Universidade de São Paulo

Organização de Computadoresres

Dr. Jorge Luiz e Silva Cap 3

Estrutura da Unidade de Processamento 8086/8088

14 Registradores

.PC →	Contador de programa	(16 bits)
.SP →	Ponteiro de Pilha	(16 bits)
.SP →	Índice Fonte	(16 bits)
.DI →	Índice destino	(16 bits)
.BP→	Ponteiro de Base	(16 bits)
.CS →	Segmento de Código	(16 bits)
.DS →	Segmento de Dados	(16 bits)
.SS →	Segmento de Pilha	(16 bits)
.ES →	Segmento Extra	(16 bits)

Registradores de Uso Geral

AH	AL	AX Acumulador
BH	BL	BX Endereço
CH	CL	CX Contador
DH	DL	DX Entra/Saída

Registrador de Status

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
				O	D	I	T	S	Z		A		P		C

O → Overflow
D → Direção
I → Interrupção abilitada /não abilitada
T → Trap
S → Sinal
Z → Zero
A → Carry
P → Paridade
C → Carry

Sinais do Registrador de Status

C → Carry (contém o carry de qualquer operação aritmética e deslocamento ou rotação).

A → Carry auxiliar (Contém o carry do bit 3 para 4 resultante da execução de uma instrução aritmética).

O → Overflow ($C \oplus C'$).

S → Sinal (contém o bit mais significativo do resultado seguinte à execução de qualquer instrução aritmética ou booleana).

P → Paridade (é 1 quando os 8 bits de menor ordem, resultante de uma operação qualquer contiver número par de 1's)

Z → Zero (1 quanto qualquer operação aritmética ou booleana gera um resultado zero. 0 caso contrário).

Sinais do Registrador de Status

D → Direção (Determina se uma operação auto-incrementa ou auto-decrementa o conteúdo dos registradores de índice.

1 – quando SI e DI serão decrementados

0 – quando SI e DI serão incrementados).

I →Interrupção habilitada / não habilitada

1 – habilitar interrupção

0 – quando todas interrupções estão desabilitadas

T → Trap (Flag de status especial que auxilia debugging do 8086 “passo-a-passo”).

Exemplos de sinais de Status

Operação aritmética

$0 \leftarrow 1 \leftarrow$

0100 0000 0000 1000 +

0100 0000 0000 1111

1000 0000 0001 0111

				O	D	I	T	S	Z		A		P		C
				1	X	X	X	1	0		1		0		0

Exemplos de sinais de Status

Operação Lógica

0← 0←

0011 0000 0000 1010 AND

1111 0000 0000 1100

0011 0000 0000 1000

				O	D	I	T	S	Z		A		P		C
				X	X	X	X	0	0		X		0		X

Modo de endereçamento no 8088/8086

Todo endereço da Memória é obtido na forma:

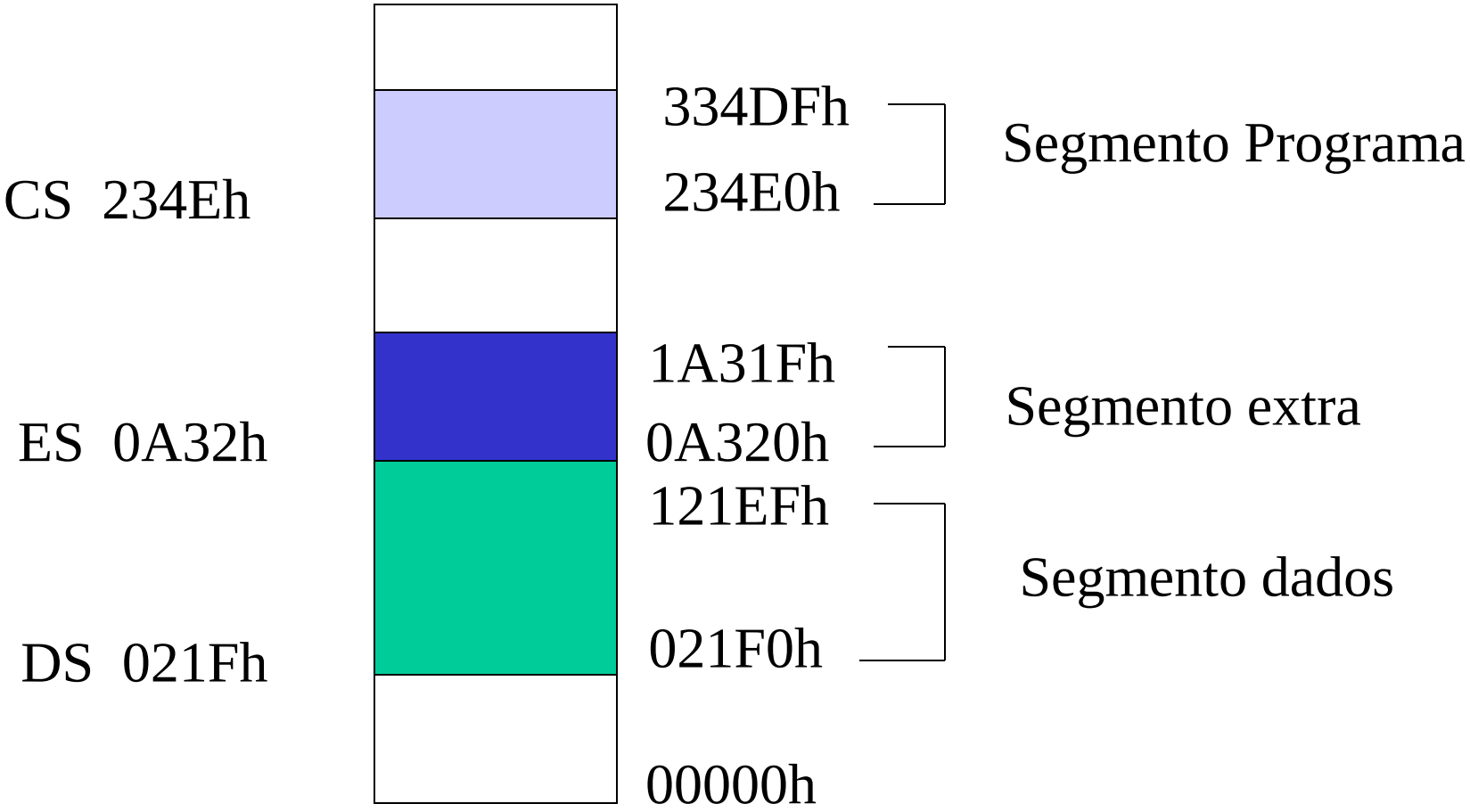
Registrador de Segmento: XXXX XXXX XXXX XXXX OOOO+
Endereço na Memória: OOOO YYYY YYYY YYYY YYYY

Endereço Real: KKKK KKKK KKKK KKKK KKKK

Endereço de Memória com 20 bits → 1 Mbytes de memória a ser endereçado diretamente.

O endereçamento do 8088/8086 é composto por dois endereços distintos: REGISTRADOR DE SEGMENTO (atua como registrador de base) + ENDEREÇO EFETIVO DA MEMÓRIA

Exemplos de segmentos



Algumas Instruções Básicas

MOV	DS,DX	8EDAH
ADD	AX, dado	81C0 ppqqH
ADD	AX, BX	01D8H
MOV	AX, dado	B8ppqqH
MOV	DX, dado	BAppqqH
MOV	AX, [end]	A1ppqqH
MOV	[end], AX	A3ppqqH
MOV	BX, AX	89C3H

Algumas instruções de Salto

JMP	disp	E8KK	jump em CS
JZ, JE	disp	74KK	jump zero
JNZ, JNE	disp	75KK	jump não zero
JL	disp	7CKK	jump less
JLE	disp	7EKK	jump lessequ
JG	disp	7FKK	jump grather
JNO	disp	71KK	jump ñ overflow
JO	disp	70KK	jump overflow

KK – para trás conta-se a partir dele até a instrução exclusive (ou seja, nº de bytes a se saltar até alcançar a instrução seguinte)

Programa Simples - P1

Fazer um programa que some dois elementos da memória e coloque o resultado em outra posição de memória.

Dados – segmento de dados (3000h)

dado1 – 1000h

dado2 – 1002h

Resultado – 2000h

Solução P1

MOV	DX, 3000	BA 3000
MOV	DS, DX	8E DA
MOV	AX, [1000]	A1 1000
MOV	BX, AX	89 C3
MOV	AX, [1002]	A1 1002
ADD	AX, BX	01 D8
MOV	[2000], AX	A3 2000

0011 0000 0000 0000 0000

0000 1000 0000 0000 0000

---

3 1 0 0 0

Atribuição de P1

01000	-	BA	01009	-	C3
01001	-	00	0100A	-	A1
01002	-	30	0100B	-	02
01003	-	8E	0100C	-	10
01004	-	DA	0100D	-	01
01005	-	A1	0100E	-	D8
04006	-	00	0100F	-	A3
01007	-	10	01010	-	00
01008	-	89	01011	-	20

Programa Simples - P2

Fazer um programa que some dois números da memória com 16 bits cada e coloque o resultado em outra posição da memória. No caso de ocorrer OVERFLOW coloque FFh co registrador CL para verificação posterior.

Segmento de dados – 3000

dado a - 1000

dado b - 1002

resultado - 2000

Solução P2

	MOV	CL,0	B1 00
	MOV	DX, 3000	BA 3000
	MOV	DS, DX	8E DA
	MOV	AX, [1000]	A1 1000
	MOV	BX, AX	89 C3
	MOV	AX, [1002]	A1 1002
	ADD	AX, BX	01 D8
	JNO	LA	71 KK
	MOV	CL, 0FF	B1 FF
LA:	MOV	[2000], AX	A3 2000

Atribuição de P2

01000	-	B1	0100C	-	A1
01001	-	00	0100D	-	02
01002	-	BA	0100E	-	10
01003	-	00	0100F	-	01
01004	-	30	01010	-	D8
01005	-	8E	01011	-	71
01006	-	DA	01012	-	02
01007	-	A1	01013	-	B1
01008	-	00	01014	-	FF
01009	-	10	01015	-	A3
0100A	-	89	01016	-	00
0100B	-	C3	01017	-	20

Programa Simples - P3

Fazer um programa que some dois elementos de 32 bits cada e coloque o resultado na memória.
Indique overflow CL – FF

Segmento de dados – 3000

dado a - 1000, 1002

dado b - 1004, 1006

resultado - 2000

Solução P3

MOV	CL, 0	B1 00
MOV	DS, 3000	BA 3000
MOV	DS, DX	8E DA
MOV	AX, [1000]	A1 1000
MOV	BX, AX	89 C3
MOV	AX, [1004]	A1 1004
ADD	AX, BX	01 D8
MOV	[2000], AX	A3 2000
JNC	LA	73 05
MOV	DX, 1	BA 0001

Solução P3 cont.

	JMP	SOMA	EB 03
LA:	MOV	DX, 0	BA 0000
	MOV	AX, [1002]	A1 1002
	MOV	BX, AX	89 C3
	MOV	AX, [1006]	A1 1006
	ADD	AX, BX	01 D8
	ADD	AX, DX	01 D0
	MOV	[2002], AX	A3 2002
	JNO	FIM	77 02
	MOV	CL, 0F	B1 FF
	JMP	S.O.	

Programa Simples - P4

Fazer um programa que some dois elementos de 32 bits cada e coloque o resultado na memória. Indique OVERFLOW CL-FF

Segmento de dados – 3000

dado a – 1000, 1002

dado b – 1004, 1006

resultado – 2000

Solução P4

	MOV	CL,0	B1 00
	MOV	DX, 3000	BA 3000
	MOV	DS,DX	8E DA
	MOV	AX,[1000]	A1 1000
	MOV	BX,[1004]	8B1E 1004
	ADD	AX,BX	01 D8
	MOV	[2000],AX	A3 2000
	MOV	AX,[1002]	A1 1002
	MOV	BX,[1006]	8B1E 1006
	ADC	AX,BX	11 D8
	JNO	LA	77 02
	MOV	CL, 0FF	B1 FF
LA:	MOV	[2000], AX	A3 2000

O Debug

Comandos

- ?

Inicialmente

- E end – insere bytes na memória
- D end – mostra a partir de end
- T end – executa end
- G end final – executa até end final
- A monta programas direto do mnemônico
- U end – mostra programa a partir de end

Debug – Comando A

Monta programa direto em mnemônico

-A end de cs:inic ou

-A inic (assume o CS anterior)

- (enter encerra a entrada)

Ex: - a100

- 1234:100 mov ax,[1000]

- 1234:103 mov bx,[1002]

- enter

Debug - comando U

Unassembly (mostra o programa em instruções mnemônico)

- U end inicial end final

Ex: -u100

-1234:100 mov ax,[1000]

-1234:103 mov bx,[1002]

Debug - comando E

Comando de Entrada de dados em hexadecimal

- E reg seg:end

Ex.:

-E DS:200

-1234:0200 00.10 00.10 00.12....

Espaço entre os dados e encerra com um enter

Debug - comando D

Mostra conteúdo de memória

-D end inicial end final

-D CS:end inicial end final
(especificarqualquer CS)

Ex: -d100

-1234:100 00 00 00 00 00 00....

-1234:110 00 00 00 00 00 00

Debug - comando G

Comando de execução de programa

-G (executa a partir de ponteiro de instrução corrente)

-G = end inicial end fina

-G end final

Ex: -g=100 103 (executa de 100 até 103 e volta ao debug)

Debug - comando T

Comando para depurar programas

-T=end inicial nº. de instruções

Ex:

-t=100 20 (vai executar, a partir de 100, as proximas 20 instruções e parar)

Debug - comando R

Mostra conteúdo de Registrador

-R nome

nome valor

: (espera por novo valor)

-R (mostra o conteúdo de todos os registradores)

Ex: -r cs

cs 1234

:

Debug - comando P

Comando P executa rotinas inteiras sem considerar instrução por instrução como é o caso do comando T (trace)

Ex: -1234:100 int21

-p=100 (vai executar a rotina inteira até o retorno)

Debug - comando W

- N especifica um nome de arquivo para disco
- W grava o programa especificado nor N posicionado na posição 100H da memória, tantos bytes quantos estiverem especificados por: BX:CX

Ex: Gravando um programa com 0100 bytes posicionando a partir do endereço 100h da memória

```
-N teste.com
-R bx
bx:0000
-R cx
cx:0100
-w
```

Debug - comando L

-L carrega o programa especificado por N posicionando-o na posição 100H da Memória e carrega BX:CX bytes do programa.

Ex: Carregando um programa com 0100 bytes a ser posicionado a partir do endereço 100h da Memória

```
-N teste.com
-R bx
bx:0000
-R cx
cx:0100
-L
```

Debug - comando Q

Comando de retorno ao D.O.S.

-Q