AC

Ficha 1

$X_{(2)} = 10010100$ $148_{(10)}$
$X_{(8)} = 377$ $255_{(10)}$
$X_{(16)} = 100$ $256_{(10)}$

$X_{(10)} = 77$ $1001101_{(2)}$
$X_{(8)} = 316$ $11001110_{(2)}$
$X_{(16)} = FF$ $11111111_{(2)}$

$X_{(2)} = 1000001$ $101_{(8)}$
$X_{(10)} = 15$ $17_{(8)}$
$X_{(16)} = 16$ $26_{(8)}$

$X_{(2)} = 10001$ $11_{(16)}$
$X_{(8)} = 33$ $1B_{(16)}$
$X_{(10)} = 46$ $2E_{(16)}$

$X_{(2)} = 1001_{(2)} + 111_{(2)} = 10000$
$X_{(2)} = 1001_{(2)} + 11111_{(2)} = 101100$
$X_{(2)} = 1101_{(2)} + 1111_{(2)} = 11100$

$X_{(8)} = 15_{(8)} + 13_{(8)} = 30$
$X_{(8)} = 15_{(8)} + 73_{(8)} = 110$
$X_{(8)} = 17_{(8)} + 43_{(8)} = 62$

$X_{(16)} = 1A_{(16)} + 1D_{(16)} = 37$
$X_{(16)} = A1_{(16)} + EF_{(16)} = 190$
$X_{(16)} = 1AF_{(16)} + 1D_{(16)} = 1CC$

$X_{(256)} = 192.168.0.0_{(256)} + 128_{(10)} = 192.168.0.128$
$X_{(256)} = 192.168.0.0_{(256)} + 254_{(10)} = 192.168.0.254$
$X_{(256)} = 192.168.0.0_{(256)} + 256_{(10)}$ 192.168.1.0
$X_{(256)} = 192.168.0.0_{(256)} + 512_{(10)}$ 192.168.2.0

Ficha 2

1

→ Em complemento para 1 trocam-se os 0 por 1 e os 1 por 0, e determina-se os valores positivos, colocando o - posteriormente

Código binário	Código binário natural	Sinal e valor absoluto	Complementos para 1	Complementos para 2
0000	0	0	0	0
0001	1	1	1	1
0010	2	2	2	2
0011	3	3	3	3
0100	4	4	4	4
0101	5	5	5	5
0110	6	6	6	6
0111	7	7	7	7
1000	8	-0	-7	-8
1001	9	-1	-6	-7
1010	16	-2	-5	-6
1011	11	-3	-4	-5
1100	12	-4	-3	-4
1101	13	-5	-2	-3
1110	14	-6	-1	-2
1111	15	-7	-0	-1

0111 (6)

2

Nº de Bits	Código binário natural		Sinal e valor absoluto		Complementos para 1		Complementos para 2	
	Máximo	Mínimo	Máximo	Mínimo	Máximo	Mínimo	Máximo	Mínimo
5	11111	00000	01111	11111	01111	10000	01111	10000
7	1111111	0000000	0111111	1111111	0111111	1000000	0111111	1000000
8	11111111	00000000	01111111	11111111	01111111	10000000	01111111	10000000

3

3 bits

Nº Inteiro	Sinal e valor absoluto	Complementos para 1	Complementos para 2
+3	011	011	011
+2	010	010	010
+1	001	001	001
0	000	111/000	000
-1	101	110	111
-2	110	101	110
-3	111	100	101
-4	Não Representável.	Não Representável	100

4

Nº Inteiro	Nº mínimo de bits		
	Sinal e valor absoluto	Complementos para 1	Complementos para 2
-64	8	8	7
+128	9	9	8
-16	6	6	5
+12	5	5	5
-250	9	9	9

5

5) Efectue, em complementos para 1 e complementos para 2, com 6 bits, as somas algébricas seguintes:

a) (-24) + (-7)
b) (-31) + (+15)
c) (-23) + (-9)
d) (+13) + (+20)
e) (+18) + (-23)
f) (-3) + (+30)

a)

C2

$24_{(10)} = 11000_{(2)}$

C1 ↳ 100111

$7_{(10)} = 111_{(2)}$

↳ 111 000 C1

```
  100111
+ 111000
--------
 1011111
+      1
--------
  100000
```

$011111_{(2)} = 31_{(10)}$ (C1)

$-31_{(10)}$

C2

$-24_{(10)} = 101000$

$-7_{(10)} = 111001$

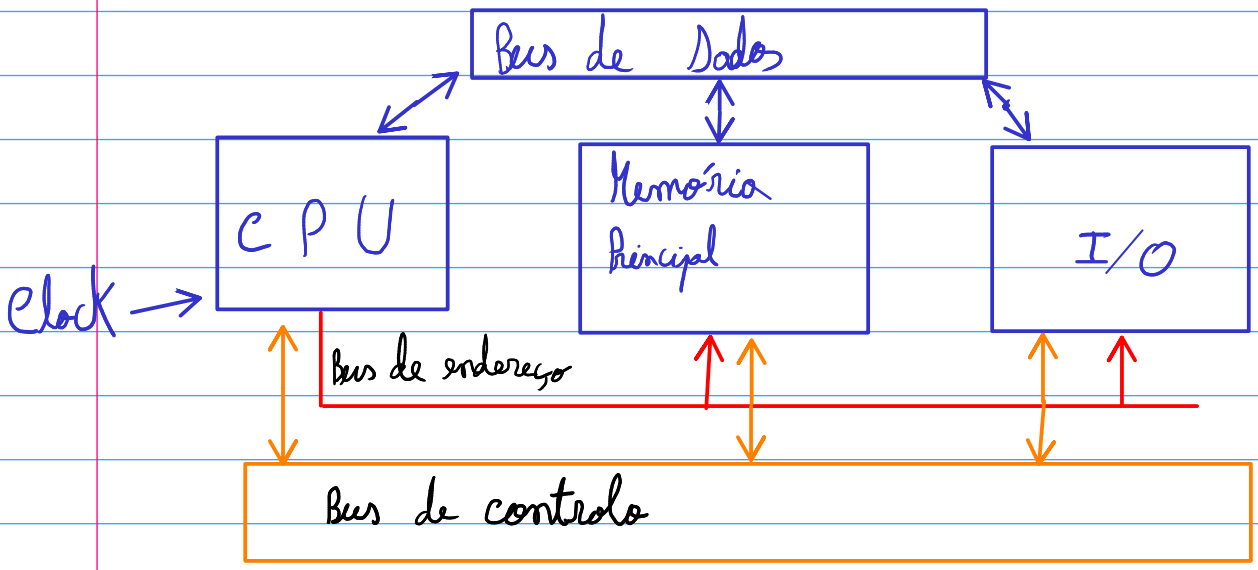
```
  101000
  111001
X 100000
```

↳ C2 $011111_{(2)}$

$-31_{(10)}$ $31_{(10)}$

1- 27/2/2023

Modelo de Arquitetura de Von Neumann



Clock → Define a frequencia de operação do sistema. É utilizado para efetuar a sincronização das operações

CPU → Unidade central de processamento (central proces unit) É o "cerebro" do computador. Lê instruções de memória, executa instruções, lê dados e escreve resultados

Memória Principal → É capaz de armazenar informação binária, normalmente organizada em celulas de 8 bits. Armazena dados, instruções e resultados. Dividida entre a RAM e a ROM

Unidades I/O → Utilizadas para estabelecer uma comunicação com o mundo exterior, como teclado, rato, etc.

Bus de Sistema → conjunto de linhas (ligações) que transportam informação. Permite a comunicação entre o CPU, Memória e Periféricos

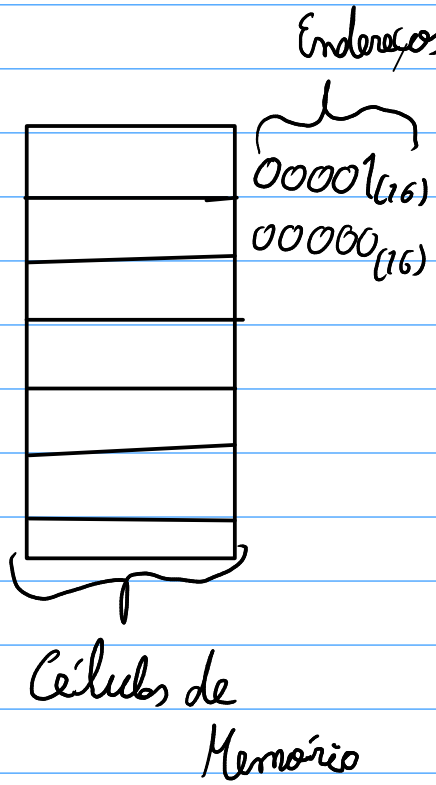
Bus de Dados → conjunto de ligações físicas que transportam informação entre o CPU, Memória e Periféricos. A largura do Bus de Dados é dada pelo nº de linhas do bus, ou nº de bits do microprocessador.

Bus de Endereços → Conjunto de ligações físicas que transportam o endereço das células de memória ou dos portos de I/O (unidirecional)
A largura do bus de endereço é dada pelo nº de linhas do bus ou pelo nº de bits, e define a capacidade de endereçamento.

Bus de controlo → Contém os sinais necessários para uma correta implementação do protocolo de comunicação.

Armazenamento da Informação Digital Binária

A organização da memória é feita em celulas.
As celulas normalmente têm 8 bits
Cada célula tem 1 endereço



Os bytes menos significativos são armazenados em endereços menos significativos.

- byte - 8 bits
- Word - 16 bits
- dWord - 32 bits

Estrutura interna do CPU (microprocessador) 8086

Registos de uso genérico

Ax registo de 16 bits
AH registo de 8 bits (8 bits + significativos)
AL registo de 8 bits (8 bits - significativos)
Registo acumulador
Implicito em algumas instruções

Bx registo de 16 bits
BH registo de 8 bits (8 bits + significativos)
BL registo de 8 bits (8 bits - significativos)
Registo de base
Normalmente utilizado para endereçar variaveis em memória

Cx registo de 16 bits
CH registo de 8 bits (8 bits + significativos)
CL registo de 8 bits (8 bits - significativos)
Registo acumulador
Implícito em algumas instruções como contador

Dx registo de 16 bits
DH registo de 8 bits (8 bits + significativos)
DL registo de 8 bits (8 bits - significativos)
Registo de Dados
Utilizado em algumas operações aritméticas
Utilizado em instruções de I/O

Registo de endereçamento

SP registo de 16 bits
Stack Pointer
Utilizado para referenciar variaveis na pilha do sistema

BP registo de 16 bits
Base Pointer
Utilizado para referenciar parâmetros e variáveis locais em subrotinas

Registos de indexação

SI registo de 16 bits
Source Index

DI registo de 16 bits
Destination Index

Registos Especiais

Registo apontador de instrução

Instruction Pointer (IP)
Registo de 16 bits
contém o endereço da próxima instrução a ser executada

Registo de Segmentos

Neste processador existem 4 segmentos:

CS - "Code Segment" - Segmento de Código - Armazena as instruções do programa

CS:[IP] - Ponteiro para a próxima instrução a ser executada

DS - "Data Segment" - Segmento de dados
armazena dados/resultados relativos a variáveis
DS:<deslocamento>

SS: "Stack Segment" - Segmento de Pilha

Suporte à Programação estruturada; Passagem de Parâmetros

SS:<deslocamento>

ES - "Extra Segment" - Segmento extra

Segmento auxiliar, utilizado por exemplo na manipulação de algumas cadeias de caracteres.

Segmentos de Memória

Endereço Lógico

O endereço lógico pode ter os seguintes formatos:

<segmento>: [<deslocamento>]
<segmento>: [<offset>]
<segmento>: [<displacement>]

→ Endereço Efetivo

Endereço físico/linear

O endereço físico/linear pode ser determinado da seguinte forma:

Segmento × 16 // multiplica um nº hexadecimal por 16 = acrescentar um "0"

\+ deslocamento

Endereço físico/linear

Ex

2000:1F00

20000

\+ 1F00 → Endereço Efetivo

21F00 → Endereço em Memória

ES - Extra Segment

Início $52B90_{(16)}$

$52B90_{(16)} + FFFF_{(16)}$

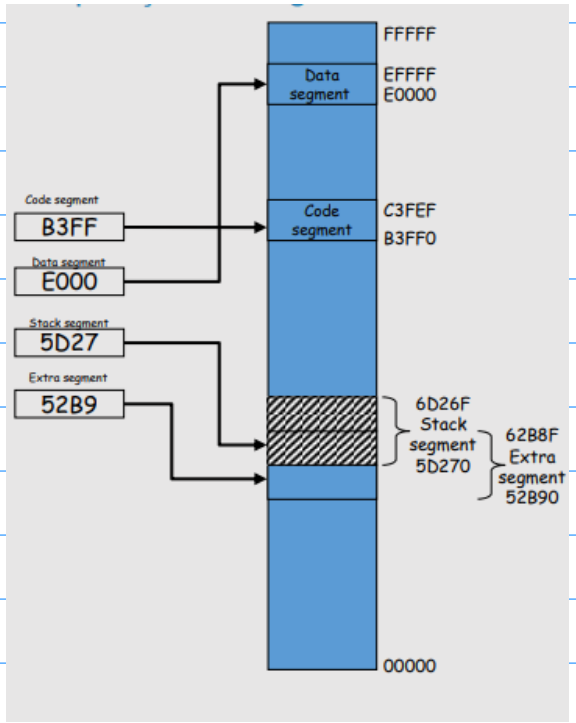
Fim $62B8F_{(16)}$

SS - Stack Segment

Início $5D270_{(16)}$

$5D270_{(16)} + FFFF_{(16)}$

Fim $6D26F_{(16)}$



Ciclo do Processador - fetch

O CPU procura as instruções na memória principal uma por uma.

Ciclo do Processador - decode

O CPU descodifica a instrução presente no program counter.

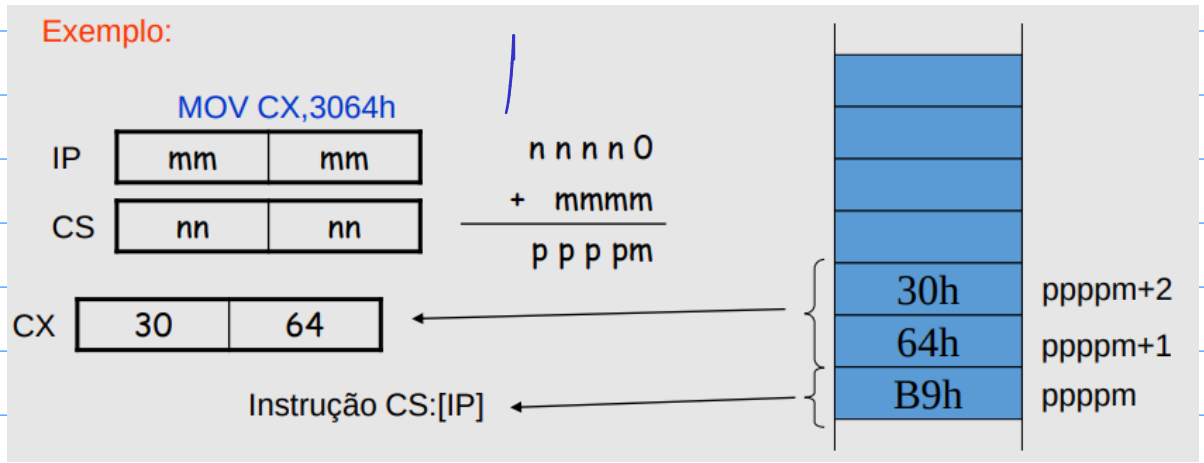
Ciclo do Processador - executa

O CPU executa a instrução anteriormente descodificada

Modos de endereçamento no 8086

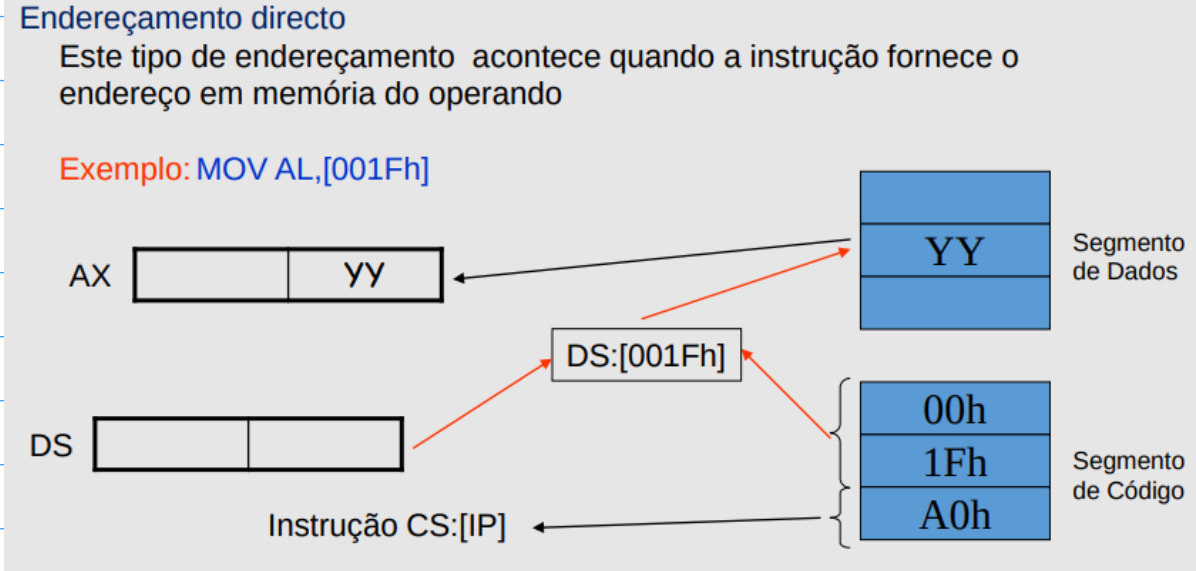
Endereçamento imediato

Este tipo de endereçamento acontece quando o operando está contido na própria instrução.



Endereçamento direto

Este tipo de endereçamento acontece quando a instrução fornece o endereço em memória do operando



Endereçamento Indireto

Este tipo de endereçamento acontece quando o endereço do operando é obtido por:

Registo de base + deslocamento: [BX + displ] ou [BX]displ ou displ[BX] ou...

Registo de indexação: [SI] ou [DI]

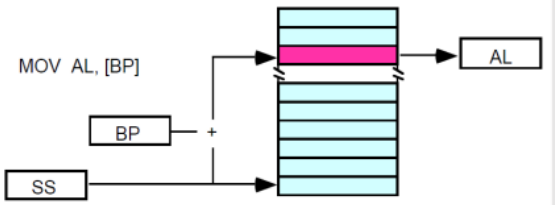
Registo de indexação + deslocamento: [SI + displ] ou [DI + displ] ou...

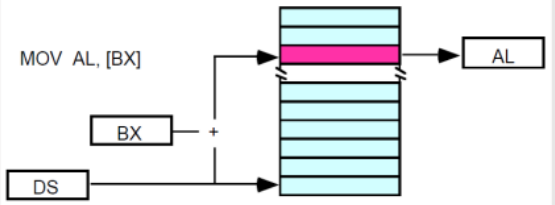
Registo de base + registo de indexação [BX + SI + displ] ou...

Endereçamento Indirecto

```
MOV AL, [BX]
MOV AL, [BP]
MOV AL, [SI]
MOV AL, [DI]
```

- [BX], [SI] e [DI] usam o segment DS por defeito
- [BP] usa o segment SS por defeito





Instruções Aritméticas

Instrução		Descrição	Bits de estado(Flags) afectados
ADD	a, b	a ← a + b	Z, C, O, N
ADC	a, b	a ← a + b + C	Z, C, O, N
NEG	a	a ← - a	Z, C, O, N
SUB	a, b	a ← a - b	Z, C, O, N
SBB	a, b	a ← a - b - C	Z, C, O, N
MUL	b	ax ← al * b } { dx:ax ← ax*b	Z, C, O, N
DIV	b	ax ← ax / b } { dx:ax ← dx:ax / b	Z, C, O, N
INC	a	a ← a + 1	Z, O, N
DEC	a	a ← a - 1	Z, O, N

Diferença entre MUL e IMUL

MUL é apenas a números inteiros > 0

IMUL é para todos os números inteiros

Multiplicação de 8 bits

<origem>

* AL

16 bits AH AL

Multiplicação de 16 bits

<origem>

* AX

32 bits DX AX

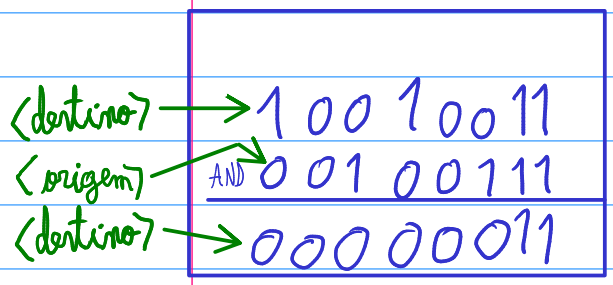
Tipos de instruções de acordo com a função que realizam

AND	a, b	$a_i \leftarrow a_i \wedge b_i$ (i ∈ 0..N-1)
OR	a, b	$a_i \leftarrow a_i \vee b_i$ (i ∈ 0..N-1)
XOR	a, b	$a_i \leftarrow a_i \oplus b_i$ (i ∈ 0..N-1)
NOT	a	$a_i \leftarrow \overline{a_i}$ (i ∈ 0..N-1)

Operadores Lógicos bit a bit

AND // multiplicação

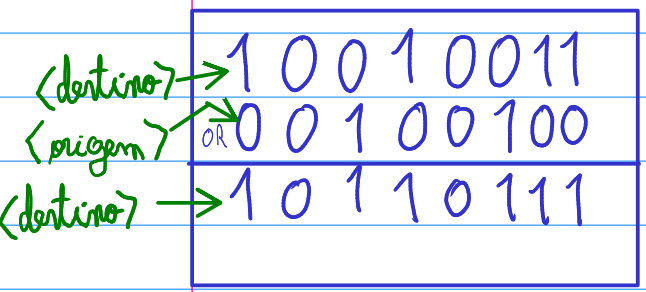
Sintaxe: AND <destino> <origem>



Neste exemplo vemos que na operação AND, o 0 é o elemento absorvente, e o 1 é o elemento neutro

OR

Sintaxe: OR <destino> <origem> // soma



Neste exemplo vemos que na operação OR o 1 é o elemento absorvente, e o 0 é o elemento neutro

XOR

Sintaxe: XOR <destino> <origem> // quando são os 2 iguais é 0 senão é 1

<destino> → 10 111 011
<origem> → XOR 00 001 111
<destino> → 01 001 110

Neste exemplo vemos que na operação XOR o 1 complementa, e o 0 é o elemento neutro

NOT

Sintaxe: NOT <destino> // inverter os bits 1→0 e 0→1

<destino> → NOT 10 110 101
<destino> → 01 001 010

Neste exemplo vemos que na operação XOR o 1 passa a 0 e o 0 passa a 1

P - 2/3/2023

6) Considere os seguintes códigos binários:
a) $1011_{(2)}$ b) $1BA_{(16)}$ c) $10_{(16)}$ d) $657_{(8)}$ e) $FF_{(16)}$
Determine os números inteiros representados em complemento para 2.

a)

$-2^3 \quad 2^1 \quad 2^0$
1 0 1 1

$-8+0+2+1 = -5$

b) $1BA_{(16)} = 110111010_{(2)}$

$-2^8 \; 2^7 \; 2^6 \; 2^5 \; 2^4 \; 2^3 \; 2^2 \; 2^1 \; 2^0$
1 10 111010

$-256+128+32+16+8+0+2+0 = -70$

c) $10_{(16)}$

$10_{(16)} = 00010000_{(2)}$

$-2^4\ 2^3\ 2^2\ 2^1\ 2^0$

00010000

$-2^4 = +16$

d) $657_{(8)} = 110\ 101\ 111_{(2)}$

$-2^8+2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

$-256+128+32+8+4+2+1 = -85$

e) $FF_{(16)} = 11111111_{(2)}$

$= 1$

7) Considere os seguintes códigos binários de 4 bits:

a) $1100_{(2)}$ b) $7_{(16)}$ c) $10_{(8)}$ d) $0000_{(2)}$

a) $1100_{(2)} \longrightarrow 11111100_{(2)}$

4 bits

b) $7_{(16)} = 0111 \rightarrow 00000111$

c) $10_{(8)} = 1000 \rightarrow 11111000$

d)

$0000_{(2)} \rightarrow 00000600$

8)

8) Considere os seguintes códigos binários de 8 bits:
a) $D1_{(16)}$ b) $FA_{(16)}$ c) $0B_{(16)}$ d) $C9_{(16)}$

a)

$D1_{(16)} = 11010001 = -47$, Não Representável

b) $FA_{(16)} = 11111010 = -6$ Representável

c) $0B_{(16)} = 0000\ 1011 = +11$ Não Representável

d) $C9_{(16)}$ $1100\ 1001 = -55$ Não Representável

---

considerando $ES = A020_{16}$, $BX = FF20_{16}$

Endereço: Es:[BX] + 20

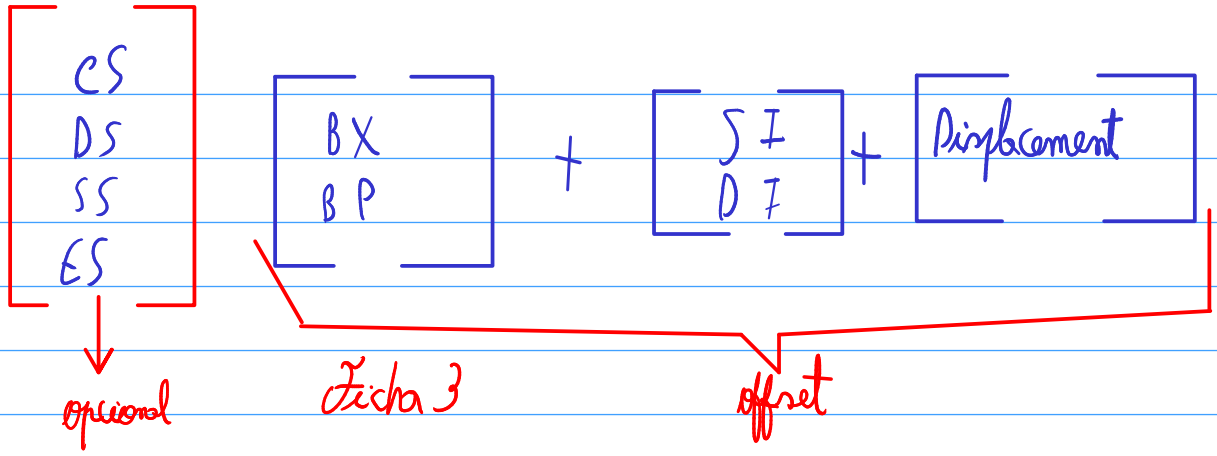
Endereço Efetivo (EE) $EE = [BX] + 20 = FF20_{16} + 14_{16} = FF34_{16}$

Endereço Lógico: $A020_{16} : FF34_{16}$

Endereço em memória: $EM = \text{segmento} \times 16_{16} + \text{offset} =$
$= A0200 + FF34 = B0134_{(16)}$

$$\begin{array}{r} 2 \\ +\ F \\ \hline 11 \end{array}$$

$$\begin{array}{r} 11 \\ A0200 \\ +\ FF34 \\ \hline B0134 \end{array}$$



1) Supondo que CS=1000h, DS=1200h, SS=0F00h, ES=14F0h, IP=0010h, BX=120Eh, BP=340Fh, SI=A000h e DI=3000h.

a) Determine o endereço em memória (endereço físico) da próxima instrução a ser executada.

b) Determine o endereço efectivo e o endereço em memória dos seguintes operandos:

[2000h]	ES:[2000h]	
[BX]	[BP]	ES:[BX]
[BX][2000h]	[BX+2]	ES:[BX][40h]
[BP][2000h]	[BP+10]	ES:[BP][40h]
[SI]	[DI]	ES:[SI]
[SI][0100h]	[DI+20]	ES:[SI+10h]
[BX][SI]	[BX+DI]	ES:[BX+SI]
[BX][SI][0100h]	[BX+DI+32]	ES:[BX+SI+32]

a) $CS[IP]$

$CS \Rightarrow 1000_{(16)}$

$IP \Rightarrow 0010_{(16)}$

$CS \times 10_{(16)} + IP = 1000 + 0010 = 1010_{(16)}$

↳ é o endereço em memória

b)

$EE \Rightarrow 2000_{(16)}$

$EM \Rightarrow DS:2000_{16} = DS \times 10 + 2000 = 1200 \times 10 + 2000 =$

$= 14000_{(16)}$

ES: $[2000_{16}]$

EI → 2000_{16}

EM → $ES:2000_{(16)} = ES \times 10 + 2000 = 14F0 \times 10 + 2000 = 14F00 + 2000 = 16F0_{(16)}$

EE → $120E_{(16)}$

EM → $1200 \times 10 + 120E = 12000 + 120E = 132E_{(16)}$

EE → $340F_{(16)}$

EM → $SS:BP = SSX + BP = 0F00 \times 10 + 340F = 0F000 + 340F = 1240F$

$$\begin{array}{r} 0F00 \\ +\ 340F \\ \hline 1240F \end{array}$$

EE → $120E + 2000_{(16)} = 320E_{(16)}$

EM → $1200 + 120E + 2000 = 3320E_{(16)}$

EE → $340E_{(16)} + 2000 = 540F_{(16)}$

EM → $0F00_{(16)} \times 10 + 540F_{(16)} = 0F000 + 540F = 1440F$

$$\begin{array}{r} 0F000 \\ +\ 540F \\ \hline 1440F \end{array}$$

[BP+10]

$10_{(10)} \rightarrow A_{(16)}$

EE → 340F+A = 3479(16)

$$\begin{array}{r} {}^{1}A \\ 340F \\ \hline 3479 \end{array}$$

$$\begin{array}{r} 0F000 \\ 3479 \\ \hline 12479 \end{array}$$

EM → SS: 3479 = 0F000 + 3479 = 12479

[BX+DI+32]

$32_{(10)} = 20_{(16)}$

EE = 100E+3000+20 = 4220$_{16}$

EM = DS:[BX+DI+20] = 1000x10+4220 = 2000+4220 = 6220$_{(16)}$