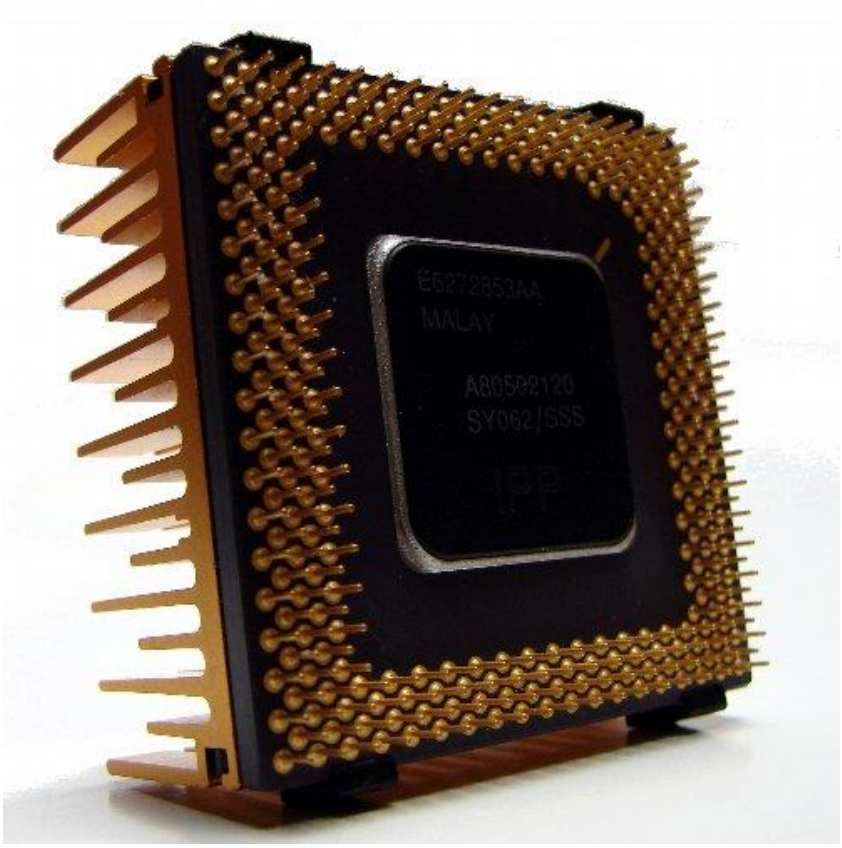
GUIA ENSAMBLADOR X86

PROYECTO



2020

Proyecto de evaluación del rendimiento

F-II. Implementación de un benchmark reducido y evaluación del procesamiento de arquitecturas PC convencionales

Arquitectura de los Computadores
Grado en Ingeniería Informática
Dpto. Tecnología Informática y Computación
Universidad de Alicante

GUÍA DE ENSAMBLADOR X86

PROYECTO DE EVALUACIÓN DEL RENDIMIENTO

I. PROGRAMADOR DE SISTEMAS Y PROGRAMADOR DE APLICACIONES

En los procesadores avanzados en muy importante saber apreciar la diferencia existente entre los dos tipos habituales de programadores que participan en la construcción del software.

Programador de aplicaciones

Es el encargado de codificar programas para los usuarios finales. En general, suelen desarrollar aplicaciones compuestas por varios programas escritos empleando un lenguaje de alto nivel comercial. Contempla la CPU como un conjunto de registros de trabajo que le permiten confeccionar dichas aplicaciones de usuario. Para este tipo de programador resultan transparentes los recursos de la CPU empleados para llevar a cabo la tarea de aplicación en coordinación con otras distintas y de acuerdo con un mecanismo de protección que controla los accesos evitando las transgresiones entre tareas.

Los conocimientos que el programador de aplicaciones debe tener sobre la máquina son imprescindibles para obtener el máximo rendimiento de las instrucciones usadas para resolver las aplicaciones. En el caso de emplear lenguaje máquina, deberá conocer los registros internos accesibles para la manipulación de datos y direcciones, el repertorio básico de instrucciones y los modos de direccionamiento. También deberá manejar el modelo de programación del coprocesador matemático.

Programador de sistemas

Es el encargado de desarrollar programas y utilidades del sistema operativo. Habitualmente la cualificación de los programadores de sistemas suele ser superior a la de los programadores de aplicaciones. La misión de este tipo de programadores es construir un sistema de explotación óptimo que sea capaz de soportar todas las aplicaciones previstas. Entre sus funciones más destacadas están:

- Organizar el sistema para el correcto tratamiento de las tareas pertenecientes a los diferentes usuarios.
- Desarrollar funciones y objetos para los sistemas operativos, depuradores, compiladores, etc.
- Asignar a cada tarea su nivel de privilegio y un sistema de protección adecuado.
- Organizar la memoria y el procesador para que las tareas consigan un mejor rendimiento.

Es indispensable que el programador de sistemas conozca profundamente la arquitectura detallada de la CPU para sí optimizar todos los recursos, obteniendo en su funcionamiento la máxima potencia, seguridad y rendimiento. Debe conocer también las prestaciones de la memoria virtual, las características de protección del entorno, los mecanismos que posibilitan la conmutación de tareas, el tratamiento de interrupciones y excepciones, etc.

Los microprocesadores Pentium disponen de una serie de registros y recursos especiales, denominados del sistema, que se encargan de gestionar el funcionamiento general, aprovechando sus prestaciones.

II. REGISTROS INTEL PENTIUM X86

En este primer ejercicio guiado se creará una nueva Solución Visual Studio .NET, denominada *EjerciciosEnsamblador*, que se utilizará para desarrollar los ejercicios de programación contenidos en esta práctica, y un nuevo Proyecto de Visual C++, denominado *Ejemplo1*. Para ello, realizar las siguientes acciones:

El microprocesador Pentium dispone de 32 registros en su arquitectura interna de los cuales 16 de ellos son para uso del programador de aplicaciones tal como se puede apreciar en la Figura 1. Estos últimos se clasifican en cuatro grandes grupos:

- Registros de propósito general.
- Registro Puntero de Instrucciones (EIP).
- Registro de estado o de Señalizadores (EFLAGS).
- Registros de Segmento.

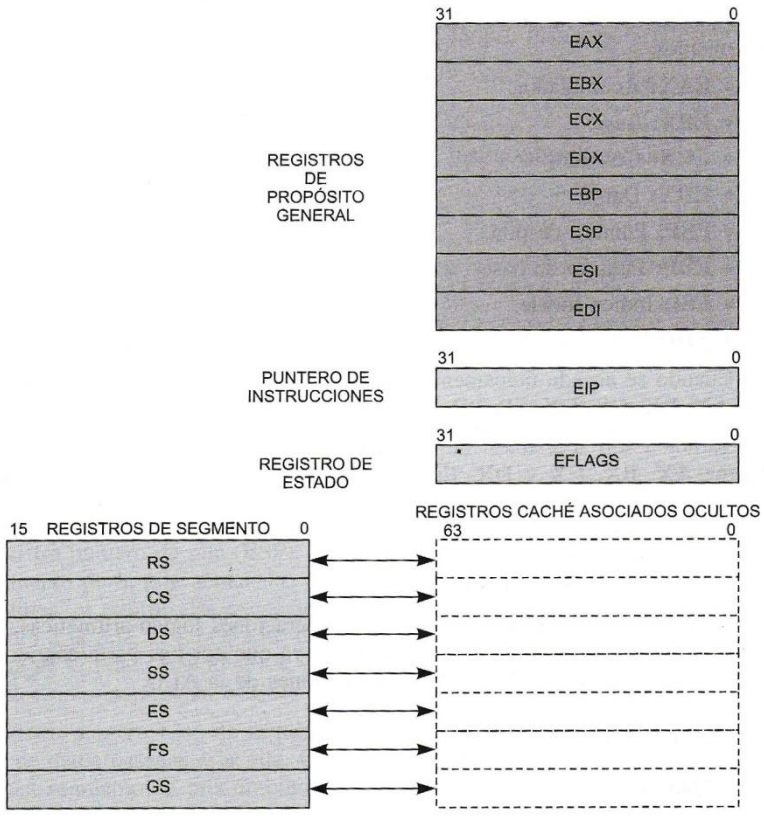


Figura 1. *Estructura general de los 16 registros del microprocesador Pentium para el programador de aplicaciones.*

Registros de propósito general

Son los mismos registros de que disponía el microprocesador 8086 de 16 bits, pero ampliados a 32 bits. Este grupo de registros consta de ocho registros capaces de trabajar con información 32 bits cuando

utilizan todo su tamaño aunque también pueden manejar datos de 16 bits, e incluso cuatro de ellos pueden trabajar con información de 8 bits.

Los registros de propósito general pueden usarse tanto para almacenar datos como direcciones. En este último caso, el contenido del registro es un desplazamiento que apunta a una dirección de memoria.

A los registros de propósito general se les ha mantenido el mismo nombre que el microprocesador 8086 pero anteponiendo la letra E de "extendido", para hacer referencia a la extensión a 32 bits de los mismos.

- EAX: Acumulador.
- EBX: Base.
- ECX: Contador.
- EDX: Datos.
- ESP: Puntero de pila.
- EBP: Puntero de base.
- ESI: Índice fuente.
- EDI: Índice destino.

Cuando se accede únicamente a los 16 bits de menos peso de estos registros, se designan por AX, BX, CX, DX, SP, BP, SI, DI, respectivamente. También son accesibles, de forma independiente, los dos bytes de menos peso de los registros AX, BX, CX y DX. En estos cuatro registros cuando se accede al byte de menos peso se les denomina AL, BL, CL y DL, respectivamente. Cuando se maneja el byte de más peso se les denomina AH, BH, CH y DH, respectivamente (Figura 2).

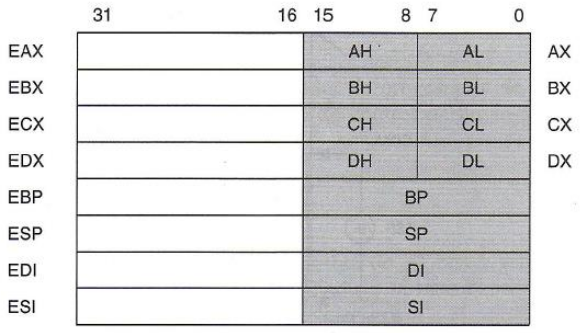


Figura 2. *Formatos y denominaciones de los registros de propósito general.*

- **Acumulador (EAX)**

  Es un registro que se emplea en todas las operaciones aritméticas y lógicas. Los procesadores de Intel hacen un uso intensivo del acumulador ya que por una parte contiene un operando y por otra se carga con el resultado de las operaciones de la Unidad Aritmético-Lógica (ALU).

- **Base (EBX)**

  Suele emplearse para contener una dirección que apunta a la base de un conjunto de datos. La longitud de las direcciones puede ser larga o corta. Se utiliza el registro EBX (32 bits) para el direccionamiento en la memoria que maneja el microprocesador Pentium (dirección larga),

mientras que se utiliza el registro BX (16 bits) para manejar la memoria del microprocesador 8086 (dirección corta).

- **Contador (ECX)**

  En el caso de iteraciones, se puede cargar con el número de veces que se ejecuta una instrucción.

- **Datos (EDX)**

  Este registro de propósito general se suele emplear para contener las direcciones de los puertos de entrada y salida (E/S) en las instrucciones que maneja el mapa de E/S.

- **Puntero de pila (ESP) y Puntero de base (EBP)**

  Se utilizan para controlar el direccionamiento de la pila. Las operaciones en la pila se realizan empleando tres registros diferentes tal como se expone a continuación (Figura 3):

  - Registro de segmento de pila (SS). Especifica las características del segmento de pila que reside en memoria.
  - Registro puntero de pila (ESP). Contiene el desplazamiento de la cima de la pila en el segmento de pila actual. Los utilizan las instrucciones PUSH y POP, las llamadas a subrutinas, el retorno de subrutinas, las excepciones y las interrupciones. Cuando se introduce un elemento en la pila, el procesador decrementa el puntero ESP, y escribe el elemento en la cima de la pila. Cuando se saca un elemento de la pila se realiza la operación contraria, es decir, se incrementa ESP.
  - Registro puntero base de pila (EBP). Se usa normalmente para acceder a estructuras de datos pasadas a la pila. Cuando el registro EBP se usa para direccionar memoria, el segmento de pila en curso es referenciado. Este registro apunta a la base de la pila y cuando existen rutinas hace el papel de ESP para no modificar el valor de este último.

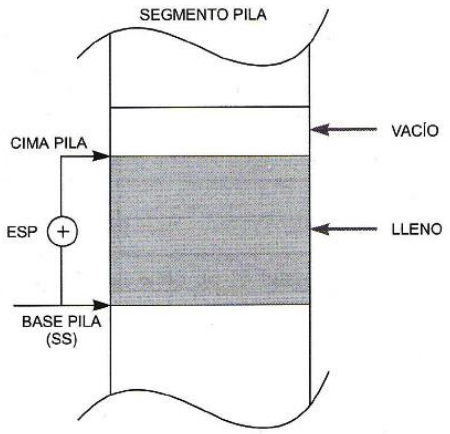


Figura 3. *La cima de la pila se direcciona sumando a la base del segmento el valor del registro ESP.*

- **Índice fuente (ESI) e Índice destino (EDI)**

  Son dos punteros de direcciones necesarios para trabajar con cadenas de caracteres. Las instrucciones que realizan operaciones entre los elementos de las cadenas se aplican a una cadena fuente y una cadena destino, cada una de ellas está direccionada por el registro ESI y el registro EDI, respectivamente (Figura 4).

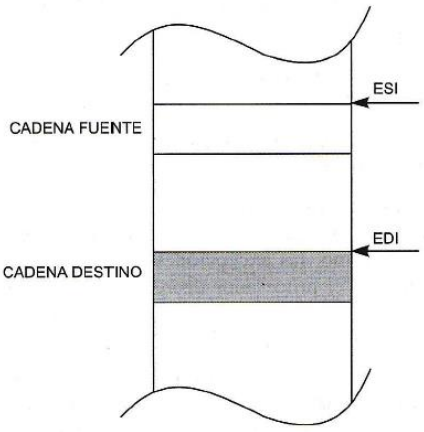


Figura 4. *Los registros ESI y EDI apuntan al elemento en curso de la cadena fuente y de la cadena destino, respectivamente.*

Tienen la propiedad de autoincremento y autodecremento. Cada vez que se ejecuta una instrucción se incrementan o decrementan los registros índices en el número de bytes que tenga cada elemento de la cadena.

EIP: Registro puntero de instrucciones

El registro EIP es el puntero de instrucciones o contador de programa y no está disponible para el programador, lo gobierna implícitamente el flujo de control de las instrucciones, las interrupciones y las excepciones. Este registro puede trabajar en dos modos:

- **En modo Nativo o Protegido**. Tiene una longitud de 32 bits y recibe el nombre de EIP. Almacena el desplazamiento que hay que añadir a la base del segmento de código para obtener la dirección donde está la siguiente instrucción a ejecutar. La base del segmento de código se obtiene a partir del valor del registro de segmento, CS. El valor máximo del desplazamiento en el segmento de código será de 4 GB.
- **En modo Real**. Se emplea un tipo de direccionamiento reducido, compatible con los procesadores 8086 y 80286, que solo precisa de 16 bits para especificar el desplazamiento en el segmento de código. Son los dos bytes de menos peso de EIP que se denominan IP. En este caso el valor máximo del desplazamiento será de 64 KB.

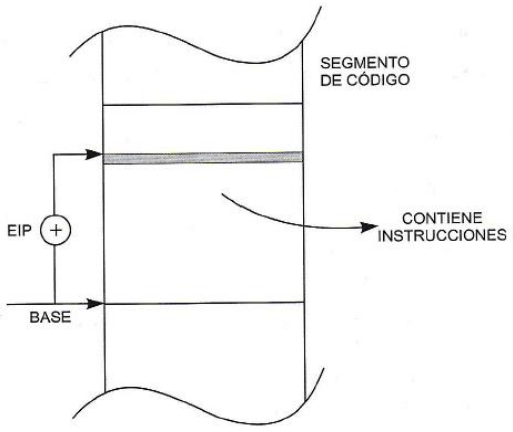


Figura 5. *En el segmento de código para direccionar la instrucción a ejecutar, se suma a la base del segmento el contenido del registro puntero de instrucciones (EIP o IP).*

Registro de estado o de señalizadores (EFLAGS)

Este registro, también conocido como registro EFLAGS, consta de 32 bits, de los cuales la mayoría son señalizadores de estado controlados por la Unidad Aritmético-Lógica (ALU): acarreo, paridad, acarreo auxiliar, cero, signo, y sobrepasamiento. Los restantes bits actúan como señalizadores del sistema ligados al mecanismo de protección y a otros recursos que dispone el sistema. Se utiliza para obtener información sobre el estado y el control de las operaciones del microprocesador.

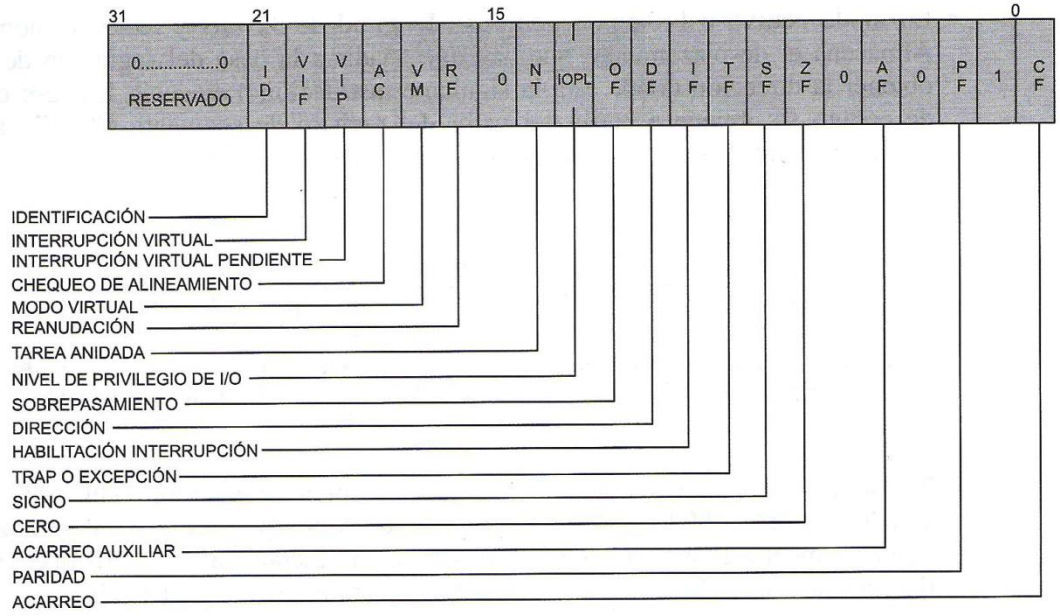


Figura 6. *Registro EFLAGS.*

A continuación, se describe cada uno de los bits que forman parte del registro de estado (Tabla 1).

Flag	**Descripción**
CF (Acarreo)	Es el señalizador de acarreo en el bit de más peso al realizarse una operación de suma aritmética. • 1: Indica que ha existido acarreo en el bit de más peso en ua operación de suma aritmética. • 0: Significa que no ha habido acarreo en el bit de más peso de una suma, o bien, que ha existido acarreo en el bit de más peso en el resultado de una resta aritmética.
PF (Paridad)	Bit de paridad impar. • 1: Toma este valor para generar la paridad impar considerando los bits que conforman el resultado de la operación. • 0: Valor para generar paridad impar.
AF (Acarreo auxiliar)	Señalizador de acarreo auxiliar o intermedio. • 1: Toma este valor cuando ha habido acarreo en el bit 3 del resultado. Se utiliza en las operaciones con números BCD. • 0: En el caso de no haber acarreo en el bit 3 del

Flag	Descripción
	resultado.
ZF (Cero)	Señalizador de cero. • 1: Cuando todos los bits del resultado son cero. • 0: En caso de que el resultado no sea cero.
SF (Signo)	Señalizador de signo. • 1: Si el bit de mayor peso del resultado de la operación es 1. • 0: Si el bit de mayor peso del resultado de la operación es 0.
TF (Excepción)	Excepción al terminar la ejecución de la instrucción. • 1: Provoca una excepción al completarse la ejecución de la instrucción en curso. Facilita la depuración de los programas al permitir la ejecución paso a paso (instrucción a instrucción). • 0: No hay excepción de depuración al final de cada instrucción.
IF (Habilitación interrupción)	Flag de habilitación de interrupciones mascarables. • 1: Permite el reconocimiento de las peticiones de interrupción mascarables provocadas por la activación de la señal INTR del microprocesador Pentium. • 0: Prohíbe el reconocimiento de la interrupción externa e ignora las peticiones de interrupción mascarables.
DF (Dirección)	Flag de dirección de exploración de las cadenas de caracteres. • 1: Postdecremento automática de lso registros ESI y EDI, que direccionan la cadena. • 0: Postincremento automático de los registros ESI y EDI.
OF (Sobrepasamiento)	Flag de desbordamiento u *overflow*. • 1: En operaciones con número enteros con signo se activa y vale 1 si el resultado es muy grande (positivo) o muy pequeño (negativo). Indica resultados erróneos. • 0: Si no existe desbordamiento.
IOPL (Nivel de privilegio de E/S)	Nivel de privilegio de las entradas y salidas. Es un campo de dos bits que se emplean en modo Protegido y determina el grado de privilegio que debe igualar o superar el segmento de código en el que se desean ejecutar instrucciones de E/S. Este tipo de instrucciones manejan datos en el espacio de E/S del procesador que está reservado para almacenar la información de los periféricos del sistema. • 11: Nivel 3, pueden acceder todos. • 10: Nivel 2. • 01: Nivel 1. • 00: Nivel 0, únicamente puede acceder el Sistema Operativo.

Flag	Descripción
NT (Tarea anidada)	Tarea anidad. Este señalizador actúa automáticamente al producirse un conmutación de tareas. • 1: La tarea en curso está anidad con la anterior. Hay que retornar obligatoriamente a la tarea previa. • 0: La conmutación de tarea es libre.
RF (Reanudación)	Flag de reanudación. Su activación provoca la ejecución de la siguiente instrucción cuando se produce una excepción de depuración en una instrucción, es decir, se ignora la excepción. • 1: Se ignoran los puntos de depuración o parada. • 0: No se ignoran los puntos de parada.
VM (Modo virtual)	Modo Virtual. Mediante este bit se permite el paso desde el modo Protegido al modo Virtual-86. • 1: Estando el procesador en modo Protegido se pasa al modo Virtual. • 0: No hay cambio al modo Virtual.
AC (Chequeo de anidamiento)	Bit de chequeo de alineamiento. • 1: Se produce una excepción cuando se encuentra una palabra o dato desalineado en la memoria. • 0: No hay excepción por desalineamiento. Para que el bus de datos de 64 líneas pueda recibir en un solo ciclo el contenido de 8 bytes, es necesario que los mismos comiencen en una dirección múltiplo de 8. Lo mismo puede suceder con las palabras de 4 bytes que deben ocupar posiciones a partir de una dirección múltiplo de 4. Si el programador no deja alineados los datos en la memoria, obligará a realizar más de un acceso para recoger toda la información.
VIF (Interrupción virtual)	Se trata de un bit que realiza una misión similar al flag IF, pero en modo Virtual. Permite o prohíbe la atención a las peticiones de interrupción mascarables. Se utiliza con el flag VIP. El procesador solo reconoce VIF cuando el flag VME o el PVI en el registro de control CR4 están activados y el flag IOLP es menor que 3.
VIP (Interrupción virtual pendiente)	Interrupción mascarable pendiente en modo Virtual. Trabaja en combinación con el flag VIF para que cada tarea en modo Virtual disponga de un flag equivalente al IF. Se activa por software para indicar que una interrupción está pendiente. El procesador lee este flag pero nunca lo modifica. El procesador solo reconoce VIF cuando el flag VME o el PVI en el registro de control CR4 están activados y el flag IOLP es menor que 3. • 1: Interrupción mascarable pendiente. • 0: No hay interrupción pendiente.
ID (Identificación)	Bit de identificación. ID informa si el procesador Pentium soporta la instrucción CPUID que sirve para su identificación. Mediante la ejecución de CPUID se muestran las características más importantes

Flag	Descripción
	del procesador. • 1: El procesador Pentium soporta la instrucción CPUID. • 0: En cado contrario.

Tabla 1. *Descripción de los bits que conforman el registro de estado (EFLAGS).*

Registros de segmento

Intel ha incorporado la segmentación como sistema principal de la organización de la memoria en la arquitectura IA-32.

El manejo de la memoria con segmentos favorece la programación estructurada y la modularidad, siendo una técnica apropiada para permitir la reubicación y soportar una estructura de protección muy segura y flexible.

Los segmentos son fragmentos de la memoria de tamaño variable que contienen el mismo tipo de información. Así, se pueden diferenciar tres tipos de segmentos en los programas de aplicación:

- Segmento de pila.
- Segmento de código.
- Segmento de datos.

El procesador Pentium controla en cada instante seis segmentos a los que hace referencia a través de los Registros de Segmento (RS). Esto no significa que solo pueda manejar seis segmentos, sino que si se desea acceder a otro que no está referenciado por dichos registros, deberá cargarse previamente en uno de ellos el valor correspondiente al nuevo segmento. Para controlar los segmentos de trabajo el microprocesador Pentium dispone de seis registros que se muestran en la Figura 7.

Figura 7. *Registros de segmento del microprocesador Pentium.*

Desde el punto de vista del programador de aplicaciones, los seis registros de segmento determinan en cada momento los segmentos que es capaz de identificar y manipular la CPU.

Cuando el procesador Pentium trabaja en modo Protegido la dirección lógica o virtual de todo elemento accesible en la memoria esta formada por una información que consta de los siguientes campos:

- **Selector.** Son los 14 bits de mayor peso del registro de segmento que referencia al segmento al cual se desea acceder y con ellos se encuentran la base donde comienza el segmento, el límite o tamaño que tiene y sus atributos. Los dos bits de menor peso del registro de segmento conforman

el campo RPL que indica el nivel de privilegio del segmento peticionario, tal como se representa en la Figura 8.

- **Desplazamiento**. Es un valor que se añade a la base del segmento para localizar la dirección a que hay que acceder en él. El tamaño del desplazamiento determina su longitud máxima que en el caso del microprocesador Pentium, es de 4 GB en el modo Protegido y de 64 KB en modo Real.

Figura 7. *Contenido de los registros de segmento*.

Cuando el procesador Pentium trabaja en modo Real el valor de la base del segmento se calcula añadiendo cuatro ceros a los 16 bits del registro de segmento correspondiente (Figura 8).

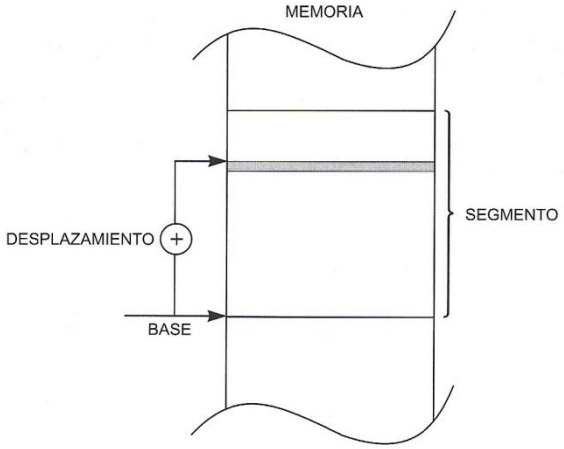


Figura 8. *Dirección lógica de un elemento es un segmento*.

Cada registro de segmento hace referencia un tipo de segmento determinado.

- **CS**: El registro CS (Segmento de Código), contiene en cada momento la información necesaria del segmento de instrucciones que está ejecutando la CPU, es decir, el segmento en curso. El desplazamiento que hay que añadir a la base del segmento de código reside en el Registro Puntero de Instrucciones, EIP.
- **SS**: El registro SS (Segmento de Pila), guarda el valor del selector del segmento de pila en curso. El registro ESP contiene el desplazamiento que debe añadirse a la base del segmento de pila para determinar la cima por donde se cargan y descargan los datos.
- **DS**: El registro DS (Segmentos de Datos), soporta el valor del selector del segmento de datos y el desplazamiento viene especificado en el modo de direccionamiento usado en la instrucción para expresar los operandos y el resultado.

Los procesadores x86 de 32 bits disponen de segmentos ES, FS y GS para poder tener activos otros tres segmentos de datos, además del proporcionado por DS (Figura 9).

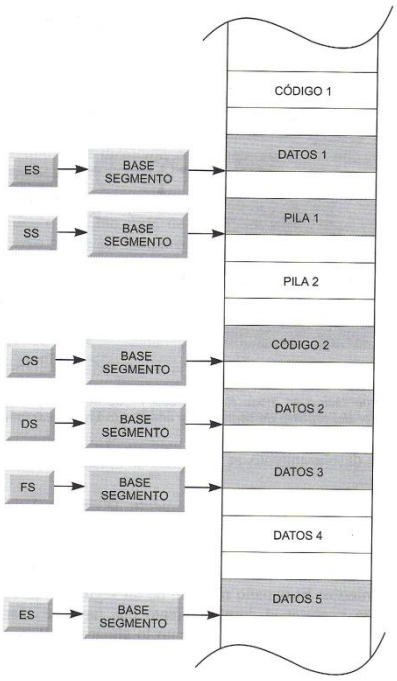


Figura 9. *A través de los registros de segmento, la CPU tiene activos seis segmentos en cada instante.*

III. REPERTORIO DE INSTRUCCIONES

En los siguientes apartados se incluye una clasificación de las instrucciones del procesador Pentium acorde a su función y características. En primer lugar, se incluyen algunos aspectos interesantes para el programador de aplicaciones.

Tipos de datos

El bus de datos externo del microprocesador Pentium tiene 64 bits, por tanto, el subsistema de memoria debe estar organizado en ocho grupos de 8 bits. Sin embargo, el microprocesador Pentium puede direccionar bytes individuales, palabras o dobles palabras.

El microprocesador Pentium está diseñado para trabajar con todos los tipos de datos que manejan los lenguajes evolucionados, aunque hay cuatro fundamentales:

1. *Byte* u octeto de ocho bits.
2. *Palabra*, compuesta por dos bytes.
3. *Doble palabra* de cuatro bytes.
4. *Cuádruple palabra* de ocho bytes.

En el procesador Pentium la unidad básica de la memoria es el byte, por lo que las palabras ocupan dos posiciones de memoria sucesivas y las dobles palabras ocupan cuatro posiciones. Existe flexibilidad en la ubicación de los datos de forma que no se precisa que la dirección de inicio de palabras y dobles palabras esté alineada ocupando direcciones pares y múltiplos de cuatro. Esta propiedad facilita la tarea del programador ya que no debe estar pendiente del direccionamiento, si bien es recomendable que alinee los datos para agilizar el acceso puesto que un correcto alineamiento permite disminuir el

número de accesos. Como se aprecia en la Figura 10, los bytes de mayor peso ocupan las direcciones más altas en la memoria.

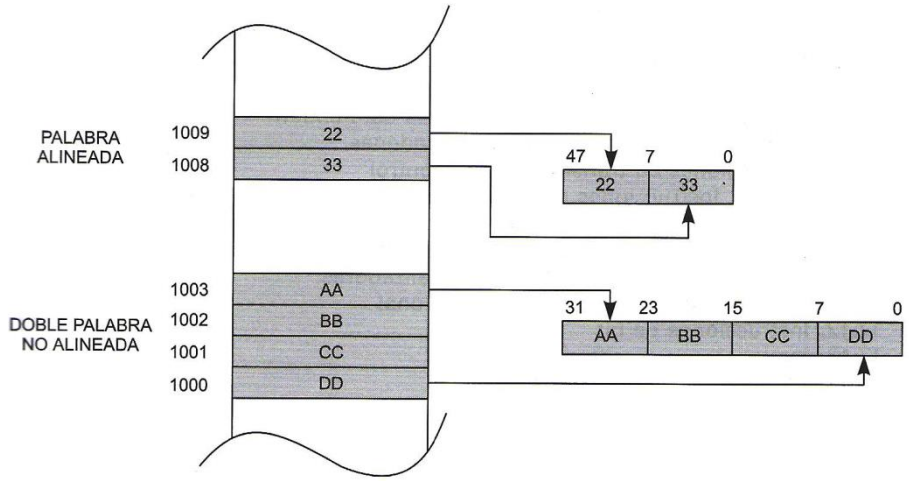


Figura 10. *Distribución de palabras y dobles palabras en una memoria organizada con tamaño byte.*

Además de los cuatro tipos de datos básicos, los datos que manejan las instrucciones del microprocesador Pentium se pueden clasificar de la siguiente forma:

- **Enteros y ordinales**

  Son los números con y sin signo que son reconocidos por todas las instrucciones aritméticas. Tiene formato Byte (8 bits), palabra (2 bytes o 16 bits) y doble palabra (4 bytes o 32 bits).

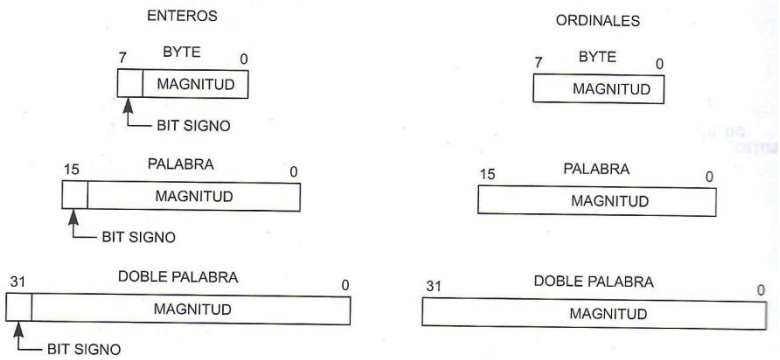


Figura 11. *Tipos de datos utilizados por el microprocesador Pentium: enteros y ordinales.*

- **Números BCD (Decimal Codificado en Binario)**

  El sistema de codificación BCD es una forma particular de emplear el sistema binario para la representación de números decimales. Los datos BCD pueden estar desempaquetados o empaquetados. En el caso de BCD empaquetado, cada byte representa dos dígitos BCD. En cambio si los datos son BCD desempaquetado, cada byte solo representa un dígito BCD utilizando los cuatro bits de menor peso. De esta última forma se desaprovecha el 50% del espacio de memoria.

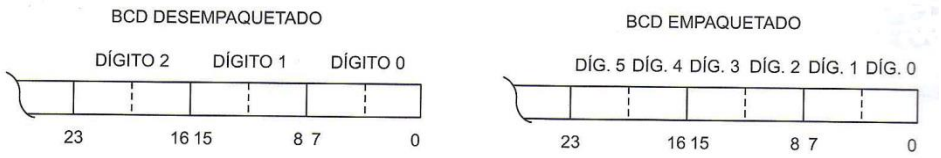


Figura 12. *Tipos de datos utilizados por el microprocesador Pentium: números BCD.*

- **Cadenas**

Existen instrucciones especializadas para manipular bits, bytes, palabras y dobles palabras concatenadas formando cadenas o *strings*, que pueden alcanzar un tamaño de 4 GB.

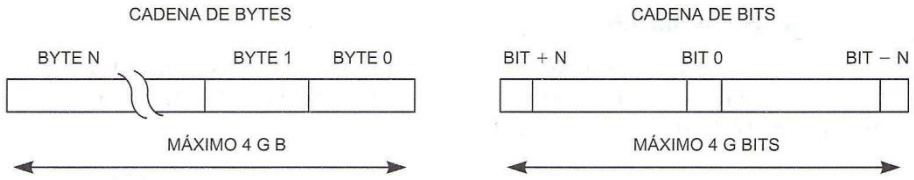


Figura 13. *Tipos de datos utilizados por el microprocesador Pentium: cadena de bytes y cadena de bits.*

- **Campo de bits**

Este formato permite a ciertas instrucciones manejar bits particulares de un campo, es decir, presenta la posibilidad de modificar únicamente ciertos bits en un campo, cuyo tamaño máximo puede alcanzar los 32 bits, manteniendo los demás con su valor anterior.

Figura 14. *Tipos de datos utilizados por el microprocesador Pentium: campo de bits.*

Las instrucciones capaces de trabajar con campos de bits y cadenas son muy interesantes en aplicaciones gráficas y de comunicaciones.

- **Puntero de direcciones**

Este tipo de datos se relaciona con los modos de direccionamiento de los operandos y con las instrucciones de salto. El puntero corto o cercano dispone de un desplazamiento de 32 bits que permite un salto dentro del segmento que se trabaja. El puntero largo o lejano, además del desplazamiento, contiene el valor de un selector para uno de los seis registros de segmento. Se puede realizar un salto a otro segmento. El tamaño de este puntero es de 48 bits.

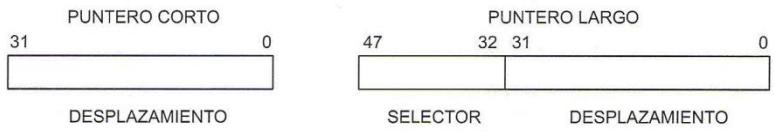


Figura 15. *Tipos de datos utilizados por el microprocesador Pentium: puntero corto y puntero largo.*

Tipos de datos del coprocesador matemático

El coprocesador matemático almacena toda la información en un formato único de coma flotante y precisión extendida con un tamaño de 80 bits que coincide con el tamaño de sus registros internos. El coprocesador debe trabajar con siete tipos de datos distintos, por lo que las instrucciones realizan conversiones en ambos sentidos. Cuando toman un dato de la memoria con un determinado formato lo convierten al formato único y viceversa. Los siete tipos de datos se clasifican en tres grandes grupos:

- **Enteros**

  En este grupo hay tres tipos: largo (64 bits), corto (32 bits) y palabra (16 bits).

Figura 16. *Tipos de datos coprocesador matemático: enteros.*

- **Decimal empaquetado**

  Guarda 18 dígitos BCD empaquetados, dos números BCD en un byte, en formato de 80 bits.

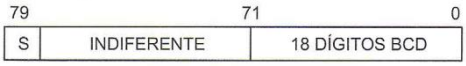


Figura 17. *Tipos de datos coprocesador matemático: decimal empaquetado.*

- **Coma flotante**

  Se distinguen los datos en simple precisión (32 bits), en doble precisión (64 bits) y de precisión extendida (80 bits), que es el formato con el que opera internamente el coprocesador.

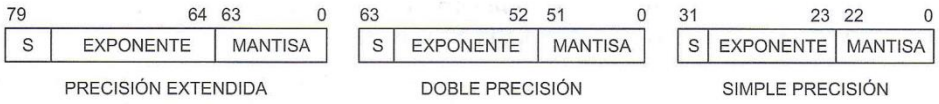


Figura 18. *Tipos de datos coprocesador matemático: coma flotante.*

Modos de direccionamiento

Las formas que dispone el microprocesador Pentium para referenciar sus instrucciones se pueden dividir en tres grades categorías:

1. **Modo de direccionamiento inmediato**

   El operando reside en la propia instrucción. Su sintaxis genérica en ensamblador sería:

   Instrucción registro, operando inmediato

   Ejemplo:

   IMUL ECX, -4

Explicación: Multiplica el entero contenido en el registro ECX por el operando -4. El segundo operando es un dato inmediato que se incluye en la propia instrucción.

2. Por registro

El operando reside en un registro interno del procesador. Su sintaxis genérica en ensamblador sería la siguiente:

Instrucción registro, registro

Ejemplo:

INC EBX

Explicación: El operando reside en el registro EBX, la instrucción incrementa su valor.

3. Modo de direccionamiento en memoria

El operando reside en la memoria y admite múltiples variantes para expresar la posición donde se encuentra. Cuando se trabaja con 32 bits, el microprocesador Pentium aplica el siguiente algoritmo para calcular la dirección efectiva del operando:

Dirección = Variable + Registro Base + Registro Índice x
Factor de Escala + Desplazamiento

Variable: Permite precisar un identificador o etiqueta que representa el desplazamiento del principio de una variable.

Registro Base: Apunta al comienzo de una estructura de datos de dirección variable (EAX, EBX, ECX, EDX, ESP, EBP, ESI, DSI).

Registro Índice: Mueve el puntero dentro de las estructuras de datos para acceder a los diferentes campos de que constan (EAX, EBX, ECX, EDX, ESP, EBP, ESI, DSI).

Factor de escala: Tiene en cuenta el tamaño del elemento.

Desplazamiento: Se usa para seleccionar un dato de dirección conocida o de una estructura de datos que responde a una dirección fija.

Ejemplo:

ADD EAX, TABLA[EBP + ESI*4 + 8]

Explicación: Se almacena en EAX el resultado de la suma del contenido de EAX con el dato de 32 bits ubicado en la posición de memoria TABLA, a la que se le suma EBP + ESI*4 + 8.

Ejemplo:

MOV EAX, [EBX]

Explicación: Se almacena en EAX el contenido de la posición de memoria apuntada por EBX.

Este completo y potente sistema de direccionamiento proporciona al procesador Pentium un acceso cómodo a gran variedad de datos en la memoria, que son tan frecuentes en los lenguajes de alto nivel. Aunque el procesador Pentium admite una eficaz flexibilidad en el empleo de los registros y se ha indicado que todos los registros generales pueden actuar como Base y como Índice, hay que matizar que

cada uno de los registros está dedicado preferentemente a ciertas misiones. Por ejemplo, el registro AX es el más solicitado para la implementación de instrucciones de tipo aritmético. Los registros ESI y EDI se emplean comúnmente como registros Índice. También es habitual encontrarse con instrucciones que seleccionan implícitamente uno de los registros, como es el caso de las instrucciones con repetición o bucles (LOOP), en el que como registro contador se suele usar ECX. No obstante, el procesador Pentium admite una eficaz flexibilidad en el empleo de los registros.

Cuando se trabaja con operandos de 16 bits se emplea el siguiente algoritmo:

Dirección = Variable + Registro Base + Registro Índice + Desplazamiento

El registro Base puede ser BX o BP y el registro índice el SI o el DI. Con este modo de direccionamiento se trabaja sobre segmentos de 64 KB de tamaño máximo.

El manejo de las puertas de E/S se realiza mediante instrucciones especializadas dedicadas a esta finalidad. El procesador Pentium dedica un espacio de 64 KB no segmentado al espacio de E/S. Las instrucciones que introducen u obtienen datos por las puertas de E/S disponen de un operando que especifica la posición de éstas en el espacio de E/S, bien directamente, si el valor de la puerta es inferior a 255, o bien colocando en DX su valor, cualquiera que sea el mismo.

Clasificación y características generales del repertorio de instrucciones

Para escribir programas en lenguaje ensamblador se necesita una descripción detallada de todas y cada una de sus instrucciones. Dicha descripción debe incluir todos los formatos de operandos que admite, así como el efecto que tiene su ejecución en el procesador y sobre los datos. Esta información se incluye en los denominados manuales de programación que acompañan a cualquier procesador. En el caso del procesador Intel Pentium, y más en concreto de la arquitectura IA-32, la descripción detallada del lenguaje máquina, su arquitectura y funcionamiento se incluye en el documento extenso que lleva por título ***Intel® 64 and IA-32 Intel Architecture Software Developer's Manual*** y cuyo contenido está dividido en los siguientes tres volúmenes:

- Volumen 1. Arquitectura básica (***Basic Architecture***): describe la arquitectura básica del procesador así como su entorno de programación.
- Volumen 2. Catálogo del juego de instrucciones (***Instruction Set Reference***): describe cada una de las instrucciones del procesador y su codificación.
- Volumen 3. Guía para la programación de sistemas (***System Programming Guide***): describe el soporte que ofrece esta arquitectura al sistema operativo en aspectos tales como gestión de memoria, protección, gestión de tareas, interrupciones, etc.

Estos y otros documentos pueden encontrarse en el apartado de documentación del sitio web de Intel:

http://www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html

A continuación, se incluye un breve resumen de las instrucciones del procesador Pentium clasificadas según su función y características. Para identificar la aportación del procesador Pentium, se han tratado separadamente las instrucciones que se han incorporado en este procesador de las conocidas en modelos precedentes.

- **Instrucciones Especiales**

  La naturaleza multitarea del procesador Pentium conlleva la necesidad de asegurar que no existan interferencias entre las distintas tareas. La mayoría de las instrucciones del procesador

Pentium se pueden ejecutar en cualquier nivel de privilegio, pero hay unas pocas que pueden comprometer la integridad del sistema y por esa razón su uso está restringido. Se clasifican en dos grandes grupos:

1. **Instrucciones privilegiadas**

   Pueden ejecutarse únicamente con el máximo nivel de privilegio (segmentos que tengan CPL = 0). Se distinguen, a su vez, cuatro grupos:

   a) Cualquier instrucción que pueda modificar el campo IOPL, como *IRET*, *POPF* y las relacionadas con la conmutación de tarea, como la instrucción *CLTS* que pone a cero el señalizador TS de tarea conmutada.

   b) Instrucciones que afectan a los registros que hacen referencia a tablas que controlan el sistema en modo Protegido. Dentro de este grupo se incluyen:

      LGDT-LIDT: Respectivamente, sirven para cargar los registros GDTR e IDTR. Solo deben cargarse en la inicialización del sistema.

      SGDT-SIDT: Almacenan los contenidos de GDTR e IDTR, respectivamente.

      LLDT-LTR: Cargan a LDT y a TR, respectivamente

      SLDT-STR: Almacenan el valor de LDTR y TR, respectivamente y pueden ejecutarse en cualquier nivel de privilegio.

   c) Instrucciones que afecten a la MSW, la palabra baja de CRO.

      LMSW: Carga en la MSW el valor del operando.

      SMSW: Almacena el valor de MSW, pudiendo ser ejecutada desde cualquier nivel de privilegio.

   d) Instrucción *HLT*. Esta instrucción lleva al procesador a un estado de parada, bloqueando la ejecución de instrucciones. De este estado se puede salir mediante un RESET o una interrupción.

2. **Instrucciones protegidas**

   Desde las aplicaciones es necesario acceder a las puertas de E/S para utilizar los periféricos, pero dicho acceso debe estar bajo control del sistema. En modo Protegido, el sistema controla el campo de dos bits IOPL del registro EFLAGS que determina a partir de qué nivel de privilegio se puede acceder a los puertos de E/S. Será necesario que el CPL del segmento que intenta usar la E/S sea igual o mayor que IOPL para poder ejecutar instrucciones de E/S. Cada tarea dispone de otro mecanismo para restringir el uso de la E/S. Es el mapa de bits de permisos de E/S, que ocupa una zona del segmento TSS y que, poniendo a 1 ó a 0 los bits del mapa, se prohíbe o se permite el acceso a cada puerta de E/S. Mediante el control del campo IOPL el sistema puede establecer dinámicamente en cada tarea, un control riguroso de la E/S. Las instrucciones protegidas son:

   IN, OUT, INS y *OUTS*: Corresponden a instrucciones de operandos y cadenas, respectivamente.

   SEI y *CLI*: Permiso y prohibición de las interrupciones enmascarables, que suelen utilizar los periféricos que manejan los puertos de E/S.

- **Instrucciones Aritméticas**

Instrucción	Descripción
AAA	Tras realizar una suma, ajusta el último byte del resultado al formato BCD.
AAD	Ajusta el dividendo que va a participar en una operación de división para que el resultado, completada la división, se proporcione en el formato Cociente:Resto.
AAM	Ajusta el resultado de una operación de dos números BCD a dicho formato.
AAS	Tras realizar una resta, ajusta el último byte del resultado a BCD.
DAA	Tras sumar dos números BCD, dejando el resultado en AL, transforma dicho resultado a dos dígitos BCD.
DAS	Al efectuar una resta de dos números BCD, dejando el resultado en AL, transforma dicho resultado en dos dígitos BCD
ADD	Realiza la suma de dos operandos.
ADC	Suma dos operandos y el acarreo.
SUB	Efectúa la resta de dos operandos.
SBB	Resta el minuendo del sustraendo más el acarreo.
DEC	Decrementa el operando.
INC	Incrementa el operando.
MUL	Multiplica el operando, sin signo, por AL, AX o EAX, según el tamaño del operando, dejando el resultado en AX, EAX ó EDX:EAX, respectivamente.
IMUL	Multiplicación de operandos con signo.
DIV	Divide a AL, AX o EAX por el operando especificado en la instrucción, dejando el resultado en dos registros en forma Resto:Cociente.
IDIV	División de operqando con signo.
CBW/CWB	Convierte AL y AX a doble tamaño, respectivamente.
NEG	Realiza el complemento a dos del operando.
CMP	Compara dos operandos, reflejando el resultado solo en los señalizadores.

Ejemplo:

```
MOV AH, R6
MOV AL, R7
MOV DH, R9
MOV DL, R10
ADD AX, DX
```

Explicación: Las instrucciones MOV mueven el contenido de los respectivos registros de la memoria principal a los registros EAX y EDX de propósito general, los valores de R6, y de R7 a las partes alta y baja de EAX, y los valores R9 y R10 a las partes alta y baja de EDX. La instrucción ADD suma los valores de los registros EAX y EDX y almacena el resultado en EAX (Figura 19).

Ejemplo:

MOV AH, R6
MOV AL, R7
MOV DH, R9
MOV DL, R10
CMP AX, DX

Explicación: Las instrucciones MOV realizan las mismas operaciones que en el caso anterior. La instrucción CMP compara AX y DX y, actualiza EFLAGS (Figura 20).

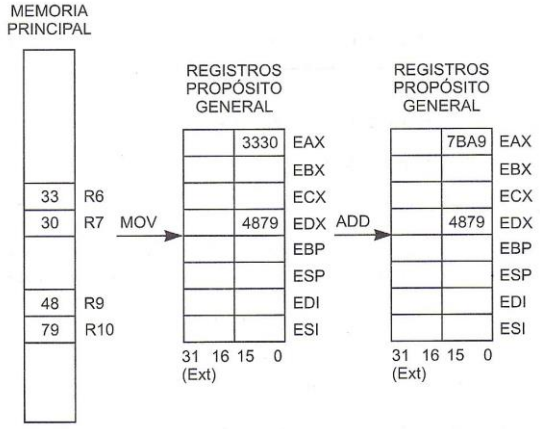


Figura 19. *Expresión gráfica del ejemplo de la suma con la instrucción ADD.*

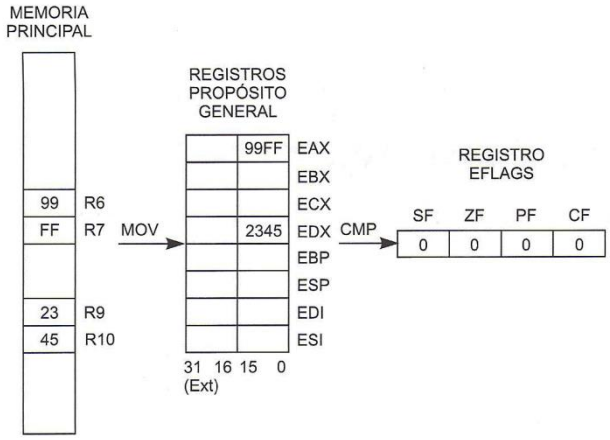


Figura 20. *Expresión gráfica del ejemplo de la comparación con la instrucción CMP.*

Nuevas instrucciones aritméticas del procesador Pentium:

Instrucción	Descripción
CWDE	Convierte una palabra a doble palabra.
CDQ	Convierte una doble palabra a cuádruple.

- **Instrucciones Lógicas**

Instrucción	Descripción
AND-OR-XOR-NOT	Realizan las operaciones lógicas AND, OR, XOR y NOT, respectivamente.
ROL/ROR	Rotación a la izquierda o derecha de los bits del primer operando, tantas veces como lo indique CL u otro operando.
RCL/RCR	Rotación a izquierda o derecha, respectivamente, de los bits del primer operando junto con el acarreo, el número de veces especificado por CL.
TEST	Realiza la operación lógica AND de los operandos, sin resultado. Solo afecta a los señalizadores.
SAL/SAR	Desplazamiento aritmético a izquierda y derecha, respectivamente.
SHL/SHR	Desplazamiento lógico sin preservar el valor del bit de signo a izquierda o derecha, respectivamente.

Nuevas instrucciones lógicas del procesador Pentium:

Instrucción	Descripción
SHLD	Desplaza el contenido del primer y segundo operando, conjuntamente, a la izquierda, el número de veces especificado en el tercer operando.
SHRD	Desplaza el contenido del primer y segundo operando, conjuntamente, a la derecha, el número de veces especificado en el tercer operando.

- **Instrucciones de Manipulación de Cadenas**

Instrucción	Descripción
CMPS-CMPSB-CMPS	Sirven para comprobar cadenas, restando el contenido del elemento apuntado por ES:DSI, con el de DS:ESI, afectando solo a los señalizadores.
MOVS	Mueve el contenido de la dirección apuntada por DS:ESI a la

Instrucción	Descripción
	referenciada por el puntero ES:EDI, incrementando el valor de los registros índice.
LODS-ODSB-LODSW	Transfiere al Acumulador es elemento apuntado por DS:ESI.
STOS-STOSB-STOSW	Transfiere el contenido del Acumulador al elemento apuntado por ES:ESI.
SCAS-SCASB-SCASW	Resta el contenido direccionado por ES:EDI al Acumulador, afectando exclusivamente a los señalizadores.
INS-INSB-INSW	Carga en la dirección apuntada por ES:EDI el valor de la puerta de E/S apuntada por DX.
OUTS-OUTSB-OUTSW	Saca por la puerta apuntada por DX, uno o dos bytes a partir de la dirección señalada por ES:EDI.
REP	Es un prefijo que se antepone a ciertas instrucciones de cadenas. Repite la ejecución de la instrucción a la que antecede tantas veces como indica el valor del contenido de ECX.
REPE-REPZ	Se repite la instrucción a la que antecede hasta que ECX valga 0, o bien, el se cumpla que el señalizador Z = 0.
REPNE-REPNZ	Igual que la anterior, excepto que termina la repetición cuando Z = 1, o bien ECX = 0.
XLAT-XLATB	Deposita en AL el contenido de la dirección obtenida al sumar EBX + AL. Sirve principalmente para la creación de ficheros.

Ejemplo:

CMPS DS, CS

Explicación: Compara la cadenas almacenadas en los registros ESI de DS con la cadena almacenada en EDI de CS y según sea el resultado actualiza el registro EFLAGS (Figura 21).

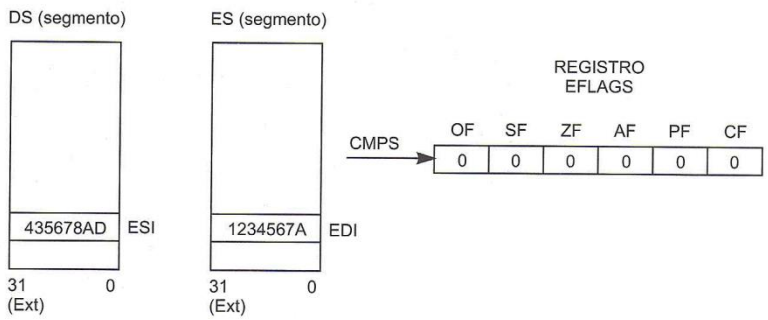


Figura 21. *Resultado de comparar dos cadenas y actualización de EFLAGS.*

- **Instrucciones de Transferencia de Control**

Instrucción	Descripción
JMP	Realiza un salto incondicional a la dirección que se indica en el descriptor referenciado.
LOOP	Provoca un salto a una etiqueta corta (situada entre -128 y + 127 posiciones).
LOOPZ-LOOPE-LOOPNZ-LOOPNE	Salto corto mientras ECX no sea 0 y Z = 0 ó 1, según sean las dos primeras o las dos últimas.
CALL	Llamada a una rutina.
RET	Retorno de una rutina.

A continuación se enumeran las instrucciones de transferencia de control especiales.

Instrucción	Descripción
JA	Salta si el primer operando es mayor que el segundo. Sin signo.
JAE	Salta si el primer operando es menor que el segundo. Sin signo.
JB	Salta si el primer operando es menor o igual que el segundo. Sin signo.
JBE	Salta si el primer operando es mayor o igual que el segundo. Sin signo.
JE	Salta si el bit Z de EFLAGS es igual a 1.
JO	Salta si el bit de OVERFLOW de EFLAGS es igual a 1.
JNO	Salta si el bit de OVERFLOW de EFLAGS es igual a 0.
JCXZ	Salta si el registro de CX es distinto de 0.
JS	Salta si el bit de S de EFLAGS es igual a 1.
JNS	Salta si el bit de S de EFLAGS es igual a 0.
JNA	Salta si el primer operando es mayor que el segundo. Con signo.
JNAE	Salta si el primer operando es menor que el segundo. Con signo.
JNB	Salta si el primer operando es menor o igual que el segundo. Con signo.
JNBE	Salta si el primer operando es mayor o igual que el segundo. Con signo.
JC	Salta si el bit de CARRY de EFLAGS es igual a 1.

JNC	Salta si el bit de CARRY de EFLAGS es igual a 0.
JP	Salta si la paridad es impar o no hay paridad.
JNP	Salta si la paridad es par o hay paridad.

Ejemplo:

CMP AX, DX
JC FF45679E

Explicación: Se comparan los contenidos de los registros AX y DX y, si son iguales, se pone el señalizador de acarreo (CARRY) de EFLAGS a 1. En este caso, la instrucción JC provoca un salto de instrucción dejando la dirección destino en el registro contador de programa.

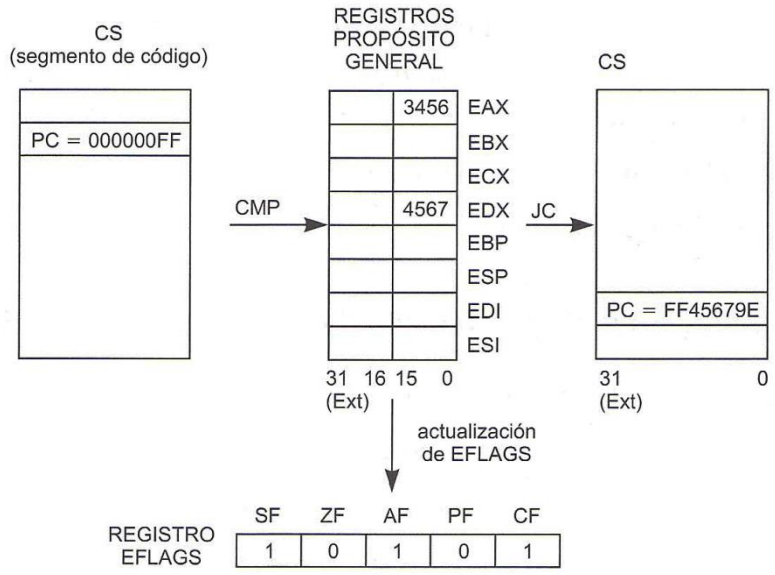


Figura 22. *Representación gráfica de un salto condicional si el bit CARRY = 1, dentro de un segmento de código.*

- **Instrucciones de Transferencia de Datos**

Instrucción	**Descripción**
IN-OUT	Entrada y salida de información desde las puertas de E/S.
POP	Se transfiere desde la cima de la pila una información a un registro.
POPA	Transfiere desde la pila ocho palabras que se cargan en ocho registros generales.
POPF	Transfiere desde la cima de la pila una palabra al registro de señalizadores.
PUSH	Transfiere un registro a la pila.
PUSHA	Transfiere a la cima de la pila ocho registros generales de 16 bits (AX, BX, CX, DX, SP, BP, SI y DI)
PUSHF	Transfiere a la pila el registro de los señalizadores.

MOV	Transfiere al primer operando el valor del segundo.
XCHG	Intercambia el contenido de los operandos.
LEA	Transfiere a un registros el valor de la dirección que corresponde al segundo operando.

Ejemplo:

IN AL, (F4)

Explicación: Lee la puerta F4 y transfiere su valor inmediato a AL.

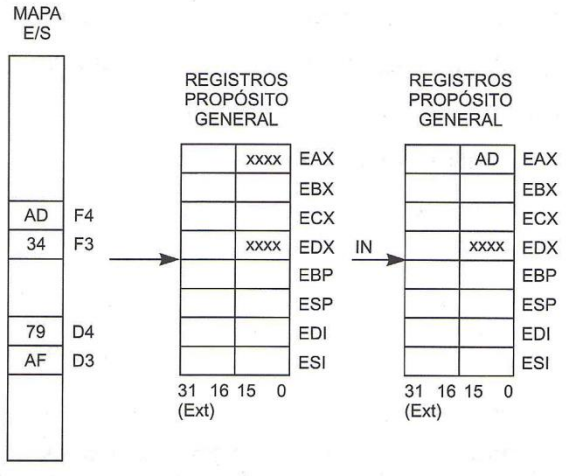


Figura 23. *Se carga un valor de una posición del mapa de E/S en AL mediante una instrucción IN.*

Ejemplo:

MOV AX, R6

MOV DX, R9

Explicación: Se mueve el contenido de los registros de memoria principal R6 y R9 a AX y DX, respectivamente.

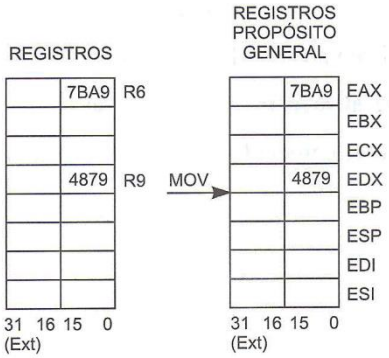


Figura 24. *Representación gráfica de movimiento de operandos mediante MOV.*

Nuevas instrucciones de transferencia de datos:

El procesador Pentium admite nuevas instrucciones de transferencia para operandos de 32 bits, como *POPAD, POPFD, PUSHFD* y *PUSHAD*. También está la instrucción *MOVSX*, que extiende el signo de un byte o una palabra a 16 o 32 bits, respectivamente.

- **Instrucciones de Control de los Señalizadores**

Instrucción	Descripción
CLC-STC	Ponen a 0 ó a 1 el señalizador de acarreo, respectivamente.
CLD-STD	Ponen a 0 ó a 1 el señalizador D, respectivamente.
CMC	Complementa el valor del señalizador C.
CLI	Pone a 0 el señalizador IF. Inhabilita interrupciones enmascarables.
LAHF	Transfiere el byte de menos peso del registro de señalizadores al registro AH.
SAHF	Transfiere el contenido del registro AH al byte de menos peso del registro de señalizadores.

- **Instrucciones de Asignación Condicional**

  Es un grupo de instrucciones nuevas en el procesador Pentium. Si se cumple la condición implícita en la instrucción, se pone a 1 el operando y, en caso contrario, se pone a 0.

Instrucción	Descripción
SETA	Pone a uno si el primer operando es mayor que el segundo. Sin signo.
SETAE	Pone a uno si el primer operando es menor que el segundo. Sin signo.
SETB	Pone a uno si el primer operando es menor o igual que el segundo. Sin signo.
SETBE	Pone a uno si el primer operando es mayor o igual que el segundo. Sin signo.
SETC	Pone a uno si el bit de CARRY del EFALGS es igual a 1.
SETNC	Pone a uno si el bit de CARRY del EFALGS es igual a 0.
SETE	Pone a uno si el bit Z de EFLAGS es igual a 1.
SETO	Pone a uno si el bit de OVERFLOW de EFLAGS es igual a 1.
SETNO	Pone a uno si el bit de OVERFLOW de EFLAGS es igual a 0.
SETCXZ	Pone a uno si el registro CX es distinto de 0.
SETS	Pone a uno si el bit S de EFLAGS es igual a 1.
SETNS	Pone a uno si el bit S de EFLAGS es igual a 0.
SETNA	Pone a uno si el primer operando es mayor que el segundo. Con signo.
SETNAE	Pone a uno si el primer operando es menor que el segundo. Sin signo.
SETNB	Pone a uno si el primer operando es menor o igual que el segundo. Sin signo.

SETNBE	Pone a uno si el primer operando es mayor o igual que el segundo. Sin signo.
SETP	Pone a uno si la paridad es impar o no hay paridad.
SETNP	Pone a uno si la paridad es par o hay paridad.

- **Instrucciones de Bit**

  Todas las instrucciones de bit son nuevas en el procesador Pentium.

Instrucción	**Descripción**
BT	Asigna al señalizador CF el valor del bit del primer operando, quedando especificada su posición por el segundo operando.
BTC	Realiza la misma función que BT, pero también complementa el bit especificado en la instrucción.
BTR	Igual que BT, pero poniendo a 0 el bit especificado en la instrucción.
BTS	Igual que BT, pero poniendo a 1 el bit especificado en la instrucción.
BSF	Si todos los bits de menor a mayor peso del primer operando, especificados por el segundo, son ceros, entonces pone el señalizador Z a 1. Si se encuentra algún bit a 1 entonces pondrá Z = 0, guardando en el primer operando la posición del primer bit a 1 que se haya encontrado.
BSR	Igual que BSF, pero la exploración se realiza desde el bit de mayor peso hacia el de menos peso.

- **Instrucciones de Alto Nivel**

Instrucción	**Descripción**
BOUND	Comprueba que un operando está comprendido entre dos límites, y en caso contrario, general la excepción 5.
ENTER	Sirve para crear una nueva pila que se usa tras una llamada a un procedimiento, un espacio reservado para el paso de parámetros. El primer operando expresa el número de bytes que se reservan en la nueva pila. El segundo operando expresa el nivel de anidamiento.
LEAVE	Elimina el espacio reservado en la pila del procedimiento saliente, asignando como nueva pila la del procedimiento anterior en anidamiento.

- **Instrucciones Multisegmento**

Implican la utilización de más de un segmento por procedimiento.

Instrucción	**Descripción**
CALL	Llamada a una rutina o procedimiento.
RET	Retorno de una rutina.
INT	Llamada a un programa de manejo de interrupción.
INTO	Llamada a la entrada 4 de la tabla de interrupciones, si OF = 1.
IRET	Retorno de un programa de interrupción.
LDS	Carga simultáneamente el registro DS y otro general con seis bytes de momoria.
LES	Igual que la anterior, pero cargando el registro ES.
JMP	Salto a un segmento de código.

Ejemplo:

IRET

Explicación: Cuando se genera una interrupción se almacena en una posición de la pila el valor del contador de programa, incrementado en uno. Al volver de la rutina de interrupción, se carga en el contador de programa el valor de esa posición de la pila.

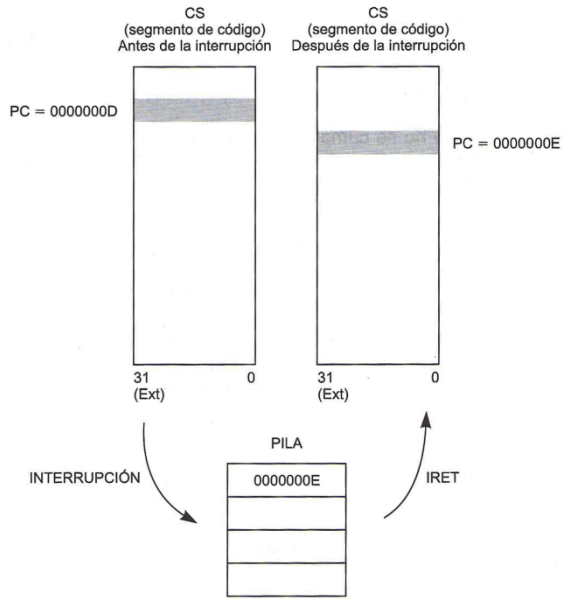


Figura 25. *Ejemplo de comportamiento de la instrucción IRET.*

Nuevas instrucciones multisegmento:

Instrucción	Descripción
LFS-LGS-LSS	Igual que LDS y LES, pero afectando a los registros FS, GS y SS.
MOV-POP-PUSH	Permiten realizar transferencias de registros de segmento a otros registros o a la pila.

- **Instrucciones del Sistema Operativo**

Instrucción	Descripción
ARPL	Comprueba el RPL de primer operando con el del segundo. Si es menor, Z = 1 e iguala los RPL al valor del segundo.
CLTS	Pone a 0 el bit TS de la MSW situada en CR0.
HLT	Detiene la ejecución del programa.
LAR	Carga en el primer operando los derechos de acceso del descripctor al que hace referencia el segundo operando.
LGDT	Carga el GDTR desde una posición de memoria.
LIDT	Carga el IDTR desde una posición de memoria.
LLDT	Carga el registro LDTR.
LMSW	Carga a MSW.
LSL	Carga el primer operando con el límite del segmento especificado por el segundo operando.
LTR	Carga el registro TR.
SGDT-SIDT-SLDT-SMSW-STR	Almacena en la memoria a los registros GDTR, IDTR, LDTR, MSW y TR.
VERR	Verifica si puede ser leído el segmento definido por el selector que se proporciona en el operando de la instrucción. En caso afirmativo, Z = 1.
VERW	Verifica si puede ser escrito un segmento.

- **Instrucciones para el Coprocesador**

Instrucción	Descripción
ESC	Precede a las instrucciones que debe procesar el coprocesador matemático. Indica a los procesadores x86 anteriores al Pentium que lo que le sigue a este prefijo lo envíe al coprocesador.
WAIT	Se detiene el procesador x86 hasta que la señal BUSY# del microprocesador se desactive. Así, el procesador espera al coprocesador. Esto se usaba en procesadores anteriores al Pentium, como el 80386 que disponía de coprocesador externo. No se usa en el procesador Pentium que tiene integrada la FPU.

- **Otras Instrucciones**

Instrucción	Descripción
NOP	No efectúa operación alguna.
LOCK	Es un prefijo de algunas instrucciones que, al ser ejecutadas, activan la señal de salida LOCK# del microprocesador, impidiendo la cesión del bus hasta la finalización de la instrucción con LOCK.

- **Nuevas Instrucciones del microprocesador Pentium**

Instrucción	Descripción
CMPXCHG8B	Compara el valor de 64 bits ubicado en EDX:EAX con un valor de 64 bits situado en memoria. Si son iguales, el valor en memoria se sustituye por el contenido de ECX:EBX, y el indicador ZF se pone a uno. En caso contrario, el valor en memoria se carga en EDX:EAX y en indicador ZF se pone a cero.
CPUID	*CPU Identification*. Le informa al software acerca del modelo de microprocesador en que se está ejecutando. Un valor cargado en EAX antes de ejecutar esta instrucción indica qué información deberá retornar CPUID. • Si EAX = 0, se cargará en este registro el máximo valor de EAX que se podrá utilizar en CPUID, para el procesador Pentium este valor es 1. Además, en la salida aparece la cadena de identificación del fabricante contenido en los registros EBX, ECX y EDX. El registro EBX contendrá los primeros cuatro caracteres, EDX los siguientes cuatro y ECX los últimos cuatro. Para los procesadores Intel la cadena es "GenuineIntel". • Si EAX = 1, entonces EAX[3:0] contendrá la identificación de la revisión del microprocesador, EAX[7:4] contendrá el modelo y EAX[11:8] contiene la familia (el valor será 5 para

Instrucción	Descripción
	el Pentium). El procesador pone el registro de características en EDX a 1BFh, indicando las características que soporta el procesador Pentium. Un bit puesto a 1 indica que esa característica está soportada.
RMDSR	*Read from Model-Specific Register*. El valor en ECX especifica uno de los registros de 64 bits específicos del modelo del procesador del procesador. El contenido de ese registro se carga en EDX:EAX. EDX se carga con los 32 bits más significativos, mientras que EAX se carga con lo 32 bits menos significativos.
RDTSC	*Read from Time Stamp Counter*. Copia el contenido del contador de tiempo (TSC) en EDX:EAX. El microprocesador Pentium mantiene un contador de tiempo de 64 bits que se incrementa por cada ciclo de reloj. Cuando el nivel de privilegio actual es cero el estado del bit TSD en el registro de control CR4 no afecta a la operación de esta instrucción. En los niveles de privilegio 1, 2 o 3 el TSC se puede leer solo si el bit TSD de CR4 vale cero.
RSM	*Resume from System Management Mode*. El estado del procesador se restaura utilizando la copia que se creó al entrar al modo de manejo del sistema (SMM). Sin embargo, los contenidos de los registros específicos del modelo no ven afectados. El procesador sale del SMM y retorna el control a la aplicación o sistema operativo interrumpido. Si el procesador detecta información inválida, entra en el estado de apagado (*shutdown*).
WRMSR	*Write to Model-Specific Register*. El valor de ECX especifica uno de los registros de 64 bits específicos del modelo de procesador. El contenido EDX:EAX se carga en ese regisro. EDX debe contener los 32 bits más significativos, mientras que EAX debe contener los 32 bits menos significativos.

IV. BIBLIOGRAFÍA

- **Angulo Usategui, J.M.; Gutiérrez Temilo, J.L.; Angulo Martínez, I**. *Arquitecturas de Microprocesadores. Los Pentium a fondo*. Editorial Paraninfo-Thomson.
- Sitio web de Intel: www.intel.com.
- Sitio web: http://lc.fie.umich.mx/~rochoa/Manuales/ENS_Arquitectura_5ium/