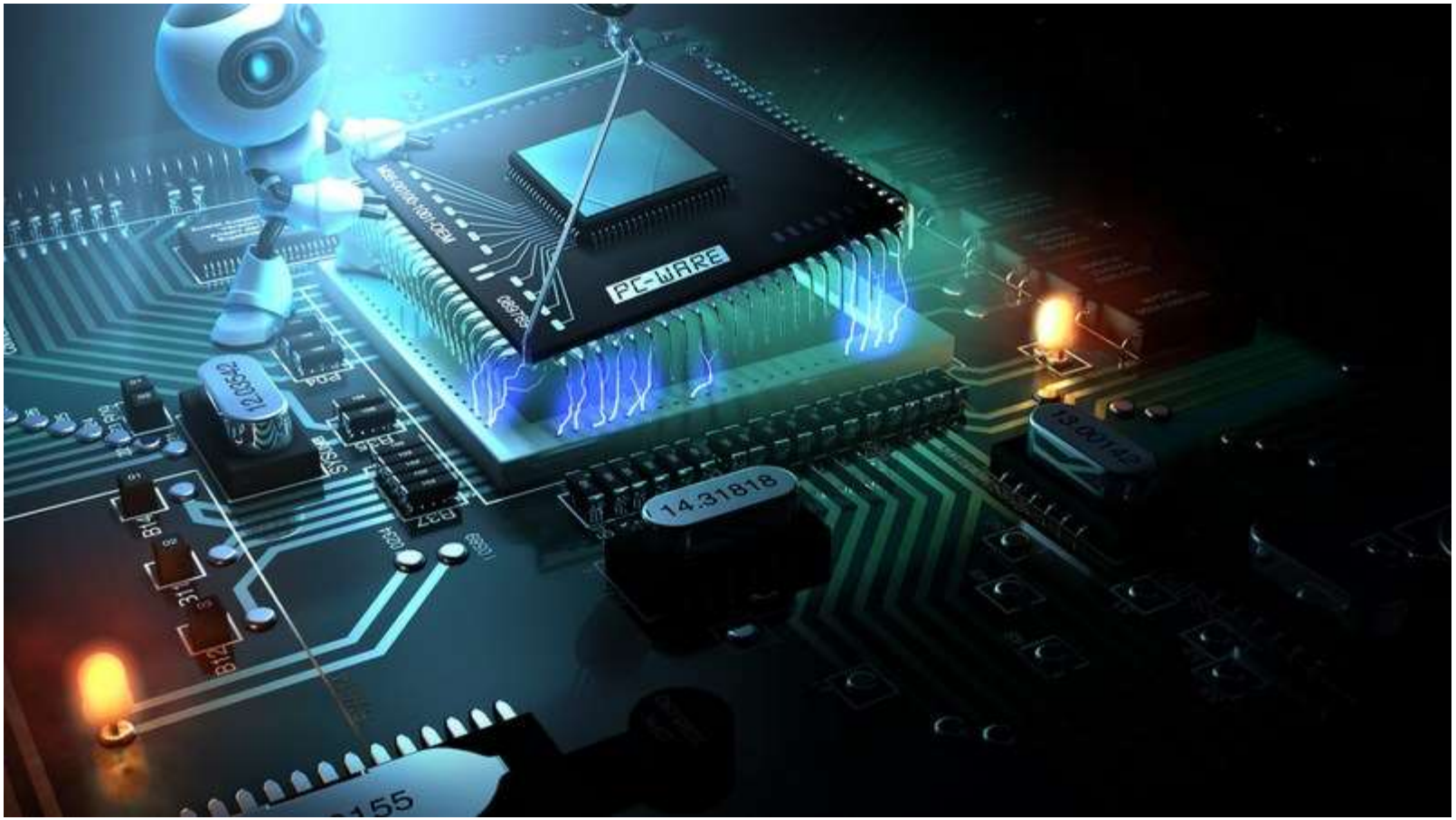
ENSAMBLADOR

(7°-ICI)

UNIVERSIDAD AUTONOMA DE AGUASCALIENTES



Mtro. en Ing. Armando Álvarez Fdez.

Sep-2024

MODOS DE DIRECCIONAMIENTO

(Continuación)

Modos de direccionamiento

De Los procesadores 8086 y 8088

- **Implícito**. El dato está implícito en la propia instrucción. Ej. `STC` , `STD` y `STI` , (Set Carry, Set Direcction y Set Interrupts) encienden el flag correspondiente indicado en la propia instrucción. `CBW` (Convert Byte to Word) extiende el bit del signo del registro AL a AX. Ni el AL ni el AX son especificados, puesto que la instrucción `CBW` implícitamente trabaja sobre ellos.
- **Inmediato**. El dato a operar está inmediatamente después del opcode de la instrucción. Ej, `MOV AX, 5`
- **Registro**. El dato está en un segundo registro. Ej. `MOV AX, BX` . Aquí, el dato está en el registro BX
- **Directo**. La dirección del dato está en el campo de la dirección del opcode. Ej. `MOV AX, [100h]` . Aquí se mueve (copia) el contenido de las direcciones 100h y 101h al registro AX. En este caso se mueven dos bytes puesto que AX es de 16 bits. Si fuera `MOV BL, [100h]` se movería solo un byte pues BL es un registro de 8 bits

- **Indirecto**. El dato es especificado mediante una combinación de registros índice y base, y puede haber un desplazamiento
  - **Base**. Un registro base (BX o BP) tienen la dirección de donde se tomará el dato. Ej. `MOV AX, [BX]`
  - **Índice**. Un registro índice (SI o DI) tienen la dirección de donde se tomará el dato. Ej. `MOV AX, [SI]`
  - **Base + Desplazamiento**. El dato se tomará de la dirección apuntada por la suma de un registro base más un desplazamiento. Ej. `MOV AX, [BP + 7]`
  - **Índice + Desplazamiento**. El dato se tomará de la dirección apuntada por la suma de un registro índice más un desplazamiento. Ej. `MOV AX, [DI + 7]`
  - **Base + Índice**. El dato se tomará de la dirección apuntada por la suma de un registro base más un registro índice. Ej. `MOV AX, [BX + SI]`
  - **Base + Índice + Desplazamiento**. El dato se tomará de la dirección apuntada por la suma de un registro base, más un registro índice, más un desplazamiento. Ej. `MOV AX, [BX + SI + 9]`

NOMBRES E INICIALES DE LOS REGISTROS DEL 8086 EN INGLES

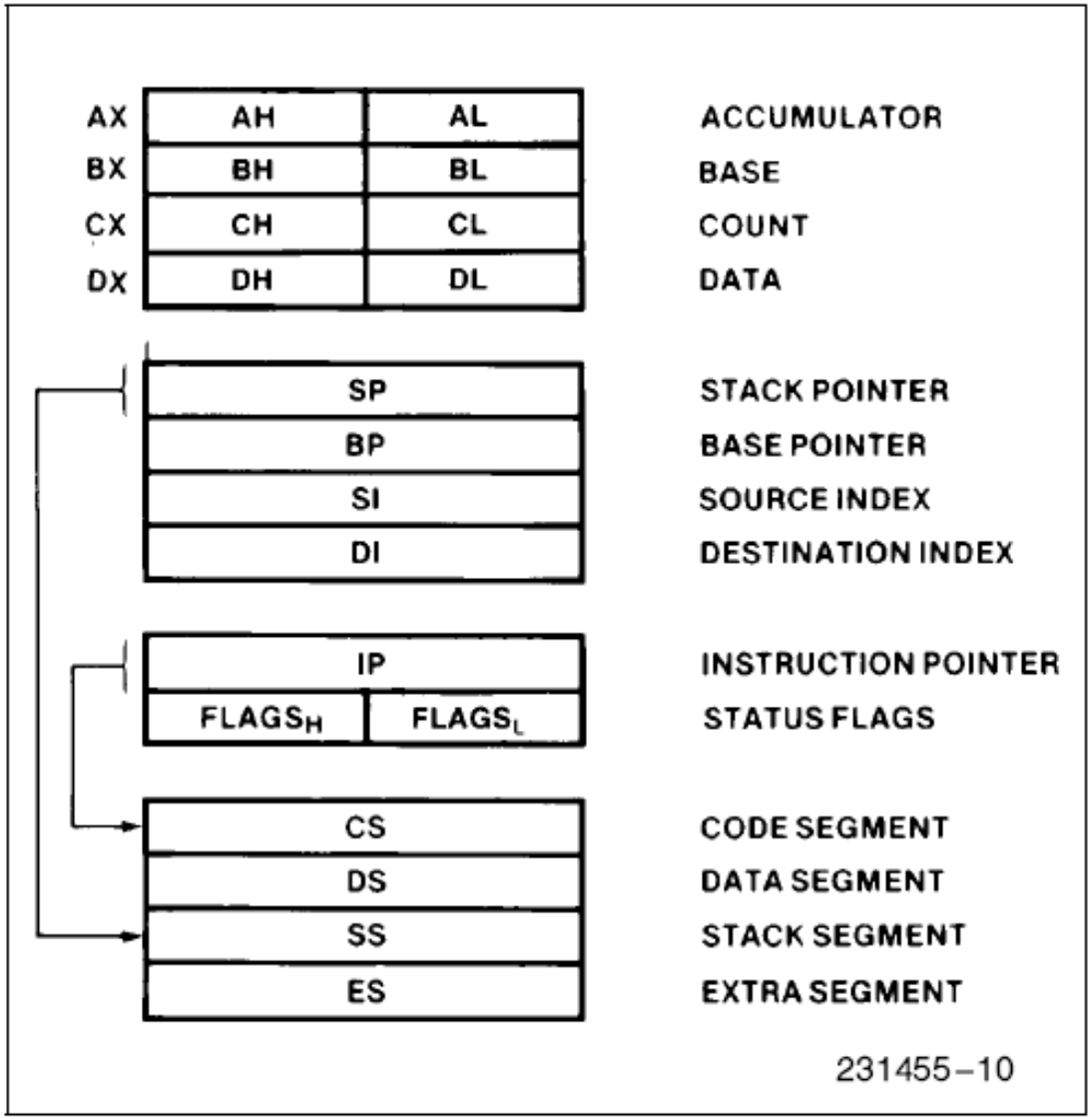


Figure . 8086 Register Model

Segunda clasificación:
(Hacia una mejor aproximación)

Modos de direccionamiento

De Los procesadores 8086 y 8088

- **Implícito**. El dato está implícito en la propia instrucción. Ej. `STC` , `STD` y `STI` , (Set Carry, Set Direcction y Set Interrupts) encienden el flag correspondiente indicado en la propia instrucción. `CBW` (Convert Byte to Word) extiende el bit del signo del registro AL a AX. Ni el AL ni el AX son especificados, puesto que la instrucción `CBW` implícitamente trabaja sobre ellos.
- **Inmediato**. El dato a operar está inmediatamente después del opcode de la instrucción. Ej, `MOV AX, 5`
- **Registro**. El dato está en un segundo registro. Ej. `MOV AX, BX` . Aquí, el dato está en el registro BX
- **Directo**. La dirección del dato está en el campo de la dirección del opcode. Ej. `MOV AX, [100h]` . Aquí se mueve (copia) el contenido de las direcciones 100h y 101h al registro AX. En este caso se mueven dos bytes puesto que AX es de 16 bits. Si fuera `MOV BL, [100h]` se movería solo un byte pues BL es un registro de 8 bits

- **Indirecto**. El dato es especificado mediante una combinación de registros índice y base, y puede haber un desplazamiento
  - **Base**. Un registro base (BX o BP) tienen la dirección de donde se tomará el dato. Ej. `MOV AX, [BX]`
  - **Índice**. Un registro índice (SI o DI) tienen la dirección de donde se tomará el dato. Ej. `MOV AX, [SI]`
  - **Base + Desplazamiento**. El dato se tomará de la dirección apuntada por la suma de un registro base más un desplazamiento. Ej. `MOV AX, [BP + 7]`
  - **Índice + Desplazamiento**. El dato se tomará de la dirección apuntada por la suma de un registro índice más un desplazamiento. Ej. `MOV AX, [DI + 7]`
  - **Base + Índice**. El dato se tomará de la dirección apuntada por la suma de un registro base más un registro índice. Ej. `MOV AX, [BX + SI]`
  - **Base + Índice + Desplazamiento**. El dato se tomará de la dirección apuntada por la suma de un registro base, más un registro índice, más un desplazamiento. Ej. `MOV AX, [BX + SI + 9]`

Ejemplos de

Modos de direccionamiento

- ***Direccionamiento Inmediato:*** Transfiere n bytes o palabras de datos inmediatos hacia el registro o localidad en la memoria en el destino. Ejemplo:

```
MOV AL, 22H
MOV EBX, 12345678H

MOV AL, 18H
MOV BX,1998H

MOV DS, 0B70H

MOV AX, 0B70H
MOV DS, AX

MOV DL, 0B800H
MOV CX, 1
MOV CX, 0001
```

Ejemplos de

Modos de direccionamiento

- ***Direccionamiento Inmediato:*** Transfiere n bytes o palabras de datos inmediatos hacia el registro o localidad en la memoria en el destino. Ejemplo:

```
MOV AL, 22H
MOV EBX, 12345678H

MOV AL, 18H
MOV BX,1998H

MOV DS, 0B70H

MOV AX, 0B70H
MOV DS, AX

MOV DL, 0B800H
MOV CX, 1
MOV CX, 0001
```

Ejemplos de

Modos de direccionamiento

- ***Direccionamiento Inmediato:*** Transfiere n bytes o palabras de datos inmediatos hacia el registro o localidad en la memoria en el destino. Ejemplo:

```
MOV AL, 22H
MOV EBX, 12345678H

MOV AL, 18H
MOV BX,1998H

MOV DS, 0B70H  Error, no se puede en direccionamiento inmediato, solo por registro.

MOV AX, 0B70H
MOV DS, AX

MOV DL, 0B800H Error.
MOV CX, 1
MOV CX, 0001
```

Ejemplos de

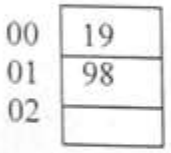
- ***Direccionamiento por Registro:*** Transfiere bytes o palabra desde el registro fuente o localidad de memoria, hasta el registro o localidad destino en la memoria. Ejemplo:

```
MOV CX, DX
MOV AH, DH
MOV BP, BX
MOV AX, BL    Error
MOV AL, BL
MOV AH, 0
CBW           Convierte de byte a word, si es positivo llena de ceros, si es neg llena de 1.
```

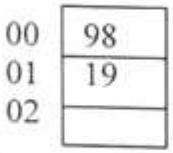
- ***Direccionamiento Directo:*** Transfiere bytes o palabra entre una localidad de memoria y un registro. Ejemplo:

```
MOV DL, [300H]
MOV DX, [200 H]
```

NOTA SOBRE FORMATOS DE ALMACENAMIENTO DE DATOS:



Formato:
Motorola
Big Endían



Formato:
Intel
Little Endian

Si se tiene el número 1998, en los procesadores motorola se guarda de la forma Big Endían, en los procesadores intel se almacena de la forma Little Endían.

Ejemplos de

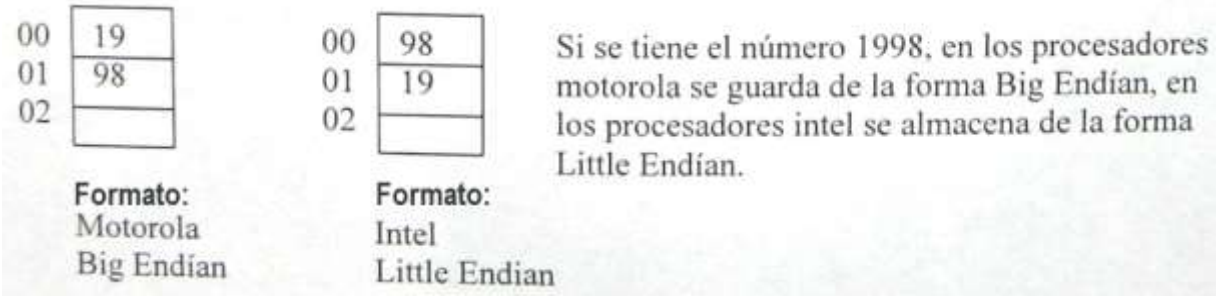
- ***Direccionamiento por Registro:*** Transfiere bytes o palabra desde el registro fuente o localidad de memoria, hasta el registro o localidad destino en la memoria. Ejemplo:

```
MOV CX, DX
MOV AH, DH
MOV BP, BX
MOV AX, BL    Error
MOV AL, BL
MOV AH, 0
CBW           Convierte de byte a word, si es positivo llena de ceros, si es neg llena de 1.
```

- ***Direccionamiento Directo:*** Transfiere bytes o palabra entre una localidad de memoria y un registro. Ejemplo:

```
MOV DL, [300H]
MOV DX, [200 H]
```

NOTA SOBRE FORMATOS DE ALMACENAMIENTO DE DATOS:



- ***Direccionamiento Indirecto por Registro (Base):*** Transfiere bytes o palabra entre un registro y una localidad de memoria direccionada por un registro índice o base. Ejemplo:

```
MOV AX, [BX]
MOV [BX], AX
```

- ***Direccionamiento Base mas Índice:*** Transfiere bytes o palabra entre un registro y una localidad de memoria direccionada por un registro base mas un registro índice. Ejemplo:

```
MOV AX, [BX + SI]
MOV [BX + DI], CX
```

- ***Direccionamiento Relativo por Registro:*** Transfiere bytes o palabra entre un registro y una localidad de memoria direccionada por un registro índice o base y además un desplazamiento. Ejemplo:

MOV AX, [BX + 4]

- ***Direccionamiento Relativo Base más Índice:*** Transfiere bytes o palabra entre un registro y una localidad de memoria direccionada por un registro base más un índice mas un desplazamiento. Ejemplo:

MOV AX, [BX + DI + 4]

EJERCICIOS: *Determine que tipo de direccionamiento es para cada instrucción.*

```
MOV [BX + 4], AX
MOV AX, [BX + SI + 4]
MOV [EBX + 2*ESI], AX
MOV AX, BX
MOV AX, [1234H]
MOV [BX], AX
MOV BL, 3AH
MOV [BX + SI], AX
```

EJERCICIOS: *Determine que tipo de direccionamiento es para cada instrucción.*

```
MOV [BX + 4], AX
MOV AX, [BX + SI + 4]
MOV [EBX + 2*ESI], AX
MOV AX, BX
MOV AX, [1234H]
MOV [BX], AX
MOV BL, 3AH
MOV [BX + SI], AX
```

EJERCICIOS: *Determine que tipo de direccionamiento es para cada instrucción.*

MOV [BX + 4], AX	Relativo por registro
MOV AX, [BX + SI + 4]	Relativo base mas índice
MOV [EBX + 2*ESI], AX	Relativo por Índice escalado
MOV AX, BX	Registro
MOV AX, [1234H]	Directo
MOV [BX], AX	Indirecto por registro
MOV BL, 3AH	Inmediato
MOV [BX + SI], AX	Relativo por Base mas índice

EJERCICIO

Ejercicio: *Implementar un programa en ensamblador que sume la localidad de memoria 200 y la 201, el resultado lo deje en la localidad 202.*

1° OPCIÓN:

```
MOV BX, 200H
MOV AL, [BX]
INC BX
MOV DL, [BX]
ADD AL, BL
INC BX
MOV [BX], AL
INT 20H
```

2° OPCIÓN:

```
MOV AL, [200H]
MOV BL, [201H]
ADD AL, BL
MOV [202H], AL
INT 20H
```

Ejercicio: *Implementar un programa en ensamblador que sume la localidad de memoria 200 y la 201, el resultado lo deje en la localidad 202.*

3° OPCION:

```
MOV AL, [200H]
ADD AL, [201H]
MOV [202H], AL
INT 20H
```

4° OPCION:

```
MOV AX, [200H]
ADD AL, AH
MOV [202H], AL
INT 20H
```


ORGANIZACIÓN DE LA MEMORIA

ORGANIZACIÓN DE LA MEMORIA

El espacio de direccionamiento de un sistema basado en un microprocesador, se denomina memoria lógica o memoria física. La estructura de la memoria lógica es diferente, en casi todos los casos, que la estructura de la memoria física. La memoria lógica es el sistema de memoria tal como lo ve el programador, mientras que la memoria física es la estructura real en el hardware en el sistema de memoria.

Memoria lógica.

El espacio básico de la memoria lógica es el mismo en todos los microprocesadores Intel. La memoria lógica se enumera por bytes. En la siguiente ilustración se observa el mapa de memoria lógica de algunoss miembros de la familia Intel. Se vera que la única diferencia es que algunos miembros contienen mas memoria que otros. Además, se debe tener en cuenta que la memoria física puede diferir de la memoria lógica en muchos sistemas.

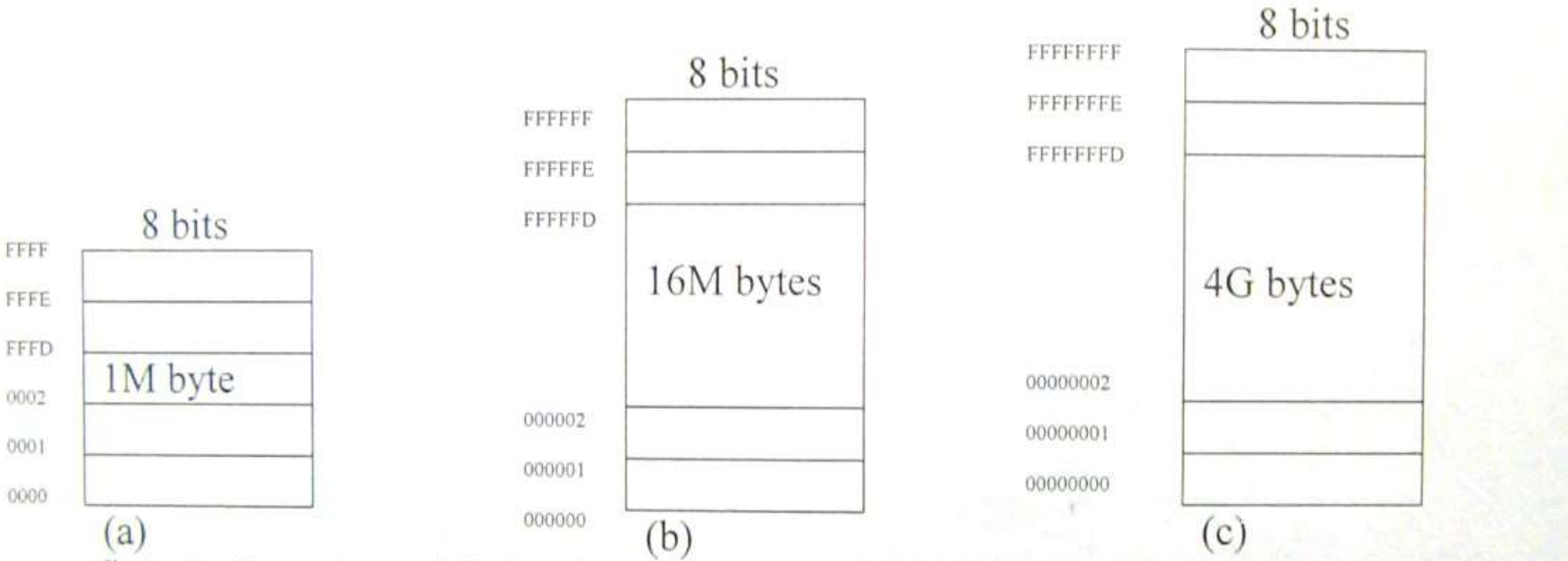


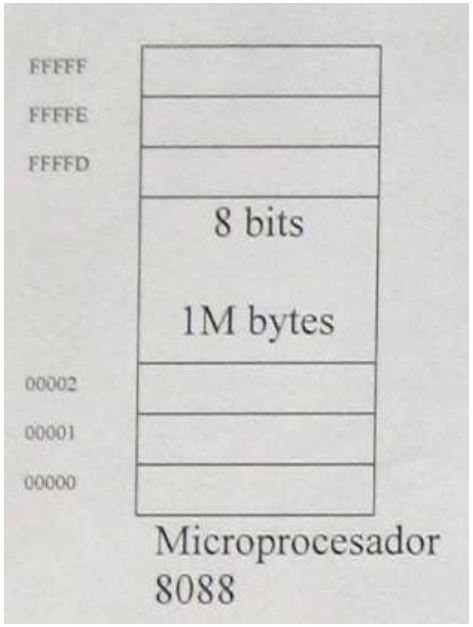
Ilustración: El mapa de memoria lógica de los microprocesadores: (a) 8086/8088/80186 (b) 80286/80386SX (c) 80386DX y 80486

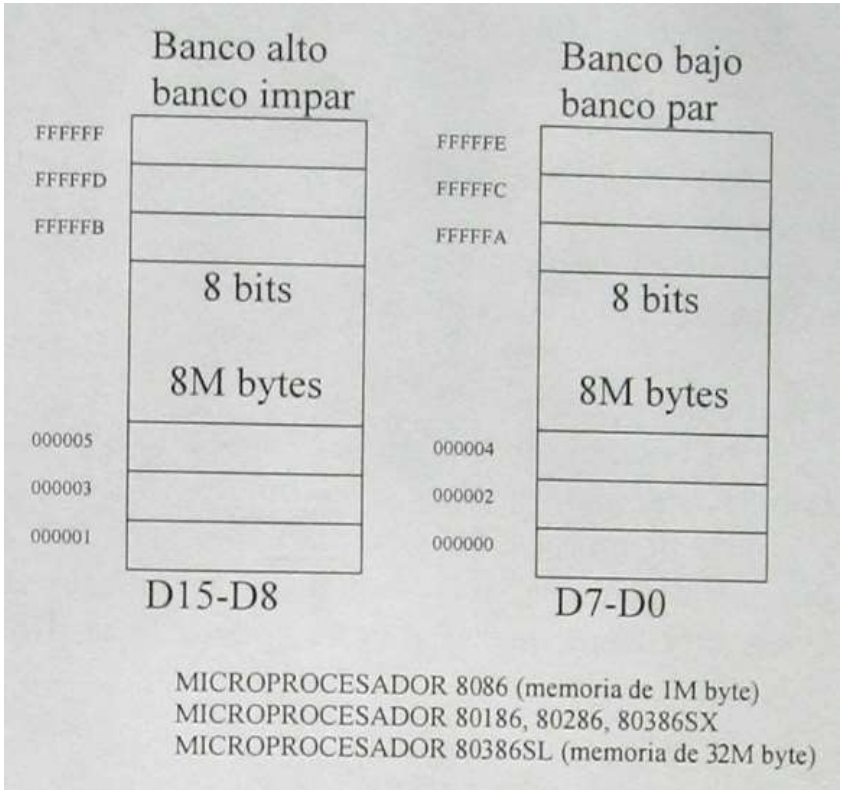
Cuando estos microprocesadores direccionan una palabra de 16 bits en la memoria, se accesan dos bytes consecutivos. Por ejemplo, la palabra en la localidad 00122H se almacena en los bytes 00122H y 00123H; el byte menos significativo se almacena en la localidad 00122H. Si se accesa a una palabra de 32 bits, esta palabra doble la contiene cuatro bytes consecutivos. Por ejemplo. La doble palabra almacenada en la localidad 00120H, se almacena en los bytes 00120H, 00121H, 00122H y 00123H; el byte menos significativo se almacena en 00120H y el byte más significativo en 00123H.

Memoria Física.

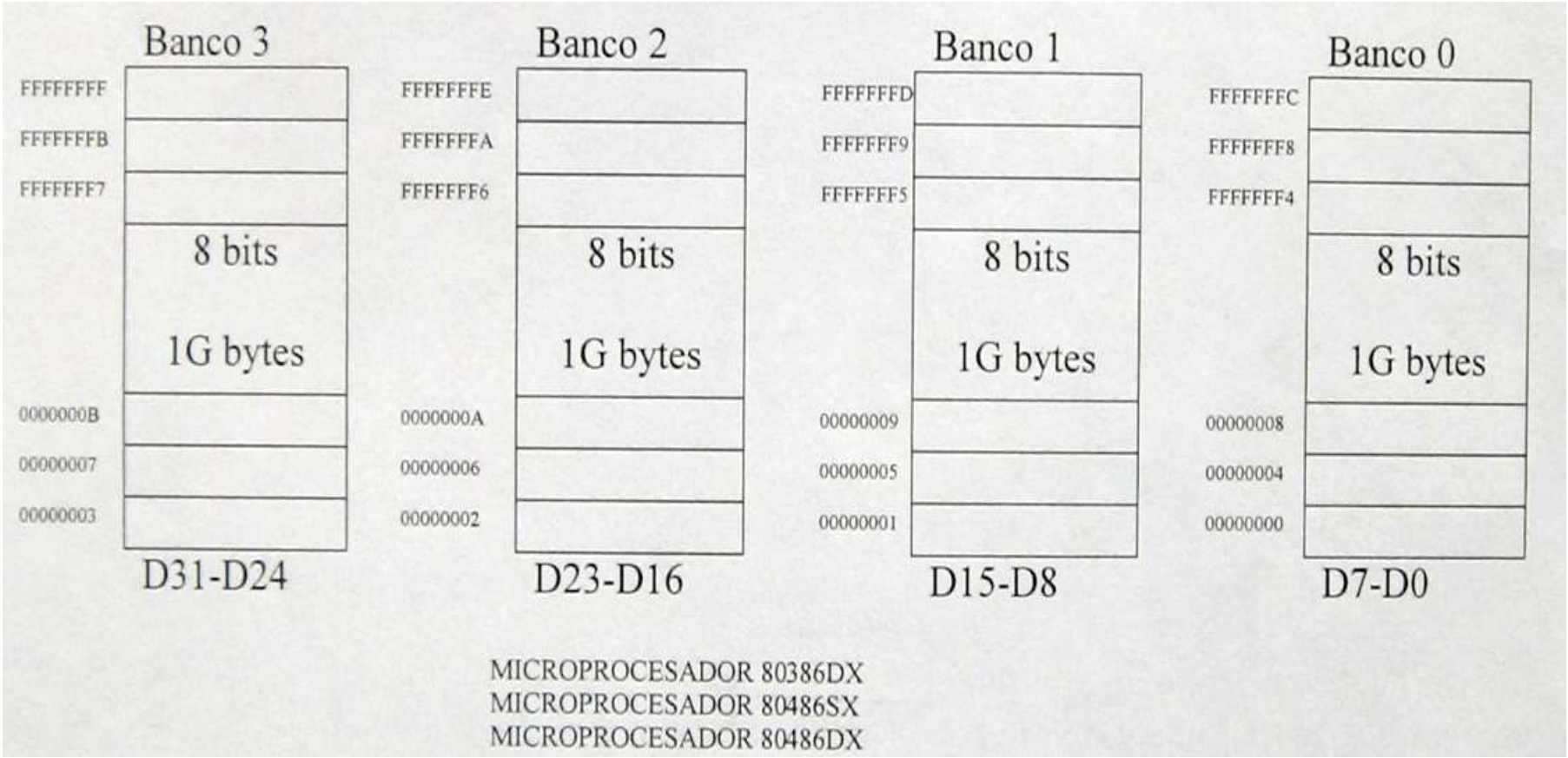
Las memorias físicas de los miembros de la familia Intel difieren del ancho. La memoria del 8088 es de 8 bits de ancho; las memorias 8086, 80186, 80286 y 80386SX tienen 16 bits de ancho; las memorias del 80386dx y 80486 son de 32 bits de ancho. Para la programación, no hay diferencia en el ancho de la memoria porque la memoria lógica siempre es de 8 bits de ancho; pero, como se puede ver en la siguiente ilustración, hay una diferencia para el diseñador del hardware.

La memoria esta organizada en bancos de memoria en todas las versiones del microprocesador excepto el 8088 que tiene un solo banco de memoria. Un *banco de memoria* es una sección de 8 bits de ancho. Los microprocesadores de 16 bits tienen dos bancos de memoria para formar una sección de memoria de 16 bits de ancho, a la cual se direcciona por bytes o por palabras.



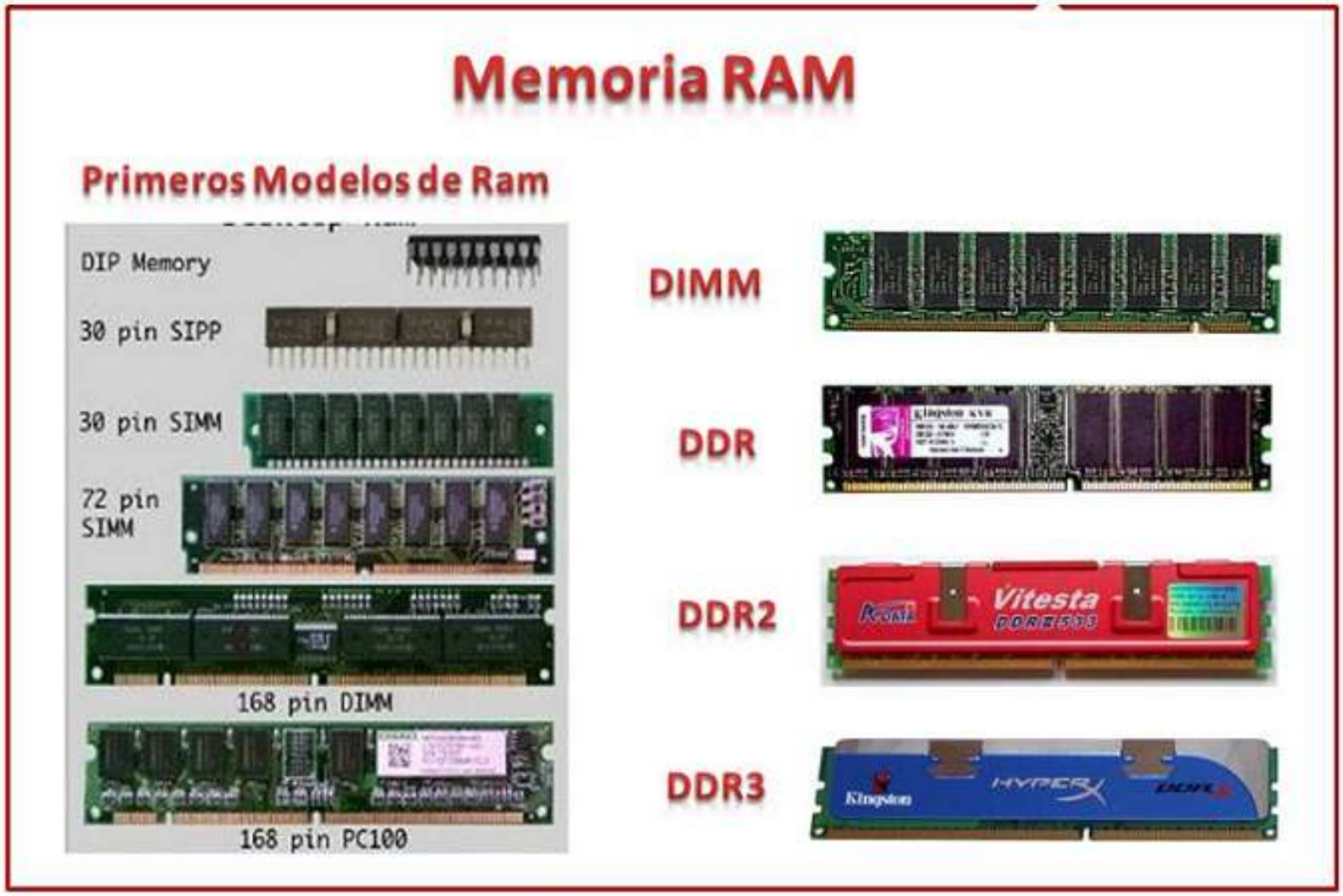


Los microprocesadores de 32 bits tienen cuatro bancos de memoria, pero se les direcciona como bytes, palabras o dobles palabras.



MICROPROCESADOR 80386DX
MICROPROCESADOR 80486SX
MICROPROCESADOR 80486DX

EJEMPLOS DE BANCOS DE MEMORIA USADOS



EJEMPLO DE MAPA DE MEMORIA DE LAS PRIMERAS IBM PCs Mod. XT con CPUs INTEL 8088/8086

PC/XT Memory Map

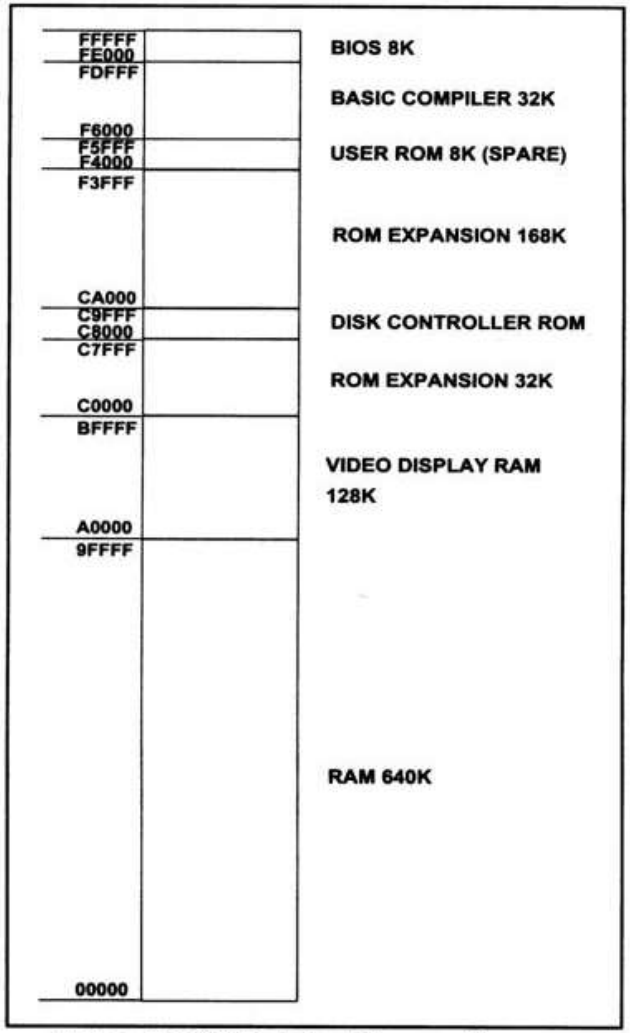
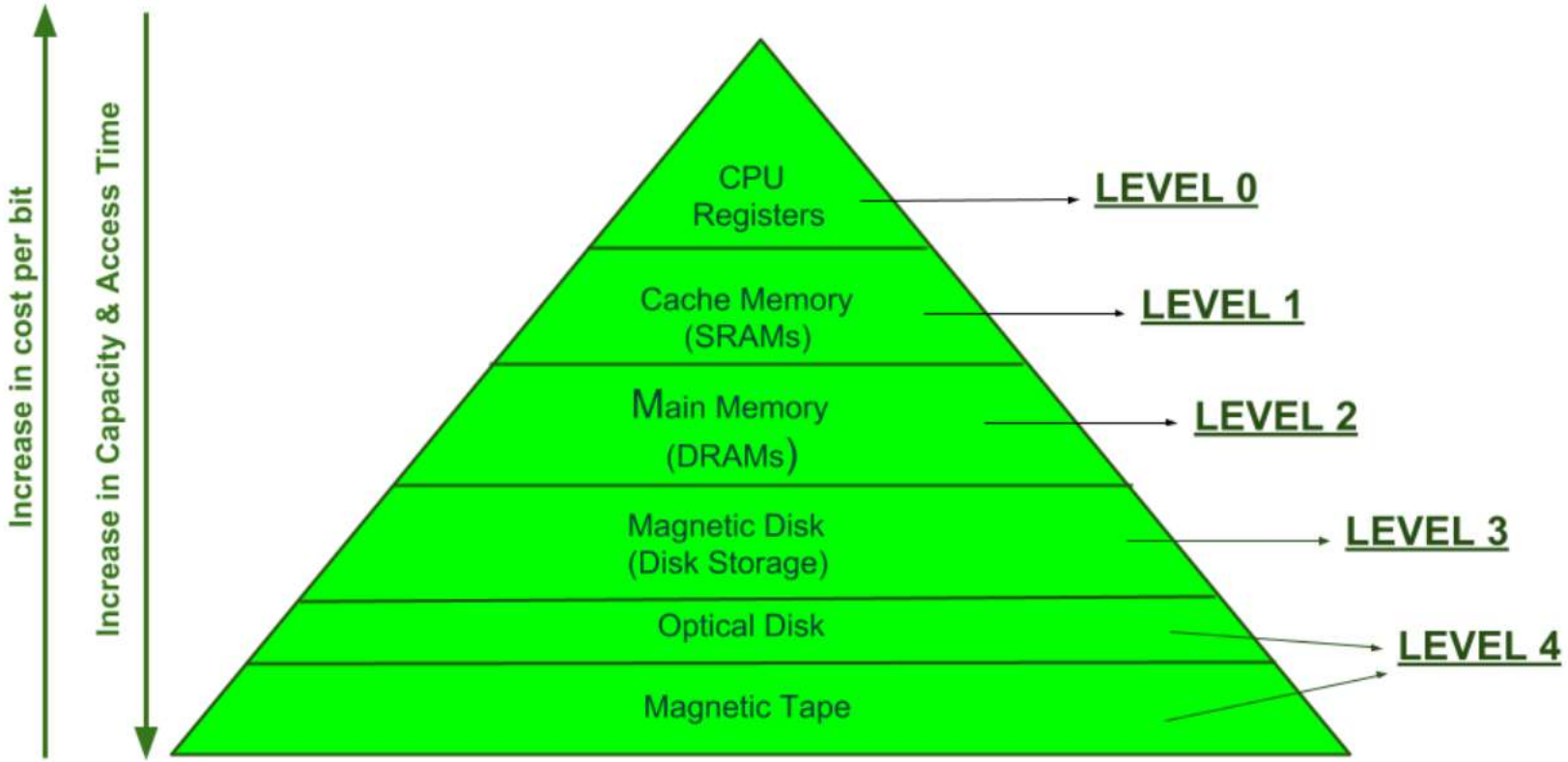


Figure . PC.XT Detailed Memory Map



MEMORY HIERARCHY DESIGN

ESTRUCTURA DE SEGMENTACIÓN DE MEMORIA

El 8086 usa un **esquema** llamado **segmentación**, para acceder correctamente a un megabyte completo de memoria, **con referencias de direcciones de sólo 16 bits**, y todo esto gracias a la utilización de registros de segmento que dividen esencialmente el espacio de memoria en **segmentos de 64KB de longitud**, que **pueden estar separados entre sí, adyacentes o superpuestos**, y que **comienzan en una dirección divisible por 16**.

La **forma como se completan los 20 bits del bus de direcciones**, disponiendo en la CPU, solamente, registros de 16 bits, se consigue **de la siguiente manera**:

Se parte del contenido de uno de los registros de segmento, que actúan como base.

Después, **se multiplica por 16** el contenido del registro de segmento, lo que, en binario, significa añadirle 4 ceros a la derecha y convertirlo en una magnitud de 20 bits.

Finalmente, **se suma un desplazamiento** al resultado de la multiplicación anterior.

Abreviadamente, la fórmula para calcular una dirección de memoria es:

Dirección Física = 16 * (registro de segmento) + desplazamiento.

FIN DE PRESENTACIÓN

APENDICES

EJERCICIOS Y NOTAS: