Universidade de São Paulo

Organização de Computadores

Dr. Jorge Luiz e Silva Cap 4

Generalizar
Problema - G1

Fazer um programa que calcule o complemento para dois de uma posição de memória e coloque o resultado em outra posição de memória.

Segmento de dados 3000

dado – 1000

resultado – 1002

Solução comum - G1

MOV	DX,3000	BA 3000
MOV	DS,DX	8E DA
MOV	AX,[1000]	A1 1000
NEG	AX	F7 D8
MOV	[1002],AX	A3 1002

Solução Generalizada - G1

MOV DX,3000	BA 3000
MOV DS,DX	83 DA
MOV BX, 1000	BB1000
MOV AX,[BX]	8B,07
INC BX	FF C2
INC BX	FF C2
NEG AX	F7 D8
MOV [BX],AX	89 07
INT 20H	

Problema - G2

Calcular o complemento para dois de um dado de 24 bits em posição de memória. Calcular o resultado na memória.

seg. de dados – 3000

dado – 1000	resultado - 1003
1001	1004
1002	1005

Solução - G2

MOV	DX,3000	BA3000
MOV	DS,DX	8E DA
MOV	DL,0	B2 00
MOV	BX,1000	BB 1000
MOV	AX,[BX]	8B 07
INC	BX	FF C2
INC	BX	FF C2
NEG	AX	F7 D8
MOV	CX,AX	8AC1
JNC	LA	73 02
MOV	DL,1	B201

Solução - G2 cont.

LA:	MOV	AL,[BX]	8A07
	INC	BX	FF C2
	NEG	AL	F6D8
	SUB	AL,1	2C01
	ADD	AL,DL	00D0
	MOV	[BX],CX	8B0F
	INC	BX	FF C2
	INC	BX	FF C2
	MOV	[BX],AL	8A 07
	INT 20H		

Problema - G3

Encontrar o maior elemento entre dois dados da memória

Seg de dados – 3000

dado a – 1000

dado b – 1002

resultado – 1004

Solução - G3

```
        MOV   DX,3000
        MOV   DS,DX
        MOV   BX,1000
        MOV   AX,[BX]
        INC   BX
        INC   BX
        CMP   AX,[BX]
        JG    LA
        MOV   AX,[BX]
LA:     INC   BX
        INC   BX
        MOV   [BX],AX
        INT 20H
```

Problema - G4

Multiplicar dois dados da memória (8 bits)

Seg. de dados – 3000

dado a – 1000

dado b – 1001

resultado – 1002

Solução - G4

```
MOV   DX,3000
MOV   DS,DX
MOV   BX,1000
MOV   AL,[BX]
INC   BX
MOV   CL,[BX]
MUL   CL
INC   BX
MOV   [BX],AX
INT 20H
```

Problema - G5

Multiplicar dois dados da memória (16 bits)

seg. de dados – 3000

dado a – 1000

dado b – 1002

resultado – 1004

Solução - G5

```
MOV   DX,3000
MOV   DS,DX
MOV   BX,1000
MOV   AL,[BX]
INC   BX
MUL   CX
INC   BX
MOV   CX,[BX]
MUL   CX      DX:AX ← AX*CX
INC   BX
INC   BX
```

Solução - G5 cont.

```
MOV   [BX],AX
INC   BX
INC   BX
MOV   [BX],DX
INT 20H
```

Programas usando LOOP
Programa - L1

Fazer um programa em Assembly que coloque o conteúdo do acumulador em posições sucessivas da memória e vá decrementando seu valor até que ele fique zero.

segmento de dados – 3000h

dado a partir de – 1000h

Solução - L1

	MOV	DX,3000	BA3000
	MOV	DS,DX	8E DA
	MOV	BX,1000	BB1000
	MOV	AL,20	B020
LA:	MOV	[BX],AL	8A07
	INC	BX	FF C2
	DEC	AL	FE C8
	JNZ	LA	75 F8
	INT 20H		

Programa - L2

Fazer um programa que some um conjunto de dados de 16 bits tal que:

tamanho da série: posição 1002

dados a partir da posição 1004

resultado na posição 1000

segmento de dados 3000

Solução - L2

	MOV	DX,3000	BA 3000
	MOV	DS,DX	8E DA
	MOV	BX,1002	BB 1002
	MOV	CX,[BX]	89 0F
	SUB	AX,AX	29C0
LA:	INC	BX	FF C2
	INC	BX	FF C2
	ADD	AX,[BX]	01 07
	DEC	CX	FF C9
	JNZ	LA	75 F6
	MOV	[1000],AX	A3 1000
	INT 20H		

Solução - L2 com loop

DEC CX e JNA LA $\rightarrow$ LOOP LA

```
        MOV   DX,3000        BA 0030
        MOV   DS,DX          8E DA
        MOV   BX,1002        BB 0210
        MOV   CX,[BX]        89 0F
        SUB   AX,AX          29C0
LA:     INC   BX             FF C2
        INC   BX             FF C2
        ADD   AX,[BX]        01 07
        LOOP  LA             E2 F8
        MOV   [1000],AX      A3 0010
        INT 20H
```

Problema - L3

Fazer um programa que encontre o maior em um conjunto de dados positivos de 16 bits tal que:

tamanho da série: posição 1002

dados a partir da posição 1004

resultado na posição 1000

segmento de dados 3000

Solução - L3

	MOV	DX,3000	BA 0030
	MOV	DS,DX	8E DA
	MOV	BX,1002	BB 0210
	MOV	CX,[BX]	89 0F
	SUB	AX,AX	29C0
T2:	INC	BX	FF C2
	INC	BX	FF C2
	CMP	AX,[BX]	01 07
	JG	T1	
	MOV	AX,[BX]	
T1:	LOOP	T2	E2 F8
	MOV	[1000],AX	A3 0010
	INT	20H	

Modificação de Endereços e Operações Lógicas

15 12	11 10	9 8	7 0
COD	REG	TE	----END----

TE	REG	X
00 – Imediato	00 – A	indexação
01 – Direto	01 – B	
10 – Indireto	10 – C	
11 – Indexado	11 – D	

Endereçamento Imediato

O dado faz parte da própria instrução

LOAD A,10 0000 00 00 00001010

Vantagem – O carregamento do dado é feito imediatamente durante a fase de execução da instrução, sem Ter que buscar o dado → menor tempo de ciclo da instrução pois não é necessário busca do dado.

Desvantagem – Os dados ficam limitados somente entre os valores : +127 a – 128, não é possível trabalhar com 16 bits.

Endereçamento Direto

O dado está contido na posição de memória apontada por end

LOAD A,[10] 0000 00 01 0000 1010

O ciclo de instrução será: busca da instrução decodificação busca do dado execução

Tempo de execução é maior

Vantagem – Dado é representado por 16 bits.

Desvantagem – Tempo maior e o dado só pode ser acessado em uma área única de memória.

Endereçamento Indireto

O dado está contido na posição de memória cujo endereço está contido na posição de memória apontado por end.

LOADI A,[10] 0000 00 10 0000 1010

O ciclo de instrução será: busca da instrução decodificação busca do endereço busca do execução

Vantagem – é possível acessar qualquer posição de memória

Desvantagem – mais lento

Endereçamento Indexado

O dado está na posição de memória cujo endereço é obtido pela soma do end e registrador indexador.

LOADX A,[10] 0000 0011 0000 1010

endereço é obtido na forma:

X	xxxx xxxx xxxx xxxx +
End	yyyy yyyy yyyy yyyy
	----------
efetivo	hhhh hhhh hhhh hhhh

O ciclo de instrução será: Busca da instrução, Decodificação, Calculo do endereço, Busca do dado e Execução.

Vantagem – utilizar registrador de índice para modificar o endereço

Desvantagem – tempo de acesso