II. Los microprocesadores 8086/8088 y sus similitudes.

Historia del 8086/8088

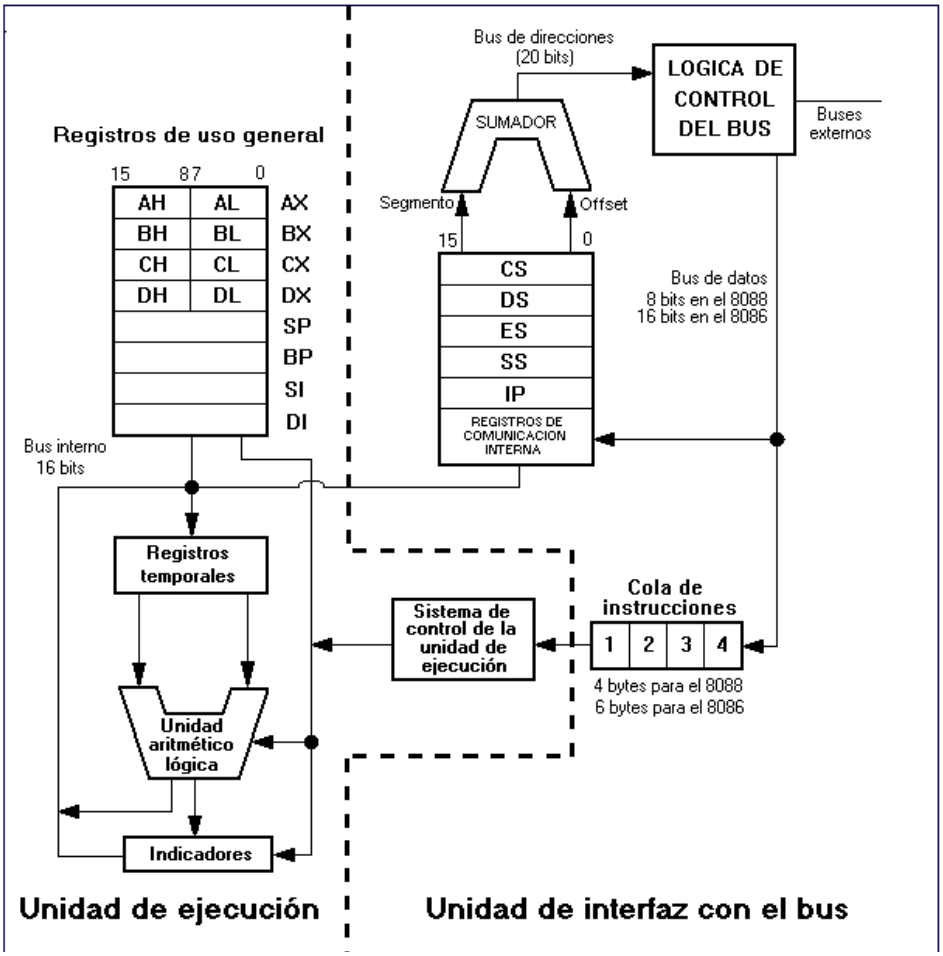
En junio de 1978 Intel lanzó al mercado el primer microprocesador de 16 bits: el 8086. En junio de 1979 apareció el 8088 (internamente igual que el 8086 pero con bus de datos de 8 bits) y en 1980 los coprocesadores 8087 (matemático) y 8089 (de entrada y salida). El primer fabricante que desarrolló software y hardware para estos chips fue la propia Intel. Reconociendo la necesidad de dar soporte a estos circuitos integrados, la empresa invirtió gran cantidad de dinero en un gran y moderno edificio en Santa Clara, California, dedicado al diseño, fabricación y venta de sus sistemas de desarrollo que, como se explicó anteriormente, son computadoras autosuficientes con el hardware y software necesario para desarrollar software de microprocesadores.
Los sistemas de desarrollo son factores clave para asegurar las ventas de una empresa fabricantes de chips. La inmensa mayoría de ventas son a otras empresas, las cuales usan estos chips en aparatos electrónicos, diseñados, fabricados y comercializados por ellas mismas. A estas empresas se las llama "fabricantes de equipo original", o en inglés, OEM (Original Equipment Manufacturer). El disminuir el tiempo de desarrollo de hardware y software para las OEM es esencial, ya que el mercado de estos productos es muy competitivo. Necesitan soporte pues los meses que les puede llevar el desarrollo de las herramientas apropiadas les pueden significar pérdidas por millones de dólares. Además quieren ser los primeros fabricantes en el mercado, con lo cual pueden asegurarse las ventas en dos áreas importantes: a
Corto plazo, ya que al principio la demanda es mucho mayor que la oferta, y a largo plazo, ya que el primer producto marca a menudo los estándares.
De esta manera la empresa Intel había desarrollado una serie completa de software que se ejecutaba en una microcomputadora basada en el 8085 llamada "Intellec Microcomputer Development System". Los programas incluían ensambladores cruzados (éstos son programas que se ejecutan en un microprocesador y generan código de máquina que se ejecuta en otro), compiladores de PL/M, Fortran y Pascal y varios programas de ayuda. Además había un programa traductor llamado CON V86 que convertía código fuente 8080/8085 a código fuente 8086/8088.
Si se observan de cerca ambos conjuntos de instrucciones, queda claro que la transformación es sencilla si los registros se traducen así: A -> AL, B -> CH, C -> CL, D -> DH, E -> DL, H -> BH y L -> BL.

Puede parecer complicado traducir LDAX B (por ejemplo) ya que el 8088 no puede utilizar el registro CX para direccionamiento indirecto, sin embargo, se puede hacer con la siguiente secuencia: MOV SI, CX; MOV AL, [SI]. Esto aprovecha el hecho que no se utiliza el registro SI. Por supuesto el programa resultante es más largo (en cantidad de bytes) y a veces más lento de correr que en su antecesor 8085. Este programa de conversión sólo servía para no tener que volver a

Escribir los programas en una primera etapa. Luego debería reescribirse el código fuente en assembler para poder obtener las ventajas de velocidad ofrecidas por el 8088. Luego debía correr el programa en la iSBC 86/12 Single Board Computer basado en el 8086. Debido al engorro que resultaba tener dos plaquetas diferentes, la empresa Godbout Electronics (también de California) desarrolló una placa donde estaban el 8085 y el 8088, donde se utilizaba un ensamblador cruzado provisto por la compañía Microsoft. Bajo control de software, podían conmutarse los microprocesadores. El sistema operativo utilizado era el CP/M (de Digital Research).
El desarrollo más notable para la familia 8086/8088 fue la elección de la CPU 8088 por parte de IBM (International Business Machines) cuando en 1981 entró en el campo de las computadoras personales.
Esta computadora se desarrolló bajo un proyecto con el nombre "Acorn" (Proyecto "Bellota") pero se vendió bajo un nombre menos imaginativo, pero más correcto: "Computadora Personal IBM", con un precio inicial entre 1260 dólares y 3830 dólares según la configuración (con 48KB de memoria RAM y una unidad de discos flexibles con capacidad de 160KB costaba 2235 dólares). Esta computadora entró en competencia directa con las ofrecidas por Apple (basado en el 6502) y por Radio Shack (basado en el Z-80).

Arquitectura de los procesadores 8088 y 8086:

El 8086 es un microprocesador de 16 bits, tanto en lo que se refiere a su estructura como en sus conexiones externas, mientras que el 8088 es un procesador de 8 bits que internamente es casi idéntico al 8086. La única diferencia entre ambos es el tamaño del bus de datos externo. Intel trata esta igualdad interna y desigualdad externa dividiendo cada procesador 8086 y 8088 en dos sub-procesadores. O sea, cada uno consta de una unidad de ejecución (EU: Execution Unit) y una unidad interfaz del bus (BIU: Bus Interface Unit). La unidad de ejecución es la encargada de realizar todas las operaciones mientras que la unidad de interfaz del bus es la encargada de acceder a datos e instrucciones del mundo exterior. Las unidades de ejecución son idénticas en ambos microprocesadores, pero las unidades de interfaz del bus son diferentes en varias cuestiones, como se desprende del siguiente diagrama en bloques:



La ventaja de esta división fue el ahorro de esfuerzo necesario para producir el 8088. Sólo una mitad del 8086 (el BIU) tuvo que rediseñarse para producir el 8088. La explicación del diagrama en bloques es la siguiente:

Registros de uso general del 8086/8088

Tienen 16 bits cada uno y son ocho:

AX = Registro acumulador, dividido en **AH** y **AL** (8 bits cada uno). Usándolo se produce (en general) una instrucción que ocupa un byte menos que si se utilizaran otros registros de uso general. Su parte más baja, AL, también tiene esta propiedad. El último registro mencionado es el equivalente al acumulador de los procesadores anteriores (8080 y 8085). Además hay instrucciones como DAA; DAS; AAA; AAS; AAM; AAD; LAHF; SAHF; CBW; IN y OUT que trabajan con AX o con uno de sus dos bytes (AH o AL). También se utiliza este registro (junto con DX a veces) en multiplicaciones y divisiones.

BX = Registro base, dividido en **BH** y **BL**. Es el registro base de propósito similar (se usa para direccionamiento indirecto) y es una versión más potente del par de registros HL de los procesadores anteriores.
.

CX = Registro contador, dividido en **CH** y **CL**. Se utiliza como contador en bucles (instrucción LOOP), en operaciones con cadenas (usando el prefijo REP) y en desplazamientos y rotaciones (usando el registro CL en los dos últimos casos).
.

DX = Registro de datos, dividido en **DH** y **DL**. Se utiliza junto con el registro AX en multiplicaciones y divisiones, en la instrucción CWD y en IN y OUT para direccionamiento indirecto de puertos (el registro
DX indica el número de puerto de entrada/salida).
.

SP = Puntero de pila (no se puede subdividir). Aunque es un registro de uso general, debe utilizarse sólo como puntero de pila, la cual sirve para almacenar las direcciones de retorno de subrutinas y los datos temporarios (mediante las instrucciones PUSH y POP). Al introducir (push) un valor en la pila a este registro se le resta dos, mientras que al extraer (pop) un valor de la pila este a registro se le suma dos.
.

BP = Puntero base (no se puede subdividir). Generalmente se utiliza para realizar direccionamiento indirecto dentro de la pila.
.

SI = Puntero índice (no se puede subdividir). Sirve como puntero fuente para las operaciones con cadenas. También sirve para realizar direccionamiento indirecto.
.

DI = Puntero destino (no se puede subdividir). Sirve como puntero destino para las operaciones con cadenas. También sirve para realizar direccionamiento indirecto. Cualquiera de estos registros puede utilizarse como fuente o destino en operaciones aritméticas y lógicas, lo que no se puede hacer con ninguno de los seis registros que se verán más adelante.
.

Unidad aritmética y lógica

Es la encargada de realizar las operaciones aritméticas (suma, suma con "arrastre", resta, resta con "préstamo" y comparaciones) y lógicas (AND, OR, XOR y TEST). Las operaciones pueden ser de 16 bits o de 8 bits.

Banderas (*flags*)

Hay nueve banderas de un bit en este registro de 16 bits. Los cuatro bits más significativos están indefinidos, mientras que hay tres bits con valores determinados: los bits 5 y 3 siempre valen cero y el bit 1 siempre vale uno (esto también ocurría en los procesadores anteriores).

Sistema de control de la unidad de ejecución

Es el encargado de decodificar las instrucciones que le envía la cola y enviarle las órdenes a la unidad aritmética y lógica según una tabla que tiene almacenada en ROM llamada CROM (Control Read Onlyn Memory).

Cola de instrucciones

Almacena las instrucciones para ser ejecutadas. La cola se carga cuando el bus está desocupado, de esta manera se logra una mayor eficiencia del mismo. La cola del 8086 tiene 6 bytes y se carga de a dos bytes por vez (debido al tamaño del bus de datos), mientras que el del 8088 tiene cuatro bytes. Esta estructura tiene rendimiento óptimo cuando no se realizan saltos, ya que en este caso habría que vaciar la cola (porque no se van a ejecutar las instrucciones que van después del salto) y volverla a cargar con instrucciones que se encuentran a partir de la dirección a donde se salta. Debido a esto las instrucciones de salto son (después de multiplicaciones y divisiones) las más lentas de este microprocesador.

Registros de la unidad de interfaz con el bus

El programador puede acceder a cinco registros de 16 bits cada uno, siendo cuatro de ellos registros de segmento y el restante el puntero de instrucción (IP). Los registros de segmento se llaman:

CS: Registro de segmento de código.
DS: Registro de segmento de datos.
ES: Registro de segmento extra.
SS: Registro de segmento de pila.

La utilización de estos registros se explica más adelante, en la sección que trata de direccionamiento a memoria.

Lógica de control del bus

El cometido de este bloque es poder unir los bloques anteriormente mencionados con el mundo exterior, es decir, la memoria y los periféricos. El 8088 tiene un bus de datos externo reducido de 8 bits. La razón para ello era prever la continuidad entre el 8086 y los antiguos procesadores de 8 bits, como el 8080 y el 8085.
Teniendo el mismo tamaño del bus (así como similares requerimientos de control y tiempo), el 8088, que es internamente un procesador de 16 bits, puede reemplazar a los microprocesadores ya nombrados en un sistema ya existente. El 8088 tiene muchas señales en común con el 8085, particularmente las asociadas con la forma en que los datos y las direcciones están multiplexados, aunque el 8088 no produce sus propias señales de reloj como lo hace el 8085 (necesita un chip de soporte llamado 8284, que es diferente del 8224 que necesitaba el

microprocesador 8080). El 8088 y el 8085 siguen el mismo esquema de compartir los terminales correspondientes a los 8 bits más bajos del bus de direcciones con los 8 bits del bus de datos, de manera que se ahorran 8 terminales para otras funciones del microprocesador. El 8086 comparte los 16 bits del bus de datos con los 16 más bajos del bus de direcciones. El 8085 y el 8088 pueden, de hecho, dirigir directamente los mismos chips controladores de periféricos. Las investigaciones de hardware para sistemas basados en el 8080 o el 8085 son, en su mayoría, aplicables al 8088. En todo lo recién explicado se basó el éxito del 8088.

Modos de direccionamiento del 8086/8088

Estos procesadores tienen 27 modos de direccionamiento (una cantidad bastante más grande que los microprocesadores anteriores) o reglas para localizar un operando de una instrucción. Tres de ellos son comunes a microprocesadores anteriores: **direccionamiento inmediato** (el operando es un número que se encuentra en la misma instrucción), **direccionamiento a registro** (el operando es un registro del microprocesador) y **direccionamiento inherente** (el operando está implícito en la instrucción, por ejemplo, en la multiplicación uno de los operando siempre es el acumulador). El resto de los modos sirve para localizar un operando en memoria. Para facilitar la explicación de estos modos, se pueden resumir de la siguiente manera:
Deben sumarse cuatro cantidades: 1) **dirección de segmento**, 2) **dirección base**, 3) una **cantidad índice** y 4) un **desplazamiento**.
La dirección de segmento se almacena en el *registro de segmento* (DS, ES, SS o CS). En la próxima sección se indica la forma en que se hace esto. Por ahora basta con saber que el contenido del registro de segmento se multiplica por 16 antes de utilizarse para obtener la dirección real. El registro de segmentación siempre se usa para referenciar a memoria.
La base se almacena en el *registro base* (BX o BP). El índice se almacena en el *registro índice* (SI o DI).
Cualquiera de estas dos cantidades, la suma de las dos o ninguna, pueden utilizarse para calcular la dirección real, pero no pueden sumarse dos bases o dos índices. Los registros restantes (AX, CX, DX y SP) no pueden utilizarse para direccionamiento indirecto. El programador puede utilizar tanto la base como el índice para gestionar ciertas cosas, tales como matrices de dos dimensiones, o estructuras internas a otras estructuras, esquemas que se utilizan en las prácticas comunes de programación. La base y el índice son variables o dinámicos, ya que están almacenadas en registros de la CPU. Es decir, pueden modificarse fácilmente mientras se ejecuta un programa.
Además del segmento, base e índice, se usa un desplazamiento de 16 bits, 8 bits o 0 bits (sin desplazamiento).
Ésta es una cantidad estática que se fija al tiempo de ensamblado (paso de código fuente a código de máquina) y no puede cambiarse durante la ejecución del programa (a menos que el programa se escriba sobre sí mismo, lo que constituye una práctica no aconsejada).

Todo esto genera los 24 modos de direccionamiento a memoria que se ven a continuación:

Registro indirecto: 1) [BX], 2) [DI]. 3) [SI].
Basado: 4) desp8[BX], 5) desp8[BP], 6) desp16[BX], 7) desp16[BP].
Indexado: 8) desp8[SI], 9) desp8[DI], 10) desp16[SI], 11) desp16[DI].
Basado−indexado: 12) [BX+SI], 13) [BX+DI], 14) [BP+SI], 15) [BX+DI].
Basado−indexado con desplazamiento: 16) desp8[BX+SI], 17) desp8[BX+DI], 18) desp8[BP+SI], 19) desp8[BX+DI], 20) desp16[BX+SI], 21) desp16[BX+DI], 22) desp16[BP+SI], 23) desp16[BX+DI].
Directo: 24) [desp16]. Aquí desp8 indica desplazamiento de 8 bits y desp16 indica desplazamiento de 16 bits. Otras combinaciones no están implementadas en la CPU y generarán error al querer ensamblar, por ejemplo, ADD CL,[DX+SI]. El ensamblador genera el tipo de desplazamiento más apropiado (0, 8 ó 16 bits) dependiendo del valor que tenga la constante: si vale cero se utiliza el primer caso, si vale entre −128 y 127 se utiliza el segundo, y en otro caso se utiliza el tercero. Nótese que [BP] sin desplazamiento no existe. Al ensamblar una instrucción como, por ejemplo, *MOV AL,[BP]*, se generará un desplazamiento de 8 bits con valor cero. Esta instrucción ocupa tres bytes, mientras que *MOV AL,[SI]* ocupa dos, porque no necesita el desplazamiento. Estos modos de direccionamiento producen algunos inconvenientes en el 8086/8088. La CPU gasta tiempo calculando una dirección compuesta de varias cantidades. Principalmente esto se debe al hecho de que el cálculo de direcciones está programado en micro código (dentro de la CROM del sistema de control de la unidad de ejecución). En las siguientes versiones (a partir del 80186/80188) estos cálculos están cableados en la máquina y, por lo tanto, cuesta mucho menos tiempo el realizarlos.

Estructura de segmentación de la memoria

Como se ha mencionado anteriormente, el 8086/8088 usa un esquema ingenioso llamado segmentación, para acceder correctamente a un megabyte completo de memoria, con referencias de direcciones de sólo 16 bits. Hay dos registros de segmento que tienen usos especiales: el microprocesador utiliza el registro CS (con el offset almacenado en el puntero de instrucción IP) cada vez que se debe acceder a un byte de instrucción de programa, mientras que las instrucciones que utilizan la pila (llamados a procedimientos, retornos, interrupciones y las instrucciones PUSH y POP) siempre utilizan el registro de segmento SS (con el offset almacenado en el registro puntero de pila SP). De ahí los nombres que toman: CS es el segmento de código mientras que SS es el registro segmento de pila.
Para acceder a datos en la memoria se puede utilizar cualquiera de los cuatro registros de segmento, pero uno de ellos provoca que la instrucción ocupe un byte menos de memoria: es el llamado segmento por defecto, por lo que en lo posible hay que tratar de usar dicho segmento para direccionar datos. Este segmento es

el DS (registro de segmento de datos) para todos los casos excepto cuando se utiliza el registro base BP. En este caso el segmento por defecto es SS.

Estructura de interrupciones del 8086/8088

Hay tres clases de interrupción: por hardware, por software e internas (a las dos últimas también se las llama "excepciones"). Veremos primeramente el caso de interrupciones por hardware: Como se mencionó anteriormente, el 8086/8088 tiene dos entradas de petición de interrupción: NMI e INTR y una de reconocimiento (INTA). La gran mayoría de las fuentes de interrupción se conectan al pin INTR, ya que esto permite enmascarar las interrupciones (el NMI no). Para facilitar esta conexión, se utiliza el circuito integrado controlador de interrupciones, que tiene el código 8259A. Este chip tiene, entre otras cosas, ocho patas para sendas fuentes de interrupción (IRQ0 − IRQ7), ocho para el bus de datos (D0 − D7), una salida de INTR y una entrada de INTA. Esto permite una conexión directa con el 8088/8086. Al ocurrir una petición de alguna de las ocho fuentes, el 8259A activa la pata INTR. Al terminar de ejecutar la instrucción en curso, el microprocesador activa la pata INTA, lo que provoca que el 8259A envíe por el bus de datos un número de ocho bits (de 0 a 255) llamado tipo de interrupción (programable por el usuario durante la inicialización del 8259A), que el 8086/8088 utiliza para saber cuál es la fuente de interrupción. A continuación busca en la tabla de vectores de interrupción la dirección del manejador de interrupción (*interrupt handler*). Esto se hace de la siguiente manera. Se multiplica el tipo de interrupción por cuatro, y se toman los cuatro bytes que se encuentran a partir de esa dirección. Los dos primeros indican el offset y los dos últimos el segmento del manejador, como se muestra a continuación.
Existen algunas interrupciones predefinidas, de uso exclusivo del microprocesador, por lo que no es recomendable utilizar estos tipos de interrupción para interrupciones por hardware o software.

Tipo 0: Ocurre cuando se divide por cero o el cociente es mayor que el valor máximo que permite el destino.

Tipo 1: Ocurre después de ejecutar una instrucción si TF (Trap Flag) vale 1. Esto permite la ejecución de un programa paso a paso, lo que es muy útil para la depuración de programas.

Tipo 2: Ocurre cuando se activa la pata NMI (interrupción no enmascarable).

Tipo 3: Existe una instrucción INT que ocupa un sólo byte, que es la correspondiente a este tipo. En los programas depuradores (debuggers) (tales como *Debug*, *CodeView*, *Turbo Debugger*, etc.), se utiliza esta instrucción como punto de parada (para ejecutar un programa hasta una determinada dirección, fijada por el usuario del depurador, se inserta esta instrucción en la dirección correspondiente a la parada y se lanza la ejecución. Cuando el CS:IP apunte a esta dirección se ejecutará la INT 3, lo que devolverá el control del procesador al

depurador). Debido a esto, si se le ordena al depurador que ejecute el programa hasta una determinada dirección en ROM (memoria de sólo lectura) (por ejemplo, para ver cómo funciona una subrutina almacenada en dicha memoria), la ejecución seguirá sin parar allí (ya que la instrucción INT 3 no se pudo escribir sobre el programa). En el 80386, con su elaborado hardware de ayuda para la depuración, se puede poner un punto de parada en ROM.

.

Tipo 4: Ocurre cuando se ejecuta la instrucción de interrupción condicional INTO y el flag OF (Overflow Flag) vale 1. Los tipos 5 a 31 (1F en hexadecimal) están reservados para interrupciones internas (también llamados "excepciones") de futuros microprocesadores.