# Seminario 1: Uso e instalación de un entorno para la programación de interrupciones en Ms-dos

# Programación de sistemas

Dos tipos de programación:

- Programación de aplicaciones
- Programación de sistemas

*El programador de sistemas requiere un conocimiento mayor sobre el hardware*. Requiere la programación a nivel de núcleo del SO, creación de controladores lógicos o drivers, etc

La programación de sistemas a bajo nivel requiere:

- Ensamblador
- Lenguaje C/C++

**Periférico** = conjunto de registros
+
posible fuente de interrupciones

# Registros internos de los periféricos

- Se accede a ellos por medio de puertos de E/S
- Tienen asignados un conjunto de direcciones

- Tipos de puertos:
  - Datos
  - Instrucciones
  - Configuración
  - Estado
  - Direcciones

- A veces un mismo puerto se comparte para acceder a varios registros:
  - Instrucciones/Datos
  - Configuración/Estado
  - etc.

# Interrupciones

**Tipos de interrupciones:**

- Externas: generadas por los periféricos (ej. Pulsación tecla)

- Internas: generadas por la CPU (ej. División por cero, reloj)

- Software: generadas por las aplicaciones o el SO (ej. Solicitud de escritura en un fichero)

En respuesta a una interrupción:

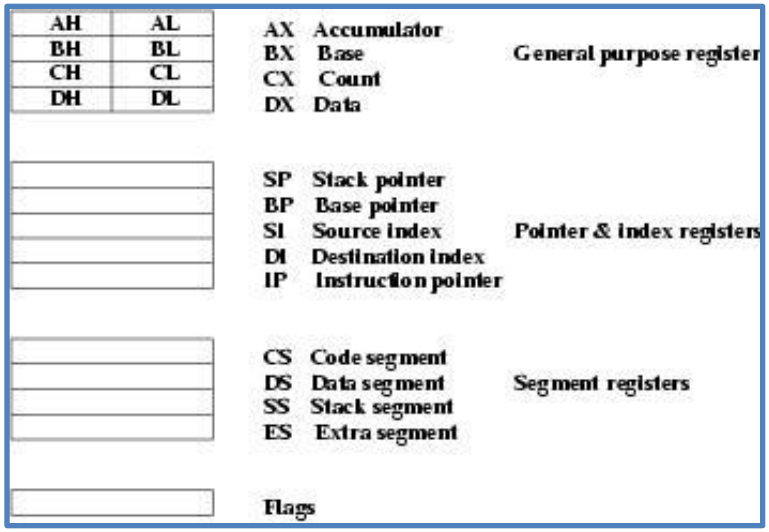
- La CPU pasa a ejecutar una ***rutina de interrupción (ISR) → BIOS***
- Los parámetros necesarios para que se ejecute la rutina se toman de los **registros internos de la CPU**
- Los valores que devuelven las rutinas de interrupción se **almacenan** en los registros internos de la CPU justo al terminar de ejecutarse la rutina

# La BIOS

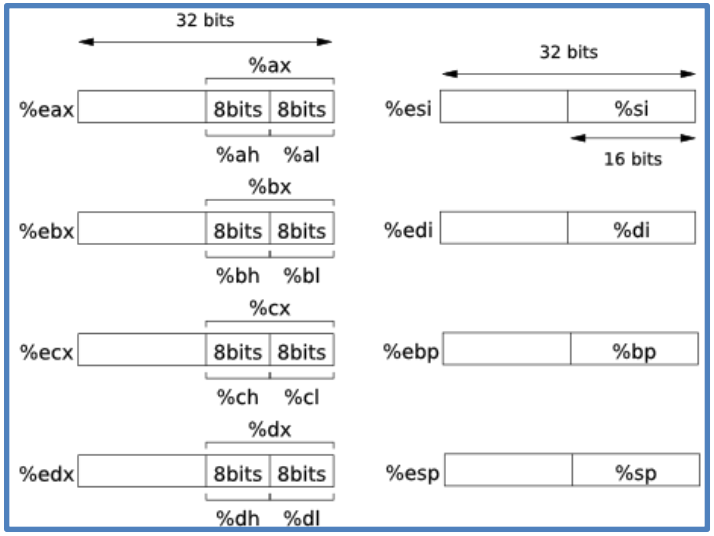
La BIOS (**Sistema Básico de Entrada/Salida**, Basic Input/Output System) de un IBM-PC contiene un conjunto de programas alojados en una memoria RAM-CMOS dentro de la placa base. Parte de esos programas son las rutinas de servicio a la interrupción (ISR).

→ Su contenido no depende del sistema operativo que ejecute el PC


# Registros del 80x86




Registros del 8086 (16 bits)



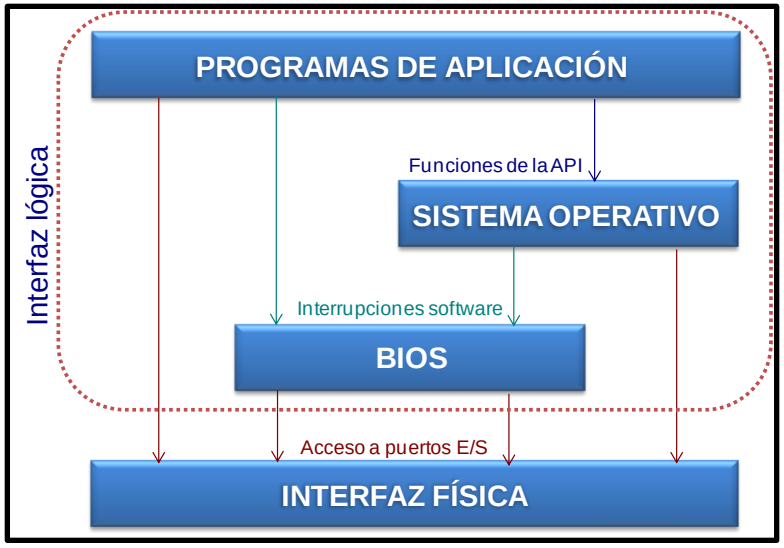

Registros IA-32

# Interrupciones software

En un PC:

- Rutinas de la ROM-BIOS:
  - Rutinas genéricas para dispositivos estándar
  - Son independientes del SO

- Rutinas de la API (*Application Program Interface*)
  - Dependen del SO
  - Amplían/particularizan/sustituyen las rutinas de la ROM-BIOS
  - Engloban tareas más complejas
  - Introducen una demora



# Ejemplo: pulsación de una tecla

Al pulsar una tecla ➔ **Interrupción externa:**

1. El teclado activa la línea de interrupción 9h
2. La unidad básica ejecuta la rutina de interrupción BIOS 9h:
   - Accede al puerto 60h para leer el código de barrido de la tecla
   - Convierte el código de barrido a código ASCII
   - Copia ese código ASCII en memoria principal (búfer del teclado)

Esta rutina se ejecuta con cada pulsación, aunque ningún programa haya solicitado que se pulse ninguna tecla.

# Ejemplo: lectura de una tecla

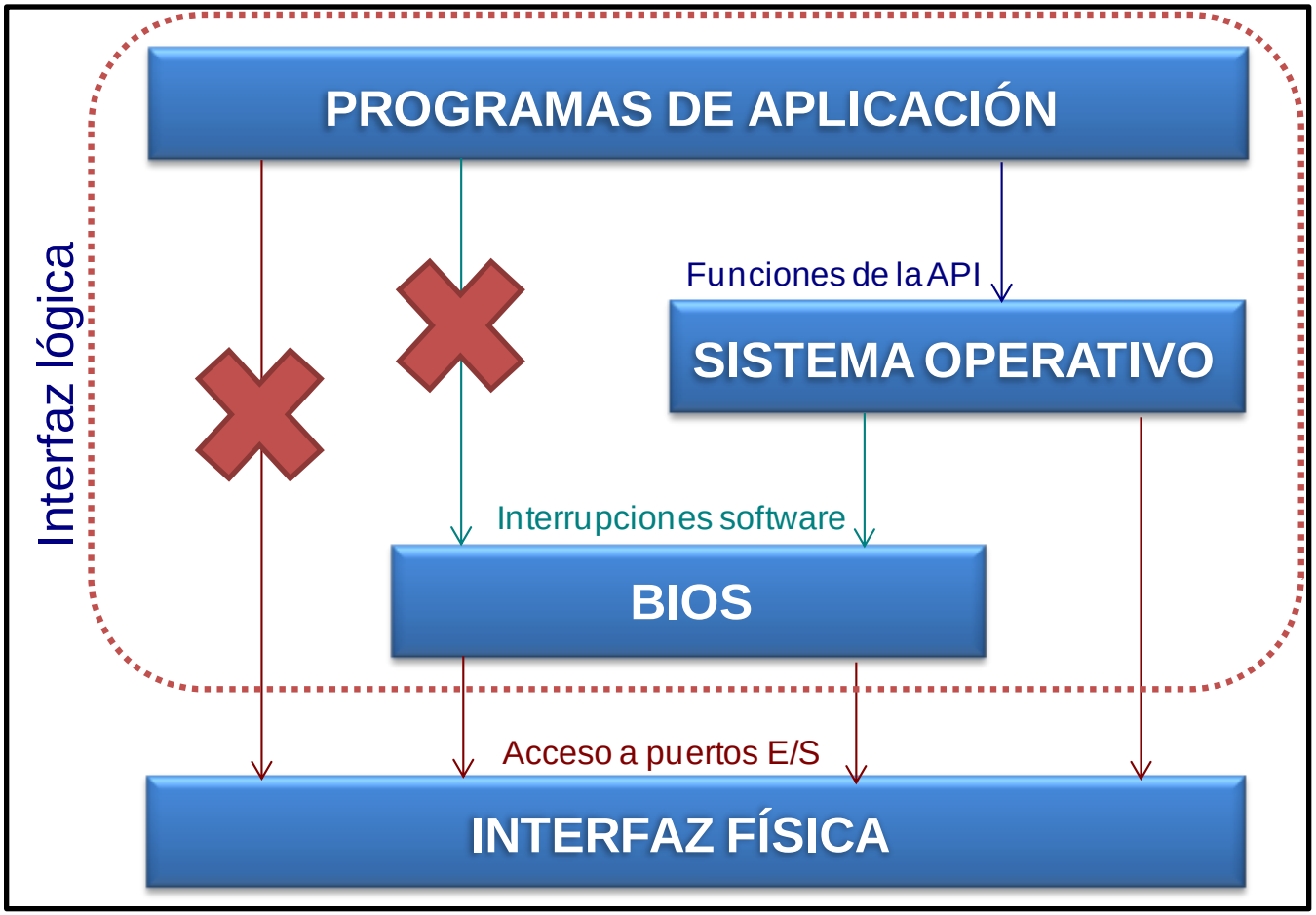
Para leer una tecla desde un programa (por ejemplo, con la función de C `getchar()`):

➔ **Interrupción software:**

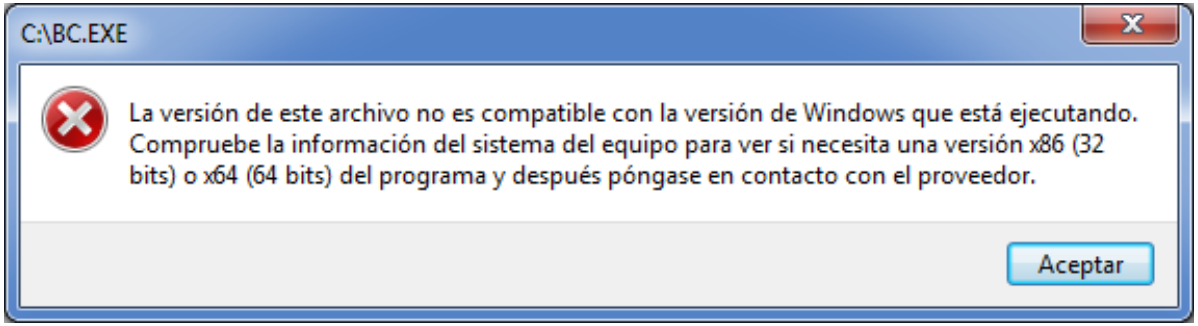
1. El programa provoca la interrupción 16h (ROM BIOS)
2. Se ejecuta la rutina de interrupción BIOS 16h:
   - Mientras el búfer del teclado esté vacío, queda en espera.
   - Extrae el primer código que encuentre en el búfer del teclado

# La comunicación con periféricos

El acceso en los sistemas operativos actuales:



# Ejecutando aplicaciones de 16 bits en S.O. actuales



# Recursos software necesarios

- Si usamos un emulador de MS-DOS, no es necesario instalar un sistema operativo en una máquina virtual o de forma nativa.
- Emulador MS-DOS: DOS-Box 

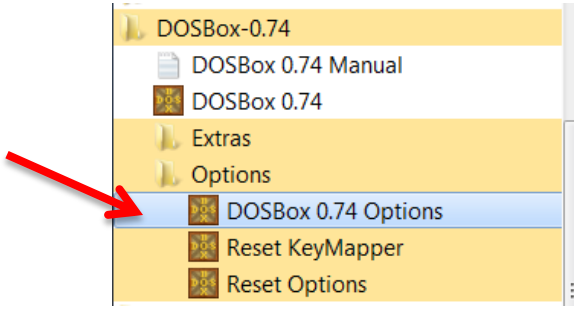
- Entorno de Programación: BorlandC 3.11

# Configuración de DOSBox

Aunque no es imprescindible, conviene configurar DosBox para no tener que repetir algunas operaciones cada vez que lo iniciamos. Básicamente queremos:

- Montar el directorio de BorlandC (de nuestro PC) en la unidad C: (virtual de DosBox). Ya de paso, lo añadimos al PATH. Debe quedar instalado en la ruta `C:\BC`
- Montar el directorio de Prácticas (de nuestro PC) en la unidad D: (virtual de DosBox).
- Configurar la ventana del programa para que se vea bien en nuestra pantalla.

Se debe cambiar el fichero "dosbox-0.74.conf". Para ello, seleccionamos el grupo de DosBox en el menú de inicio y abrimos "DosBox Options"



# Configuración de DOSBox

1) Para desbloquear el ratón de la ventana usar CTRL+F10.
2) Para finalizar → teclear EXIT.
3) Por defecto CTRL+F9 en DOSBox cierra el programa (¡sin salvar nada!).

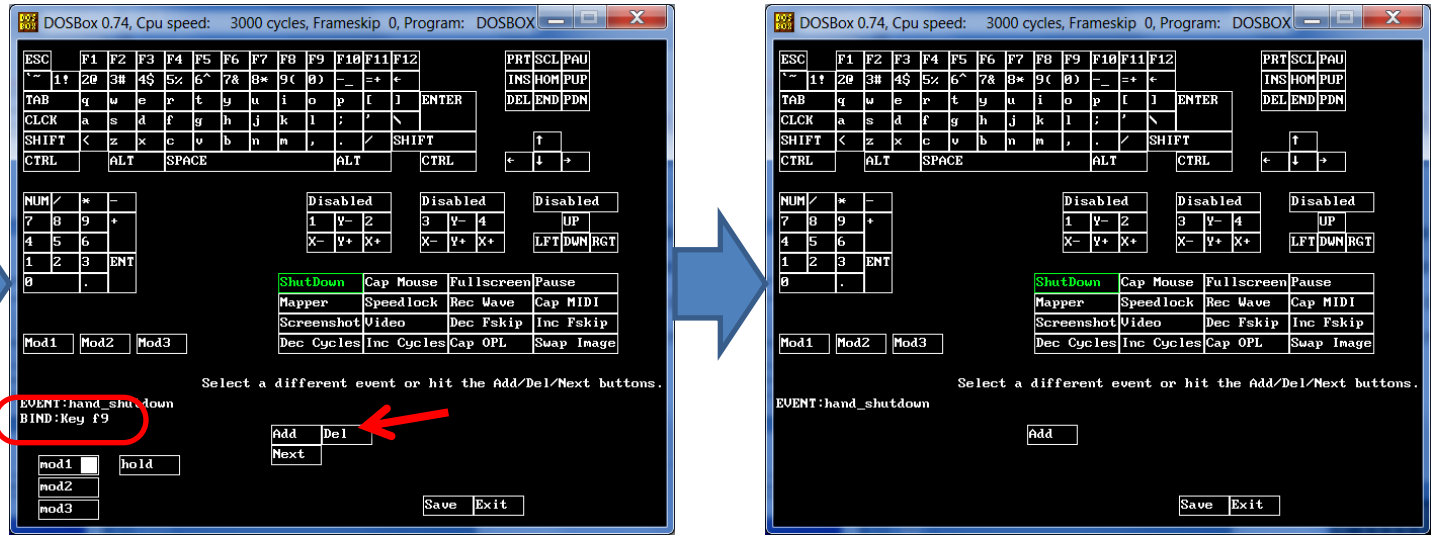
   Es muy recomendable deshabilitarlo así como otras combinaciones de teclas que usemos normalmente en BorlandC

   → pulsar CTRL+F1 y quitar el "map" de F9).

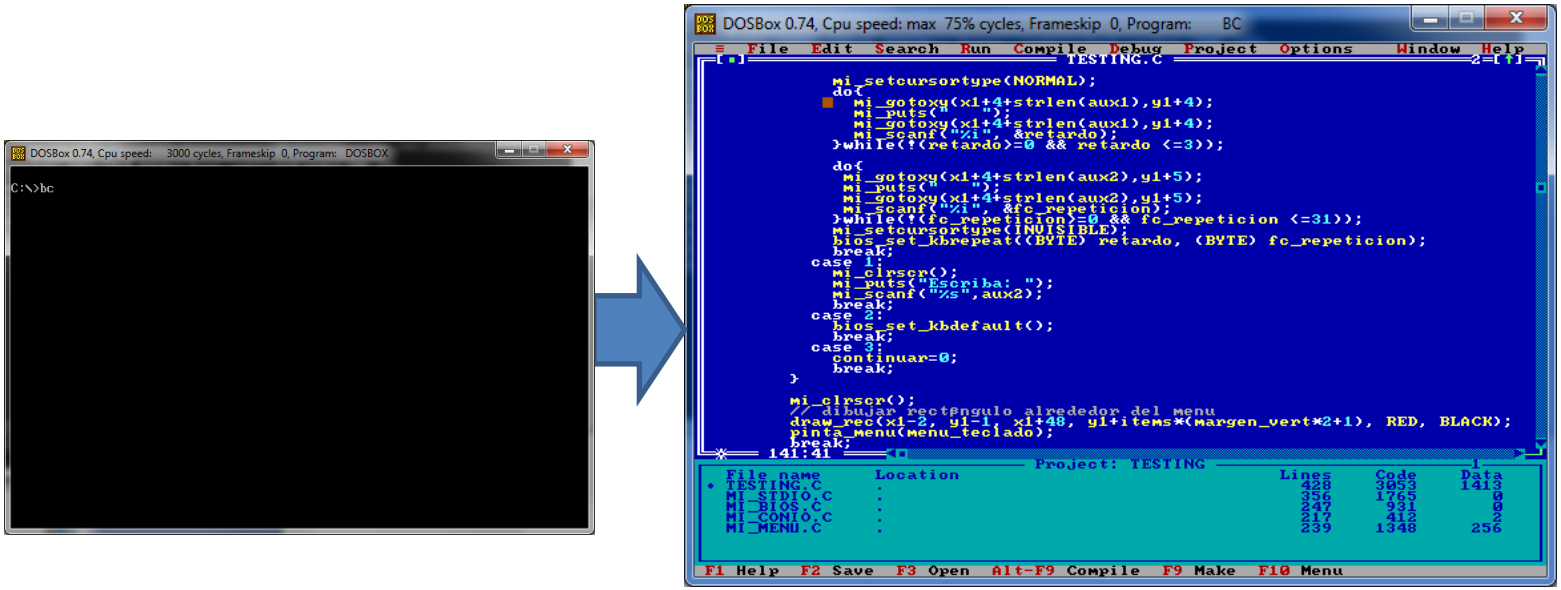
   → los cambios de guardan en el fichero: `mapper-0.74.map`

   (idea: guárdalo por si tienes que exportarlo a otra máquina)

CTRL+F1



# IDE de Programación: Borland C




Aunque es un IDE antiguo, permite las operaciones principales de programación:

- Edición de código fuente → aunque también puedes editar desde el s.o. anfitrión.
- Resaltado de sintaxis
- Herramientas de depuración
- Creación de proyectos
- …

# Interrupciones software en C

```
#include <dos.h>

int86 (int intno, union REGS *inregs,
       union REGS *outregs)
```

1. Copia inregs en los registros internos (valor de los parámetros necesarios para la rutina).

2. Ejecuta la rutina de interrupción intno

3. Copia los registros internos en outregs ( valores devueltos por la rutina).

# Interrupciones software en C

La unión `REGS` está definida en `dos.h`. Permite acceder tanto a los registros de 8 bits como a los de 16.

```
union REGS {
  struct WORDREGS x;
  struct BYTEREGS h;
};
struct BYTEREGS {
  unsigned char al, ah, bl, bh;
  unsigned char cl, ch, dl, dh;
};
struct WORDREGS {
  unsigned int ax, bx, cx, dx;
  unsigned int si, di, cflag, flags;
};
```

Se puede hacer: `inregs.x.ax=0x0305;`

o bien:
```
inregs.h.ah=0x03;
inregs.h.al=0x05;
```

# Interrupciones software en C

## Ejemplo: función en C que permite modificar el modo de video

```
#define BYTE unsigned char

/* Selecciona el modo de video deseado */
void selecciona_modo_video(BYTE modo)
{
 union REGS inregs, outregs;

 inregs.h.ah = 0x00;
 inregs.h.al = modo;
 int86(0x10,&inregs,&outregs);
 return;
}
```

# Ejercicios:

## A evaluar la semana que viene (26 de febrero)

1. Instala el software "DOSBox" en tu PC. http://www.dosbox.com/. Estudia la forma en que puedes ejecutar aplicaciones de MS-DOS como juegos clásicos (en SWAD: Arkanoid y Golden Axe). Otros ejecutables en http://www.clasicosbasicos.org/ y https://archive.org/details/softwarelibrary_msdos_games.
2. Configurar el inicio de DOSBox para que monte en la unidad C: el directorio donde se encuentra el entorno de programación Borland C. Añadir a la variable “PATH” de inicio el directorio “bin” donde se encuentra el ejecutable “bc.exe”.
3. Configurar el inicio de DOSBox para que monte en la unidad D: el directorio donde se irán almacenando los subdirectorios con los proyectos en C que se vayan generando.
4. Configurar el “keymapper” de DOSBox para que el programa Borland C no se cierre al pulsar CTRL+F9 y para que funcionen correctamente las teclas F7 y F8 para el modo de depuración paso a paso de Borland C.
5. Realizar un programa que llame a la función de cambio de modo de vídeo especificado en la página 8. Debe pedir un valor numérico al usuario y cambiar a dicho modo de vídeo. Guardar los ficheros de este ejercicio dentro de un proyecto (.PRJ).