



# **UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA**

## **Facultad de Ciencias Químicas e Ingeniería**

Materia: Organización de Computadoras y Lenguaje Ensamblador  
Docente: Lara Camacho Evangelina

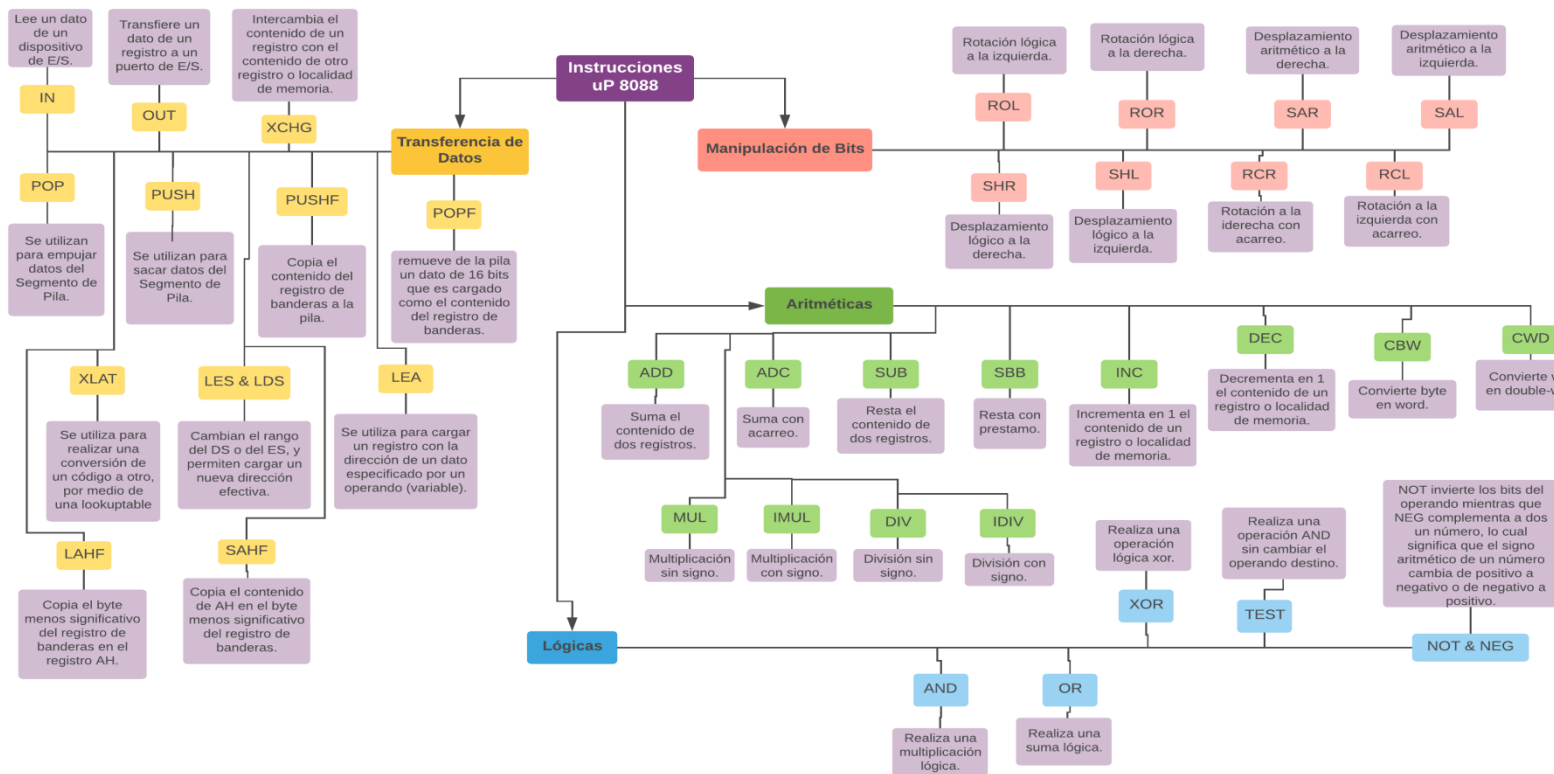
### **Practica 6 Instrucciones del procesador**

Alumno:

Morales Rosales Iván A. 1231098

# Teoría

## Mapa conceptual.



## Desarrollo

**a) Instrucciones lógicas y de manipulación de bits:** NOT, AND, OR, XOR, TEST, SHL, SHR, SAR, ROL, ROR, RCL, RCR.

## NOT

Se carga el registro ax con el valor 1234 para después aplicar la operación lógica "NOT", que es invertir los bits del registro.

```

073F:0100 mov ax,1234
073F:0103 not ax

AX=1234 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0103 NU UP EI PL NZ NA PO NC
073F:0103 F7D0 NOT AX
-t

AX=EDCB BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0105 NU UP EI PL NZ NA PO NC
073F:0105 809B07F98 SBB BYTE PTR [BX+DI+7FB0],98 DS:
  
```

## AND

Se desactivan los bits 0, 3, 4, 7 y 15 del registro ax (AND AX,6D24).

```
1110 1101 1100 1011 → EDCB
0111 1111 0110 0110 → 7F66
-----
0110 1101 0100 0010 → 6D42
```

```
073F:0105 and ax,7f66
073F:0108
-t
AX=6D42 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0108 NU UP EI PL NZ NA PE NC
073F:0108 7F98 JG 00A2
```

## OR

Los bits 0, 1, 3, 4, y 15 del contenido del registro ax se activan mediante la operación OR AX, 801B.

```
0110 1101 0100 0010 → 6D42
1000 0000 0001 1011 → 801B
-----
1110 1101 0101 1011 → ED5B
```

```
073F:0108 or ax,801b
073F:010B
-t
AX=ED5B BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010B NU UP EI NG NZ NA PO NC
073F:010B 7F34 JG 0141
```

## XOR

El contenido del registro ax, se le aplica un enmascaramiento mediante la instrucción XOR AX, D915.

```
1110 1101 0101 1011 → ED5B
1101 1001 0001 0101 → D915
-----
0011 0100 0100 1110 → 344E
```

```
073F:010B xor ax,d915
073F:010E
-t
AX=344E BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010E NU UP EI PL NZ NA PE NC
073F:010E B87F34 MOV AX,347F
```

## TEST

La instrucción **TEST** realiza una operación **AND**, pero sin afectar el operando, esta solo afecta las banderas de signo, zero y paridad.

```
AX=344E BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010E NU UP EI PL NZ NA PE NC
073F:010E B87F34      MOV     AX,347F
```

Después de ejecutar la instrucción la única bandera afectada fue la de zero.

```
073F:0103 test ax,0000
073F:0106
-t
AX=344E BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0106 NU UP EI PL ZR NA PE NC
073F:0106 0000      ADD     [BX+SI],AL      DS:0000=CD
```

## SHL

Para demostrar esta instrucción se ingresó el valor de 2 en el registro cl, ya que por medio de este se indica el total de corrimientos a la izquierda a realizar, tomando en cuenta que en este caso la bandera de carry quedo en cero.

```
073F:010E mov cl,2
073F:0110 shl ax,cl
073F:0112
-t
AX=344E BX=0000 CX=0002 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0110 NU UP EI PL NZ NA PE NC
073F:0110 D3E0      SHL     AX,CL
-t
AX=D13B BX=0000 CX=0002 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0112 NU UP EI NG NZ AC PO NC
073F:0112 0000      ADD     [BX+SI],AL      DS:0000=CD
```

## SHR

De igual manera que el caso anterior (SHL AX, CL), aplico dos corrimientos a la derecha y no quedo activado el carry.

```
073F:0112 shr ax,cl
073F:0114
-t
AX=344E BX=0000 CX=0002 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0114 NU UP EI PL NZ AC PE NC
073F:0114 0000      ADD     [BX+SI],AL      DS:0000=CD
```

## SAR

Esta instrucción realiza corrimientos a la derecha, a diferencia de la instrucción **shr**, esta conserva el signo más significativo del registro. En este caso el carry se activo.

```
073F:0114 mov cl,4
073F:0116 sar ax,cl
073F:0118
-t
AX=344E BX=0000 CX=0004 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0116  NU UP EI PL NZ AC PE NC
073F:0116 D3F8 SAR AX,CL
-t
AX=0344 BX=0000 CX=0004 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0118  NU UP EI PL NZ AC PE CY
073F:0118 0000 ADD [BX+SI],AL DS:0000=CD
```

## ROL

Con esta instrucción se realizan rotaciones a la izquierda. El bit más significativo es ingresado al carry al mismo tiempo que se copia hacia la parte menos significativa.

```
073F:0118 rol ax,cl
073F:011A
-t
AX=3440 BX=0000 CX=0004 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=011A  NU UP EI PL NZ AC PE NC
073F:011A 0000 ADD [BX+SI],AL DS:0000=CD
```

## ROR

Realiza lo contrario de la instrucción **rol**, rotaciones hacia la derecha. El bit menos significativo es ingresado al carry al mismo tiempo que es copiado a la parte más significativa del valor del registro.

```
073F:011A ror ax,cl
073F:011C
-t
AX=0344 BX=0000 CX=0004 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=011C  NU UP EI PL NZ AC PE NC
073F:011C 3400 XOR AL,00
```

## RCL

Con esta instrucción se realizan corrimientos hacia la izquierda tomando en cuenta el valor del carry. El valor actual del carry se ingresa en la parte menos significativa mientras que el MSB es ingresado en el carry.

```
073F:011C mov cl,3
073F:011E rcl ax,cl
073F:0120
-t 2

AX=0344 BX=0000 CX=0003 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=011E NU UP EI PL NZ AC PE NC
073F:011E D3D0 RCL AX,CL

AX=1A20 BX=0000 CX=0003 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0120 NU UP EI PL NZ AC PE NC
073F:0120 0000 ADD [BX+SI],AL DS:0000=CD
```

## RCR

Con esta instrucción se realizan corrimientos hacia la derecha tomando en cuenta el valor del carry. El valor actual del carry se ingresa en la parte mas significativa mientras que el LSB es ingresado en el carry.

```
073F:0120 rcr ax,cl
073F:0122
-t

AX=0344 BX=0000 CX=0003 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0122 NU UP EI PL NZ AC PE NC
073F:0122 0000 ADD [BX+SI],AL DS:0000=CD
```

**b) Instrucciones aritméticas: NEG, MUL, IMUL, DIV, IDIV, CBW y CWD.**

## NEG

Invierte el signo del valor aplicando complemento a dos. Esta instrucción afecta las banderas carry, zero, signo, overflow, aux carry y paridad.

```
0001 0010 0011 0100
1110 1101 1100 1011 → not ax
+
1
-----
110 1101 1100 1100 → EDCC
```

```
073F:0106 mov ax,1234
073F:0109 neg ax
073F:010B
-t

AX=1234 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0109 NU UP EI PL ZR NA PE NC
073F:0109 F7DB NEG AX

AX=EDCC BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010B NU UP EI NG NZ AC PE CY
073F:010B 0000 ADD [BX+SI],AL DS:0000=CD
```

## MUL

Se ingresa un valor al registro bx, el cual sera multiplicado por el registro ax, quien siempre contiene al multiplicando. EL resultado de la operación fue mayor de 2 bytes, por esta razón, se usa el registro dx.

```
073F:0105 mov bx,2
073F:0108 mul bx
073F:010A
-t 2

AX=EDCC BX=0002 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0108 NU UP EI NG NZ AC PE CY
073F:0108 F7E3 MUL BX

AX=DB98 BX=0002 CX=0000 DX=0001 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010A OU UP EI NG NZ AC PE CY
073F:010A 0000 ADD [BX+SI],AL DS:0002=3E
```

## IMUL

Con este nemónico se multiplican números con signo. El valor del registro bx se multiplica con el registro ax. El producto es almacenado en registro dx (parte MSB) y registro bx(LSB).

```
073F:010A mov bx,ff
073F:010D imul bx
073F:010F
-t 2

AX=DB98 BX=00FF CX=0000 DX=0001 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010D OU UP EI NG NZ AC PE CY
073F:010D F7EB IMUL BX

AX=BC68 BX=00FF CX=0000 DX=FFDB SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010F OU UP EI NG NZ AC PE CY
073F:010F FE00 INC BYTE PTR [BX+SI] DS:00FF=00
```

## DIV

Se realiza la division de numeros positivos. El cociente es almacenado en AL y el residuo en AH.

```
073F:0100 mov ax,128
073F:0103 mov bx,2
073F:0106 div bx
073F:0108
-t 3

AX=0128 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0103 NU UP EI PL NZ NA PO NC
073F:0103 BB0200 MOV BX,0002

AX=0128 BX=0002 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0106 NU UP EI PL NZ NA PO NC
073F:0106 F7F3 DIV BX

AX=0034 BX=0002 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0108 NU UP EI PL NZ NA PO NC
073F:0108 0000 ADD [BX+SI],AL DS:0002=3E
```

## IDIV

Si la división se realizara con números con signo se utiliza IDIV.

```
073F:0108 mov ax,10
073F:010B mov bl,fd
073F:010D idiv bl
073F:010F
-t 3

AX=0010 BX=0002 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010B NU UP EI PL NZ NA PO NC
073F:010B B3FD          MOV     BL,FD

AX=0010 BX=00FD CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010D NU UP EI PL NZ NA PO NC
073F:010D F6FB          IDIV    BL

AX=01FB BX=00FD CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010F NU UP EI PL NZ NA PO NC
073F:010F FE00          INC     BYTE PTR [BX+SI]          DS:00FD=00
```

## CBW

Cuando se quiere convertir un byte a palabra se utiliza esta instrucción, a la vez, esta respeta el signo del byte.

```
073F:0100 mov ax,80
073F:0103 cbw
073F:0104
-t 2

AX=0080 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0103 NU UP EI PL NZ NA PO NC
073F:0103 98          CBW

AX=FF80 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0104 NU UP EI PL NZ NA PO NC
073F:0104 0200          ADD     AL,[BX+SI]          DS:0000=CD
```

## CWD

Para convertir una palabra a doble palabra, el cambio se puede realizar con esta instrucción, de igual manera respeta el signo. LA parte MSB se almacena en el registro dx mientras que la parte LSB en el registro origen.

```
073F:0104 mov bx,ff12
073F:0107 cwd
073F:0108
-t 2

AX=FF80 BX=FF12 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0107 NU UP EI PL NZ NA PO NC
073F:0107 99          CWD

AX=FF80 BX=FF12 CX=0000 DX=FFFF SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0108 NU UP EI PL NZ NA PO NC
073F:0108 B81000          MOV     AX,0010
```



c) Instrucciones de movimiento de datos: XLAT, LEA, LDS y LES.

## XLAT

Esta instrucción suma el contenido de al con el contenido dl registro bx para formar una dirección del segmento de datos y almacenarla en el registro ax. La dirección resultante sería  $34+1000=1034$ , el valor contenido en esta dirección del segmento de datos se copia al registro ax.

```
073F:0108 mov ax,0034
073F:010B mov bx,1000
073F:010E xlat
073F:010F
-t 3

AX=0034 BX=FF12 CX=0000 DX=FFFF SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010B NU UP EI PL NZ NA PO NC
073F:010B BB0010 MOV BX,1000

AX=0034 BX=1000 CX=0000 DX=FFFF SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010E NU UP EI PL NZ NA PO NC
073F:010E D7 XLAT

AX=0000 BX=1000 CX=0000 DX=FFFF SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010F NU UP EI PL NZ NA PO NC
073F:010F FE00 INC BYTE PTR [BX+SI] DS:1000=00
```

Usando el comando dump, se accede a la dirección para comprobar que el valor de ax corresponde al de la dirección.

```
d 1034:00
1034:0000 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
1034:0010 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
1034:0020 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
```

## LEA

Se utiliza para cargar un registro con la dirección de un dato especificado por un operando.

En el siguiente ejemplo se carga el contenido del registro di en bx (di contiene la dirección).

```
073F:0118 lea bx,[di]
073F:011A
-t 2

AX=34F1 BX=1000 CX=0000 DX=FFFF SP=00FD BP=0000 SI=0000 DI=14BC
DS=073F ES=073F SS=073F CS=073F IP=0118 NU UP EI PL NZ NA PO NC
073F:0118 8D1D LEA BX,DI DS:14BC=

AX=34F1 BX=14BC CX=0000 DX=FFFF SP=00FD BP=0000 SI=0000 DI=14BC
DS=073F ES=073F SS=073F CS=073F IP=011A NU UP EI PL NZ NA PO NC
```

## LDS y LES

Estas instrucciones permiten cargar un registro de 2 bytes que representara una dirección y cargar DS o ES con una nueva dirección.

LDS: el valor de 2 bytes que se encuentra en memoria direccionada por el registro di, se almacena en el registro bx, y los siguientes 2 bytes son almacenados en el registro ds.

```
073F:0100 mov di,123
073F:0103 lds bx,[di]
073F:0105
-t 2

AX=0000 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0123
DS=073F ES=073F SS=073F CS=073F IP=0103  NU UP EI PL NZ NA PO NC
073F:0103 C51D          LDS     BX,[DI]                DS:0123=0000

AX=0000 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0123
DS=0000 ES=073F SS=073F CS=073F IP=0105  NU UP EI PL NZ NA PO NC
073F:0105 0000          ADD     [BX+SI],AL            DS:0000=60
```

Para comprobar, se ejecuto el comando dump con la dirección obtenida ( $73F0H \times 10H + 123 = 7513$ ).

```
-d 0751:00
0751:0000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0751:0010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0751:0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

LES: el valor de 2 bytes que se encuentra en memoria direccionada por el registro di, se almacena en el registro bx, y los siguientes 2 bytes son almacenados en el registro ds.

```
073F:0105 mov si,121
073F:0108 les cx,[si]
073F:010A
-t 2

AX=0000 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0121 DI=0123
DS=0000 ES=073F SS=073F CS=073F IP=0108  NU UP EI PL NZ NA PO NC
073F:0108 C40C          LES     CX,[SI]                DS:0121=0010

AX=0000 BX=0000 CX=0010 DX=0000 SP=00FD BP=0000 SI=0121 DI=0123
DS=0000 ES=60F0 SS=073F CS=073F IP=010A  NU UP EI PL NZ NA PO NC
073F:010A 0000          ADD     [BX+SI],AL            DS:0121=10
```

2. Escriba y ejecute en Debug las instrucciones necesarias para:

a) Colocar en el registro AX el valor 0xA357 y por medio de rotaciones obtener 0x8D5E.

```
073F:0100 mov ax,a357
073F:0103 mov cl,2
073F:0105 rol ax,cl
073F:0107
-t 3

AX=A357 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0103 NU UP EI PL NZ NA PO NC
073F:0103 B102 MOV CL,02

AX=A357 BX=0000 CX=0002 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0105 NU UP EI PL NZ NA PO NC
073F:0105 D3C0 ROL AX,CL

AX=8D5E BX=0000 CX=0002 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0107 OV UP EI PL NZ NA PO NC
073F:0107 0000 ADD [BX+SI],AL DS:0000=CD
```

b) Colocar en el registro BL el valor 0x7E y por medio de corrimientos obtener 0xF.

```
073F:0107 mov bl,7e
073F:0109 mov cl,3
073F:010B shr bl,cl
073F:010D
-t 3

AX=8D5E BX=007E CX=0002 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0109 OV UP EI PL NZ NA PO NC
073F:0109 B103 MOV CL,03

AX=8D5E BX=007E CX=0003 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010B OV UP EI PL NZ NA PO NC
073F:010B D2EB SHR BL,CL

AX=8D5E BX=000F CX=0003 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=010D NU UP EI PL NZ AC PE CY
073F:010D 0000 ADD [BX+SI],AL DS:000F=03
```

c) Colocar en el registro CX el valor 0x94F2 y por medio de enmascaramiento invertir los bits 0,3 y 13, sin modificar los demás.

```
073F:010D mov cx,94f2
073F:0110 xor cx,2009
073F:0114
-t 2

AX=8D5E BX=000F CX=94F2 DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0110 NU UP EI PL NZ AC PE CY
073F:0110 81F10920 XOR CX,2009

AX=8D5E BX=000F CX=B4FB DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0114 NU UP EI NG NZ NA PO NC
073F:0114 0000 ADD [BX+SI],AL DS:000F=03
```

d) Colocar en el registro AH el valor 0x57 y por medio de enmascaramiento activar los bits 3 y 5, sin modificar los demás.

```

073F:011C mov ah,57
073F:011E or ah,28
073F:0121
-t 2

AX=575E BX=000F CX=B4FB DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=011E NU UP EI PL NZ NA PE NC
073F:011E 80CC28 OR AH,28

AX=7F5E BX=000F CX=B4FB DX=0000 SP=00FD BP=0000 SI=0000 DI=0000
DS=073F ES=073F SS=073F CS=073F IP=0121 NU UP EI PL NZ NA PO NC
073F:0121 0000 ADD [BX+SI],AL DS:000F=03

```

e) Colocar en el registro DI el valor 0xFA61 y por medio de enmascaramiento desactivar los bits 0, 9, 13 y 15, sin modificar los demás.

```

073F:0100 mov di,fa61
073F:0103 and di,5dfe
073F:0107
-t 2

AX=0000 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=FA61
DS=073F ES=073F SS=073F CS=073F IP=0103 NU UP EI PL NZ NA PO NC
073F:0103 81E7FE5D AND DI,5DFE

AX=0000 BX=0000 CX=0000 DX=0000 SP=00FD BP=0000 SI=0000 DI=5860
DS=073F ES=073F SS=073F CS=073F IP=0107 NU UP EI PL NZ NA PE NC
073F:0107 0000 ADD [BX+SI],AL DS:0000=CD

```

f) Colocar en el registro AL el valor 0x8E y por medio de la instrucción CBW convertirlo a una palabra que se almacene en AX, respetando el signo.

```

073F:012F mov al,8e
073F:0131 cbw
073F:0132
-t

AX=008E BX=000F CX=B4FB DX=0000 SP=00FD BP=0000 SI=0000 DI=5860
DS=073F ES=073F SS=073F CS=073F IP=0131 NU UP EI PL NZ NA PE NC
073F:0131 98 CBW

AX=FF8E BX=000F CX=B4FB DX=0000 SP=00FD BP=0000 SI=0000 DI=5860
DS=073F ES=073F SS=073F CS=073F IP=0132 NU UP EI PL NZ NA PE NC
073F:0132 0000 ADD [BX+SI],AL DS:000F=03

```

g) Colocar en el registro AL el valor 0x49 y por medio de la instrucción CBW convertirlo a una palabra que se almacene en AX, respetando el signo.

```

073F:0132 mov al,49
073F:0134 cbw
073F:0135
-t 2

AX=FF49 BX=000F CX=B4FB DX=0000 SP=00FD BP=0000 SI=0000 DI=5860
DS=073F ES=073F SS=073F CS=073F IP=0134 NU UP EI PL NZ NA PE NC
073F:0134 98 CBW
AX=0049 BX=000F CX=B4FB DX=0000 SP=00FD BP=0000 SI=0000 DI=5860
DS=073F ES=073F SS=073F CS=073F IP=0135 NU UP EI PL NZ NA PE NC
073F:0135 0000 ADD [BX+SI],AL DS:000F=03

```

h) Colocar en el registro AX el valor 0xA61D y por medio de la instrucción CWD convertirlo a una doble palabra que se almacene en DX-AX, respetando el signo.

```

073F:0138 mov ax,a61d
073F:013B cwd
073F:013C
-t 3

AX=A61D BX=000F CX=B4FB DX=0000 SP=00FD BP=0000 SI=0000 DI=5860
DS=073F ES=073F SS=073F CS=073F IP=0138 NU UP EI PL NZ NA PE NC
073F:0138 B81DA6 MOV AX,A61D
AX=A61D BX=000F CX=B4FB DX=0000 SP=00FD BP=0000 SI=0000 DI=5860
DS=073F ES=073F SS=073F CS=073F IP=013B NU UP EI PL NZ NA PE NC
073F:013B 99 CWD
AX=A61D BX=000F CX=B4FB DX=FFFF SP=00FD BP=0000 SI=0000 DI=5860
DS=073F ES=073F SS=073F CS=073F IP=013C NU UP EI PL NZ NA PE NC
073F:013C 0000 ADD [BX+SI],AL DS:000F=03

```

i) Colocar en el registro AX el valor 0x7320 y por medio de la instrucción CWD convertirlo a una doble palabra que se almacene en DX-AX, respetando el signo.

```

073F:013C mov ax,7320
073F:013F cwd
073F:0140
-t 2

AX=7320 BX=000F CX=B4FB DX=FFFF SP=00FD BP=0000 SI=0000 DI=5860
DS=073F ES=073F SS=073F CS=073F IP=013F NU UP EI PL NZ NA PE NC
073F:013F 99 CWD
AX=7320 BX=000F CX=B4FB DX=0000 SP=00FD BP=0000 SI=0000 DI=5860
DS=073F ES=073F SS=073F CS=073F IP=0140 NU UP EI PL NZ NA PE NC
073F:0140 0000 ADD [BX+SI],AL DS:000F=03

```

## **Conclusiones y comentarios**

Sin duda alguna reafirme lo visto en clase. Agregue el diagrama con todas las instrucciones vistas anteriormente.

## **Dificultades en el desarrollo**

Surgieron algunas dudas que resolví leyendo de nuevo los pdf de clase.

## **Referencias**

PDF's de clase.

Assembly Language For x86 Processors. Kip R. Irvine. 6th Edition