LAB 3

i8086

16 bits para registros ⇒ 64K

can access to 1MB ⇒ 2^{20} memory size dividiendo en trozos de 64K

El programa debe saber 2 cosas

data y code segment address

BaseRegister x 10h + offset

tengo DS, XXXX

lo multiplico por 10h

XXXX0

se sumo el offser YYYYY

me da como resultado

ZZZZZ

Segment registers are:

- CS: code segment register
- DS: data segment register
- SS: stack segment register
- ES: extra segment (data) register

Addressing modes		µP 8086/88	Examples
Inmediate		Inmediate	MOV AX,15h
Direct	To register	Register	MOV AX,BX
	To memory	(with segmentation)	
	To page	Direct	MOV CX,ETIQUETA
Relative	To program counter	(for jumps only)	
	To register	Relative to base register	MOV [BX]+ARTÍCULO,AL
	To index register	Relative to index register	MOV DL,VECTOR[SI]
		Relative to index and base registers	MOV AH,[BX][SI]+ARRAY
	To stack	(relative to stack)	
Indirect		(it doesn't exist)	
Implícit		Some instructions only	

- Relative addresing modes: ejemplos

- ADD DX, [BX]
- MOV DL, [SI]
- SUB AX,[BP][SI]
- XOR red[BX][DI], DX
- AND AL,[BP+8]

- shift instructions

ShiftInstruction target, times

times si es 1 lo puedo poner directamente pero si son mas, lo tengo que poner en CL

- SAL

- SAR

- SHL

- SHR

- ROL
- ROR

- RCL

- RCRA

Operaciones

- AND target, source

  AND ax,bx

  AND ax,8000h

  AND ax,1
- OR
- XOR
- NOT

```
DOSSEG
.MODEL SMALL
.STACK 100h
.DATA
numeros  DB 1,2,3,4,5,6 ; String numbers

.CODE
Inicio:
MOV AX, @DATA
MOV DS, AX

LEA BX, numeros      ; DS:BX string address

MOV CX, 6                ; Number of times
```

```
Bucle:
  MOV DL, [BX]     ; Memory position is
                          ; stored on DL
  ADD DL, 7          ; We add 7 to it
  MOV [BX], DL     ; Store new value on
                          ;current memory
                           ; position
  INC BX           ; BX is incremented by 1
LOOP Bucle
MOV AH, 4Ch          ; program end
   service
INT 21h      : requested
```

```
DOSSEG
.MODEL SMALL
.STACK 100h
.DATA
numeros  DB  1,2,3,4,5,6; String numbers

.CODE
Inicio:
MOV AX, @DATA

MOV DS, AX

MOV  SI, 0      ; SI is initializated to 0
MOV  CX, 6      ; Number of times

Bucle:
    ADD numeros[SI], 7
    INC SI
LOOP Bucle
MOV AH, 4Ch   ; Program end service
INT 21h       ; requested

END Inicio
```