

Transferência de Dados

- POPF - Recupera o estado de cada flag (16 bits flags).
  - Sintaxe:

POPF

EX.: POPF

Transferência de Dados

- POPFD- Recupera o estado de cada flag (32 bits flags).

  – Sintaxe:

  POPFD

  EX.: POPFD

Transferência de Dados

- IN – Transfere um dado de uma porta de entrada para o registrador AL.
  - Sintaxe:

IN ACUMULADOR, PORTA (Entrada)

EX.: IN AL,60H

Transferência de Dados

- OUT – Transfere um dado de AL para uma porta de saída.
  - Sintaxe:

OUT PORTA, ACUMULADOR

EX.: OUT 80H,AX

Transferência de Dados

- XCHG – Permuta o dado do operador destino com o dado do operador fonte.

  – Sintaxe:

  XCHG OP_DEST,OP_FONTE

  EX.: XCHG AL,AH
  XCHG AL,[BX]

Transferência de Dados

- XCHG
  - Ex.: Mov ax, 20
    Mov bx,30
    XCHG ax,bx

Transferência de Dados

- XLAT - Converter um valor presente em AL acessando uma tabela previamente endereçada por BX, com no máximo 256 valores, usando o valor em AL como índice desta tabela.

  – Sintaxe:

  XLAT

  EX.: MOV AL,3
  MOV BX,0400H
  XLAT

Transferência de Dados

- LEA - Carregar o *offset* de um endereço ou deslocamento de um operando na memória.

  – Sintaxe:

  LEA OP_DEST,LABEL

  EX.: LEA BX,[40H]
  LEA = 40

  Qual a diferença entre MOV BX,[40H] e LEA BX,[40H] ??

Transferência de Dados

- LDS - Transfe quatro bytes consecutivos de um operando fonte para um par de registradores de 16 bits.
  - Sintaxe:

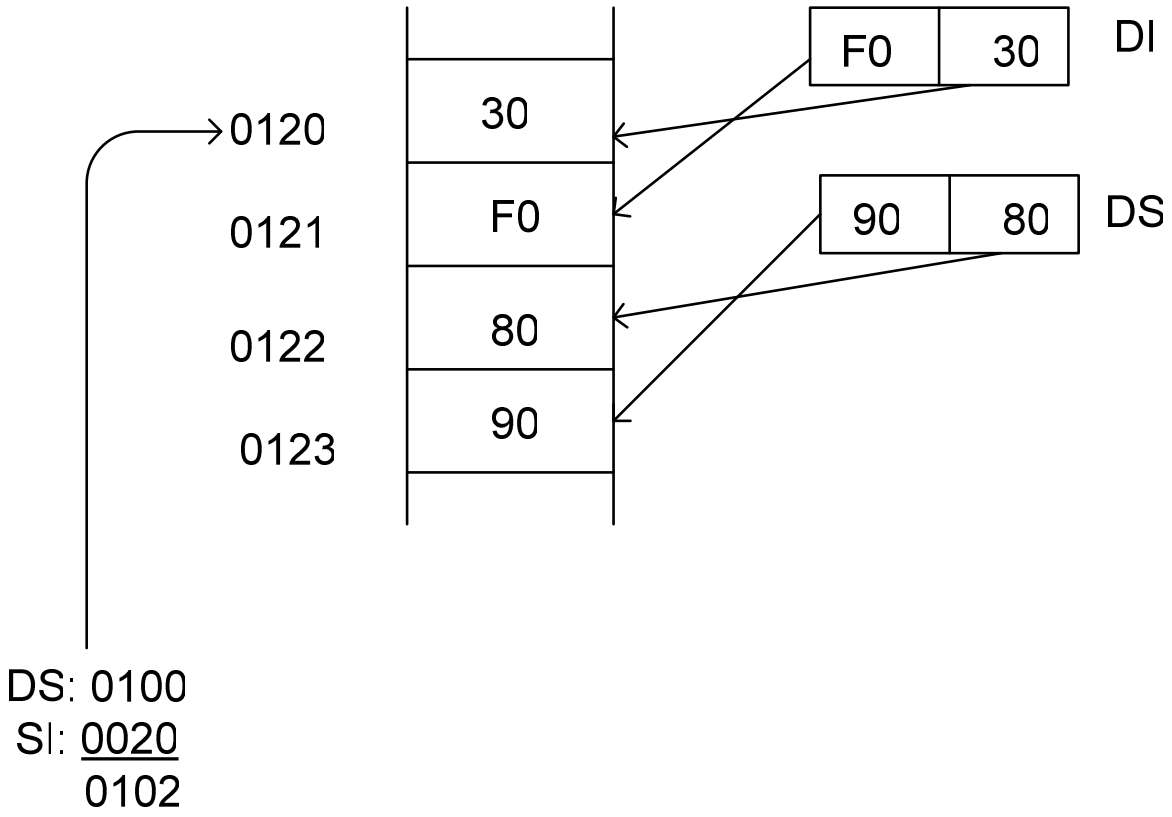
LDS OP_DEST,OP_FONTE

LES,LFS,LGS,LSS - são similares

EX.: LDS DI,[SI]

Transferência de Dados

- LDS
  - Exemplo



Transferência de Dados

- BSWAP – Realiza a troca (swapping) dos bytes dos registradores (32 bits).

  – Sintaxe:

  BSWAP OP_DEST

  EX.: MOV EAX,12345678H
  BSWAP EAX

  EAX = 78563412H

Transferência de Dados

- LAHF - Carrega o registrador AH com os dados dos flags.

  – Sintaxe:

  LAHF

  EX.: POPF
  LAHF

Transferência de Dados

- SAHF - Armazena nos flags o registrador AH.

  – Sintaxe:

  SAHF

  EX.: MOV AH,0FFH
  SAHF

Instruções

ADD	Adição
ADC	Adição C/ Carry
INC	Incremento
SUB	Subtração
SBB	Subtração C/ Carry
DEC	Decremento
CMP	Comparação
MUL	Multiplicação
IMUL	Multiplicação C/ Sinal
DIV	Divisão
IDIV	Divisão C/ Sinal

NEG	Complemento a 2
CBW	Mudança de módulo
CWD	Mudança de módulo
DAA	Ajuste BCD
DAS	Ajuste BCD
AAA	Ajuste ASCII
AAS	Ajuste ASCII
AAM	Ajuste ASCII
AAD	Ajuste ASCII

Aritmética

- ADD – Realiza adição normal entre dois operandos, um fonte e outro destino.

  – Sintaxe:

ADD OP_DEST,OP_FONTE

EX.: ADD AL,CH

ADD CX,[SI]

ADD DX,4

Flags: AF, CF, OF, PF ,SF, ZF

Aritmética

- ADC - Realiza adição normal entre dois operandos, um fonte e outro destino mais o valor do *flag carry*.
  - Sintaxe:

ADC OP_DEST,OP_DEST

EX.:ADC AL,CH

ADC [BX],DX

ADC DX,4

Aritmética

- ADC
  - Ex.: Mov BL, FFH ;255
    Mov CL, 02
    ADD BL,CL ; 257 CF = 1
    LAHF
    ADC BX,CX
    AX = 4

Aritmética

- INC – Incrementa 1 ao conteúdo do operando.

  – Sintaxe:

  INC OP_DEST

  EX.: INC AL
  INC [BX]
  INC [0400H]

Aritmética

- INC
  - Ex.:

```
T:  INC CX
    CMP CX,5
    JNE T       ; Salta se não igual a 5
    MOV AX,15
```

Aritmética

- SUB - Realiza a subtração normal entre dois operandos, um fonte e outro destino.
  - Sintaxe:

SUB OP_DEST,OP_DEST

Flags: AF, CF, OF, PF ,SF, ZF

EX.: SUB AL,CH

SUB CX,[0300H]

SUB DX,4

Aritmética

- SBB - Realiza a subtração normal entre dois operandos, um fonte e outro destino menos o valor do *flag carry*.
  - Sintaxe:

SBB OP_DEST,OP_FONTE

EX.: SBB AL,CH
SBB CX,[0300H]
SBB DX,4

Aritmética

- SBB
  - Ex.: Mov BL, FFH ;255
    Mov CL, 02
    ADD BL,CL ; 257 CF = 1
    LAHF
    Mov AX,8
    SBB AX,CX
    AX = 5

Aritmética

- DEC - Decrementa 1 ao conteúdo do operando.
  - Sintaxe:

DEC OP_DEST

EX.: DEC AL
DEC [BX]
DEC [0400H]

Aritmética

- DEC
  - Ex.: XOR BL,BL
    MOV AL,5
    T: INC BL
    DEC AL
    JNZ T ; Salta se não Zero.
    MOV, CX 100

Aritmética

- CMP - Subtrai o operador origem do destino, mas não armazena o resultado da operação, apenas afeta o estado das flags de estado.
  - Sintaxe:

CMP OP_DEST,OP_ORIGEM

EX.: CMP AL,CH
CMP CX,[0300H]
CMP DX,4

Flags: AF, CF, OF, PF, SF, ZF

Aritmética

- CMP
  - Ex.: Mov al,5
    CMP al,5

Aritmética

- CMPXCHG - Compara o operando destino com o fonte e caso o operando fonte seja igual ao operando destino, o operando fonte é copiado no destino. Caso sejam diferentes, o valor do fonte é substituído pelo valor do destino.
  - Sintaxe:

CMPXCHG OP_DEST, OP_FONTE

EX.: CMPXCHG BL,CL

CMPXCHG CX,[0300H]

CMPXCHG EDX,EBX

Aritmética

- MUL - Realiza a mutiplicação entre dois operandos, um fonte e outro acumulador (AL ou AX) e atribui o resultado ao acumulador (AX ou DX).
  - Sintaxe:

MUL OP_FONTE

Flags: CF, OF

EX.: MUL BL
MUL DX
MUL ECX

Aritmética

- MUL
  - Ex.: MOV AL, 83H
    MOV CL, 44H
    MUL CL ; realiza o produto entre AL e CL, resultado em AX.

    MOV AX,1234H
    MOV BX,6000H
    MUL BX ; realiza o produto de 16 bits entre AX e BX, resultado AX.

Aritmética

- IMUL - Realiza a mutiplicação sinalizada entre dois operandos, um fonte e outro acumulador (AL ou AX) e atribui o resultado ao acumulador (AX ou DX).
  - Sintaxe:

IMUL OP_FONTE

EX.: IMUL BL
IMUL DX
IMUL ECX

Aritmética

- IMUL
  - Ex.: Mov al, -5
    Mov bl, -5
    IMul bl
    al = 25 (19h)

Aritmética

- DIV - Realiza a divisão entre o conteúdo do acumulador (AL ou AX) pelo operador fonte e atribuindo o resultado ao acumulador (Al ou AX).
  - Sintaxe:

DIV OP_FONTE

Flags: AF, CF, OF, PF, SF, ZF

EX.: DIV BL

DIV DX

DIV ECX

Aritmética

- IDIV - Realiza a divisão sinalizada entre o conteúdo do acumulador (AL ou AX) pelo operador fonte e atribuindo o resultado ao acumulador (AX ou DX).
  - Sintaxe:

IDIV OP_FONTE

EX.: IDIV BL
IDIV DX
IDIV ECX

Aritmética

- NEG – Realiza o complemento de dois de um operando.

  – Sintaxe:

  NEG OP_DEST

  EX.: MOV AL, 5 (+5)
  NEG AL
  AL = FB

Aritmética

- Sintaxe:
  - **CBW** - Converte byte de AL em palavra (16 bits) estendendo-se o sinal.
  - **CWD** - Converte a palavra de AX em dupla-palavra estendendo-se o sinal (DX:AX).
  - **CWDE** - Converte a palavra de AX em dupla-palavra estendendo-se o sinal (EAX)
  - **CDQ** - converte a dupla-palavra de EAX em quádrupla-palavra estendendo-se o sinal (EDX:EAX)

EX.: CBW CWDE

CWD CDQ

Aritmética

- *Instruções de Ajuste*
  - Estas instruções são utilizadas quando os dados manipulados pelo programa estão em formato ASCII ou BCD.
  - Sintaxe:
    - DAA- Ajuste Decimal para adição.
    - DAS – Ajuste Decimal para Subtração.
    - AAA - Ajuste ASCII para adição.
    - AAS - Ajuste ASCII para subtração.
    - AAM - Ajuste ASCII para multiplicão.
    - AAD - Ajuste ASCII para divisão.

Aritmética

- DAA- Corrige o resultado presente no acumulador AL, após uma soma entre dois valores BCD.
  - 4 bits menos significativos de Al estiverem entre A e F soma 06H.
  - 4 bits mais significativos de Al estiverem entre A e F soma 60H.
  - Sintaxe:

```
DAA

EX.: MOV AL,15H
     MOV BL,25H
     ADD AL,BL          ;AL=3AH
     DAA                ;AL=40H
```

Lógica e Manipulação de Bit

Instruções Lógicas

NOT

AND

OR

XOR

TEST

Instruções Manipulação de bit

SHL/SAL

SHR

SAR

ROL

ROR

RCL

RCR

Lógica

- NOT – Realiza a negação do operador de destino bit a bit.
  - Sintaxe:

NOT OP_DEST

EX.: NOT AL

NOT [0200H]

NOT DX

Lógica

- AND – Realiza a função lógica "AND" entre cada bit de um operador fonte com um operador destino.
  - Sintaxe:

AND OP_DEST,OP_FONTE

EX.: AND AL,CH

AND CX,[SI]

AND DX,FFFFH

Fonte	Destino	Destino
1	1	1
1	0	0
0	1	0
0	0	0

Lógica

- OR - Realiza a função lógica "OR" entre cada bit de um operador fonte com um operador destino.
  - Sintaxe:

OR OP_DEST,OP_FONTE

EX.: OR AL,00H
OR CX,[SI]
OR DX,FFFFH

Fonte	Destino	Destino
1	1	1
1	0	1
0	1	1
0	0	0

Lógica

- XOR - Realiza a função lógica "XOR" (ou exclusivo) entre cada bit de um operador fonte com um operador destino.
  - Sintaxe:

XOR OP_DEST,OP_FONTE

EX.: XOR AL,CH
XOR CX,[SI]
XOR DX,DX

Fonte	*Destino*	*Destino*
1	*1*	*0*
0	*0*	*1*
0	*1*	*1*
0	*0*	*0*

Lógica

- TEST – Realiza um AND entre os operandos, atualizando os flags e sem armazenar o resultado.
  - Sintaxe:

TEST OP_DEST,OP_FONTE

EX.: TEST AL,CH
TEST CX,[SI]
TEST DX,FFFFH

Lógica

- SETcc - Atribui 01 ao operando destino se a condição testada for verdadeira. Caso contrário, atribui-se 00.
  - Sintaxe:

SETcc OP_DEST

EX.: SETZ AL

SETLE AX

SETNC DX (CF = 0)

Lógica

- BSF/BSR - Procura pelo 1°bit igual a um no operando fonte a partir do LSB ou MSB. A posição encontrada será informada em operando destino.
  - Sintaxe:

BSF/BSR OP_DEST,OP_FONTE

BSF/BSR OP_DEST,OP_FONTE OP - Deve ser REG. ou memória de 16 ou 32 bits

EX.: BSF EAX,EBX

Lógica

- BT/BTC,BTS,BTR - Testa, complementa, seta ou reseta um bit do operando destino especificado no operando fonte (16 ou 32 bits).

  – Sintaxe

  BT/BTC,BTS,BTR OP_DEST,OP_FONTE

  EX.: BT AX,BX
  BTC AX,15
  BTS AX,1
  BTR AX,0

Lógica e Manipulação de Bit

Algumas Instruções Manipulação de bit

SHL/SAL

SHR

SAR

ROL

ROR

RCL

RCR

Manipulação de bit

- SHL/SAL – Realiza o deslocamento do conteúdo de um registrador à esquerda. O flag carry conterá o valor do último bit deslocado.
  - Sintaxe:

SHL/SAL OP_DEST, COUNT

EX.: SAL DX,1 ;Desloca o conteúdo de DX
uma posição a esquerda.
SAL BH,CL ;Desloca o conteúdo de BH
o número de vezes especificado em CL.

Manipulação de bit

- SHL
  - Ex.: MOV ax, 8000 ; 8000h=10000000-00000000

    SHL ax,1 ; 00000000-00000000

    ax = 0

Manipulação de bit

- SHR – Realiza o deslocamento do conteúdo de um registrador à direita. O flag carry conterá o valor do último bit deslocado.
  - Sintaxe:

SHR OP_DEST, COUNT

EX.: SAR DX,1 ;Desloca o conteúdo de DX uma posição a direita.

SAR BH,CL ;Desloca o conteúdo de BH o número de vezes especificado em CL.

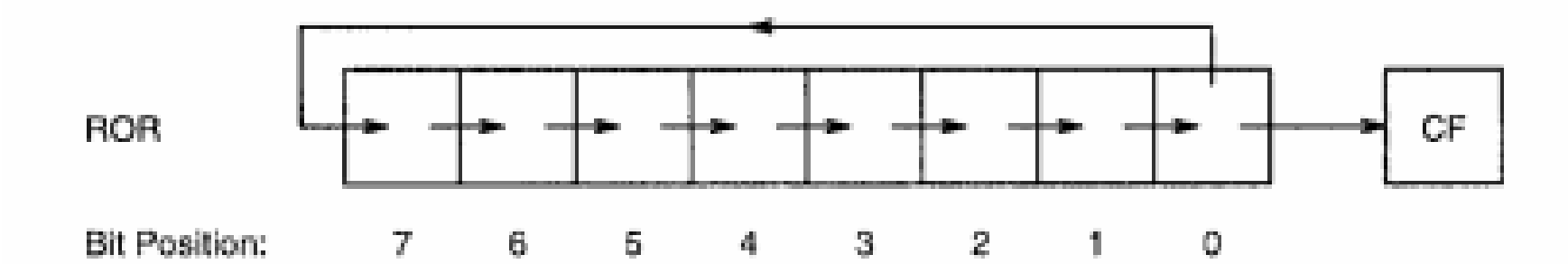
Manipulação de bit

- SHR
  - Ex.: MOV ax,1 ; 00000000 - 00000001
    SHR ax,1 ; 00000000 - 00000000
    ax = 0

Manipulação de bit

- ROL – Realiza o rotacionamento do conteúdo de um registrador à esquerda.
  - Sintaxe:

ROL OP_DEST, COUNT



Manipulação de bit

- RCL

EX.: ROL DX,1 ;Rotaciona o conteúdo de DX uma posição a direita.

ROL BH,CL ; Rotaciona o conteúdo de BH o número de vezes especificado em CL.

Manipulação de bit

- ROR – Realiza o rotacionamento do conteúdo de um registrador à direita.
  - Sintaxe:

ROR OP_DEST, COUNT

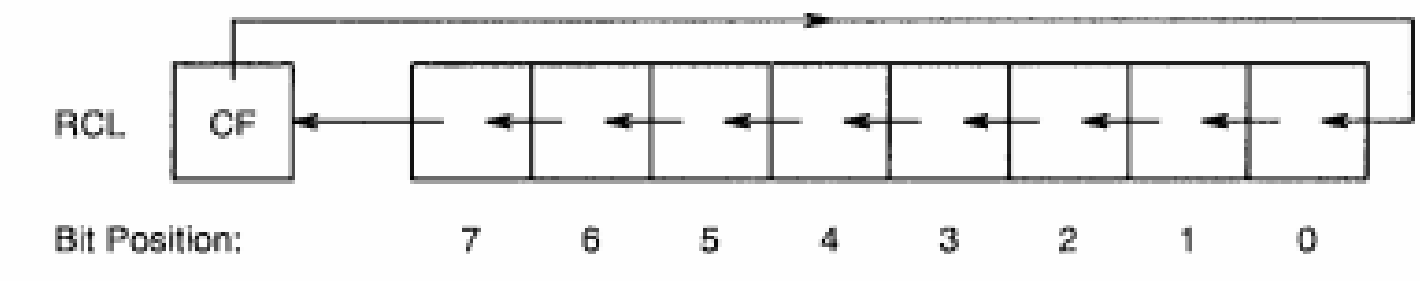
EX.: ROR DX,1 ;Rotaciona o conteúdo de DX uma posição a direita.

ROR BH,CL ; Rotaciona o conteúdo de BH o número de vezes especificado em CL.

Manipulação de bit

- RCR – Realiza o rotacionamento do conteúdo de um registrador à direita através do flag carry.
  - Sintaxe:

RCR OP_DEST, COUNT



Manipulação de bit

- RCL – Realiza o rotacionamento do conteúdo de um registrador à esquerda, através do flag carry.
  - Sintaxe:

RCL OP_DEST, COUNT

EX.: RCL DX,1 ;Rotaciona o conteúdo de DX uma posição a direita.

RCL BH,CL ; Rotaciona o conteúdo de BH o número de vezes especificado em CL.

Manipulação de bit

- RCR

EX.: RCR DX,1 ;Rotaciona o conteúdo de DX uma posição a direita.

RCR BH,CL ; Rotaciona o conteúdo de BH o número de vezes especificado em CL.

Laços, Desvios, Subrotinas e Interrupções

Laços e Desvios

JMP - Desvio Incondicional

Jcc - Desvio Condicional

LOOP - Laço

- **LOOPE** - ZF = 1
- **LOOPNE** - ZF = 0

Subrotinas e Interrupções

CALL - Subrotinas

INT - Interrupções

RET - Retorno de Subrotinas

IRET - Retorno de Interrupções

Laços e Desvios

- **Desvios Incondicionais** – São sempre executados.

- **Desvio Condicional** – Depende da combinação de estados das flags.

Laços e Desvios

- JMP – Realiza um desvio incondicional no fluxo de processamento, transferindo a execução para o endereço destino.
  - Sintaxe:

```
Label:  --------
        JMP Label

Ex.:    XOR al, al
        Mov al,10
        T:  INC al
        JMP  T
```

Laços e Desvios

- Jcc - Realiza um desvio no fluxo de processamento, transferindo a execução para o endereço destino, caso uma condição seja satisfeita.

  – Sintaxe:

Label: --------

Jcc Label

Laços e Desvios

CMP OP_DEST, OP_FONTE

JE – Desvia se OP_DEST for igual ao OP_FONTE;

JG – Desvia se OP_DEST for maior que o OP_FONTE;

JL – Desvia se OP_DEST for menor que o OP_FONTE;

JGE – Desvia se OP_DEST for maior ou igual ao OP_FONTE;

JLE – Desvia se OP_DEST for menor que o OP_FONTE;

JNE – Desvia se OP_DEST não for igual ao OP_FONTE;

MUL OP_FONTE

JZ – Desvia se zero (ZF = 1);

JNZ - Desvia se não zero (ZF = 0);

JC - Desvia se Carry (CF = 1);

JNG – Desvia se não Carry (CF = 0);

Laços e Desvios

- Jcc
  - Ex.: mov bx,10
    cmp ax,bx
    JE Label ;Desvia se OP_DEST for igual ao OP_FONTE;
    mov ax,10
    Label:

  - Ex.: mov ax,10
    cmp bx,ax
    JNE Label ;Desvia se OP_DEST não for igual ao OP_FONTE;
    mov bx,10
    Label:

Laços e Desvios

- LOOP – Decrementa o valor de CX e realiza um desvio no processamento para o endereço destino, caso CX ainda não for zero.
  - Sintaxe

LOOP end_destino

```
Ex.:    MOV cx,5
        XOR al,al
     T: INC al
        LOOP T
```

Laços e Desvios

- LOOPE – Decrementa o valor de CX e realiza um desvio no processamento para o endereço destino, caso CX ainda não for zero e ZF = 1.

  – Sintaxe

  LOOPE end_destino

  Ex.: T: -----

  LOOPE T

Laços e Desvios

- LOOPNE – Decrementa o valor de CX e realiza um desvio no processamento para o endereço destino, caso CX ainda não for zero e ZF = 0.

  – Sintaxe

  LOOPNE end_destino

  Ex.: T: -----

  LOOPNE T

Controle

CLC - Limpa CF

STC - Faz CF=1

CMC - Complementa CF

CLD - Limpa DF

STD - Faz DF=1

CLI - Limpa IF

STI - Faz IF=1

Controle

- HLT - Para o sistema. O processador fica neste estado até ocorrer um reset ou uma interrupção NMI ou INTR.
  - Sintaxe:

    HLT

    Exemplo.: HLT

Controle

- LOCK - Bloqueia o barramento.
  - Sintaxe:

    LOCK

    EX.: LOCK XCHG

- NOP – Não executa nada.

  - Sintaxe:

    NOP

Instruções

MOVS – Move String

CMPS - Compara String

SCAS – Busca String

LODS – Carrega String

STOS – Armazena String

Strings

- Prefixo Repetição.
  - Decrementa o registrador CX, sem afetar as flags, e repete a instrução de string, enquanto CX ≠ 0.

Ex.: REP instrução_string

Strings

- REP - Repete enquanto CX for diferente de zero. Deve ser usada antes das instruções MOVS, STOS e LODS

– Sintaxe:

REP instrução_string

EX.: REP MOVSB

Strings

- REPE/REPZ - Repete enquanto CX for diferente de zero e ZF = 1. Deve ser usada antes das instruções CMPS e SCAS.

– Sintaxe:

REPE/REPZ - instrução_string

EX.: REPE CMPSB

Strings

- REPNE/REPNZ - Repete enquanto CX for diferente de zero e ZF = 0. Deve ser usada antes das instruções CMPS e SCAS.

– Sintaxe:

REPNE/REPNZ - instrução_string

EX.: REPNE SCASW

Strings

- MOVS/ MOVSB/ MOVSW/ MOVSD - Move o elemento apontado por SI no segmento de dados para área apontada por DI no segmento extra. DI e SI serão incrementados ou decrementados de 1, 2, ou 4 dependendo do flag DF e do tipo do dado.

  – Sintaxe:

  MOVS/ MOVSB/ MOVSW/ MOVSD

  EX.:
  ```
  MOV SI,300
  MOV DI,200
  MOV CX,10
  CLD ; incrementa DI
  REP MOVSW
  NOP
  ```

Strings

- CMPS/ CMPSB/ CMPSW/ CMPSD - Compara o elemento apontado por DI no segmento extra com o elemento apontado por SI no segmento de dados. As flags são alterados de acordo com o resultado da subtração mas o resultado é desprezado. DI e SI serão incrementados ou decrementados de 1, 2, 0u 4 dependendo do flag DF e do tipo do dado.

  - Sintaxe:

CMPS/ CMPSB/ CMPSW/ CMPSD

```
EX.:   MOV SI,0300H
       MOV DI,200
         MOV CX,10
           CLD
                REP CMPSW
```

Strings

- SCAS/ SCASB/ SCASW/ SCASD - Compara o elemento apontado por DI no segmento extra com o conteúdo do acumulador (AL,AX ou EAX). O DI será incrementado ou decrementado de 1, 2, ou 4 dependendo do flag DF e do tipo do dado.
  - Sintaxe:

SCAS/ SCASB/ SCASW/ SCASD

```
EX.:    MOV AL,FF
        MOV DI,200
        MOV CX,A
        CLD
        REP SCASB
        NOP
```

Strings

- LODS/ LODSB/ LODSW/ LODSD - Carrega o elemento apontado por SI no segmento de dados no acumulador (AL,AX ou EAX). SI será incrementado ou decrementado de 1, 2, ou 4 dependendo do flag DF e do tipo do dado.
  - Sintaxe:

LODS/ LODSB/ LODSW/ LODSD

EX.: MOV AX,0510H
MOV SI,5H
CLD
LODSW

Strings

- STOS/ STOSB/ STOSW/ STOSD - Armazena na área apontada por DI no segmento extra o conteúdo do acumulador (AL,AX ou EAX). DI será incrementado ou decrementado de 1, 2, ou 4 dependendo do flag DF e do tipo do dado.

  – Sintaxe:

STOS/ STOSB/ STOSW/ STOSD

```
EX.:    XOR AX,AX
        MOV DI,100
        MOV CX,6H
        CLD
        REP STOSW
        NOP
```