



## Modos de Direcccionamiento

Modo de direccionamiento admitidos en el 8086.

### ***Direcccionamiento por Registro***

La fuente es un registro

```
mov AX,BX
```

```
mov BL,AH
```

No esta permitido asignaciones entre registro de segmento.

Mov ES,DS ; no está permitido

### ***Direcccionamiento inmediato***

Copia una palabra o byte a un registro

```
mov AX,10      10 → AX
```

```
mov AL,10      10 → AL
```

Registro a los cuales no esta permitido una asignación inmediata.

- registro de segmento **es,ds,cs,ss**
- puntero de instrucción **ip**

### ***Direcccionamiento directo***

Solo se aplica para la instrucción mov cuando es usada con los registro AL o AX.

Implica copiar en forma directa lo que se encuentra en una posición de memoria al registro AL o AX.

```
mov AL,[100]      ; copia DS:[100] → AL
```

```
mov [100],AX      ; copia AX → DS:[100]
```

```
mov AX,[100]      ; copia DS:[100] → AX
```

```
mov [100],AL      ; copia AL → DS:[100]
```

Esto forma un pequeño grupo de instrucciones que por su uso muy a menudo en los programas se decidió hacerlas de 3 bytes de longitud contra 4 o mas que posee las por desplazamiento.

### ***Direcccionamiento por desplazamiento***

Es similar al “direccionamiento directo” pero mas flexible y es el que se aplica en todas las instrucciones y con todo los demás registros, usando en este caso 4 bytes de longitud cada instrucción.

```
mov CL,[200]      ; copia DS:[200] → CL
```

```
mov BX,[203]      ; copia DS:[203] → BX
```

El único registro al que no esta permitido usar este tipo de direccionamiento es el IP.

### ***Direcccionamiento indirecto por Registro***

Permite direccionar una localidad de memoria apuntada por un registro

Los registros que se pueden usar son:

- BP
- BX
- DI
- SI

Cuando son utilizados cualquiera de los registros arriba indicados, utiliza por defecto al DS como segmento para completar la dirección, con excepción del BP que utiliza el SS.

Ejemplo: copiar a AX lo que se encuentra en DS:100.



```
mov BX,100
mov AX,[BX]
```

En algunos casos es necesario aclarar el tamaño del dato a mover, por ejemplo cuando se copia una cte a un posición de memoria.

```
mov BX,100
mov [BX],22 ; el compilador no sabe si 22 es un byte o word
mov BYTE PTR [BX],22 ; se copiará el 22 en una pos de memoria de un byte
mov WORD PTR [BX],22 ; se copiará el 22 en una pos de memoria de un word
```

### ***Direcccionamiento Base mas Índice***

Es un caso similar al “direcccionamiento por registro”, nada mas que la dirección a acceder en vez de estar dado por un registro, lo está por dos, un registro base ( BP o BX ) mas un registro índice ( DI o SI ).

Este tipo de direcccionamiento se utiliza para acceder a tablas donde por ejemplo la base contiene el comienzo de la tabla y el índice la posición relativa del elemento a acceder.

```
mov AX,[BX+DI] copia DS:[BX+DI] → AX
mov AX,[BP+DI] copia SS:[BP+DI] → AX
```

Al igual que el de “direcccionamiento indirecto” si se utiliza a BP como registro índice, se utilizará al segmento de Stack (SS) para formar la dirección completa.

### ***Direcccionamiento Relativo por Registro***

Para este direcccionamiento utilizamos para acceder a una posición de memoria el valor de un registro ( como en los casos anteriores ) y un valor de desplazamiento constante.

```
mov AX,[BX+100] copia DS:[BX+100] → AX
```

En este caso el valor de la memoria a que apunta es el valor contenido en BX mas 100.

Los registros que se pueden usar son los mismos que en el “indirecto por registro” BP, BX, DI, SI, el segmento usado para completar la dirección será DS para BX, DI y SI y SS para BP como en los casos anteriores.

Este direcccionamiento es usado para acceder a un vector el cual tiene fijo su dirección de comienzo, si por ejemplo TABLA es el nombre que le pusimos a un vector de 100 posiciones, para acceder entonces a cualquier valor de esta tabla y cargarla en el registro AX, se deberá efectuar lo siguiente:

```
mov AX,TABLA[BX] ; copia en AX el contenido de la posición BX de TABLA
```

### ***Direcccionamiento Relativo por Base mas Índice***

La dirección a acceder será la suma de un registro base mas un índice mas un desplazamiento cte.

```
mov AX,[BX+DI+100]; copia en AX el contenido de la posición DS:[BX+DI+100]
```

Al igual que en el caso anterior puedo declarar en el compilador un vector de 100 posiciones llamado TABLA, y luego escribir.

```
mov TABLA[BX+DI]
```



## Resumen

### Direcccionamiento por Registro

La fuente es un registro.

```
mov AX, BX
```

### Direcccionamiento inmediato

La fuente es una constante.

```
mov AX, 100
```

### Direcccionamiento directo

Se direcciona una posición de memoria constante

Caso especial de AX y AL junto con la instrucción mov.

```
mov AX, [100]
```

### Direcccionamiento por desplazamiento

Igual a Direcccionamiento directo pero para los otros registros.

```
mov BX, [100]
```

### Direcccionamiento indirecto por Registro

Se direcciona una posición de memoria mediante un registro.

```
mov AX, [BX]
```

### Direcccionamiento Base mas Índice

Se direcciona una posición de memoria mediante un registro base mas un registro indice.

```
mov AX, [BX+DI]
```

### Direcccionamiento Relativo por Registro

Se direcciona una posición de memoria mediante un registro mas un desplazamiento.

```
mov AX, [BX+100]
```

```
mov AX, TABLA[BX]
```

### Direcccionamiento Relativo por Base mas Indice

Se direcciona una posición de memoria mediante un registro base mas un registro indice mas un desplazamiento constante.

```
mov AX, [BX+DI+100]
```

```
mov AX, TABLA[BX+DI]
```



## Modificación de Código

En el siguiente ejemplo se a ingresado el siguiente código, en el cual se cometió un error de tipeo al cargar la 2da línea.

Una vez ingresado el código completo, se intenta subsanar el error modificando únicamente esa línea mediante una instrucción -a 103.

Esa aparente pequeña modificación, modifica la longitud de la instrucción de una de 3 bytes a otra de 4 bytes, utilizando para completar su longitud el primer byte de la siguiente instrucción, ahora el octavo byte del código el cual era parte integrante de la 3ra instrucción pasa a ser el primer byte de una nueva instrucción en este caso una instrucción PUSH CS, y así sucesivamente.

### Código Fuente a modificar

|           |    |    |    |               |               |
|-----------|----|----|----|---------------|---------------|
| 1         | 2  | 3  |    |               |               |
| 0DA8:0100 | A1 | 00 | 01 | MOV AX,[0100] |               |
|           | 4  | 5  | 6  |               |               |
| 0DA8:0103 | A1 | 02 | 01 | MOV AX,[0102] |               |
|           | 7  | 8  | 9  | 10            |               |
| 0DA8:0106 | 8B | 0E | 04 | 01            | MOV CX,[0104] |
|           | 11 | 12 |    |               |               |
| 0DA8:010A | 01 | D8 |    |               | ADD AX,BX     |
|           | 13 | 14 |    |               |               |
| 0DA8:010C | 01 | C8 |    |               | ADD AX,CX     |
|           | 15 |    |    |               |               |
| 0DA8:010E | CC |    |    |               | INT 3         |

### Código Fuente Modificado

|           |    |    |    |               |               |
|-----------|----|----|----|---------------|---------------|
| 1         | 2  | 3  |    |               |               |
| 0DA8:0100 | A1 | 00 | 01 | MOV AX,[0100] |               |
|           | 4  | 5  | 6  | 7             |               |
| 0DA8:0103 | 8B | 1E | 02 | 01            | MOV BX,[0102] |
|           | 8  |    |    |               |               |
| 0DA8:0107 | 0E |    |    |               | PUSH CS       |
|           | 9  | 10 |    |               |               |
| 0DA8:0108 | 04 | 01 |    |               | ADD AL,01     |
|           | 11 | 12 |    |               |               |
| 0DA8:010A | 01 | D8 |    |               | ADD AX,BX     |
|           | 13 | 14 |    |               |               |
| 0DA8:010C | 01 | C8 |    |               | ADD AX,CX     |
|           | 15 |    |    |               |               |
| 0DA8:010E | CC |    |    |               | INT 3         |

Ejemplo de si en vez de haber sido mov cx,[104] hubiera sido mov cx,[106]

|           |          |      |                   |
|-----------|----------|------|-------------------|
| 0DA8:0100 | A10002   | MOV  | AX,[0200]         |
| 0DA8:0103 | 8B1E0201 | MOV  | BX,[0102]         |
| 0DA8:0107 | 0E       | PUSH | CS                |
| 0DA8:0108 | 06       | PUSH | ES                |
| 0DA8:0109 | 0101     | ADD  | [BX+DI],AX        |
| 0DA8:010B | D801     | FADD | DWORD PTR [BX+DI] |
| 0DA8:010D | C8       | DB   | C8                |
| 0DA8:010E | CC       | INT  | 3                 |