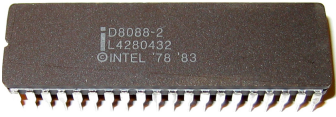


Intel 8086 y 8088

Intel 8088 Microprocesador	
 <p>Microprocesador Intel 8088</p>	
Producción	1979 — 1982
Fabricante(s)	Intel, AMD, NEC, Fujitsu, Harris (Intersil), OKI, Siemens AG, Texas Instruments, Mitsubishi.
Frecuencia de reloj de CPU	5 MHz a 10 MHz
Conjunto de instrucciones	x86-16
Zócalo(s)	40 pin DIP

El 8088 fue el microprocesador usado para el primer computador personal de IBM, el IBM PC, que salió al mercado en agosto de 1981.

Historia

Antecedentes

En 1972, Intel lanzó el 8008, el primer microprocesador de 8 bits.^[1] El 8008 implementó un conjunto de instrucciones diseñado por la corporación Datapoint la cuál tenía en mente hacer terminales de CRT programables. El 8008 también demostró ser bastante de propósito general. El dispositivo necesitó varios ICs adicionales para producir un computador funcional, en parte debido a su pequeño "paquete de memoria" de 18 pines, que eliminó el uso de un bus de direcciones separado (En ese tiempo, Intel era primariamente un fabricante de DRAM).

Dos años más tarde, en 1974, Intel lanzó el 8080,^[2] empleando los nuevos paquetes DIL de 40 pines desarrollados originalmente para ICs de calculadora para permitir un bus de direcciones separado. Tenía un conjunto de instrucciones extendido que era compatible a nivel de código fuente, no de código de máquina binario, con el 8008 y también incluyó algunas instrucciones de 16 bits para hacer la programación más fácil. El dispositivo 8080, con frecuencia descrito como el primer microprocesador verdaderamente útil, fue eventualmente substituido por el 8085, basado en tecnología depletion-load NMOS (1977) que podía trabajar con una sola fuente de alimentación de 5V en vez de los tres diferentes voltajes de funcionamiento de los chips anteriores.^[3] Otros microprocesadores de 8 bits bien conocidos que emergieron durante estos años fueron el Motorola 6800 (1974), MOS Technology 6502 (1975), Zilog Z80 (1976), y Motorola 6809 (1978), así como otros.

El primer diseño x86

El proyecto 8086 comenzó en mayo de 1976 y fue pensado originalmente como un sustituto temporal para el ambicioso y retrasado proyecto del iAPX 432. Era un intento de robar la atención de los menos retrasados procesadores de 16 y 32 bits de los otros fabricantes (tales como Motorola, Zilog, y National Semiconductor) y al mismo tiempo contrarrestar la amenaza del Zilog Z80, que llegó a ser muy exitoso (diseñado por anteriores empleados de Intel). Por lo tanto, tanto la arquitectura y el chip físico fueron desarrollados algo rápidamente por un pequeño grupo de personas, y usando los mismos elementos básicos de la microarquitectura y técnicas físicas de implementación que fueron empleadas para el ligeramente más viejo 8085 (y para el cuál el 8086 también funcionaría como una continuación).

Mercadeado como compatible a nivel de código fuente, el 8086 fue diseñado de modo que el lenguaje ensamblador para el 8008, 8080, o el 8085 pudiera ser convertido automáticamente en (subóptimo) código fuente equivalente del 8086, con poca o ninguna edición a mano. Para hacer esto posible, el modelo de programación y el conjunto de instrucciones fueron (flojamente)

basados en el 8080. Sin embargo, el diseño del 8086 fue ampliado para soportar el completo procesamiento de 16 bits, en vez de las bastante básicas capacidades de 16 bits del 8080/8085.

También fueron agregadas nuevas clases de instrucciones; soporte total para enteros con signo, direccionamiento de base + offset, y las operaciones auto-repetidas fueron semejantes a las del diseño del Z80,^[4] pero todas fueron hechas levemente más generales en el 8086. También fueron agregadas instrucciones que soportaban las funciones anidadas de la familia de lenguajes ALGOL tales como Pascal y PL/M. De acuerdo al principal arquitecto Stephen P. Morse, esto fue un resultado de un acercamiento más centrado en el software que en el que hubo en el diseño de procesadores anteriores de Intel (los diseñadores tenían experiencia trabajando con implementaciones de compiladores). Otras mejoras incluyeron instrucciones de multiplicación y división por microcódigo y una estructura de bus mejor adaptada para futuros coprocesadores (tales como el 8087 y el 8089) y a los sistemas de multiprocesadores.

La primera revisión del conjunto de instrucciones y la arquitectura de alto nivel estaba lista después de cerca de tres meses,^[5] y como no fue usada casi ninguna herramienta CAD, cuatro ingenieros y 12 personas de diagramación estaban simultáneamente trabajando en el chip.^[6] El 8086 tomó un poco más de dos años desde la idea hasta el producto trabajando, lo que fue considerado algo rápido para un complejo diseño en 1976-1978.

El 8086 fue secuenciado^[7] usando una mezcla al azar de lógica y microcódigo y fue implementado usando circuitería de depletion load nMOS con aproximadamente 20.000 transistores activos (29.000 contando todos los sitios del ROM y el PLA). Pronto fue movido a un nuevo proceso de fabricación refinado de nMOS llamado HMOS (por High performance MOS) que Intel desarrolló originalmente para la fabricación de productos de RAM estática rápida.^[8] Esto fue seguido por versiones de HMOS-II, HMOS-III, y, eventualmente, una versión completamente estática de CMOS para dispositivos energizados con baterías, manufacturados usando los procesos de CHMOS de Intel.^[9] El chip original midió 33 mm² y el mínimo tamaño fue de 3.2 µm.

La arquitectura fue definida por Stephen P. Morse con cierta ayuda y asistencia de Bruce Ravenel (el arquitecto del 8087) en refinar las revisiones finales. Los diseñadores de la lógica Jim McKeivitt y John Bayliss fueron los ingenieros que encabezaban el equipo de desarrollo de nivel de hardware,^[10] y William Pohlma el gerente para el proyecto. La herencia del 8086 perdura en el conjunto de instrucción básico de los computadores personales y servidores de hoy; el 8086 también prestó sus últimos dos dígitos a posteriores versiones extendidas de diseño, tales como el Intel 286 y el Intel 386, todas las cuales eventualmente serían conocidas como la familia x86. (Otra referencia es que la identificación de vendedor del PCI para los dispositivos de Intel es 8086_h !)

```

-u 100 1a
OCFD:0100 BA0B01 MOV DX,010B
OCFD:0103 B409 MOV AH,09
OCFD:0105 CD21 INT 21
OCFD:0107 B400 MOV AH,00
OCFD:0109 CD21 INT 21
-d 10b 13f
OCFD:0100 20 65 73 74 65 20 65 73-20 75 6E 20 70 72 6F 67 48 6F 6C 61 2C
OCFD:0110 72 61 6D 61 20 68 65 63-68 6F 20 65 6E 20 61 73
OCFD:0120 73 65 6D 62 6C 65 72 20-70 61 72 61 20 6C 61 20
OCFD:0130 57 69 6B 69 70 65 64 69-61 24
OCFD:0140

Hola,
este es un prog
rama hecho en as
sembler para la
Wikipedia$

```

Lenguaje de máquina del Intel 8088. El código de máquina en hexadecimal se resalta en rojo, el equivalente en lenguaje ensamblador en magenta, y las direcciones de memoria donde se encuentra el código, en azul. Abajo se ve un texto en hexadecimal y ASCII.

Estructura interna

La Unidad de Interfaz del Bus y la Unidad de ejecución

El 8086 y el 8088 tienen internamente dos componentes, la Unidad de Interfaz del Bus y la Unidad de ejecución (Bus Interface Unit (BIU) y Execution Unit (EU)).

- La Unidad de Ejecución procesa las instrucciones del CPU. Está conformada por los registros generales, los registros índice y apuntadores, los flags, la unidad aritmético lógica, y la lógica de control que maneja todo el proceso para ejecutar las instrucciones.
- La Unidad de Interfaz del Bus maneja la lectura y escritura desde y hacia la memoria y los puertos de entrada/salida. Está conformada por los registros de segmento, una cola de 4 bytes para instrucciones en el 8088 y de 6 en el 8086, y lógica para controlar los buses externos del microprocesador.

En la figura de la derecha, la Unidad de Ejecución se encuentra en la parte de abajo y la Unidad de Interfaz del Bus está en la parte superior. Las dos están interconectadas mediante un bus interno.

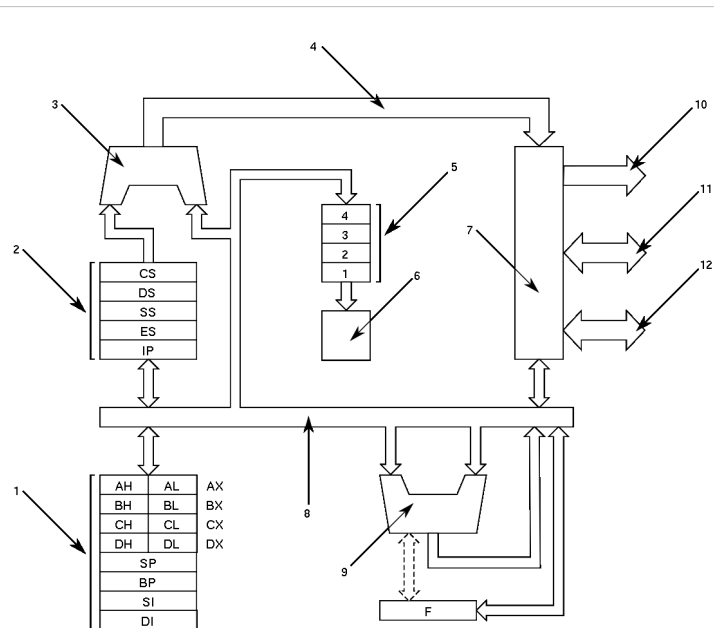


Diagrama de bloque de los microprocesadores Intel 8086 y 8088.

- 1 Bloque de registros de propósito general
- 2 Bloque de registros de segmento y registro IP
- 3 Sumador de direcciones
- 4 Bus de direcciones interno
- 5 Cola de instrucciones (4 bytes para el 8088 y 6 bytes para el 8086)
- 6 Unidad de control (muy simplificada)
- 7 Interfaz del bus
- 8 Bus de datos interno
- 9 Unidad aritmético lógica (ALU)
- 10, 11, 12 Bus de direcciones, datos y control externos

Registros

Registros de propósito general		
AH	AL	AX (Acumulador)
BH	BL	BX (Base)
CH	CL	CX (Contador)
DH	DL	DX (Datos)
Registros índices		
	SI	Source Index (Índice origen)
	DI	Destination Index (Índice Destino)
	BP	Base Pointer (Puntero Base)
	SP	Stack Pointer (Puntero de Pila)
Registro de Bandera		
- - - -	O D I T S Z - A - P - C	Flags (Banderas)
Registros de Segmentos		
	CS	Code Segment (Segmento de Código)
	DS	Data Segment (Segmento de Datos)
	ES	ExtraSegment (Segmento Extra)
	SS	Stack Segment (Segmento de Pila)
Registro apuntador de instrucciones		
	IP	Instruction Pointer
Modelo de los registros		

Los registros del i8086 e i8088 se basaron en el diseño del Intel 8080 y el Intel 8085, y de hecho son compatibles a nivel de lenguaje ensamblador con el i8080. El conjunto de registros también es similar al del i8080, pero ampliados a 16 bits. Tanto el i8086 como el i8088 tienen cuatro registros de propósito general de 16 bits, que también pueden ser accedidos como ocho registros de 8 bits, y tienen cuatro registros índice de 16 bits (incluyendo el puntero de pila). Los registros de datos se usan a veces de forma implícita por las instrucciones, haciendo más difícil la organización de los registros para emplearlos con valores temporales.

Los registros del procesador, se usan para contener los datos con que se está trabajando puesto que el acceso a los registros es mucho más rápido que los accesos a memoria. Se pueden realizar operaciones aritméticas y lógicas, comparaciones, entre otras. Se pueden hacer estas operaciones con todos los registros excepto los de segmento, el IP, y los flags.

Registros de Propósito General

Los registros de propósito general son el AX, BX, CX, y DX, de 16 bits. Cada uno de ellos se divide en dos registros de 8 bits, llamados AH y AL, BH y BL, CH y CL, y, DH y DL, H significando Hight (alto) y L significando Low (bajo), indicando la parte alta o la parte baja del registro correspondiente de 16 bits (ver esquema). Un programa podía usar tanto los registros de 16 bits como los registros de 8 bits. Aparte del uso general de los registros para hacer cálculos aritméticos y lógicos, existen instrucciones que usan estos registros con un uso particular especializado, como se indica a continuación:

- **Registro AX:** El registro AX es el registro acumulador, es utilizado para operaciones que implican entrada/salida, y multiplicación y división (estas dos últimas en conjunto con el registro DX)
- **Registro BX:** El registro BX es el registro base, y es el único registro de propósito general que puede ser un índice para direccionamiento indexado
- **Registro CX:** El registro CX es conocido como el registro contador. Puede contener un valor para controlar el número de veces que un ciclo se repite o un valor para corrimiento de bits
- **Registro DX:** El registro DX es el registro de datos. En algunas operaciones se indica mediante este registro el número de puerto de entrada/salida, y en las operaciones de multiplicación y división de 16 bits se utiliza junto con el acumulador AX

Registros Índice

Los registros SI y DI están disponibles para direccionamiento indexado y para operaciones de cadenas de caracteres.

- **Registro SI:** El registro índice fuente de 16 bits es requerido por algunas operaciones con cadenas de caracteres. El SI está asociado con el segmento DS.
- **Registro DI:** El registro índice destino también es requerido por algunas operaciones con cadenas de caracteres. El DI está asociado con el segmento ES.

Registros Apuntadores

Los registros SP (apuntador de pila) y BP (apuntador base) están asociados con el registro SS y permiten al sistema acceder a datos en el segmento de la pila.

- **Registro SP:** El apuntador de pila de 16 bits está asociado con el segmento SS y proporciona un valor de desplazamiento que se refiere a la palabra actual que está siendo procesada en la pila. El sistema maneja de manera automática este registro, aunque el programa puede hacer ciertas manipulaciones con él.
- **Registro BP:** El apuntador base de 16 bits facilita la referencia de parámetros dentro de la pila.

Registros de Banderas

Es un registro de 16 bits, de los cuales nueve sirven para indicar el estado actual de la máquina y el resultado del procesamiento. Muchas instrucciones aritméticas y de comparación cambian el estado de las banderas y apoyándose en ellas se pueden tomar decisiones para determinar la acción subsecuente.

La tabla contiene 16 posiciones (de 0 a 15), que son los 16 bits del registro de banderas, numeradas de derecha a izquierda. La posición 0 la encontraremos a la derecha y la posición 15 a la izquierda.

-	-	-	-	OF	DF	IF	TF	SF	ZF	-	AF	-	PF	-	CF
---	---	---	---	----	----	----	----	----	----	---	----	---	----	---	----

Los bits de las banderas son las siguientes:

- **OF (overflow, desbordamiento):** Indica desbordamiento del bit de mayor orden después de una operación aritmética de números con signo (1=existe overflow; 0=no existe overflow). Para operaciones sin signo, no se toma en cuenta esta bandera.
- **DF (dirección):** Controla la selección de incremento o decremento de los registros SI y DI en las operaciones con cadenas de caracteres (1=decremento automático; 0=incremento). La bandera DF se controla con las instrucciones STD y CLD.
- **IF (interrupción):** Controla el disparo de las interrupciones (1=habilita las interrupciones; 0=deshabilita las interrupciones). La interrupción no enmascarable es la única que no puede ser bloqueada por esta bandera. El estado de la bandera IF se controla con las instrucciones STI y CLI.
- **TF (trampa):** Permite la operación del procesador en modo de depuración (paso a paso)
- **SF (signo):** Contiene el signo resultante de una operación aritmética (0=positivo; 1=negativo).

- **ZF (cero):** Indica el resultado de una operación aritmética o de comparación (0=resultado diferente de cero; 1=resultado igual a cero).
- **AF (acarreo auxiliar):** Contiene el acarreo del bit 3. Esta bandera se prueba con las instrucciones DAA y DAS para ajustar el valor de AL después de una suma o resta BCD.
- **PF (paridad):** Indica si el número de bits 1, del byte menos significativos de una operación, es par (0=número de bits 1 es impar; 1=número de bits 1 es par).
- **CF (acarreo):** Contiene el acarreo del bit de mayor orden después de una operación aritmética; también almacena el contenido del último bit en una operación de desplazamiento o de rotación.

Registros de Segmento

Definen áreas de 64 Kb dentro del espacio de direcciones de 1 Mb del 8086. Estas áreas pueden solaparse total o parcialmente. No es posible acceder a una posición de memoria no definida por algún segmento: si es preciso, habrá de moverse alguno.

- **Registro CS:** El DOS almacena la dirección inicial del segmento de código de un programa en el registro CS. Esta dirección de segmento, más un valor de desplazamiento en el registro apuntador de instrucción (IP), indica la dirección de una instrucción que es buscada para su ejecución. Para propósitos de programación normal, no se necesita referenciar el registro CS.
- **Registro DS:** La dirección inicial de un segmento de datos de programa es almacenada en el registro DS. Esta dirección, más un valor de desplazamiento en una instrucción, genera una referencia a la localidad de un byte específico en el segmento de datos.
- **Registro SS:** El registro SS permite la colocación en memoria de una pila, para almacenamiento temporal de direcciones y datos. El DOS almacena la dirección de inicio del segmento de pila de un programa en el registro SS. Esta dirección de segmento, más un valor de desplazamiento en el registro del apuntador de la pila (SP), indica la palabra actual en la pila que está siendo direccionada. Para propósitos de programación normal, no se necesita referenciar el registro SS.
- **Registro ES:** Algunas operaciones con cadenas de caracteres utilizan el registro extra de segmento para manejar el direccionamiento de memoria. El registro ES está asociado con el registro DI (Índice). Un programa que requiere el uso del registro ES puede inicializarlo con una dirección de segmento apropiada.

Registro Apuntador de Instrucciones

El registro IP de 16 bits contiene el desplazamiento de dirección de la siguiente instrucción que se ejecuta. El IP está asociado con el registro CS en el sentido de que el IP indica la instrucción actual dentro del segmento de código que se está ejecutando actualmente en la memoria.

Acceso a memoria y a puertos

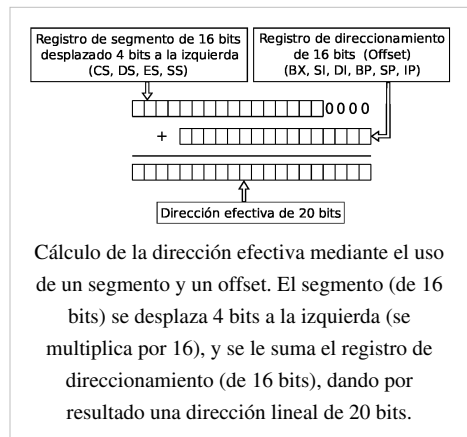
Ambos microprocesadores tienen un rango de 65536 direcciones de entrada/salida que se pueden acceder como puertos de 8 ó 16 bits. En la parte baja de la memoria hay 256 vectores de interrupción.

Estos procesadores usaban 20 bits de dirección que les permitía acceder hasta 1 MB de memoria. Sus registros internos eran de sólo 16 bits, por lo que se desarrolló un mecanismo usando registros de segmento para poder acceder al megabyte de memoria. El 8086 y el 8088 tenían cuatro registros de segmento de 16 bits (CS, DS, ES y SS). En lugar de suministrar los 4 bits faltantes para direccionar los 20 bits, como la mayoría de los procesadores segmentados, el 8086 y el 8088 desplazan el contenido del registro de segmento 4 bits hacia la izquierda y lo suman a una dirección de memoria de 16 bits proveniente de registros índice (BX, SI, DI, BP, SP, IP) y opcionalmente una constante, siendo el resultado la dirección efectiva. Esto suele ser considerado como un mal diseño, aunque puede ser aceptable, e incluso útil en el lenguaje ensamblador. Por el contrario, provoca confusión cuando se hace un uso extensivo de los punteros (como en el lenguaje C), debido a que es posible que dos punteros con diferentes valores apunten a la misma dirección de memoria. Este esquema de segmentos se cambió en el Intel 80286 y luego en el

Intel 80386.

Segmentación

Había también cuatro registros de 16 bits de segmento que permitían al 8086 y 8088 tener acceso a un megabyte de memoria de una manera inusual. En vez de concatenar el registro de segmento con el registro de dirección, como en la mayoría de los procesadores cuyo espacio de dirección excedía su tamaño de registro, el 8086 y el 8088 desplazaban el segmento de 16 bits sólo cuatro bits hacia la izquierda antes de sumarlo un offset de 16 bits ($16 \times \text{segmento} + \text{offset}$), produciendo una dirección externa (efectiva o física) de 20 bits a partir del par segmento:offset de 32 bits. Consecuentemente, cada dirección externa podía ser referida por $2^{12} = 4096$ diferentes pares segmento:offset. La separación de 16 bytes entre las bases del segmento (debido al desplazamiento de 4 bits) fue llamada un *párrafo*. Aunque fue considerado complicado e incómodo por muchos programadores, este esquema también tenía ventajas; un pequeño programa (menos de 64 KB) podía ser cargado comenzando en un offset fijo (como 0) en su propio segmento, evitando la necesidad de relocación, con a lo más 15 bytes de alineación desperdiciados.



Los compiladores para la familia 8086 comúnmente soportaban dos tipos de punteros, cerca y lejos (near y far). Los punteros near eran offset de 16 bits implícitamente asociados con el segmento del código y/o de datos del programa y así podían ser usados sólo dentro de partes de un programa suficientemente pequeño para caber en un segmento. Los punteros far eran pares segmento:offset de 32 bits que se resolvían en direcciones externas de 20 bits. Algunos compiladores también soportaban punteros enormes (huge), que eran como los punteros far salvo que la aritmética de puntero en un puntero huge lo trataba como un puntero lineal de 20 bits, mientras que la aritmética de puntero en un puntero huge daba vueltas (wrapped around) dentro de su offset de 16 bits sin tocar la parte del segmento de la dirección.

Para evitar la necesidad de especificar near y far en numerosos punteros, estructuras de datos, y funciones, los compiladores también soportan los "modelos de memoria" que especifican tamaños de puntero por defecto. Los modelos minúsculo (tiny) (64K máximo), pequeño (small) (128K máximo), compacto (compact) (datos > 64K), medio (medium) (código > 64K), grande (large) (código y datos > 64K), y enorme (huge) (arreglos individuales > 64K), cubrían las combinaciones de punteros near, far y enorme para código y datos. El modelo tiny significaba que el código y los datos estaban compartidos en un sólo segmento, justo como en la mayoría de los procesadores basados en 8 bits, y podía ser usado para construir archivos .com por ejemplo. Las bibliotecas precompiladas vinieron a menudo en varias versiones compiladas para diversos modelos de memoria.

Según Morse y otros, los diseñadores contemplaban realmente usar un desplazamiento de 8 bits (en vez de 4 bits), para crear un espacio de dirección física de 16 MB. Sin embargo, pues que esto habría forzado a los segmentos comenzar límites de 256 bytes, y, alrededor de 1976, 1 MB era considerado muy grande para un microprocesador, la idea fue descartada. También, no había suficientes pines disponibles en un paquete barato de 40 pines.^[11]

En principio, el espacio de dirección de la serie x86 pudo haberse extendido en procesadores posteriores, aumentando el valor del desplazamiento, mientras las aplicaciones obtuvieran sus segmentos del sistema operativo y no hicieran asunciones sobre la equivalencia de diferentes pares segmento:offset.^[12] En la práctica el uso de punteros "huge" y de mecanismos similares era amplio y la dirección plana de 32 bits se hizo posible, con los registros de offset de 32 bits en el 80386, eventualmente extender el límite del rango de direcciones en una forma más general.

Portando software viejo

Los pequeños programas podían ignorar la segmentación y sólo usar la dirección plana de 16 bits. Este permitía al software de 8 bits ser portado con bastante facilidad al 8086 y 8088. Los autores del MS-DOS tomaron ventaja de esto proporcionando una Interface de programación de aplicaciones muy similar a la del CP/M así como incluyendo el simple formato de archivo ejecutable.com, idéntico al del CP/M. Esto era importante cuando el 8086/8088 y el MS-DOS eran nuevos, porque permitió que muchas aplicaciones existentes de CP/M (y otros) hacerse rápidamente disponibles, facilitando grandemente la aceptación de la nueva plataforma.

Dirección de la primera instrucción ejecutable: FFFF:0

Los procesadores 8088 y 8086, por diseño de su hardware, ejecutaban su primera instrucción en la dirección FFFF:0 (16 bytes por abajo del tope de su capacidad de memoria de 1 MB con sus 20 bits de direccionamiento). En esta área debe haber una ROM para poder ejecutar sus instrucciones al encender o reiniciar el computador (o dispositivo). En el caso del IBM PC, en esa área estaba el IBM PC ROM BIOS, y la primera instrucción que éste ejecutaba era un salto (JMP) al inicio del código del Power On Self Test (POST), donde había código para revisar el CPU y revisar e inicializar los diferentes componentes del hardware y el propio BIOS del computador, y al final se ejecutaba Boot Strap Loader, que iniciaba el Bootstrap.^[13] Los microprocesadores de la línea x86 han heredado esta dirección de memoria (FFFF:0) para la primera instrucción ejecutable.

Modos de direccionamiento

Los procesadores 8086 y 8088 tenían los siguientes modos de direccionamiento:

- **Implícito.** El dato está implícito en la propia instrucción. Ej. STC, STD y STI, (Set Carry, Set Direction 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 sólo 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]

Patillaje

Pines del 8086 y del 8088. Las líneas del bus de direcciones se ven en rojo, las del bus de datos en azul y las del bus de control en verde. Las líneas del bus de energía se ven en negro. Estos procesadores multiplexan en tiempo el bus de direcciones, con el bus de datos y control. En el 8086 se ven los pines del 2 al 16 y los pines 35 al 39 con doble funcionalidad, en un momento determinado transporta la dirección y en otro momento entran o salen los datos (o sale información de algunas líneas del bus de control).

En el 8088 se comparten los pines 9 al 16 entre el bus de direcciones y de datos, y los pines 35 al 38 entre el bus de direcciones y el de control.

Pines del 8086

+---_/---+			
GND	1	40	Vcc (+5V)
<--- A14 <-> D14	2	39	A15 --> D15 <->
<--- A13 <-> D13	3	38	A16 --> S3 -->
<--- A12 <-> D12	4	37	A17 --> S4 -->
<--- A11 <-> D11	5	36	A18 --> S5 -->
<--- A10 <-> D10	6	35	A19 --> S6 -->
<--- A9 <-> D9	7	34	!BHE/S7 -->
<--- A8 <-> D8	8	33	MN/!MX <--
<--- A7 <-> D7	9	Intel 32	!RD -->
<--- A6 <-> D6	10	8086 31	HOLD, !RQ/!GTO <->
<--- A5 <-> D5	11	30	HLDA, !RQ/!GT1 <->
<--- A4 <-> D4	12	29	!WR, !LOCK -->
<--- A3 <-> D3	13	28	M/!IO, !S2 -->
<--- A2 <-> D2	14	27	DT/!R, !S1 -->
<--- A1 <-> D1	15	26	!DEN, !S0 -->
<--- A0 <-> D0	16	25	ALE, QS0 -->
--> NMI	17	24	!INTA, QS1 -->
--> INTR	18	23	!TEST <--
--> CLK	19	22	READY <--
GND	20	21	RESET <--

Pines del 8088

Además del bus externo de datos, que se reduce a 8 bits, la diferencia con el 8086 es mínima)

+---_/---+			
GND	1	40	Vcc (+5V)
<--- A14	2	39	A15 -->
<--- A13	3	38	A16 --> S3 -->
<--- A12	4	37	A17 --> S4 -->
<--- A11	5	36	A18 --> S5 -->
<--- A10	6	35	A19 --> S6 -->
<--- A9	7	34	!SSO....HIGH, -->
<--- A8	8	33	MN/!MX <--
<-> D7 <--- A7	9	Intel 32	!RD -->
<-> D6 <--- A6	10	8088 31	HOLD....!RQ/!GTO <->
<-> D5 <--- A5	11	30	HLDA....!RQ/!GT1 <->

```

<-> D4 <-- A4 12 |          | 29 !WR.....!LOOK  -->
<-> D3 <-- A3 13 |          | 28 M/!IO...!S2    -->
<-> D2 <-- A2 14 |          | 27 DT/!R...!S1    -->
<-> D1 <-- A1 15 |          | 26 !DEN.....!S0    -->
<-> D0 <-- A0 16 |          | 25 ALE.....QS0    -->
      --> NMI 17 |          | 24 !INTA...QS1    -->
      --> INTR 18 |          | 23 !TEST              <--
      --> CLK 19 |          | 22 READY              <--
      GND 20 |          | 21 RESET              <--
      +-----+

```

Desempeño

Aunque fue opacado en parte por otras opciones de diseño en este particular chip, el bus multiplexado limitaba el desempeño ligeramente; las transferencias de cantidades de 16 bits o de 8 bits fueron hechas en ciclos de acceso a memoria de cuatro ciclos del reloj, que comparado a los típicos CPU contemporáneos de 8 bits, era más rápido cantidades de 16 bits, aunque más lento en cantidades de 8 bits. Como las instrucciones variaban de uno a seis bytes, la lectura (fetch) y la ejecución fueron hechos concurrentemente (tal y como sigue siendo en los procesadores x86 de hoy): La Unidad de Interface del Bus alimentó el flujo de instrucciones a la Unidad de Ejecución a través de una cola prefetch de 6 bytes para el 8086 y 4 bytes para el 8088 (una forma débilmente acoplada de pipeline), acelerando operaciones en los registros y con los operandos inmediatos, mientras que las operaciones de memoria desafortunadamente llegaron a ser más lentas; cuatro años más tarde, este problema de desempeño fue corregido con el 80186, 80188, y el 80286.

Sin embargo, el 8086 y el 8088 vinieron con una completa arquitectura de 16 bits, con una ALU de ancho completo, significando que las instrucciones aritméticas de 16 bits ahora podían ser realizadas en un simple ciclo del ALU, en vez de los dos ciclos usando acarreo usadas por el 8080 y el 8085, acelerando tales instrucciones considerablemente. Igualmente se tenía un mejor desempeño con las operaciones lógicas de 16 bits. Combinado con la ortogonalización de las operaciones versus los tipos de operandos y modos de direccionamiento, así como con otras mejoras, hizo bastante significativo el aumento del desempeño sobre el 8080 o el 8085, a pesar de los casos donde los chips más viejos podían ser más rápidas.

Tiempos de ejecución para las instrucciones típicas (en ciclos de reloj)^[14]

Instrucción	Registro-Registro	Registro Inmediato	Registro-Memoria	Memoria-Registro	Memoria-Inmediato
MOV	2	4	8+EA	9+EA	10+EA
ALU	3	4	9+EA,	16+EA,	17+EA
JMP	Registro => 11; Etiqueta => 15; Condición, Etiqueta => 16				
Multiplicación entera	70~160 (dependiendo del operando <i>dato</i> como del tamaño) más EA				
División entera con signo	80~190 (dependiendo del operando <i>dato</i> como del tamaño) más EA				

- EA = tiempo para computar la dirección efectiva (Effective Address (EA)), extendiéndose de 5 a 12 ciclos en el 8086 y hasta 19 en el 8088.
- Los tiempos son el mejor caso, dependiendo de estado del prefetch, la alineación de la instrucción, y de otros factores.

Como puede verse de estas tablas, las operaciones en los registros y los operandos inmediatos eran rápidas (entre 2 y 4 ciclos), mientras que las instrucciones con operandos en memoria y los JMP (salto incondicional) eran

absolutamente lentas; los saltos tomaron más ciclos que en los simples 8080 y 8085. El 8088 (utilizado en el IBM PC) fue adicionalmente obstaculizado por su más estrecho bus de datos de 8 bits. Las razones por las que la mayoría de las instrucciones relacionadas con la memoria eran lentas eran triples:

- Las unidades de lectura (fetch) y de ejecución débilmente acopladas son eficientes para el prefetch de la instrucción, pero no para los saltos (JMP) y el acceso de datos al azar (sin usar medidas especiales).
- No se ofreció un sumador dedicado para el cálculo de las direcciones; las rutinas de microcódigo tuvieron que usar el ALU principal para esto (aunque había un sumador dedicado para el segmento + offset).
- Los buses de dirección y datos estaban multiplexados, forzando un ciclo de bus levemente más largo (33~50%) que en los típicos procesadores de 8 bits contemporáneos.

Sin embargo, el desempeño del acceso a la memoria fue drásticamente mejorado con los chips de la siguiente generación de Intel. El 80186, 80188, y 80286 tenían hardware dedicado para de cálculo de direcciones, ahorrando muchos ciclos, y el 80286 también tenía adicionalmente buses de dirección y datos separados (no-multiplexados).

Chips de soporte

El 8086/8088 usan algunos circuitos integrados de soporte fabricados por Intel:

- Intel 8284. Generador de reloj
- Intel 8282. Octal Latch
- Intel 8286. Octal Bus Transceiver
- Intel 8288. Controlador de bus
- Intel 8289. Árbitro de bus
- Intel 8237. Controlador programable de DMA
- Intel 8259. Controlador programable de interrupciones (PIC)
- Intel 8253. Temporizador programable de intervalos (PIT)
- Intel 8250 UART. Comunicaciones seriales, RS-232
- Intel 8255. Interface programable de periféricos (PPI)
- Intel 8089. Coprocesador de entrada/salida

La funcionalidad de la mayoría de estos chips de soporte sigue existiendo en los computadores de la arquitectura x86 hasta nuestros días, sólo que desde hace tiempo se fueron integrando y dejaron de ser chips individuales y ahora forman parte del chipset del computador (la mayoría están integrados en el southbridge).

Coprocesador numérico

El 8086/8088 no tenía ninguna instrucción de coma flotante y para realizar operaciones con números reales se requerían bibliotecas con rutinas de software de coma flotante. Los computadores con el 8086/8088, generalmente tenían un socket de 40 pines en donde se podía enchufar un coprocesador matemático opcional para tener capacidad de coma flotante mucho más rápida basada en hardware/microcódigo. El Intel 8087 era el coprocesador matemático estándar para el 8086 y el 8088, operando internamente en números de 80 bits. Fabricantes como Cyrix (compatible con el 8087) y Weitek (no compatible con el 8087) eventualmente vinieron con un coprocesador de coma flotante de alto desempeño que competía con el 8087, así como con el subsecuente Intel 80387 de más alto desempeño.

Versiones del chip

Originalmente la frecuencia de reloj estaba limitada a 5 MHz (El IBM PC usaba 4.77 MHz, 4/3 de la frecuencia estándar del colorburst del NTSC, pero las últimas versiones en HMOS fueron especificadas para 10 MHz. Las versiones de HMOS-III y CMOS fueron fabricadas durante mucho tiempo (por lo menos un tiempo en los años 1990) para los sistemas empotrados, aunque sus sucesores, los 80186/80188 (que incluye algunos periféricos en el chip), han sido más populares para el uso empotrado.

Derivados y clones

Versiónes compatibles, y en muchos casos mejoradas, fueron fabricadas por Fujitsu, Harris/Intersil, OKI, Siemens AG, Texas Instruments, NEC, Mitsubishi, y AMD. Por ejemplo, el par NEC V20 y NEC V30 eran compatibles a nivel de hardware con el 8088 y el 8086 respectivamente, pero incorporaron el conjunto de instrucciones del 80186 junto con algunas (pero no todas) las mejoras en velocidad del 80186, proporcionando con sólo sustituir el chip, la capacidad para aumentar el conjunto de instrucciones y la velocidad de procesamiento sin que los fabricantes tuvieran que modificar sus diseños. Tales procesadores en CMOS, relativamente simples y de baja potencia, compatibles con el 8086/8088, todavía se usan en sistemas empujados.

La industria de la electrónica de la Unión Soviética fue capaz de replicar el 8086 con espionaje industrial e ingeniería inversa. El chip resultante, el K1810BM86, era binaria y compatible a nivel de pines con el 8086, pero no era mecánicamente compatible porque usó medidas métricas.

Los 8088 y 8086 eran los núcleos respectivos de los computadores de escritorio compatibles con el PC, ES1840 y ES1841, hechos por los soviéticos. Sin embargo, estas computadoras tenían significativas diferencias de hardware de sus auténticos prototipos, y el trazado de circuito del bus de datos/direcciones fue diseñado independientemente de los productos de Intel.^[cita requerida] El ES1841 era el primer PC compatible con tamaño de bus dinámico (patente de los E.E.U.U. No 4.831.514). Posteriormente, algunos de los principios del ES1841 fueron adoptados en el PS/2 (patente de los E.E.U.U. No 5.548.786) y algunas otras máquinas (solicitud de patente británica, publicación No. GB-A-2211325, publicada el 28 de junio de 1989).



El clon soviético KP1810BM86.



El QFP-56 M80C86A de OKI

Microcomputadores que usaron el i8086 o el i8088

IBM PC y XT

El IBM PC original fue el microcomputador más influyente en usar el 8088. Utilizó una frecuencia de reloj de 4.77 MHz (4/3 la frecuencia de colorburst del NTSC). Algunos de los ingenieros y otros empleados de IBM quisieron usar el procesador IBM 801, algunos hubieran preferido el nuevo Motorola 68000,^[15] mientras que otros argumentaron por un pequeño y simple microprocesador similar al que había sido usado en computadores personales anteriores (tales como el TRS-80 o el Apple II).^[16] Sin embargo, IBM tenía ya una historia de usar los chips de Intel en sus productos y también había adquirido los derechos de manufacturar la familia 8086.^[17] Otro factor era que el 8088 permitía que la computadora fuera basada en un diseño modificado del 8085, pues podría fácilmente interconectarse con la mayoría de los chips nMOS con buses de datos de 8 bits, es decir, componentes existentes y madurados, y por lo tanto económicos. Esto incluyó los ICs previstos originalmente para soporte y funciones periféricas alrededor del 8085 y procesadores similares (no exclusivamente de Intel) que ya eran bien conocidos por muchos ingenieros, reduciendo adicionalmente los costos.^[18]

El sucesor inmediato del IBM PC, el IBM XT, usaba el 8088. Tenía una arquitectura muy similar al PC, sólo añadiendo tres slots de expansión a los cinco del PC, un disco duro, memoria RAM adicional en la tarjeta madre, y una fuente de poder de más capacidad; y eliminando la interface para cassette.

Los descendientes del 8088 incluyen el 80188, 80186, 80286, 80386, 80486, Pentium, y los posteriores procesadores compatibles en software, que están en uso hasta hoy.

Clones

La mayoría de los clones del IBM PC y XT usaron el Intel 8088 corriendo a 4.77 ó más MHz. Algunos llegaron a correr a 8, 10 y hasta en 12 MHz.

Otros



De los microcomputadores que usaron el 8086 tenemos los siguientes:

- El primer microcomputador comercial basado en el 8086 fue el Mycron 2000.
- El procesador de texto IBM Displaywriter.^[cita requerida]
- El Wang Professional Computer, manufacturado por Wang Laboratories
- El AT&T 6300 PC (construido por Olivetti)
- El NEC PC-9801 (construido por NEC)
- El primer Compaq Deskpro usaba un 8086 corriendo a 7.14 MHz, pero fue capaz de correr tarjetas de expansión designadas para el IBM XT de 4.77 MHz
- El FLT86 es un bien establecido sistema de entrenamiento para el CPU 8086, que todavía está siendo manufacturado por *Flite Electronics International Limited*^[19] en Southampton, Inglaterra.
- El IBM PS/2 modelos 25 y 30 fueron construidos con un 8086 de 8 MHz
- La serie SL del Tandy 1000
- El Amstrad PC1512, PC1640, PC2086, PC3086 y PC5086 usaron el 8086 a 8 MHz.
- Hasta el año 2002, la NASA todavía estaba usando el CPU 8086 original en equipo para mantenimiento en tierra del Transbordador espacial Discovery, para prevenir la regresión de software que pudiera resultar por actualizar o cambiar a clones imperfectos.^[20]

Referencias

- [1] using enhancement load pMOS logic (demanding 14V, achieving TTL-compatibility by having V_{CC} at +5V and V_{DD} at -9V)
- [2] using non-saturated enhancement load nMOS logic (demanding a higher gate voltage for the load transistor-gates)
- [3] made possible with depletion load nMOS logic (the 8085 was later made using HMOS processing, just like the 8086)
- [4] Birth of a Standard: The Intel 8086 Microprocessor. Thirty years ago, Intel released the 8086 processor, introducing the x86 architecture that underlies every PC-Windows, Mac, or Linux-produced today (http://www.pcworld.com/article/146957/birth_of_a_standard_the_intel_8086_microprocessor.html), PC World, June 17, 2008
- [5] Rev.0 of the instruction set and architecture was ready in about 3 months, according to Morse.
- [6] Using rubylith, light boards, rulers, electric erasers, and a digitizer (according to Jenny Hernandez, member of the 8086 design team, in a statement made on Intel's web-page for its 25th birthday).
- [7] 8086 used less microcode than many competitors designs, such as the MC68000 and others
- [8] Fast static RAMs in MOS technology (as fast as bipolar RAMs) was an important product for Intel during this period.
- [9] CHMOS is intel's name for CMOS circuits manufactured using processing steps very similar to HMOS.
- [10] Other members of the design team were Peter A.Stoll and Jenny Hernandez.
- [11] Intel 8008 to 8086 by Stephen P. Morse et al.
- [12] Some 80186 clones did change the shift value, but were never commonly used in desktop computers.
- [13] IBM PC Technical Reference Manual (Número de parte 6025008), pág A-80 y A-5
- [14] (Similarly for iAPX 286, 80386, 80387.)
- [15] *Later used for the IBM Instruments 9000 Laboratory Computer*
- [16] *There were some rumours that (the then small) Microsoft somehow managed to persuade IBM to use the 8088, because it had more and better 16-bit capabilities than most other "8-bit" CPUs.*
- [17] *In exchange for giving Intel the rights to its bubble memory designs. However, due to fierce competition from Japanese manufacturers who were able to undercut by cost, Intel soon left this market and changed focus to microprocessors*
- [18] *68000 components were not widely available at the time, though it could use Motorola 6800 components to an extent.*
- [19] <http://www.flite.co.uk>
- [20] For Old Parts, NASA Boldly Goes... on eBay (<http://www.nytimes.com/2002/05/12/technology/ebusiness/12NASA.html?pagewanted=2>), May 12, 2002

Enlaces externos

-  Wikimedia Commons alberga contenido multimedia sobre **intel 8086**. Commons
-  Wikimedia Commons alberga contenido multimedia sobre **intel 8088**. Commons
- Historia de los microprocesadores Intel del 8008 al 8086 (<http://stevemorse.org/8086history/8086history.doc>)
(en inglés)
- The 8086/8088 Primer. An Introduction to Their Architecture, System Design, and Programming (<http://stevemorse.org/8086/index.html>). Libro de Stephen P. Morse, arquitecto principal del 8086/8088 (en inglés)
- Hoja de datos del Intel 80C86 (<http://www.datasheetcatalog.org/datasheet/Intel/mXtxqyu.pdf>)
- Hoja de datos del Intel 8088 (<http://www.datasheetcatalog.org/datasheet/Intel/mXrysuv.pdf>)
- Hoja de datos del 8086 (http://www.datasheetcatalog.org/datasheets/2300/499305_DS.pdf)
- x86 instruction set (<http://web.archive.org/20051121163920/home.comcast.net/~fbui/intel.html>)
- 80x86 instruction set (<http://www.penguin.cz/~literakl/intel/intel.html>)

Fuentes y contribuyentes del artículo

Intel 8086 y 8088 *Fuente:* <http://es.wikipedia.org/w/index.php?oldid=75969550> *Contribuyentes:* AldanaN, Aleposta, Alpertron, Atope36, Avm, Baiji, Beto29, Bryant1410, Chrihern, DarAR92, David0811, Dodo, Ealmagro, Elimedina, Er Komandante, Felixdavid, Focojoaco, GermanX, JWBE, Jkbw, Kizar, Leonpolanco, LordT, Martin Rizzo, Matdrones, Murphy era un optimista, Netito777, Nicolasm03, Piero71, Rastrojo, Roberpl, Rotlink, Shalbat, Shooke, Sr Beethoven, Takumi 1, Thunderbird2, VARGUX, VR0, Yrithinnd, 108 ediciones anónimas

Fuentes de imagen, Licencias y contribuyentes

Archivo:I8088.jpg *Fuente:* <http://es.wikipedia.org/w/index.php?title=Archivo:I8088.jpg> *Licencia:* GNU Free Documentation License *Contribuyentes:* Denniss, EugeneZelenko, Poil, Shooke

Archivo:Código de maquina.png *Fuente:* http://es.wikipedia.org/w/index.php?title=Archivo:Código_de_maquina.png *Licencia:* Public Domain *Contribuyentes:* German

Archivo:Intel 8086 block scheme.svg *Fuente:* http://es.wikipedia.org/w/index.php?title=Archivo:Intel_8086_block_scheme.svg *Licencia:* Creative Commons Attribution-ShareAlike 3.0 Unported *Contribuyentes:* Harkonnen2

Archivo:Intel 8088 segmento y offset.svg *Fuente:* http://es.wikipedia.org/w/index.php?title=Archivo:Intel_8088_segmento_y_offset.svg *Licencia:* Creative Commons Attribution-ShareAlike 3.0 *Contribuyentes:* User:German

Archivo:KL USSR KP1810BM86.jpg *Fuente:* http://es.wikipedia.org/w/index.php?title=Archivo:KL_USSR_KP1810BM86.jpg *Licencia:* GNU Free Documentation License *Contribuyentes:* Konstantin Lanzet

Archivo:Oki 80c86a.jpg *Fuente:* http://es.wikipedia.org/w/index.php?title=Archivo:Oki_80c86a.jpg *Licencia:* Public Domain *Contribuyentes:* Alecv

Archivo:Commons-logo.svg *Fuente:* <http://es.wikipedia.org/w/index.php?title=Archivo:Commons-logo.svg> *Licencia:* Public Domain *Contribuyentes:* SVG version was created by User:Grunt and cleaned up by 3247, based on the earlier PNG version, created by Reidab.

Licencia

Creative Commons Attribution-Share Alike 3.0
[//creativecommons.org/licenses/by-sa/3.0/](http://creativecommons.org/licenses/by-sa/3.0/)