UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

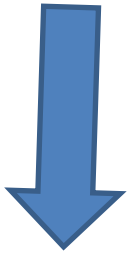
Microprocesadores

Registros

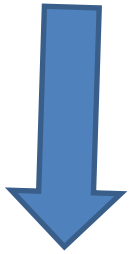
Palabra

- Una palabra es una cadena finita de bits
- **Una Palabra son bits manejados como un conjunto por la máquina o componente**.
- El tamaño de una palabra se refleja en muchos aspectos de la estructura y las operaciones de las computadoras. La mayoría de los registros en un ordenador normalmente tienen el tamaño de la palabra. El valor numérico típico manipulado por un ordenador es probablemente el tamaño de palabra.
- Los ordenadores modernos normalmente tienen un tamaño de palabra de 32 o 64 bits.
- No confundir esta definición con las estructuras de datos o tipos de datos

Microprocesador vs Microcontrolador

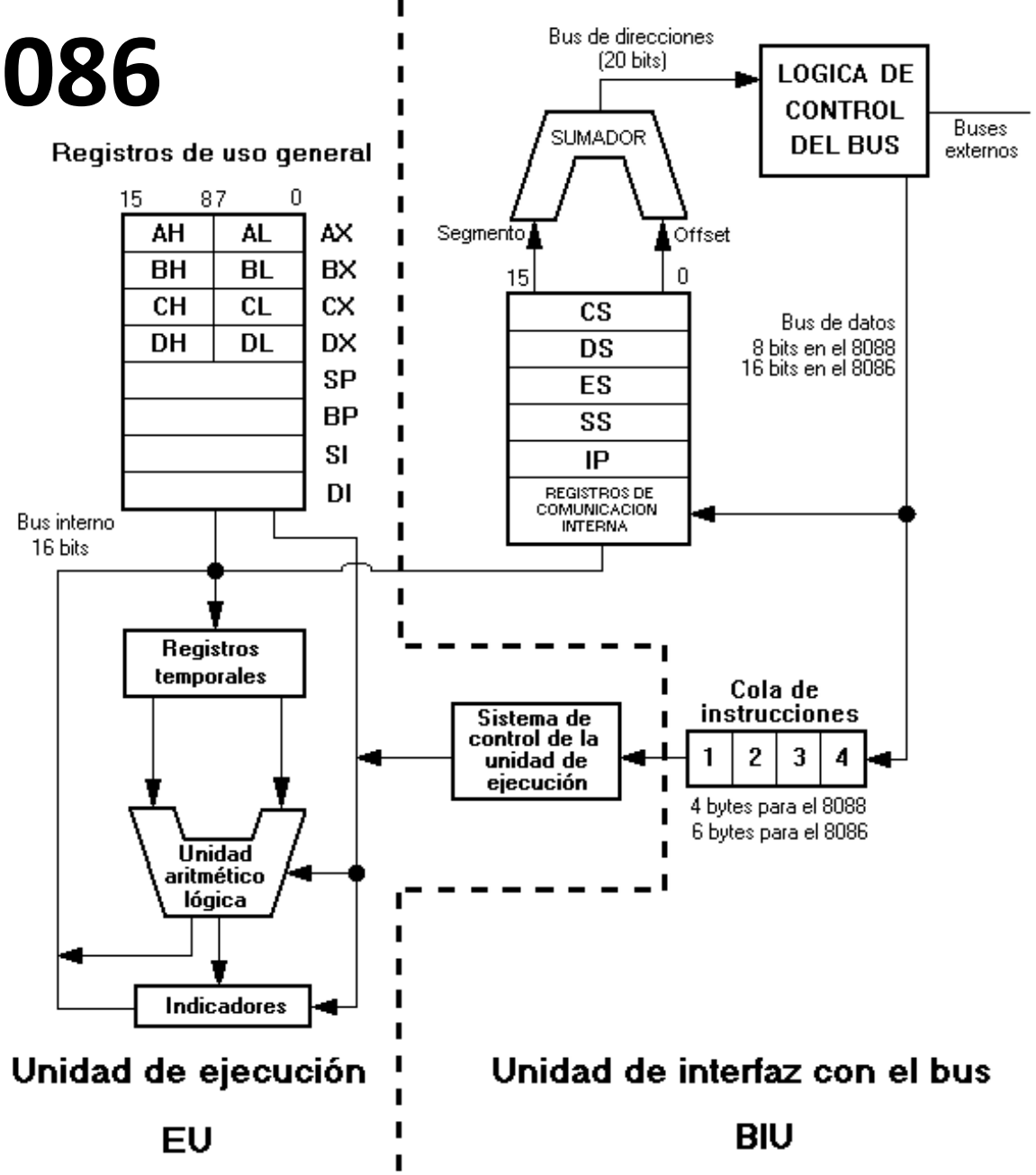


Arquitectura Abierta de propósito general y configurable de acuerdo con el negocio



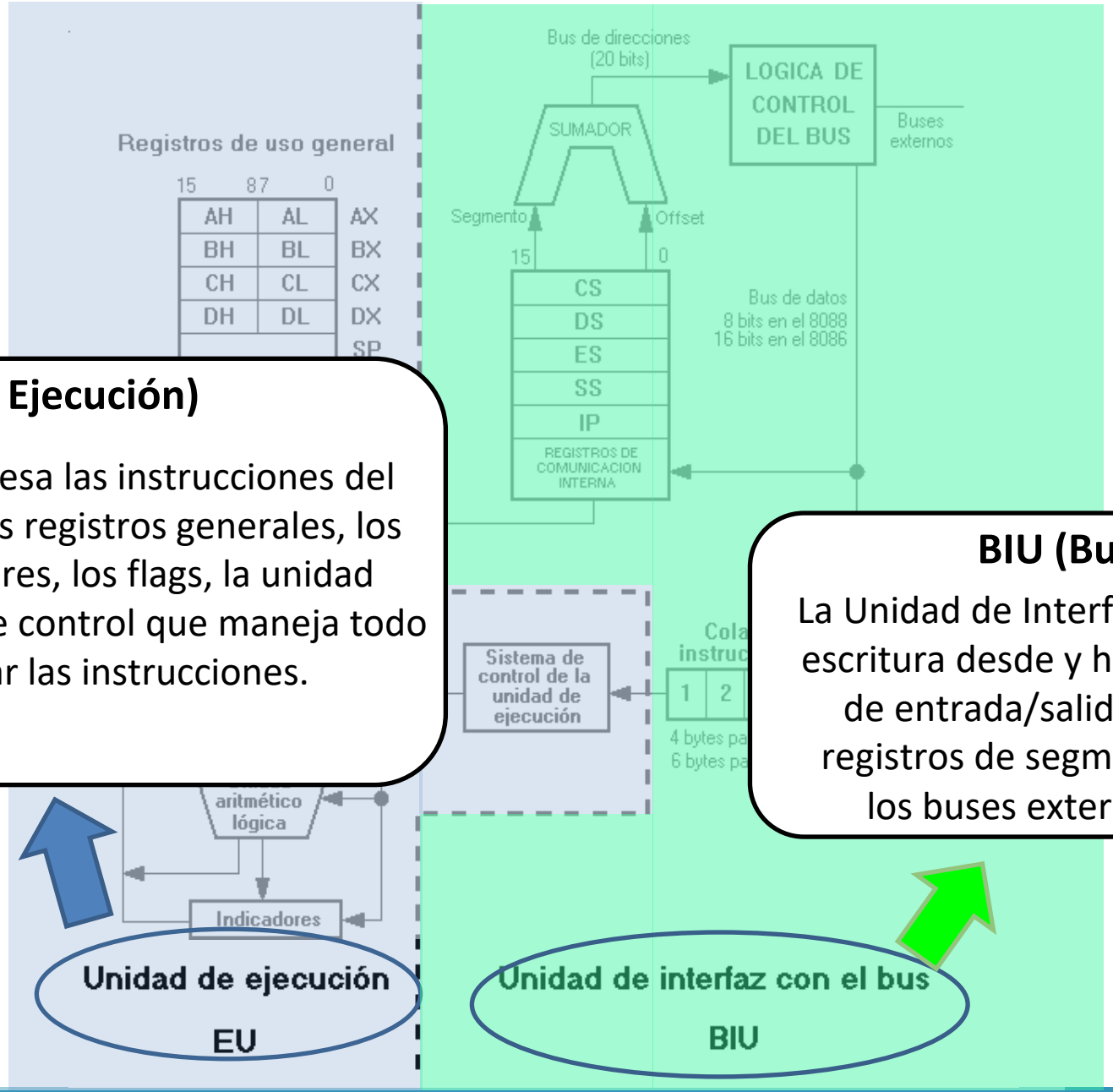
Propósito específico, arquitectura cerrada.

Arquitectura 8086



EU (Unidad de Ejecución)

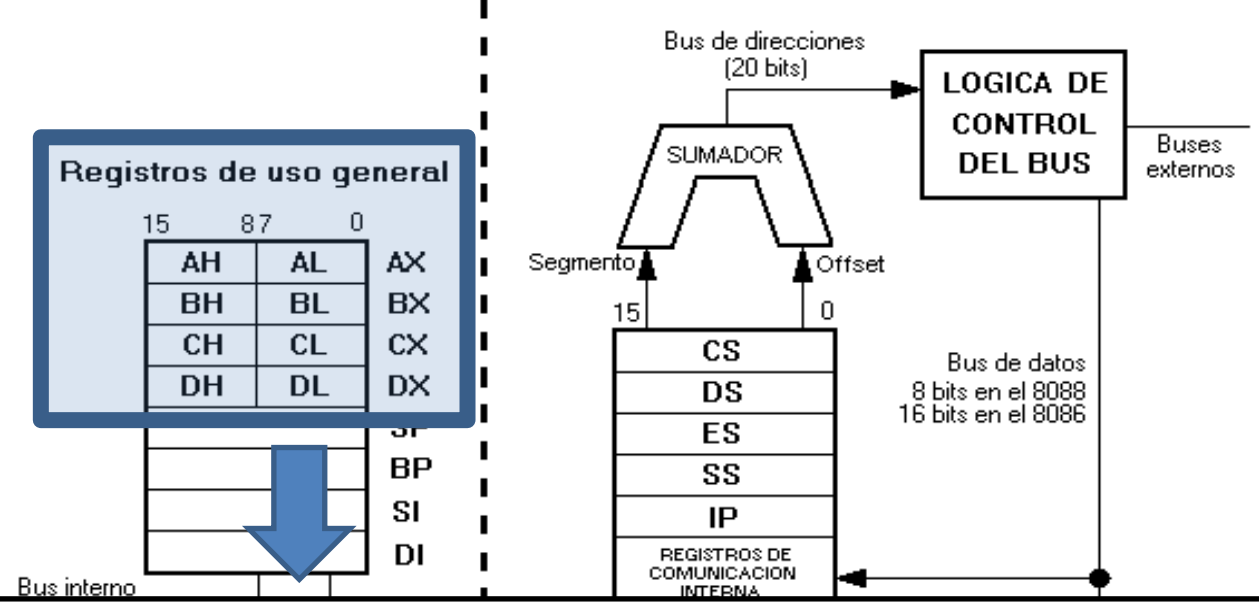
La Unidad de Ejecución procesa las instrucciones del CPU. Está conformada por los registros generales, los registros índice y apuntadores, los flags, la unidad aritmético lógica, y la lógica de control que maneja todo el proceso para ejecutar las instrucciones.



BIU (Bus Interface Unit)

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, y lógica para controlar los buses externos del microprocesador.

Registros



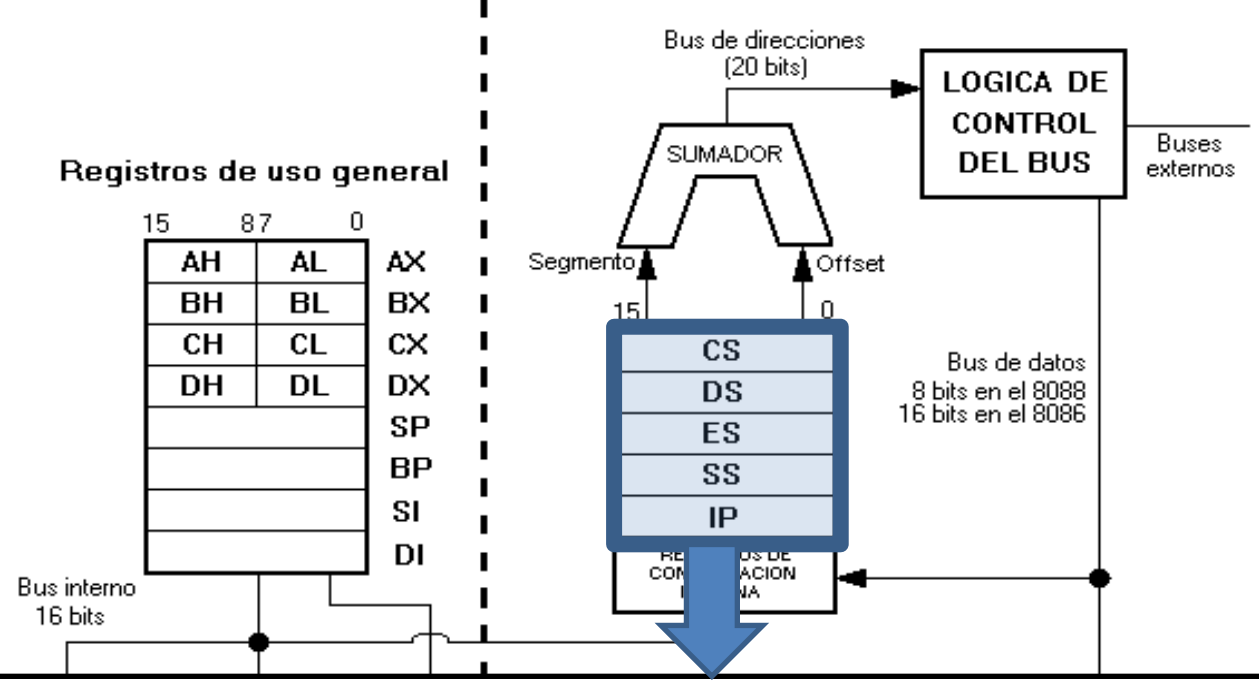
Registros de Propósito General

AX: Usualmente utilizado para guardar el resultado de las sumas. Es un registro que se emplea en todas las operaciones lógico-aritméticas. 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 ALU.

BX: Usualmente guarda la base sobre la cual se va a aplicar un desplazamiento de acceso a memoria. Contiene 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 Pentium (dirección larga), mientras el registro BX (16 bits) para manejar la memoria del 8086 (dirección corta).

CX: Usualmente utilizado como contador. Se carga con el número de veces que se ejecuta una instrucción (iteraciones).

DX: De datos. Este registro de propósito general se emplea para contener las direcciones de los puertos de entrada y salida en las instrucciones que manejan el mapa de E/S.



Registros de Segmento

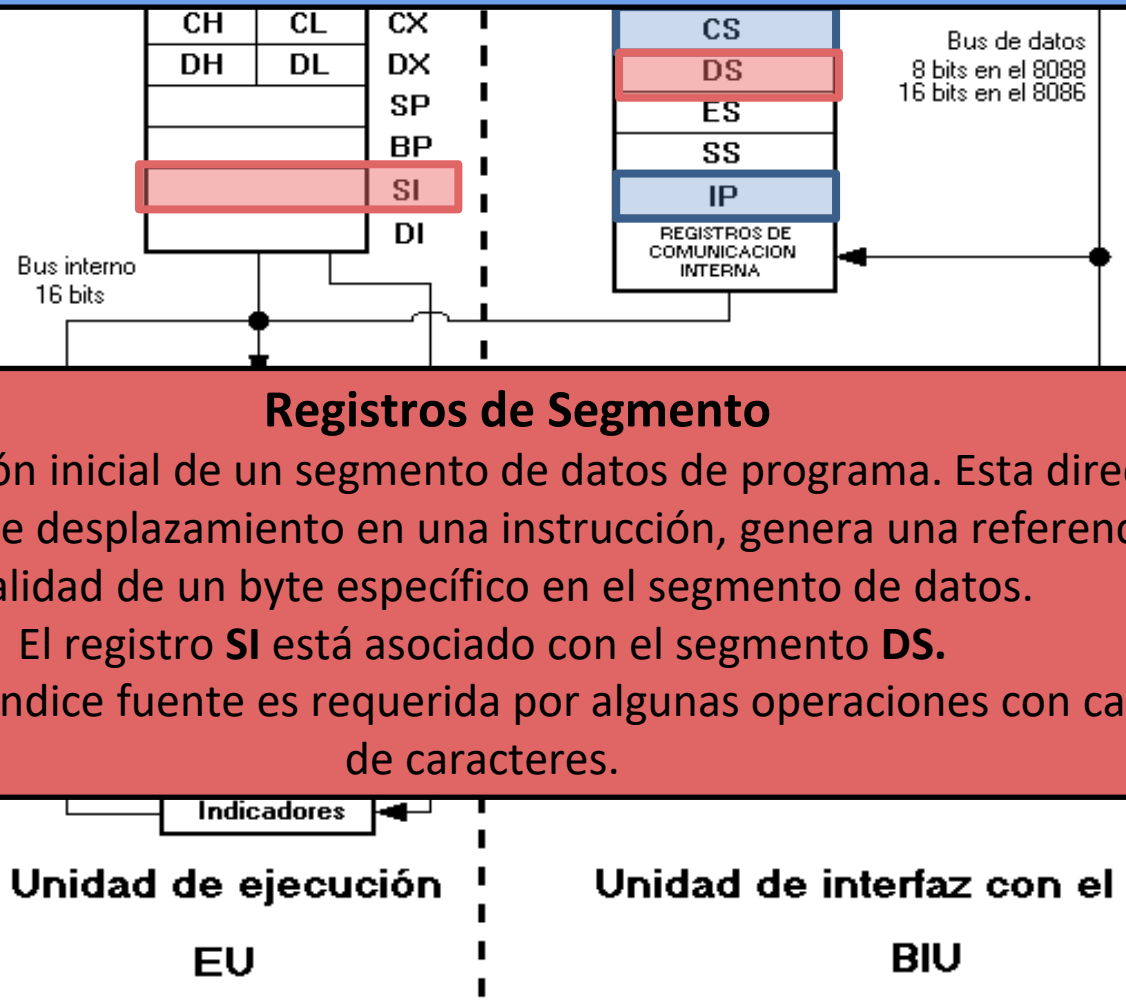
Definen áreas tamaño variable dentro del espacio de direcciones del 8086. Estas áreas pueden solaparse total o parcialmente. No es posible acceder a una posición de memoria no definida por algún segmento.

Unidad de ejecución — **EU**

Unidad de interfaz con el bus — **BIU**

Registros de Segmento

CS: Dirección inicial del segmento de código de un programa. 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.



Registros de Segmento

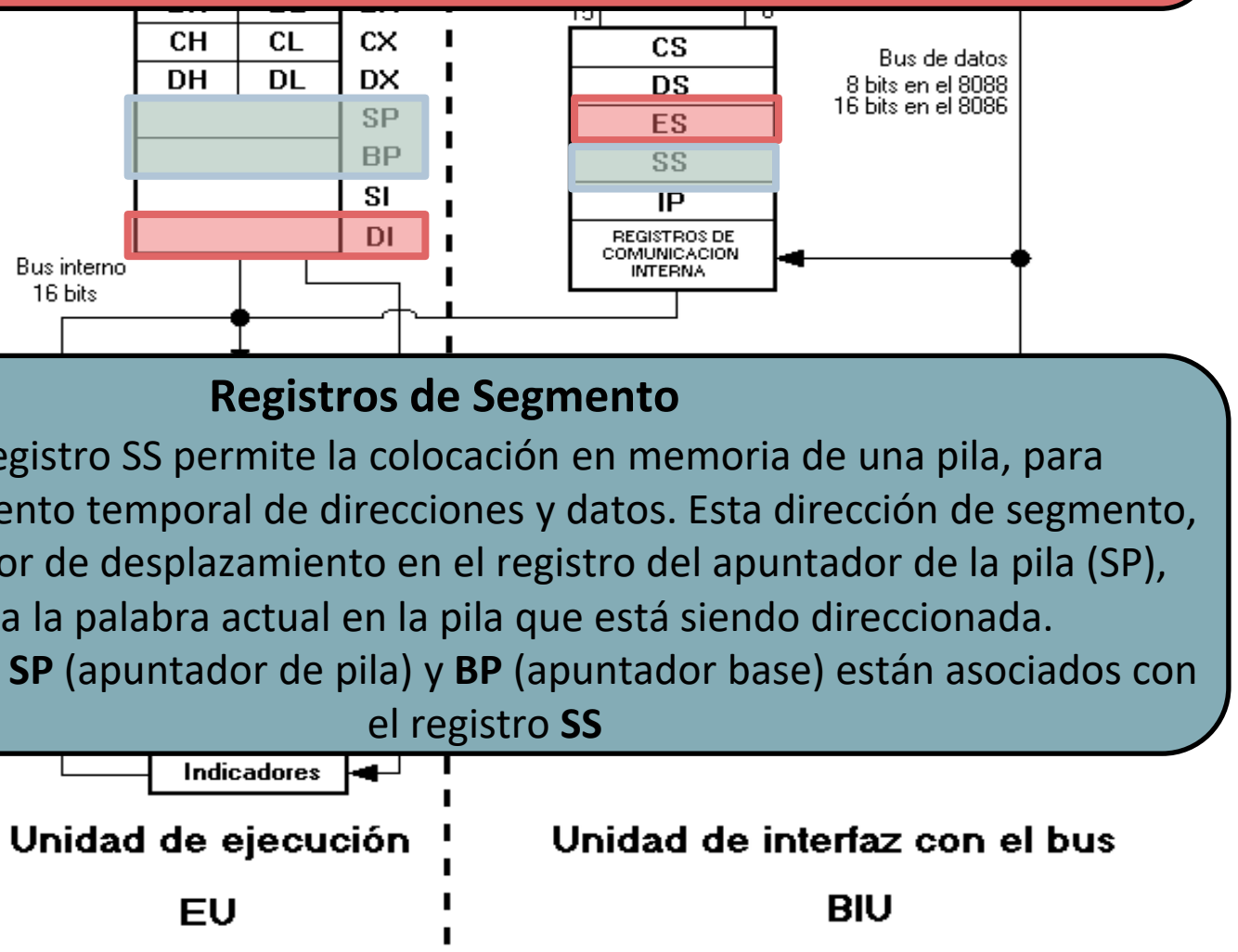
DS: La dirección inicial de un segmento de datos de programa. Esta dirección, más un valor de desplazamiento en una instrucción, genera una referencia a la localidad de un byte específico en el segmento de datos.

El registro **SI** está asociado con el segmento **DS.**

SI: El registro índice fuente es requerida por algunas operaciones con cadenas de caracteres.

Registros de Segmento

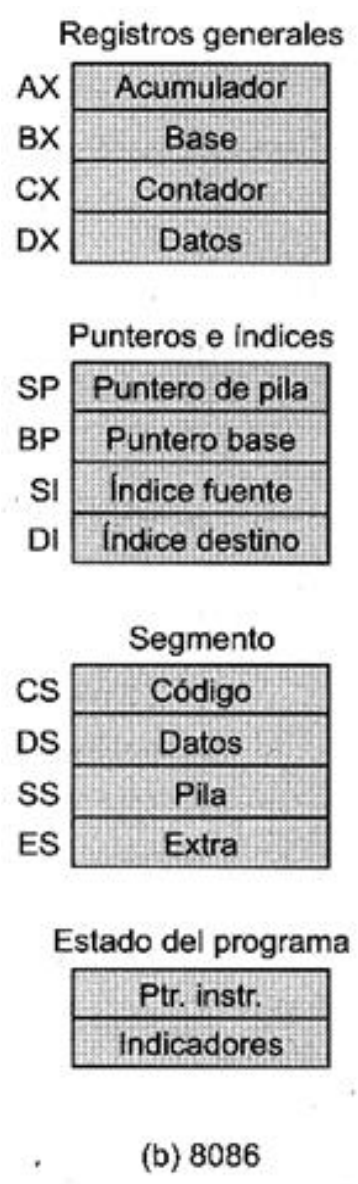
ES: Algunas operaciones con cadenas de caracteres utilizan el registro extra de segmento para manejar el direccionamiento de memoria. El registro **ES** está asociado con el registro **DI** (Índice).



Registros de Segmento

SS: El registro SS permite la colocación en memoria de una pila, para almacenamiento temporal de direcciones y datos. 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.
Los registros **SP** (apuntador de pila) y **BP** (apuntador base) están asociados con el registro **SS**

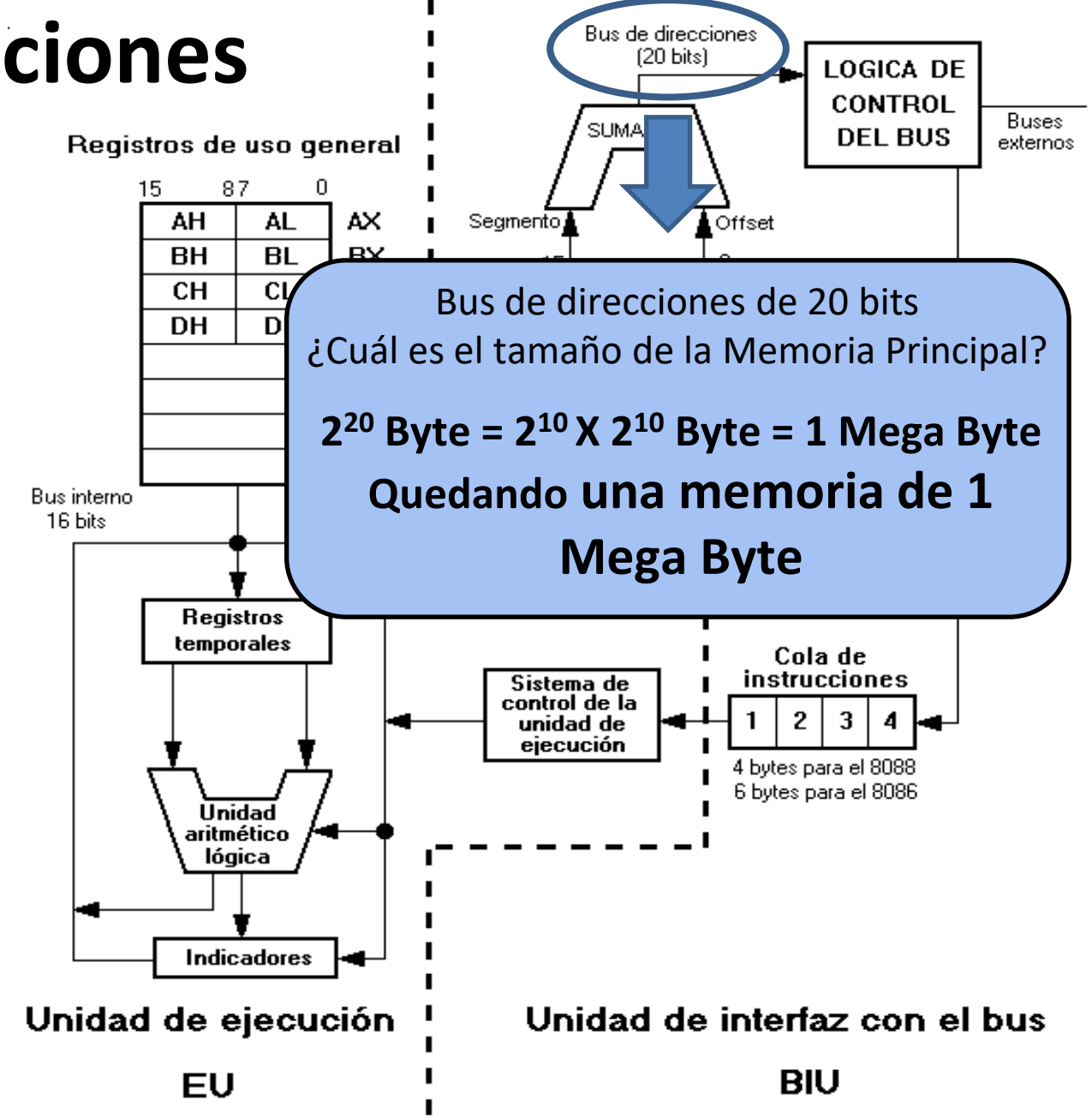
Evolución de los registros



Registros de Segmento

En el 386 se incrementaron a 32 bits.
Actualmente son de 64 bits.

Bus de Direcciones



Bus de direcciones de 20 bits
¿Cuál es el tamaño de la Memoria Principal?

2^{20} Byte = 2^{10} X 2^{10} Byte = 1 Mega Byte

Quedando una memoria de 1 Mega Byte

Direccionamiento Segmentado

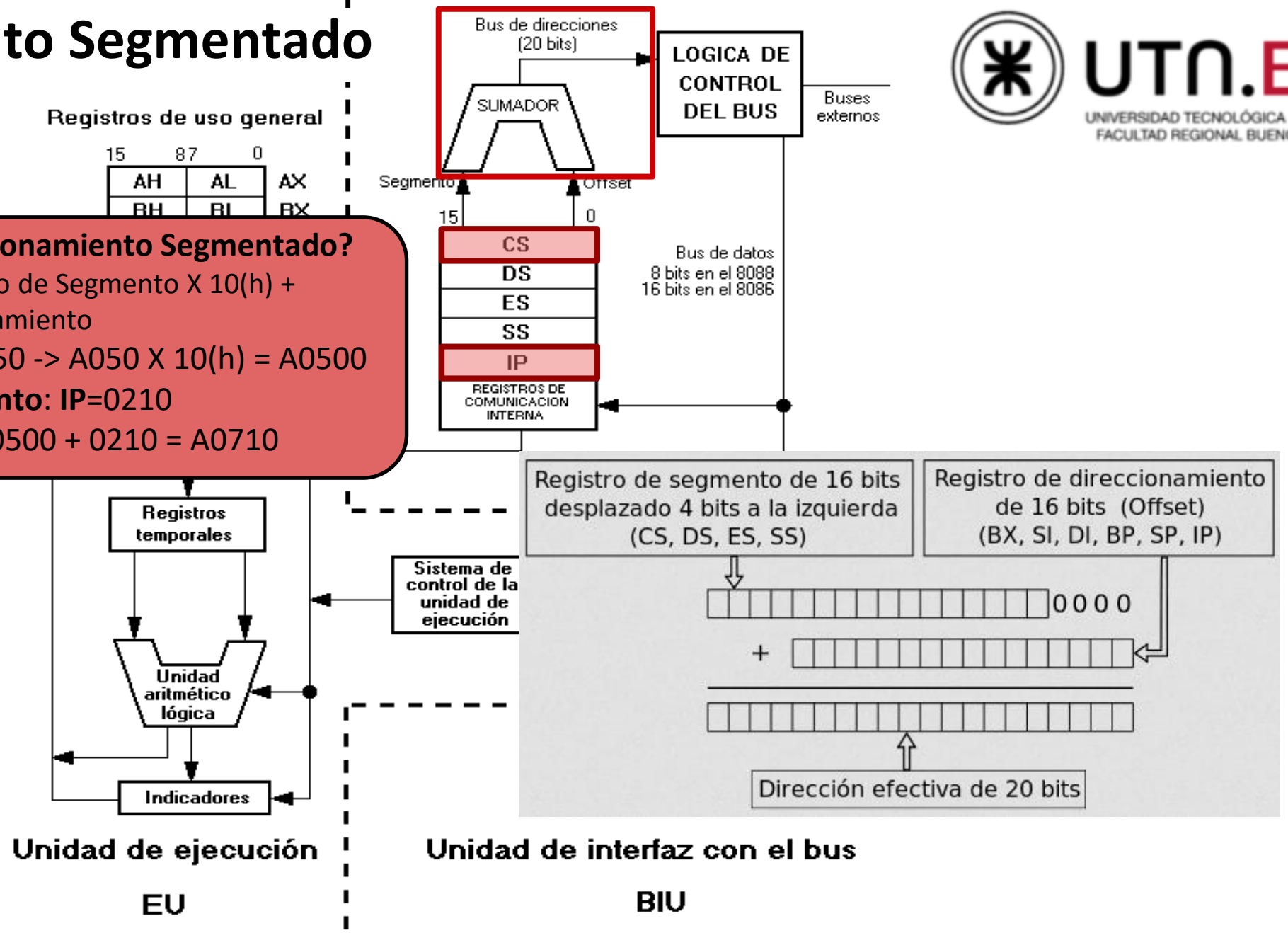
¿Cómo funciona el Direccionamiento Segmentado?

Dirección Física = Registro de Segmento X 10(h) + Desplazamiento

Registro Segmento: **CS**=A050 -> A050 X 10(h) = A0500

Desplazamiento: **IP**=0210

Dirección Física -> A0500 + 0210 = A0710



Registro de Estado 8086

C : acarreo en la suma y arrastre en la resta
P : paridad del dato (0, impar y 1, par)
A : acarreo auxiliar. Indica el acarreo o arrastre entre los bits 3 y 4
Z : indicación de resultado igual a cero
S : indicador de signo del resultado. 0, positivo y 1, negativo
T : trampa. Habilita la característica de depuración del procesador
I : habilitación de interrupciones de hardware
D : selección de incremento o decremento en los índices
O : sobreflujo.

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Set de instrucciones

La cantidad de instrucciones diferentes que un microprocesador "puede entender" o, en términos técnicos, decodificar, y por lo tanto ejecutar, en la mayoría de los casos implica su mejor capacidad para "hacer cosas distintas".

CISC vs RISC

•En arquitectura computacional, CISC (del inglés Complex Instruction Set Computer, en español Computador con Conjunto de Instrucciones Complejas) es un modelo de arquitectura de computadores. Los microprocesadores CISC tienen un conjunto de instrucciones que se caracteriza por ser muy amplio y permitir operaciones complejas entre operandos situados en la memoria o en los registros internos, en contraposición a la arquitectura RISC.

•En arquitectura computacional, RISC (del inglés Reduced Instruction Set Computer, en español Computador con Conjunto de Instrucciones Reducidas) es un tipo de diseño de CPU generalmente utilizado en microprocesadores o microcontroladores con las siguientes características fundamentales:

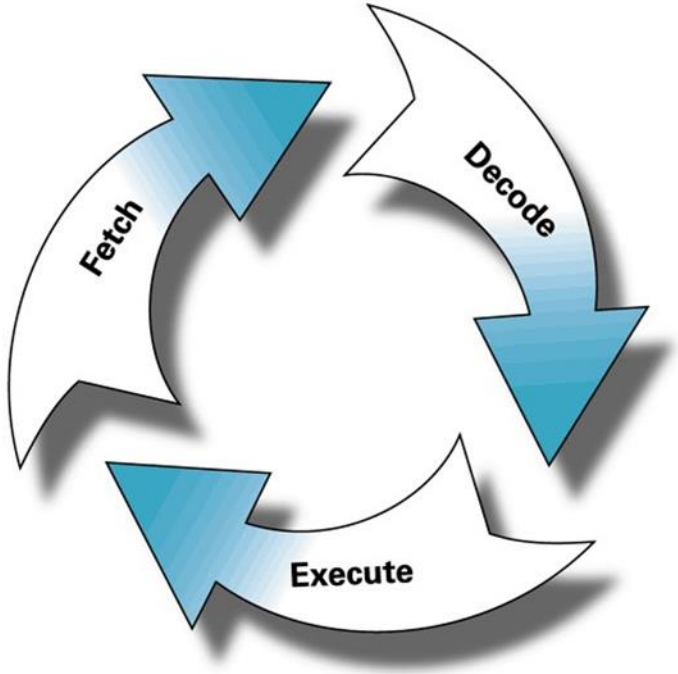
- –Instrucciones de tamaño fijo y presentadas en un reducido número de formatos.
- –Sólo las instrucciones de carga y almacenamiento acceden a la memoria de datos.

EPIC

- Procesamiento de instrucciones explícitamente en paralelo (del inglés EPIC: Explicitly Parallel Instruction Computing) es un paradigma de programación que comenzó a investigarse a principios de los años 80.
- El objetivo de EPIC era aumentar la capacidad de los microprocesadores para ejecutar instrucciones de software en paralelo mediante el uso del compilador, o sea que la dependencia entre instrucciones sea encontrada y manejada por el compilador y no por el procesador.
- De esta manera el programa ejecutable ya está "organizado" antes de su ejecución.
- Esto permitiría escalar el rendimiento más rápidamente en los futuros diseños de procesadores, sin tener que recurrir a frecuencias de reloj cada vez más altas, las cuales se han convertido desde ese momento en una problemática importante debido a problemas de alimentación y refrigeración.

Etapas de Ejecución

1. Recuperar la siguiente instrucción desde memoria (apuntada por el *program counter*) y luego incrementar el *program counter*.



2. Decodificar el patrón de bits en el registro de instrucción IR

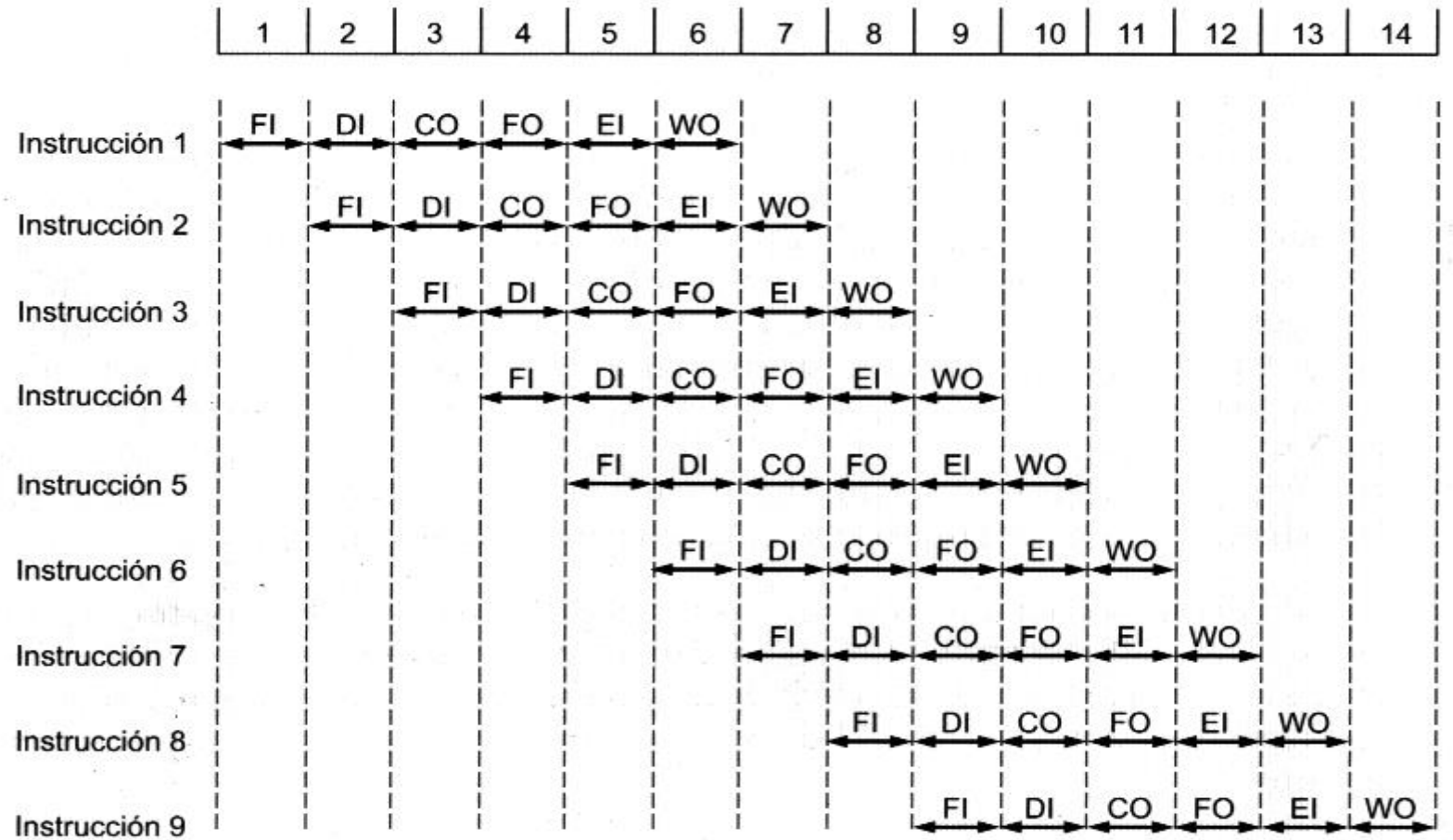
3. Ejecutar la instrucción indicada en el registro de instrucción IR

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

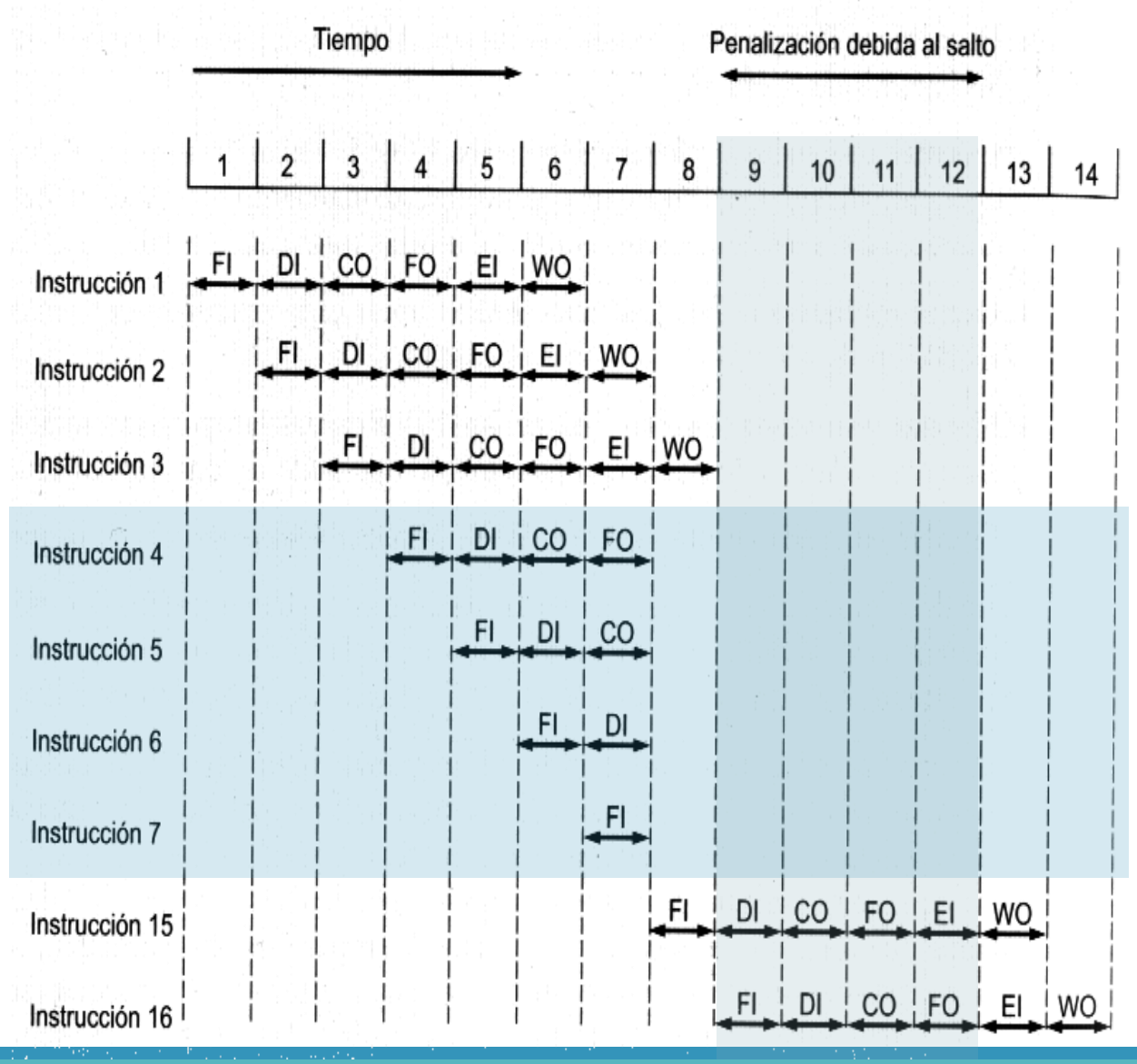
Estrategia de Segmentación

Captar la instrucción (FI)	Leer la supuesta siguiente instrucción.
Decodificar instrucción (DI)	Determinar el código de operación y los campos del operando.
Calcular operandos (CO)	Calcular la dirección efectiva de cada operando fuente. Esto puede involucrar direccionamiento mediante un desplazamiento indirecto, a través del registro indirecto, u otras formas de calcular la dirección.
Captar operandos (FO)	Captar cada operando de memoria. Los operandos en registros no tienen que ser captados.
Ejecutar instrucción (EI)	Realizar la operación indicada y almacenar el resultado, si lo hay, en la posición de operando destino especificada.
Escribir operando (WO)	Almacenar el resultado en memoria.

Carga de un PIPELINE



Efecto de un salto condicional



La instrucción Nº 3 es una bifurcación condicional a la instrucción 15. Hasta que no se ejecuta la instrucción no hay forma de saber qué instrucción vendrá a continuación. El cauce, en este ejemplo, simplemente carga la siguiente instrucción secuencialmente (instrucción 4) y continúa. Éste no se determina hasta el final de la unidad de tiempo 7. En ese momento, el cause debe limpiarse de instrucciones inútiles. Durante la unidad de tiempo 8, la instrucción 15 entra en el cauce.

Ninguna instrucción termina durante las unidades de tiempo de la 9 a la 12; ésta es la penalización por no haber podido prever el salto.

Programador de sistemas y programador de aplicaciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Programador de aplicaciones

- Es el encargado de **codificar programas** para los usuarios finales.
- En general suelen **desarrollar aplicaciones compuestas** por uno o más programas en un lenguaje de alto nivel comercial.
- Con una **visión limitada del procesador**
- Para este 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.

Programador de sistemas y programador de aplicaciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Programador de sistemas

- Se encarga de **desarrollar programas y utilidades del sistema operativo**.
- Su misión es **construir un sistema operativo óptimo, 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.
  - Confección de **herramientas para sistemas operativos, como: depuradores, compiladores**, etc.
- Es indispensable que **conozca profundamente la arquitectura detallada de la CPU** para **optimizar** los recursos.
- Debe conocer **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.

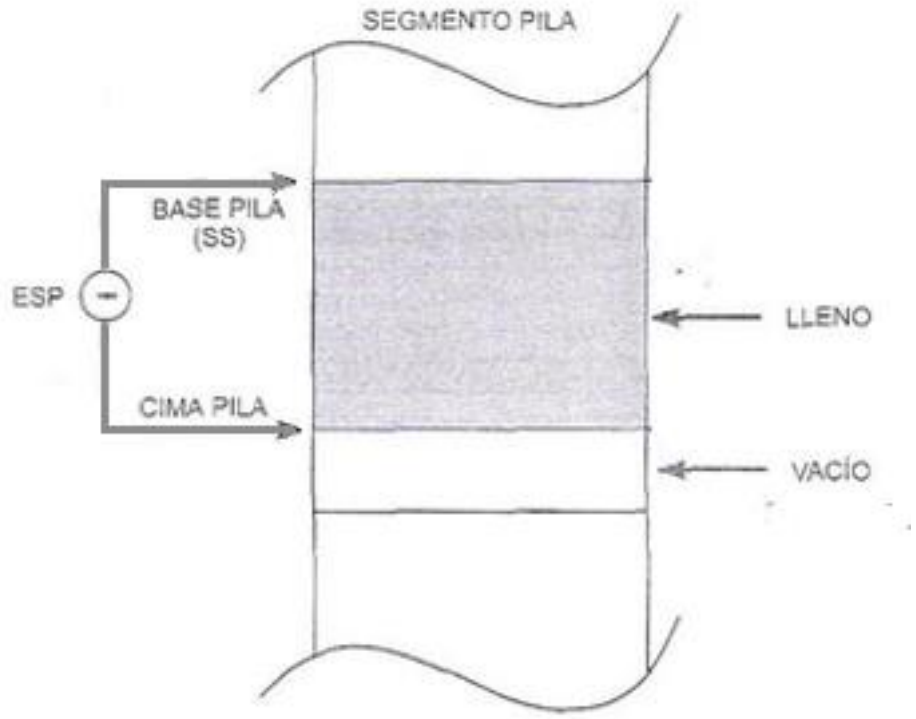
Registros internos para el programador de aplicaciones

- **Puntero de pila (ESP, SP) y Puntero de base (EBP, BP)**

Sirven para controlar el direccionamiento de la pila.

Esas operaciones en la pila son soportadas por tres registros diferentes que se exponen a continuación:

1. Registro de segmento de pila (SS). Especifica las características del segmento de pila que reside en memoria.
2. Registro puntero de pila (ESP). Contiene el desplazamiento de la cima de la pila en el segmento de la pila actual. Lo usan las operaciones PUSH y POP, las llamadas a subrutinas. el retorno, las excepciones y las interrupciones. Cuando se introduce un elemento a 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 hace la operación contraria, es decir, se incrementa el ESP.
3. Registro puntero base de la 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.



Registros internos para el programador de aplicaciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

- **Índice fuente (ESI. SI) e índice destino (EDI. DI)**

Son dos punteros de direcciones necesarios para trabajar con cadenas de caracteres.

Las instrucciones que realizan operaciones entre los elementos de las cadenas o «strings» se aplican a una cadena fuente y a una cadena destino, cada una de ellas está díreccionada por el registro ESI y el EDI, respectivamente (Figura 7.4).

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.

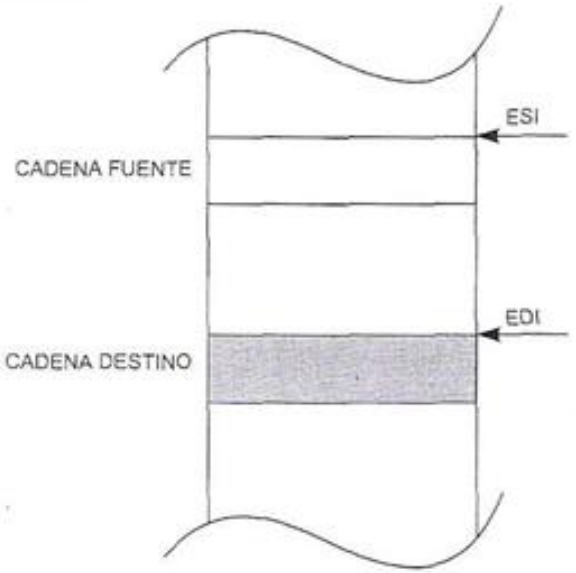


Figura 7.4. *Los registros ESI y EDI apuntan al elemento en curso de la cadena fuente y la cadena destino, respectivamente.*

Registros internos para el programador de aplicaciones: Registro puntero de instrucciones

EIP: Registro puntero de instrucciones

El EIP es el puntero de las instrucciones o contador de programa y no está disponible para el programados 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.

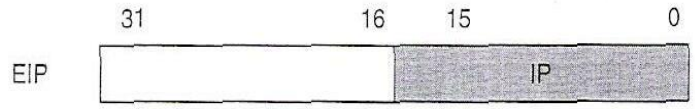


Figura 7.5. *El Puntero de Instrucciones tiene un tamaño de 32 bits (EIP) en modo Protegido y de 16 bits (IP), cuando soporta el direccionamiento «reducido» del 8086.*

Registros internos para el programador de aplicaciones: Registro EFLAGS

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 ALU (acarreo, paridad, acarreo auxiliar, cero, signo y overflow), actuando los restantes como señalizadores del sistema, ligados al mecanismo de protección y a otros recursos que dispone el sistema.

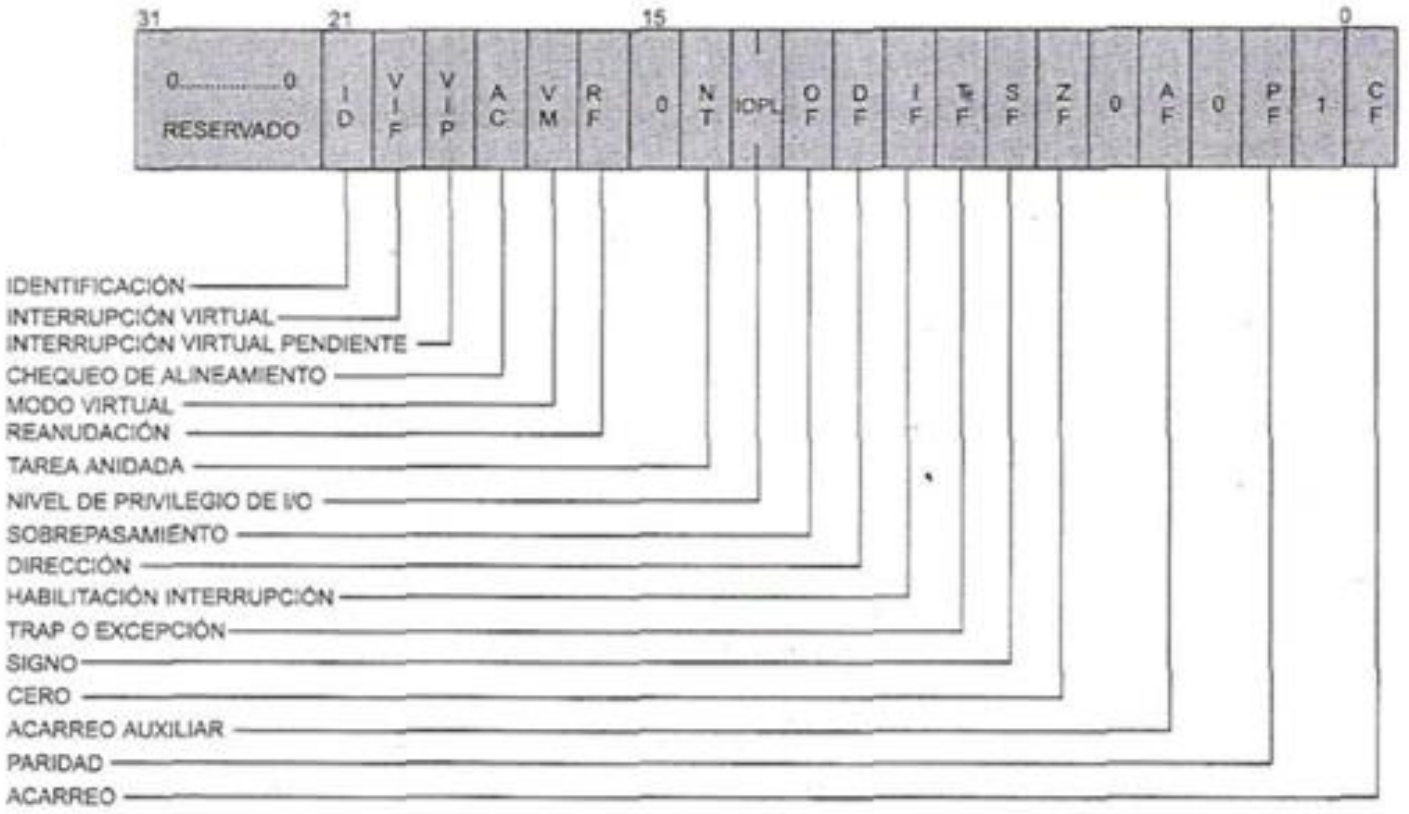


Figura 7.7. *Registro EFLAGS.*

Registros internos para el programador de aplicaciones: Registro EFLAGS

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Registro de estado o de señalizadores (EFLAGS)

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

• CF: 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 una operación de suma aritmética.
  - 0: Cuando el señalizador CF toma este valor significa que no ha habido acarreo en el bit de más peso de una suma, o bien, que ha existido «llevada» en el bit de más peso de una resta aritmética.

• PF: Bit de paridad impar.
  - 1: Toma este valor para generar la paridad impar con los bits que conforman el resultado de una operación.
  - 0: Valor para generar paridad impar.

• AF: 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 caso de no haber acarreo en el bit 3 del resultado.

Registros internos para el programador de aplicaciones: Registros EFLAGS

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Registro de estado o de señalizadores (EFLAGS)

• ZF: Señalizador de cero.
  - 1: Cuando todos los bits del resultado son ceros.
  - 0: En caso de que el resultado no sea 0.
• SF: Señalizador de signo.
  - 1: Si el bit de más peso del resultado de la operación es 1.
  - 0: Si el bit de más peso del resultado de la operación es 0.
• TF: Excepción al terminar la ejecución de la instrucción.
  - 1: Provoca una excepción al completarse la ejecución de la instrucción en curso. Posibilita la depuración de los programas al permitir la ejecución paso a paso, es decir, instrucción por instrucción.
  - 0: No hay excepción de depuración al terminar cada instrucción.
• IF: Flag de habilitación de interrupciones mascarables.
  - 1: Permite el reconocimiento de las peticiones de interrupción mascarables provocadas por la activación de la patita ÍNTR del Pentium.
  - 0: Prohíbe el reconocimiento de la interrupción externa e ignora las peticiones de interrupción mascarables.

Registros internos para el programador de aplicaciones: Registros EFLAGS

Registro de estado o de señalizadores (EFLAGS)

• DF: Flag de dirección de exploración de las cadenas de caracteres o strings.
- 1: Postdecremento automático de los registros ESI y EDI, qne direccionan la cadena.
- 0: Postincremento automático de ESI. EDI.

• OF: Flag de desbordamiento (u o vertió wt.
- 1: En operaciones con números enteros con signo se activa y vale 1 si el resultado es muy grande (es positivo) o muy pequeño (si es negativo). Indica resultados erróneos.
- 0: Si no existe desbordamiento.

• IOPL: Nivel de privilegio de las entradas y salidas. Es un campo de 2 bits que se emplean en modo Protegido y determinan el nivel de privilegio que debe igualar o superar el segmento de código en el que se desean ejecutar instrucciones de *E/S* que manejan datos en el espacio de E S del procesador que está reservado a guardar la información de los periféricos del sistema.

Los valores que se pueden tomar son:
- 11: nivel 3 (pueden acceder todos).
- 10: nivel 2.
- 01: nivel 1.
- 00: nivel O (únicamente puede acceder el SO).

• NT: Tarea anidada. Este señalizador actúa automáticamente al producirse una conmutación de tareas.
- 1: La tarea en curso está anidada con la anterior. Hay que retornar obligatoriamente a la tarea previa.
- 0: La conmutación de tareas es libre.

Registros internos para el programador de aplicaciones: Registros EFLAGS

Registro de estado o de señalizadores (EFLAGS)

• RF: 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.
  - 1: Se ignoran los plintos de depuración o parada,
  - 0: No se ignoran los puntos de parada.

• VM: 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 a modo Virtual.
  - 0: No hay paso al modo Virtual 86.

• AC: 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 daros en la memoria, obligará a realizar más de un acceso para recoger toda la información.

Registros internos para el programador de aplicaciones: Registros EFLAGS

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Registro de estado o de señalizadores (EFLAGS)

• VIF: Se trata de un bit que realiza una misión similar al IF, pero en el modo Virtual 86. Permite o prohíbe la atención a las peticiones de interrupción mascarables. Se utiliza con el flag VIP. El procesador sólo reconoce VIF cuando el flag VME o el PVI del registro de control CR4 están activados y el IOPL es menor que 3.

• VIP: 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 sólo reconoce el flag VIP cuando el flag VME o el PVI en el registro de control CR4 están activados y el IOPL es menor que 3.

- 1: Interrupción mascarable pendiente.
- 0: No hay interrupción pendiente.

• ID: Bit de identificación. ID informa si el Pentium soporta la instrucción CPU1D que sirve para su identificación. Mediante la ejecución de CPUID se muestran las características más importantes del procesador.

- 1: El Pentium soporta la instrucción CPUID.
- 0: En caso contrario.

Registros internos para el programador de aplicaciones: Registros de segmento

Registros de segmento

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

Los segmentos son trozos de la memoria de tamaño variable que contienen el mismo tipo de información. Así, hay tres tipos de segmentos en los programas de aplicación.

- **De pila.**
- **De código.**
- **De datos.**

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.

El Pentium controla en cada instante seis segmentos a los que referencia a través de los Registros de Segmento (RS). Esto no significa que sólo pueda manejar seis segmentos, sino que. si 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 Pentium dispone de seis registros de 16 bits, que se muestran en la Figura 7.8

Figura 7.8. *Registros de segmento en el Pentium.*

Modo real vs Modo Protegido

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Modo real

- 20 bits de espacio de direcciones segmentado (2^{20} = 1 MB de memoria máximo)
- Acceso directo del software a las rutinas del BIOS y el hardware periférico
- No tiene conceptos de protección de memoria o multitarea a nivel de hardware.
- Palabra de 16 bits.
- La memoria es segmentada. (Tamaño máximo de segmento 2^{16} = 64 KB).
- Rango de direcciones de un segmento de 0000 a FFFF en Hexadecimal.
- Cada dirección identifica 8 bits de almacenamiento.

Modo Protegido

En este modo existe la Memoria Virtual.

- Trabaja con Memoria Principal de hasta 4GB y Memoria Virtual de hasta 64TB (a partir del 80386).
- La memoria es segmentada, con o sin paginación.
- Protección de programas.
- Capacidad de Multitareas.
- Posible Conexión a Memoria Cache.

Segmentación en modo real

Cuando el Pentium funciona en **modo real (monotarea),** compatible con el 8086, y sin tipo alguno de protección ni posibilidad de manejo de memoria virtual, un segmento queda definido básicamente por los siguientes elementos:

- Base o dirección de comienzo de 20 bits.

- Desplazamiento o tamaño de 16 bits. El tamaño máximo que se admite en este modo para mantener la compatibilidad con el 8086, es de 64 KBytes y la capacidad máxima de la memoria principal sólo es de 1 MByte.

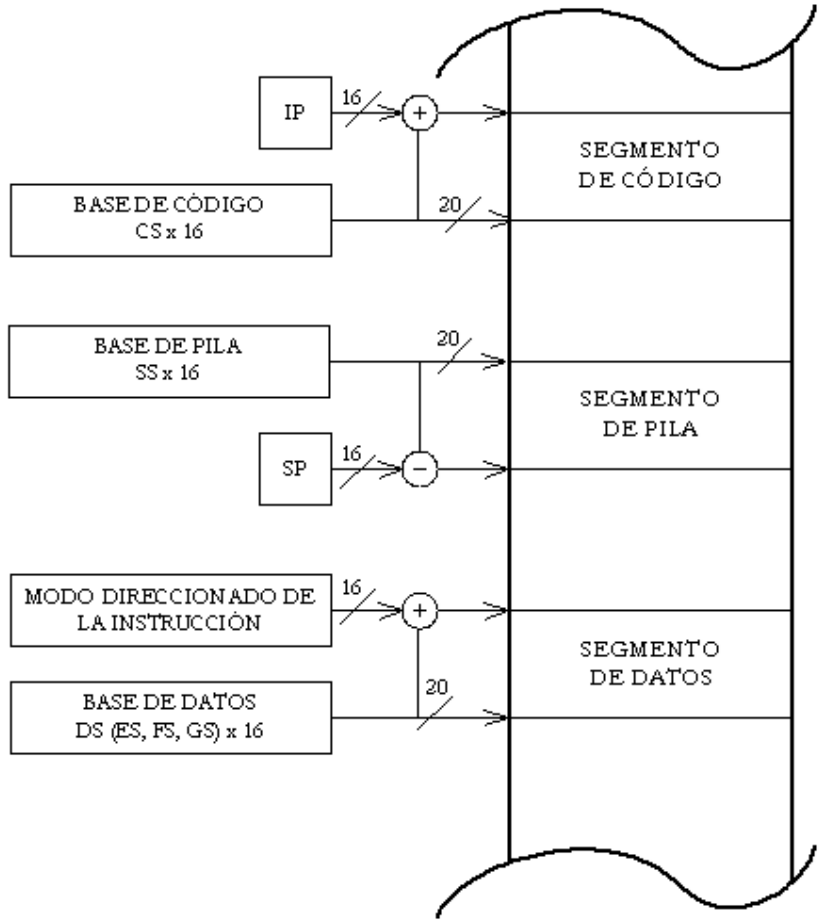


Figura 7.13 – Segmentación en modo real. En este modo se accede a elementos de memoria multiplicando por 16 el valor del registro de segmento y añadiendo un desplazamiento de 16 bits al resultado.

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Segmentación en modo real

En modo real todo segmento de código, datos o pila está especificado por una dirección lógica, compuesta por dos campos de 16 bits cada uno.

- **Selector:** Referencia la base del segmento, la cual se deduce a partir del valor contenido en el registro de segmento apropiado. Como el 8086 sólo maneja una memoria de 1 MByte (2^{20}) de capacidad máxima, para obtener la base del segmento se añaden cuatro ceros a los 16 bits del registro de segmento o selector, es decir, se multiplica dicho valor binario por 10 en Hexadecimal (16 en decimal).

- **Desplazamiento:** El tamaño máximo del segmento en el 8086 es de 64 KBytes , por lo tanto bastan 16 bits para expresar el desplazamiento que hay que añadir a la base. En el caso del segmento de código, el desplazamiento lo almacena IP, que está formado por los 16 bits de menos peso de EIP. El desplazamiento correspondiente al segmento de pila está guardado en SP, y el del segmento de datos lo expresa el modo de direccionamiento de los operandos o del resultado en la instrucción en curso, por ejemplo (MOV AX, ES : 555 hex).

Segmentación en modo real

Por tanto, una dirección quedaría:

Dirección efectiva/física = RS x 10_h + Desplazamiento.
Esto queda especificado en la siguiente tabla. Dependiendo de si es un segmento de código, de pila o de datos, se conseguirá la base utilizando CS, SS, DS... tal y como se muestra a continuación.

Observaciones: Multiplicar por 10 en hexadecimal es lo mismo que 16 en decimal

	BASE	DESPLAZAMIENTO
CÓDIGO	CS X 16	IP
PILA	SS X 16	SP
DATOS	DS X 16 ES X 16 FS X 16 GS X 16	DESPLAZAMIENTO

Tabla 7.1. – Tabla para la determinación de la base y el desplazamiento de los segmentos en modo real

Segmentación en modo protegido

Descriptor de segmento:
Se denomina **descriptor de segmento** al conjunto de los 2 parámetros base (32 bits), límite (20 bits) y atributos (12 bits).

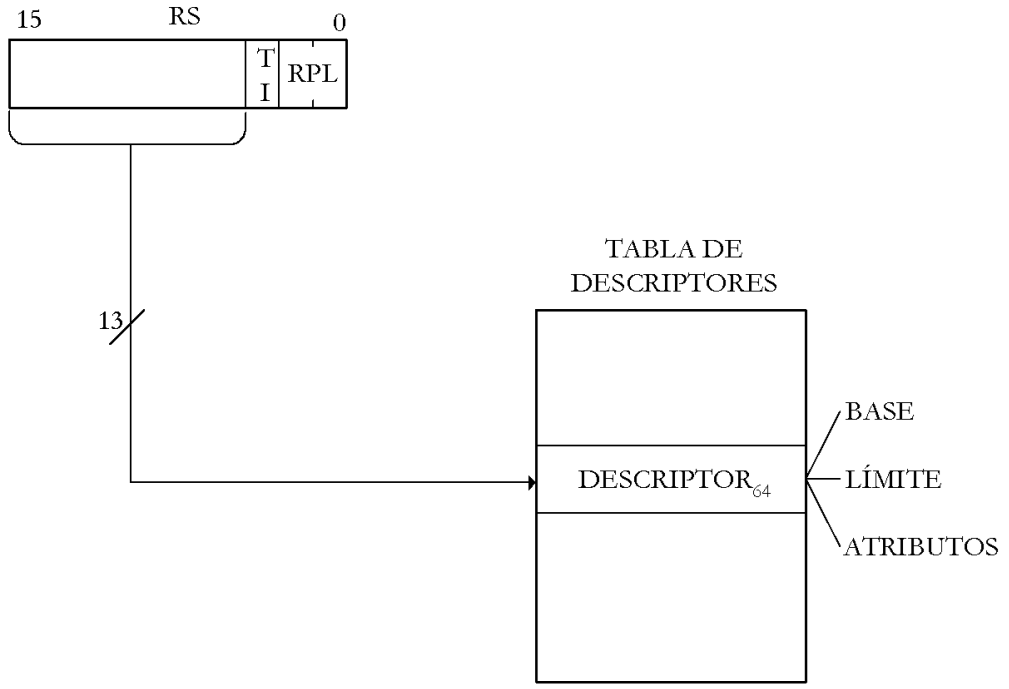


Figura 7.14 – Se accede a la tabla de descriptores consultado el selector y comprobando el índice de tabla.

Registros internos para el programador de aplicaciones: Registros de segmento

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

- **Selector:** Son los 14 bits de más peso del registro de segmento que referencia al segmento al que 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 menos peso del registro de segmento conforman el campo RFL que indica el nivel de privilegio del segmento peticionario (Figura 7.9).
- **Desplazamiento:** Es un valor que se añade a la base del segmento para localizar la dirección que hay que acceder en él. El tamaño del desplazamiento determina su longitud máxima que, en el caso del Pentium, es de 4 GB en el modo Protegido y 64 KB en modo Real.

Figura 7.9. *Contenido de los registros de segmento.*

Segmentación en modo protegido

Cuando el Pentium trabaja en modo protegido (multitarea), un segmento queda caracterizado por tres parámetros fundamentales que son comprobados automáticamente por el sistema de protección cada vez que se utiliza. Dichos parámetros son:

Base, es la dirección lineal donde comienza el segmento. Está formada por 32 bits, que es la longitud de la dirección de la memoria física que puede alcanzar un tamaño máximo de 2^{32} = 4 GBytes.

Límite, consta de 20 bits que determinan con exactitud el tamaño del segmento usado por el programador y en el que residen informaciones válidas. Si está expresado en bytes, el límite máximo sería de 2^{20} = 1 MByte y si está expresado en páginas de 4 KByte, un segmento puede ser tan grande como la memoria principal, es decir 4 GBytes.

Atributos o derechos de acceso, se trata de un campo de 12 bits, que proporciona las características relevantes del segmento como:

- Tipo de segmento, admitiendo las variantes de legible, escribible, ejecutable o una combinación de estos.
- Nivel de privilegio, que oscila entre 0 y 3. Es el grado de seguridad que tiene el contenido del segmento en el sistema.
- Indicadores sobre aspectos relacionados con la gestión de la memoria virtual, como el que indica si el segmento se halla cargado o no en la memoria física.

Segmentación en modo protegido

Descriptor de segmento:

Los descriptores de segmento se guardan en una “tabla de descriptores” que residen en la MP y se crean y manejan desde el SO.

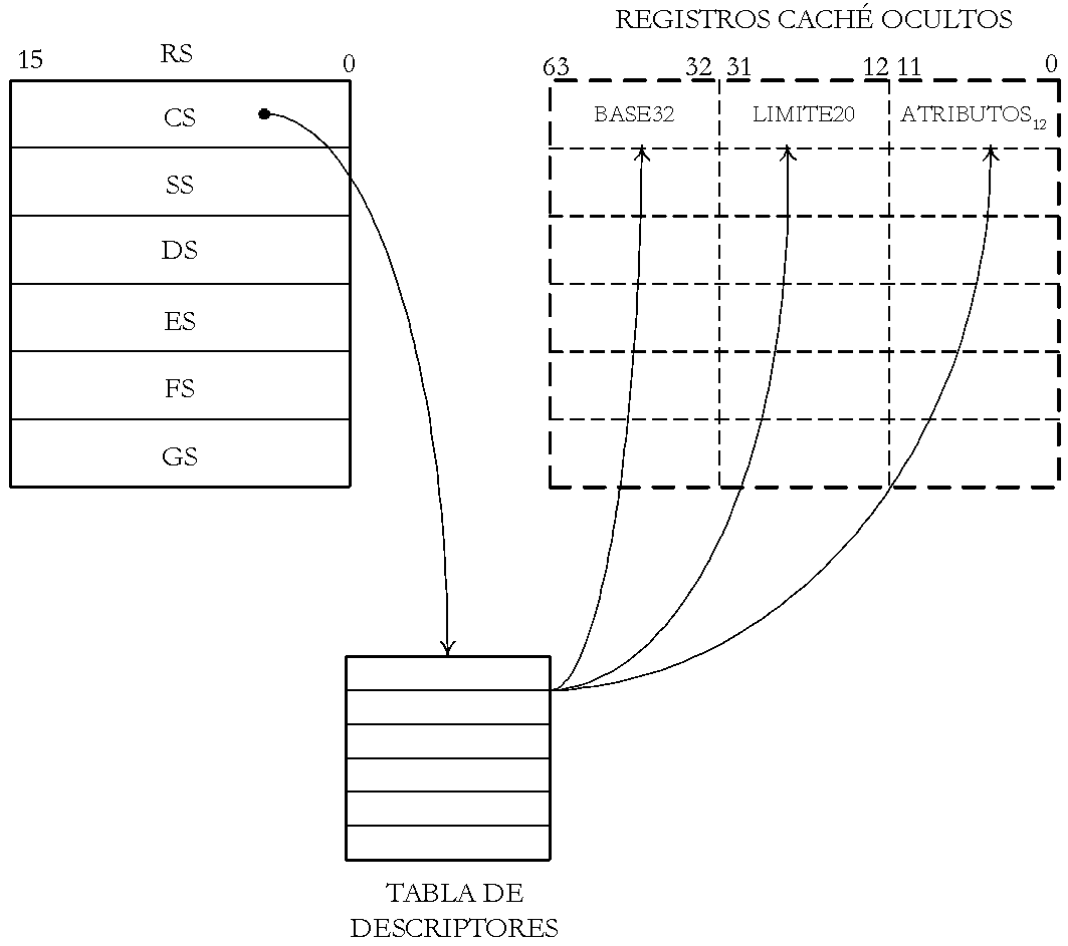


Figura 7.15 – En modo protegido, los registros de segmento actuan como selectores “visibles”, con los que se accede a tablas en las que se hallan los parámetros que definen al segmento, es decir, la base, el límite y los atributos. Estos parámetros los carga automáticamente la CPU en los registros caché asociados, que son invisibles, es decir, inaccesibles al programador de sistemas y al de aplicaciones.

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Concepto de Pila o Stack e Interrupciones

Repaso Ciclo de instrucción

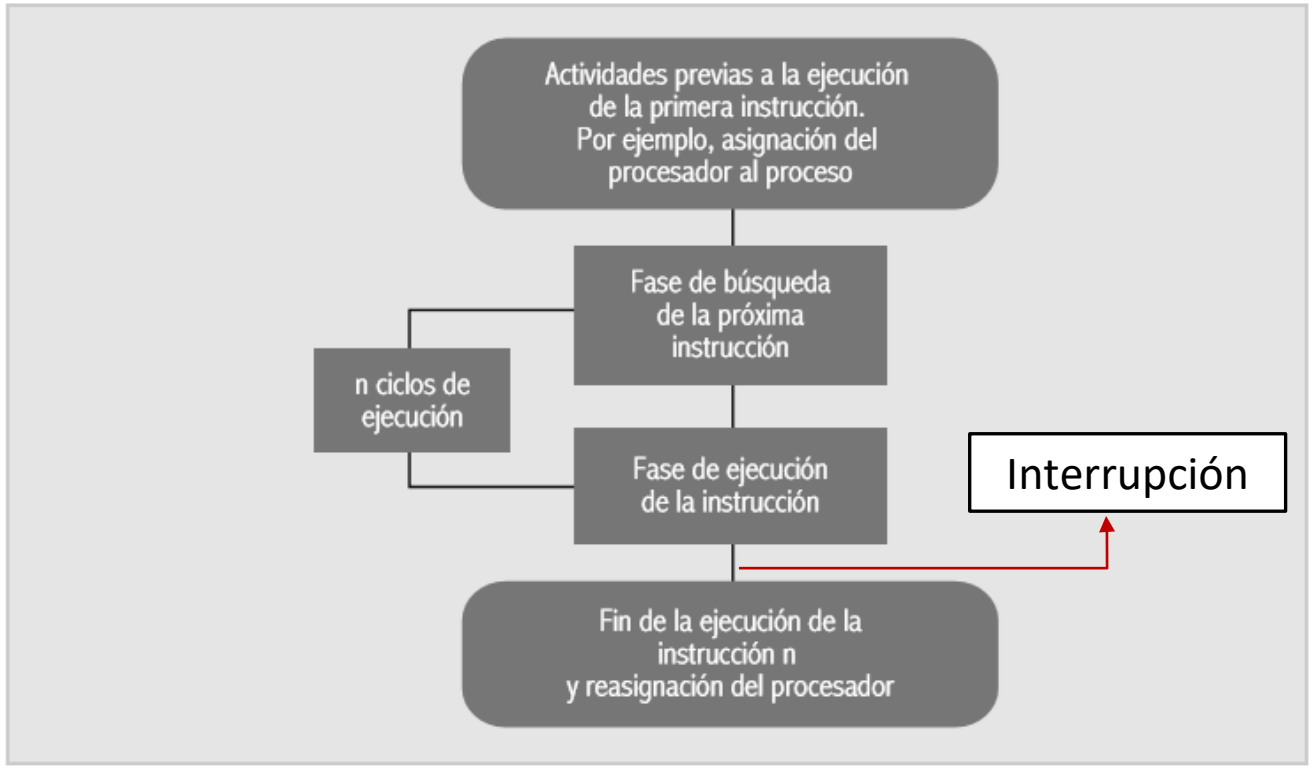


Fig. 8.2. Ciclo de instrucción para las n instrucciones de un programa.

Fase de búsqueda:

- Cálculo de la dirección física de la instrucción.
- Dar orden de lectura RD
- Se carga el registro IR

Fase de ejecución:

- Interpretar el código de la instrucción (Decodificar)
- Incrementar el IP
- Búsqueda del dato **(RD)** o Guardar el dato **(WR)** (si afecta)
- Generar orden al módulo para que opere el dato.

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Concepto de Pila

Segmento de Pila:

Zona de memoria de tamaño variable destinado a almacenar o agrupar mediante el criterio de datos referenciados a la pila para determinadas instrucciones y procesos de ejecución.
Principalmente en el manejo de Subrutinas.

Estructura de la Pila:

La misma es del tipo LIFO (Last In First Out – Ultimo en entrar primero en salir).

Registros asociadoa para acceso a la Pila:

SS (Registro Segmento de Stack o Pila)
SP (Puntero de Stack o Pila)
BP (Puntero Base)

Concepto de Pila

Funciones básicas:

- Almacenar la dirección de retorno (IP) y, eventualmente, el CS (segmento de código) cuando se llama a un procedimiento, también conocido como subrutina.
- Almacenar el estado del procesador cuando se produce una interrupción.

  Los registros obligados que apila son el CS, el IP y el estado de los *flags* (*status register*) (conocido también como el PSW o Palabra de Estado del Programa y, eventualmente, los registros de cálculo, etcétera.
- Para pasar parámetros entre dos procedimientos.

Concepto de pila

El acceso a la pila se realiza a través de los registros punteros SP y BP, donde SP contiene la dirección de la cima de la pila. Es decir, SS:SP

La escritura o lectura de datos de la pila es un procedimiento software y se realizan incrementado o decrementando el registro SP.

Si la pila trabaja a nivel palabra (2 bytes) por ejemplo en modo real, cada vez que se extraiga un dato se autoincrementara el SP = SP + 2 y cada vez que se cargue se autodecrementara (SP = SP – 2) en 2 unidades.

Concepto de funcionamiento:

Si se va a almacenar o guardar un dato en la pila, previamente el SP se actualizará decrementando su valor y luego se almacena el valor en la pila.

Si se realiza lectura o sale un dato de la pila, luego de dicha operación el puntero SP se actualizará incrementando su valor

Concepto de pila

La encargada de acceder a la pila es la CPU, ejecutando instrucciones propias para esta estructura, como **PUSH y POP**.

- PUSH, decrementa el SP y luego almacena una palabra en la pila.
- POP, extrae una palabra de la pila y desafecta las posiciones de memoria incrementando el SP.

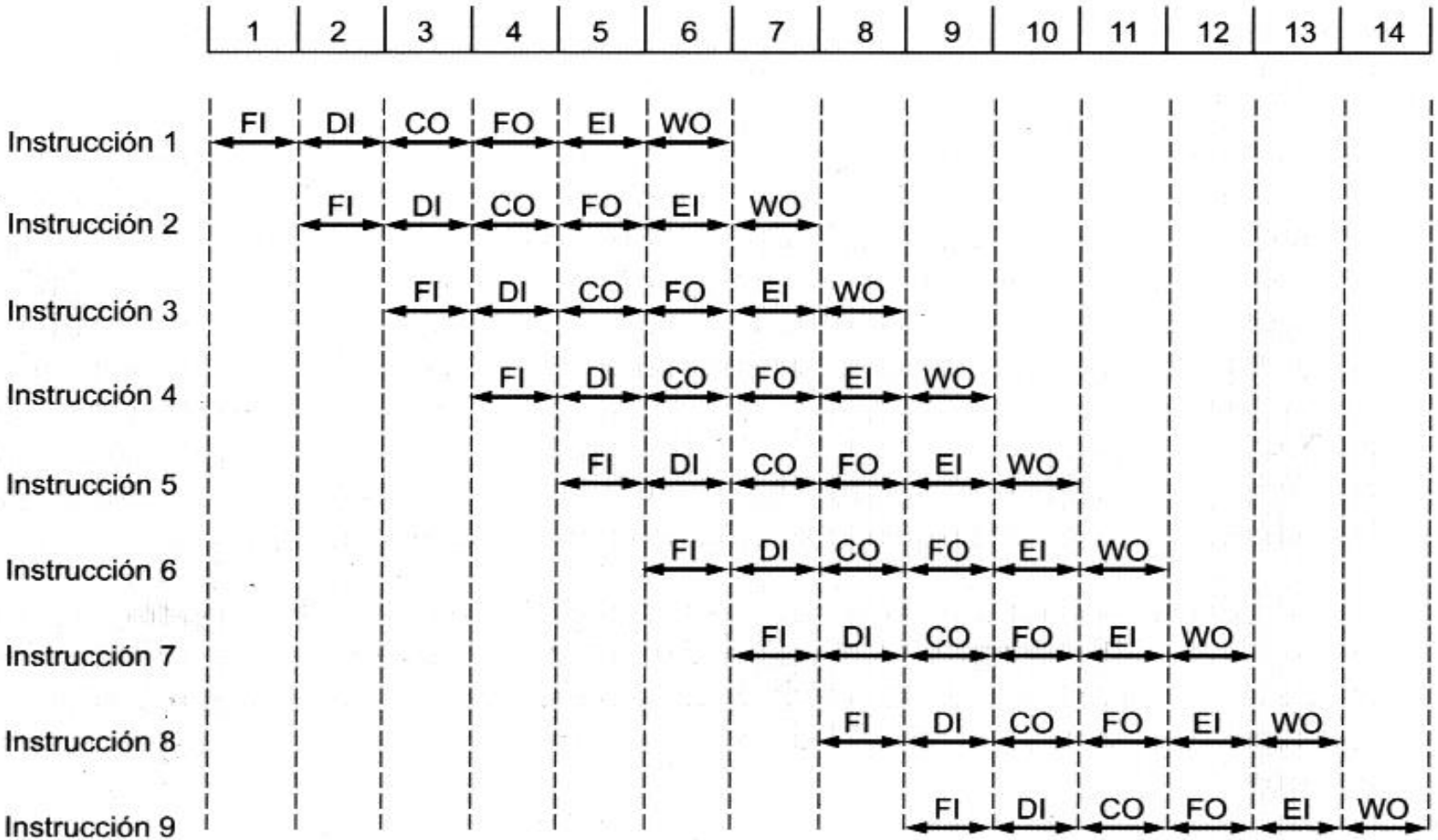
Las instrucciones que a causa de su ejecución afectan a la pila son **CALL, INT, RET o IRET**.

- **CALL y Ret** son instrucciones que sirven para invocar y dar el retorno a un procedimiento o subrutina.
- **INT e IRET** cumplen la misma función cuando se invoca una subrrutina de interrupción.

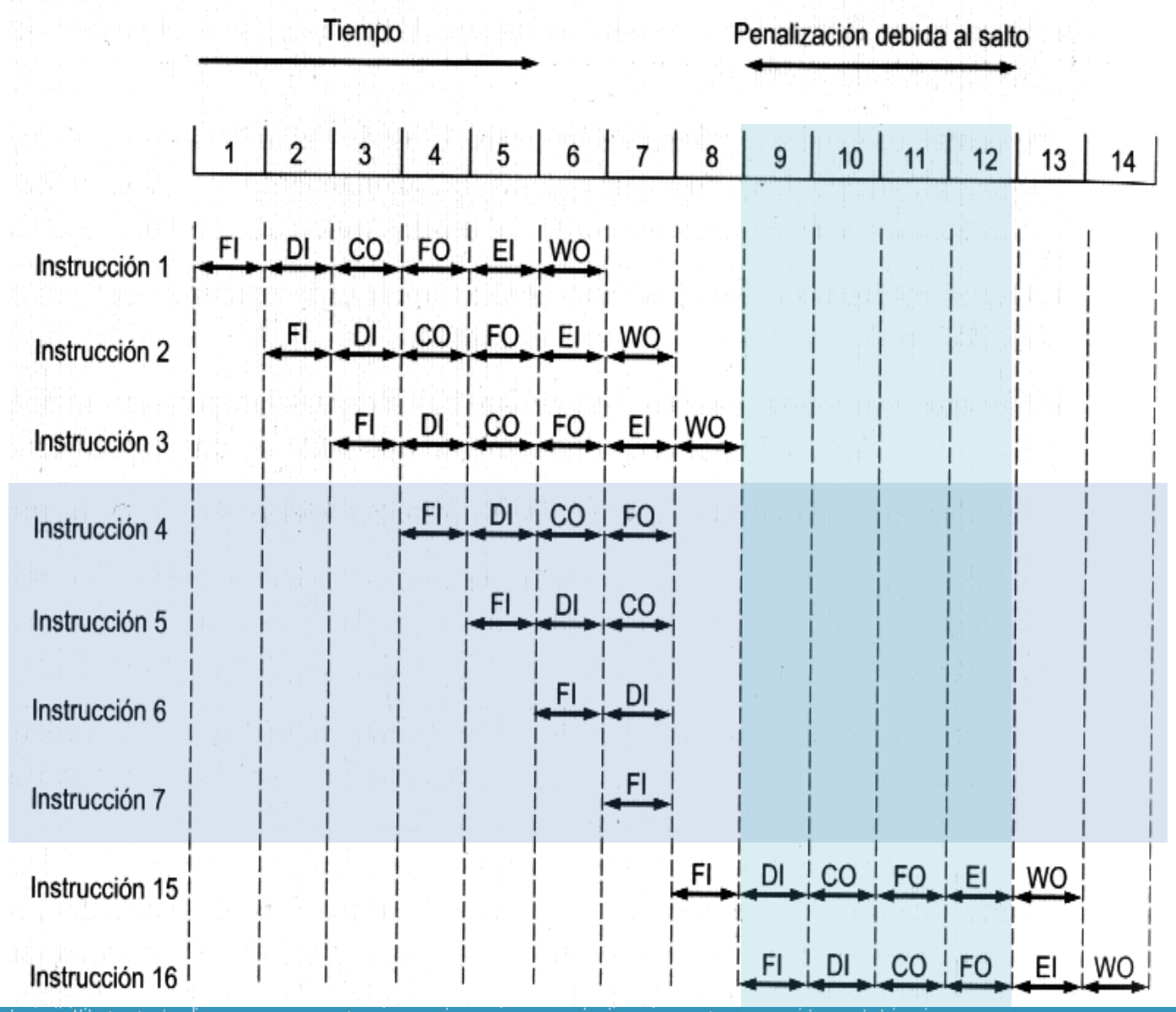
UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Interrupciones

Carga de un PIPELINE



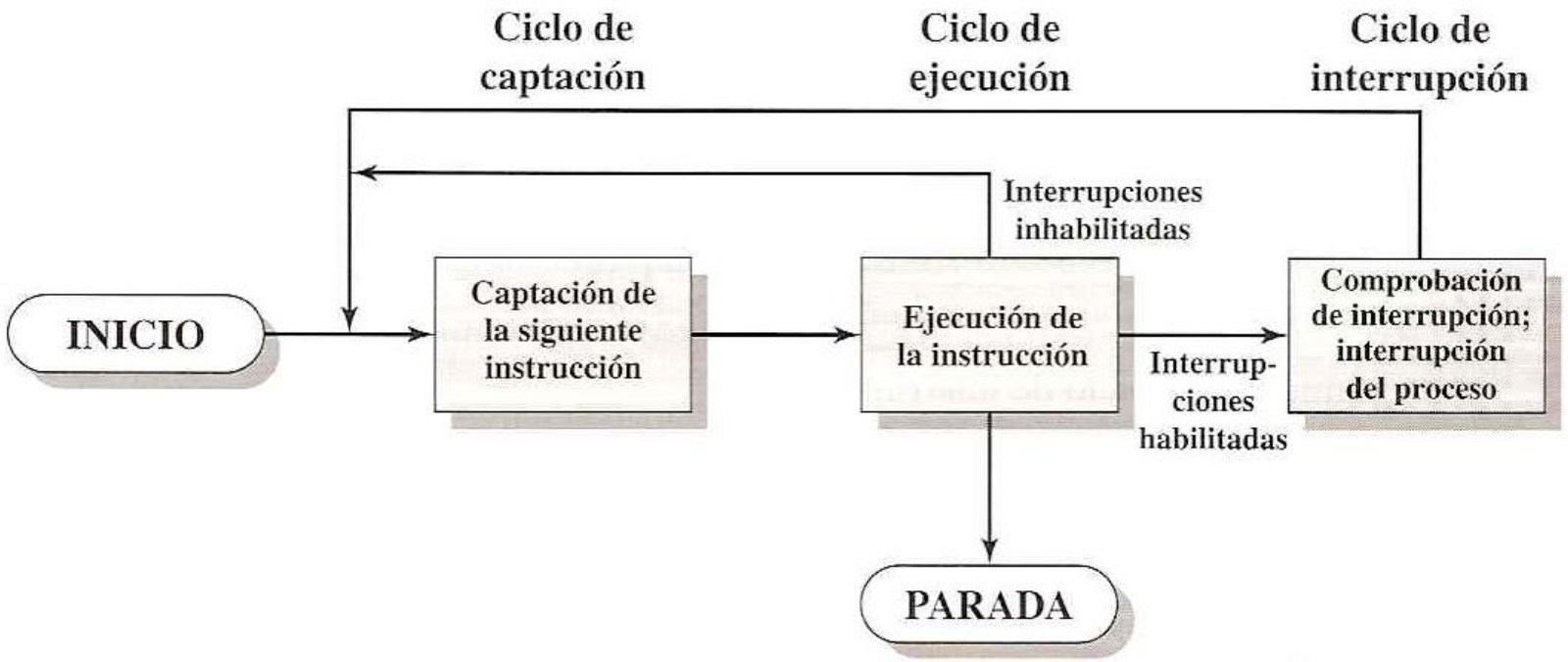
Efecto de un salto condicional



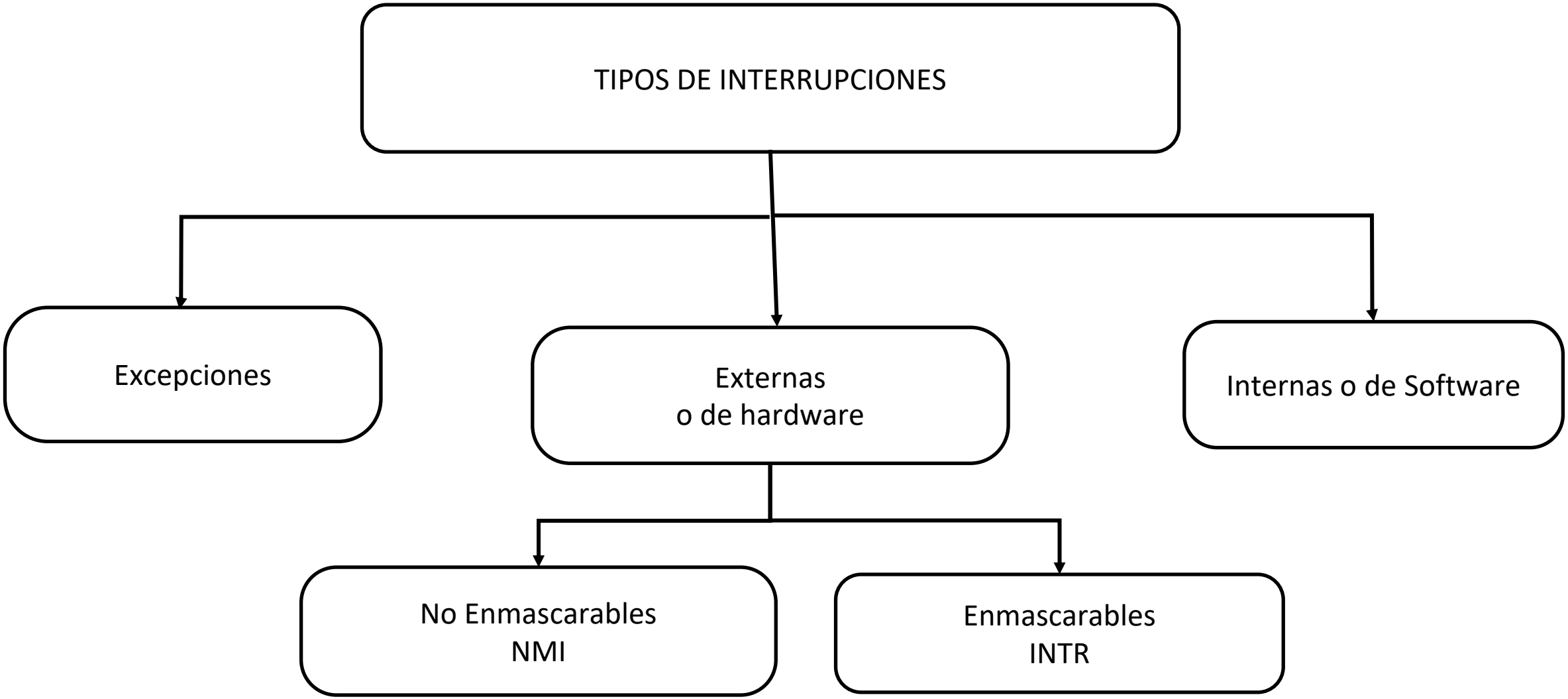
La instrucción Nº 3 es una bifurcación condicional a la instrucción 15. Hasta que no se ejecuta la instrucción no hay forma de saber qué instrucción vendrá a continuación. El cauce, en este ejemplo, simplemente carga la siguiente instrucción secuencialmente (instrucción 4) y continúa. Éste no se determina hasta el final de la unidad de tiempo 7. En ese momento, el cause debe limpiarse de instrucciones inútiles. Durante la unidad de tiempo 8, la instrucción 15 entra en el cauce.

Ninguna instrucción termina durante las unidades de tiempo de la 9 a la 12; ésta es la penalización por no haber podido prever el salto.

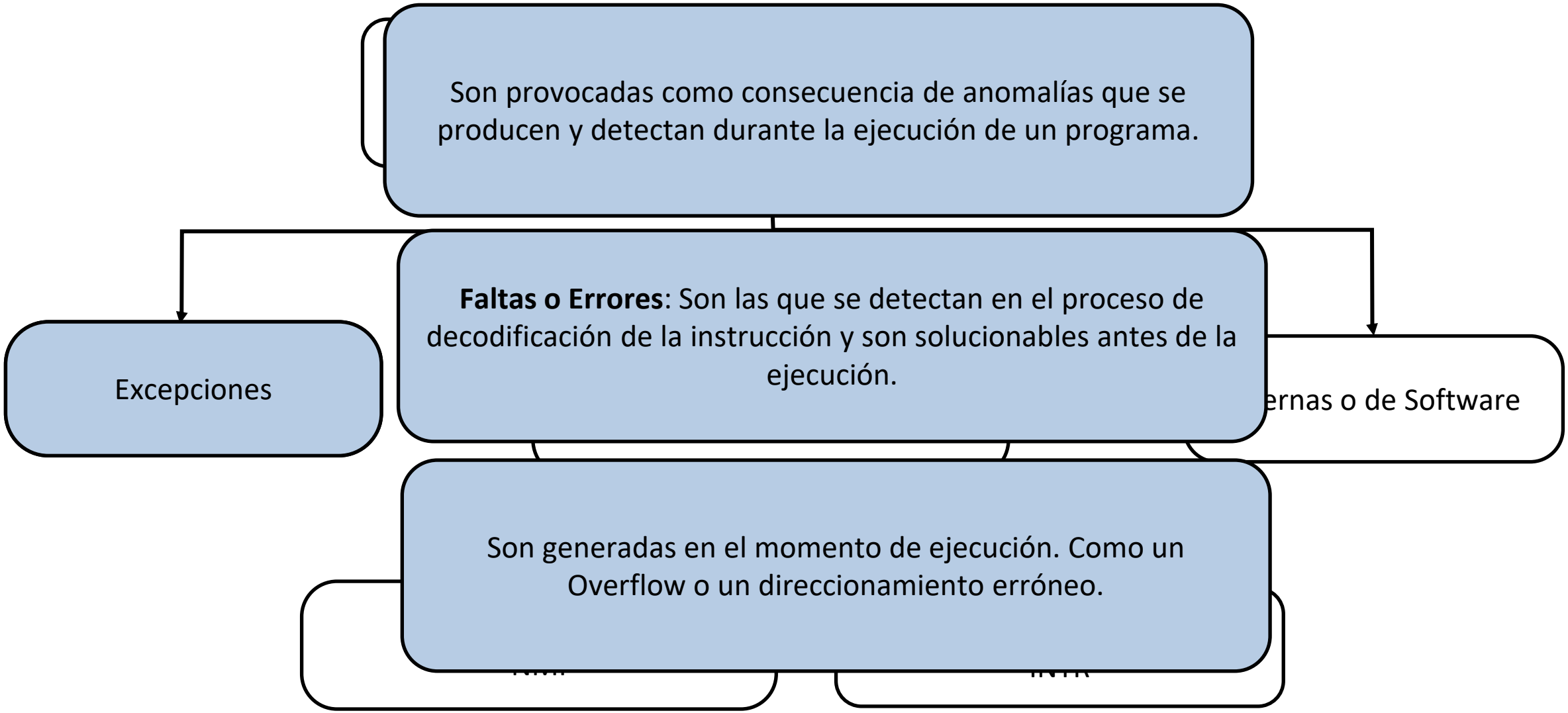
Ciclo de ejecución con interrupción



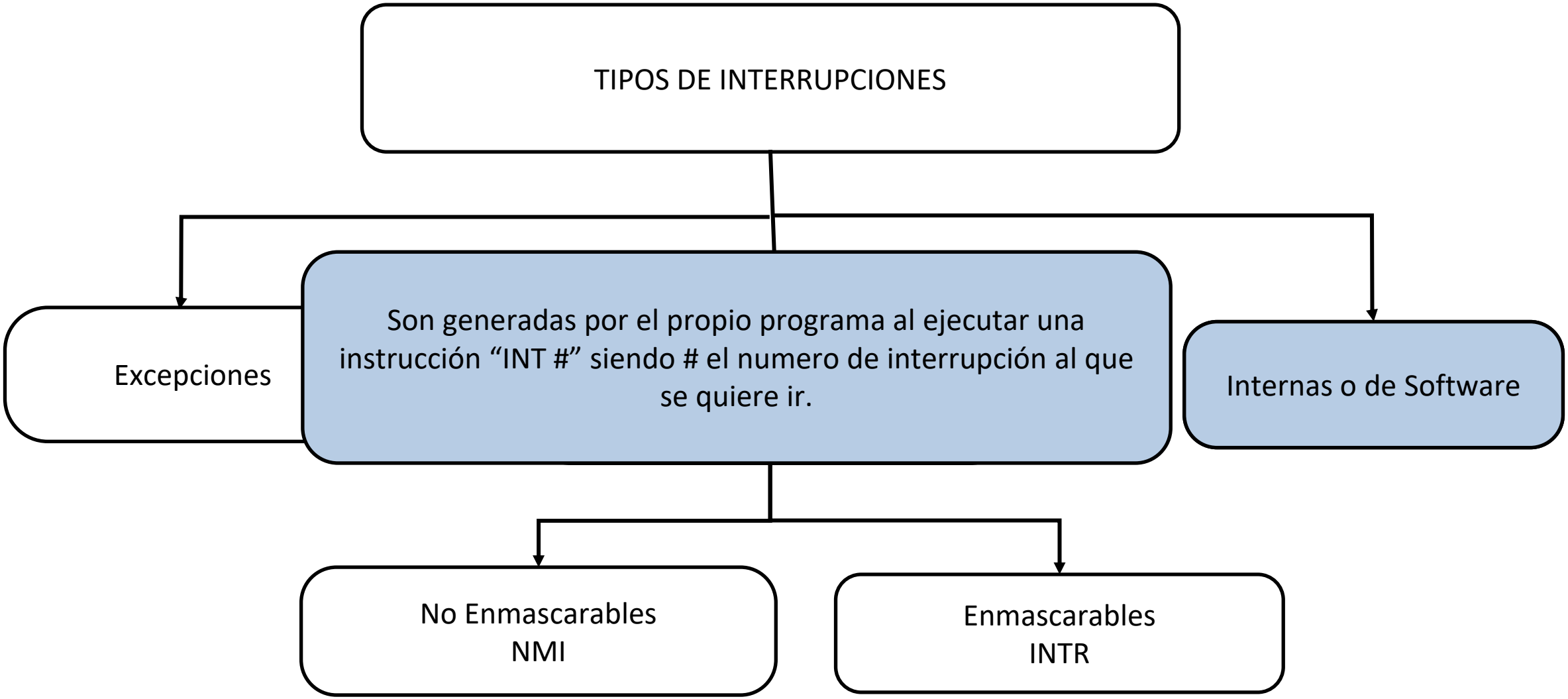
Interrupciones



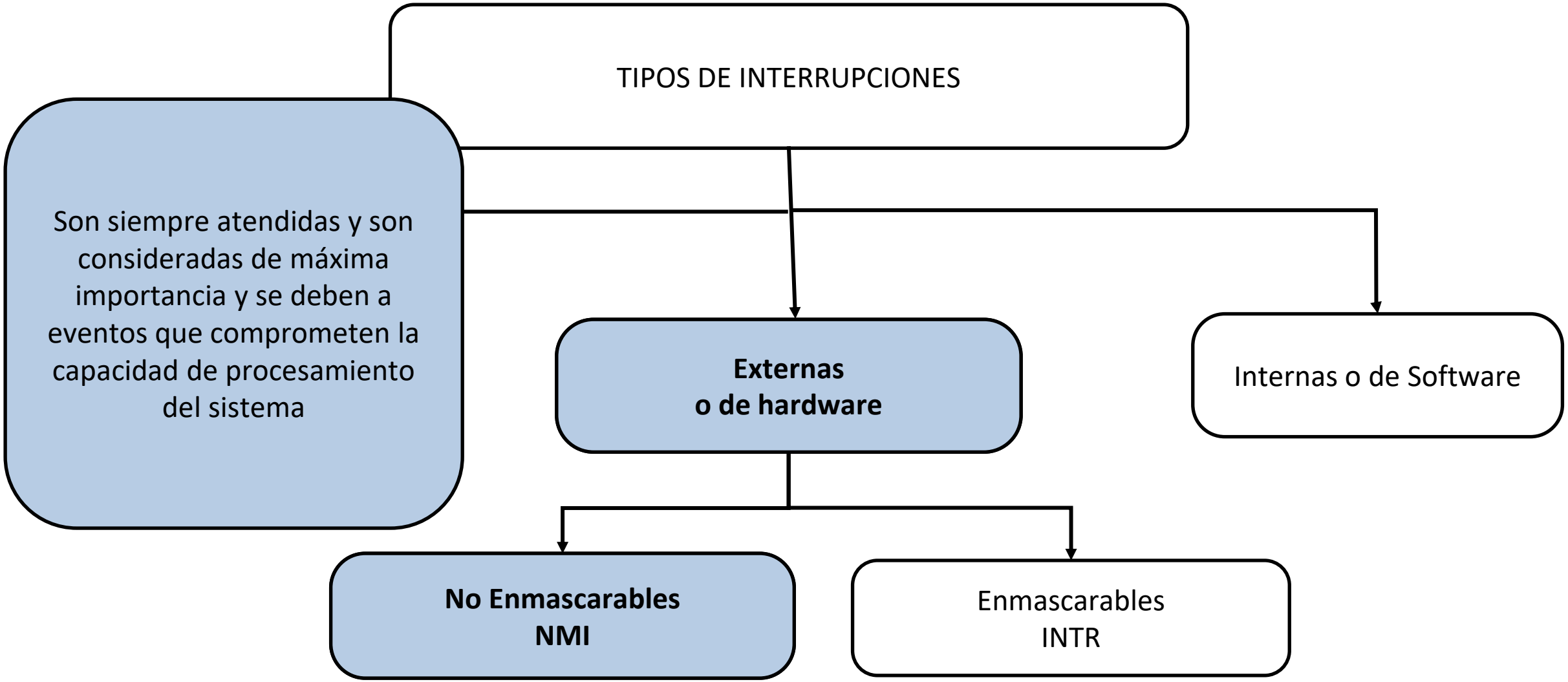
Interrupciones



Interrupciones



Interrupciones



Interrupciones

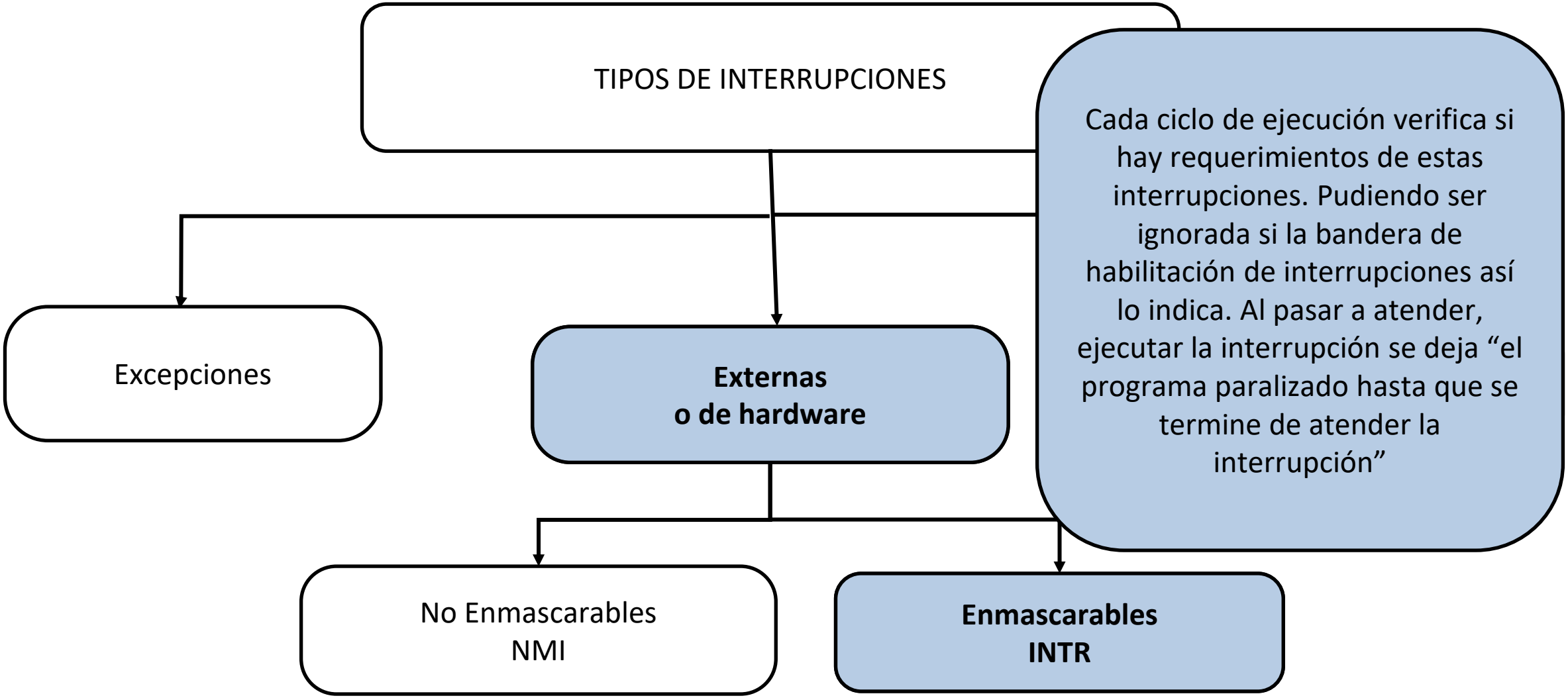
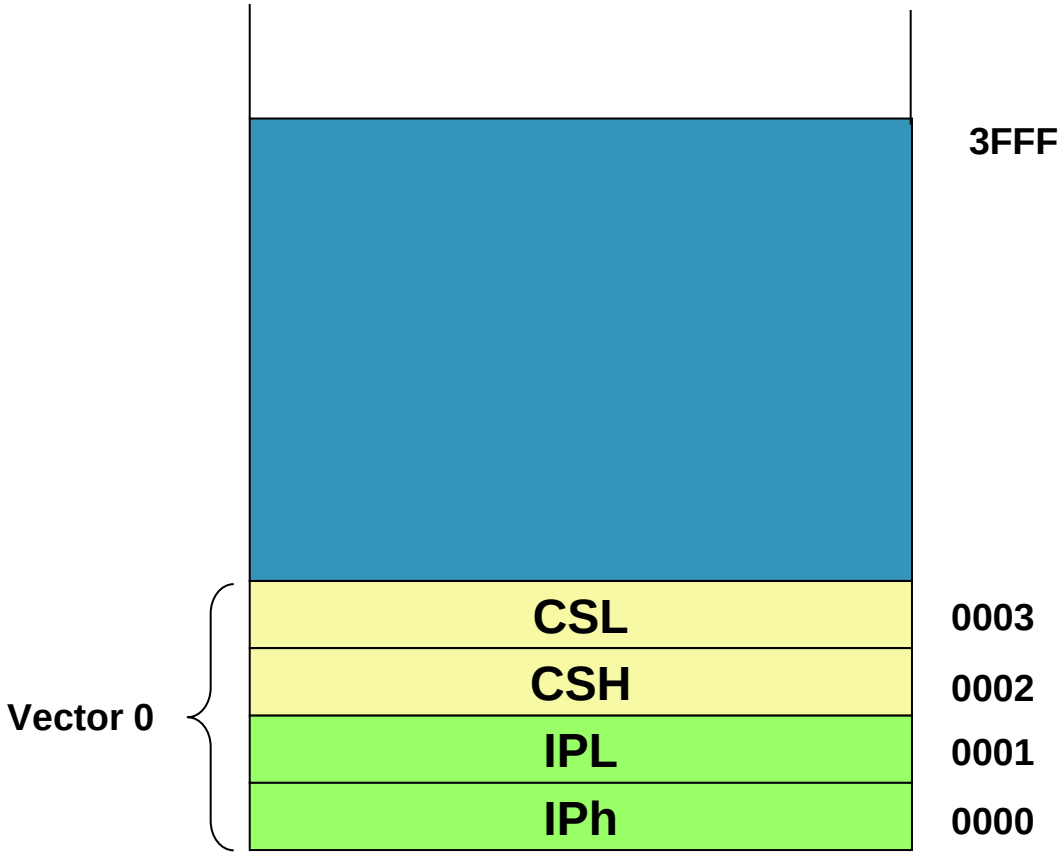
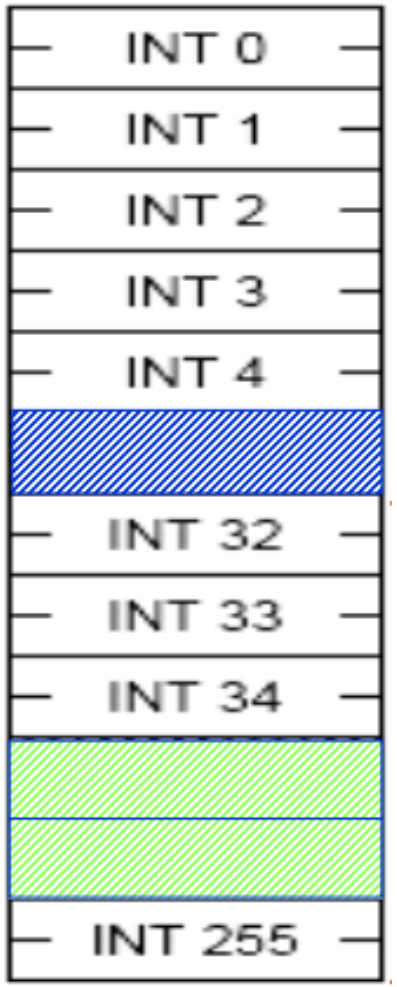


Tabla de Vectores de Interrupción



- La tabla de vectores de interrupción contiene las direcciones de las rutinas de servicio de las interrupciones que realizan las funciones asociadas con las interrupciones.
- Las rutinas del POST del BIOS inicializan la tabla de vectores, al momento de “bootear”, con las direcciones de las rutinas suministradas por el código en la ROM BIOS y después el DOS y nuestros programas de aplicación agregan sus respectivos vectores a los vectores de esta tabla, conforme son cargados.
- Un vector de interrupción está formado por la dirección de inicio de la rutina de servicio de la interrupción ISR (2 bytes para CS y 2 bytes para IP)

Tabla de Vectores de Interrupción



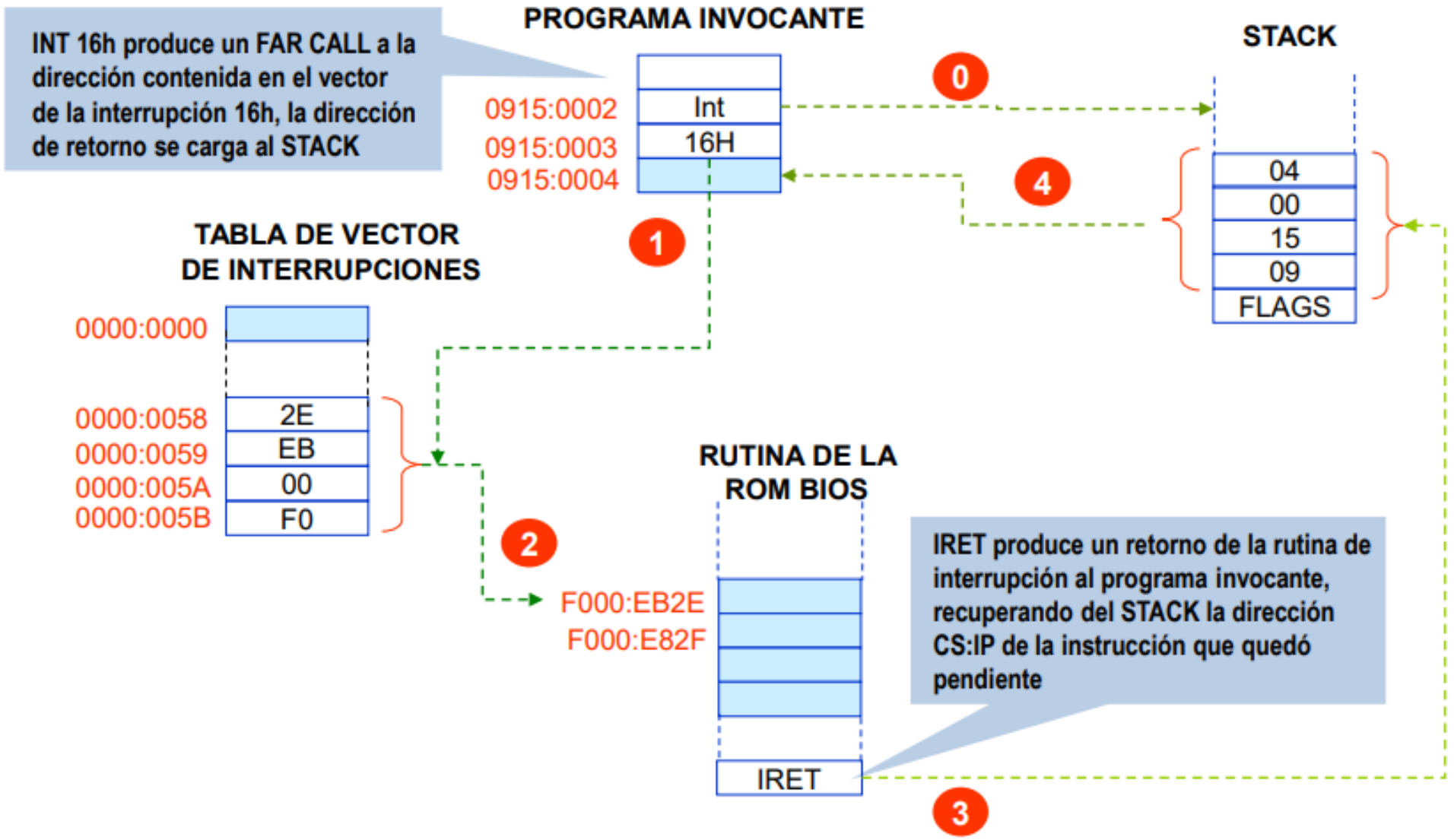
Error de división por 0
Ejecución paso a paso
NMI (No enmascarable)
Instrucción INT
Desbordamiento (INTO)

Reservadas por INTEL (modo protegido)

Utilizables por el usuario

Cada vector: 4 bytes
Dirección del vector: 0: INT x 4
La tabla ocupa 256x4 = 1024 bytes (1 Kbyte)

Tabla de Vectores de Interrupción



Interrupciones

Cuando una interrupción ocurre, independientemente de la fuente, el 80x86 realiza lo siguiente:

1. El CPU carga (push) el registro de banderas al STACK
2. La CPU carga al STACK la dirección de retorno (segmento:offset) primero el valor del segmento.
3. La CPU determina la causa de la interrupción (esto es, lee el número o tipo de la interrupción) y toma los 4 bytes del vector de interrupción de la dirección 0000:vector*4.
4. La CPU transfiere el control a la rutina especificada por la tabla de vectores de interrupción.

Después de completados estos pasos, la rutina de servicio de la interrupción toma el control. Cuando la interrupción desea regresar el control, debe ejecutar una instrucción IRET (Interrupt Return). El retorno de una interrupción recupera del STACK la dirección de retorno y las banderas.

Capacidad de interrupción

- ✓Las interrupciones y excepciones son acontecimientos que provocan un desvío en el flujo de control de la CPU.
- ✓Interrupciones son acontecimientos externos (en este caso activarán una patita (entrada) del microprocesador, que desvían el flujo de control (por ejemplo, averías en el hardware). También pueden ser generadas por alguna instrucción del procesador.
- ✓Excepciones son internas y se producen como consecuencia de una anomalía dentro de la CPU (por ejemplo, una división por cero) durante la ejecución de un programa.

IDT – Modo real

- ✓Para el manejo de interrupciones y excepciones, el Pentium dispone de una tabla o vector de interrupciones llamada IDT. (Tabla de descriptores de Interrupciones).
- ✓La IDT está formada por 256 entradas, cada una asociada a los diferentes tipos de interrupciones.
- ✓Cada entrada apuntará al comienzo de la subrutina que atenderá el tipo de interrupción.
- ✓La IDT ocupa un segmento, cuya base y limite están contenidos en el Registro de Tablas de descriptores de Interrupciones (IDTR).
- ✓Tanto en modo protegido como en modo real se utiliza la IDT, aunque la forma de operar es distinta en cada modo, el objetivo es el mismo.

IDT – Modo real

- ✓ En la inicialización o reset del Pentium (modo real), el SO carga la base del registro IDTR con el valor 0000 0000_H, para ubicar la IDT en el mismo lugar que si fuese un 8086. Previamente ha sido cargada en la memoria la tabla IDT. Esto lo hace el SO.

- ✓ En cada entrada de la IDT, se apunta la dirección dentro del segmento de código donde reside la rutina que atiende a la interrupción.

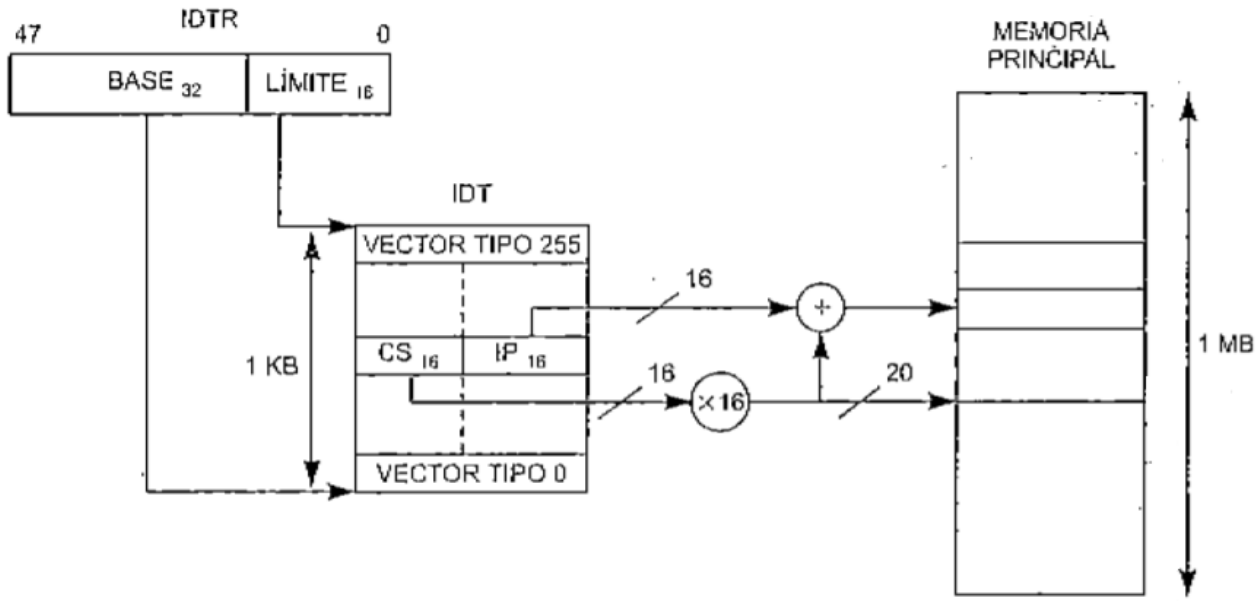


Figura 14.8. *Acceso a las interrupciones y excepciones en modo Real.*

- ✓ El tamaño máximo de la IDT es de 1 KB, o sea, 256 entradas x 4 bytes/entrada, por lo tanto, el límite es $03FF_H$

IDT - Vector

Relación entre el número de vector y su función.

Hay tantas entradas en la IDT como servicios definidos, cada vector (puntero) contiene la posición del servicio.

Función	Número de vector
Error de división	0
Interrupción paso a paso	1
Interrupción NMI	2
Interrupción punto de parada	3
Excepción por sobrepasamiento	4
Sobrepasamiento de los límites	5
Código OP inválido	6
Coprocesador no disponible	7
Reservadas (Modo protegido)	8-15
Error de coprocesador	16
Reservados INTEL	17-31
Definidas por el usuario	32-255

Figura 14.9. *Relación de vectores de interrupción comunes en modo Real y modo Protegido.*

Tipos de interrupciones y excepciones

Tabla 8-6. Diferencias entre distintos tipos de interrupciones.

Externas o hardware: son convocadas en forma asincrónica, no dependen del programa en ejecución.	No enmascarables (NMI) La CPU es avisada por una señal de control llamada comúnmente NMI (*Non Mascarable Interrupt*).	Siempre son atendidas. Se consideran de máxima importancia, pues las provoca el hardware ante acontecimientos que no pueden ser por lo menos "avisados" (p. ej., la caída de tensión de la alimentación).
	Enmascarables La CPU es avisada por otra señal que la diferencia de la NMI.	Se consulta la señal de interrupción por cada ciclo de ejecución de instrucción. Como no siempre debe ser atendida, se consulta también una bandera de estado de habilitación de interrupciones. Si la bandera lo autoriza, entonces la CPU suspende de manera momentánea la ejecución de programa activo y ejecuta el servicio de interrupción. Una función hardware externa, determina la **prioridad** que tiene esta interrupción respecto de otras peticiones pendientes y le da curso (p. ej., recepción de un byte en un puerto). Si la bandera de interrupción está desactivada, la CPU no la tiene en cuenta y continúa la ejecución del programa activo; de ahí deriva el nombre de "mascarable" o "enmascarable".

Tipos de interrupciones y excepciones

Internas o software: son convocadas por el programa.		Se pueden producir por la ejecución de una instrucción específica dentro de un programa, para solicitar una interrupción que cumpla con alguna función determinada. La forma de convocarla es INT #, donde # corresponde al número que identifica la interrupción.
Excepciones: son provocadas como consecuencia de anomalías que se producen y detectan durante la ejecución del programa y a causa de ella.	Faltas o errores	Son las que se pueden detectar y corregir antes de que se produzca la ejecución de una instrucción determinada (p. ej., se produce una falla al invocar una página que no está en memoria principal). La atención de esta interrupción provoca que el sistema operativo localice la página faltante y la cargue.
	Trampas	Son las que se detectan una vez ejecutada la instrucción que las provoca (p. ej., las que de alguna manera incorpora el usuario en el programa, como sobreflujo u *overflow*).
	Abortos	Son las que se detectan sin localizar la instrucción que las provoca, abortando la ejecución del programa (p. ej., un valor no válido en un registro de sistema).

Fases de atención de una interrupción o excepción

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

✓**Fase 1**

Se comprueba si hay interrupciones pendientes para ser atendidas. Si hay varias peticiones de interrupciones pendientes, se atenderán por orden de prioridad.

✓**Fase 2**

Al comenzar la interrupción se **resguarda el entorno actual de la CPU** (context switch) para que al finalizar la interrupción o excepción se retorne al mismo punto, es decir, se guardan el contenido de los registros del procesador en la pila.

Por ultimo IF = 0 prohíbe las interrupciones mascarables, evitando que una interrupción de menor importancia interrumpa a la rutina en ejecución. (ejemplo, periférico)

Fases de atención de una interrupción o excepción

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

✓**Fase 3**

Se busca la entrada correspondiente en la IDT y se cargan los registros CS e IP con el valor contenido en dicha entrada, de esta forma, se obtiene el inicio de la rutina y comienza la ejecución de la misma.

✓**Fase 4**

La rutina de interrupción finaliza con la instrucción IRET, **se restaura el contexto de la CPU** (context switch). De esa manera, el programa continua su ejecución en la siguiente instrucción a la que se produjo la interrupción o excepción.

El flag IF=1, es decir, se vuelven a activar las interrupciones mascarables.

Interrupciones Externas

Se dan cuando es necesaria la comunicación entre un dispositivo (Periférico) y la CPU.

Los mismos generan una señal de IRQ (Interrupt Request) donde las mismas acceden a un Controlador de Interrupciones (PIC).

El Controlador de Interrupciones es el que administra las mismas. (Habilita las líneas de Interrupción y asigna prioridades), salvo las no enmascarables ó NMI.

Las IRQ vienen acompañadas de un número para saber luego que Subrutina de Atención de Interrupción corresponderá ejecutar para atender la misma.