

Práctica : *Frontón*

Estructura de Computadores

Curso 2º

I. Informática
I.T. Informática (Sistemas)
I.T. Informática (Gestión)

Departamento de Arquitectura de Computadores

Curso 2003-2004

1. Objetivos.

El objetivo de esta práctica es conseguir que el alumno siga trabajando en base a los conocimientos adquiridos durante la realización de las prácticas obligatorias de la asignatura Estructura de Computadores. En especial en lo que se refiere al manejo de algunos de los periféricos más importantes de un PC: pantalla (memoria de vídeo), teclado y ratón. Así mismo se pretende que el alumno profundice más en la programación del sistema de interrupciones, trabajando con otros periféricos como el puerto paralelo y el temporizador 8244.

El trabajo a realizar se ajusta a una serie de especificaciones que se detallan a continuación. De ellas, las especificaciones mínimas deberán ser cumplidas *obligatoriamente*.

2. Especificaciones.

2.1. Especificaciones mínimas

Se propone el desarrollo de un programa escrito en lenguaje ensamblador del i8086, que simule el virus de la pelota y un frontón con la particularidad de que existan numerosos “virus pelota” actuando en la pantalla.

Sumarizamos los requisitos a los que se habrá de ajustar la práctica:

1. El programa funcionará en modo de vídeo texto 25×80 color.
2. A continuación aparecerá un mensaje en el que se solicitará al usuario el número de “virus pelota” con el que trabajará el programa. Este número se leerá desde el teclado y estará limitado en el rango de 1 a 5.

El programa debe permitir también la lectura de argumentos (un número de 1 a 5) desde la línea de comandos de MS-DOS. Por ejemplo, la invocación del programa:

`C:>Fronton 2`

daría lugar a la ejecución directa del juego, sin que aparezca el mensaje inicial de entrada. En caso de que no se especifiquen argumentos, o éstos no sean correctos, se procedería a comenzar el juego con el mensaje inicial, tal como se ha especificado anteriormente.

3. Conocido el número de “virus pelota”, aparecerán moviéndose por pantalla tantas “pelotas” como indique este número. Dichos “virus pelota” se implementarán con el carácter con código ASCII = 1 y deben tener colores diferentes entre sí.

La posición inicial y la dirección y sentido de movimiento de cada “virus” será aleatoria.

4. Existirán 3 modos básicos de funcionamiento que se pueden seleccionar en cualquier momento mediante una tecla según se especifica en la siguiente tabla. El modo inicial al entrar en el programa es el correspondiente a F1.

Tecla	Modo
F1	“Virus múltiple transparente”
F2	“Virus múltiple con rebote”
F3	“Frontón”
Pulsador	Finalizar programa

5. Si se pulsa una tecla distinta de las permitidas, el programa se queda en el modo en el que está y la(s) pelota(s) continuarán en la misma posición en que estuvieran y se moverán con la dirección que tuvieran.
6. Al actuar sobre el pulsador del puerto paralelo, el programa finaliza y, esté donde esté, restaura la pantalla original que había antes de comenzar la ejecución del programa.
7. Para que el usuario no olvide cuáles son los diferentes modos de juego se escribirá en la primera línea de la pantalla (fila 0) un breve menú con la funcionalidad de las teclas según se ha especificado. Este menú estará presente en todo momento. También se presentará un indicativo de cuál es el modo actual (por ejemplo cambiando el atributo de los caracteres que representen el modo de juego en el que se está en ese momento).
8. En cualquier momento el usuario podrá decidir cambiar de un modo a cualquier otro, o bien finalizar el programa.

2.2. Modos de juego

2.2.1. Virus múltiple transparente (F1)

En este modo los “virus” pasan por encima de los caracteres que se encuentren en la pantalla, restaurando el contenido (ascii y atributo) de las posiciones que atraviesen. En este modo el rebote de las pelotas se produce únicamente con los bordes de la pantalla. En caso de que el usuario seleccione más de un “virus pelota”, no rebotarán entre sí en este modo.

2.2.2. Virus múltiple con rebote (F2)

En el modo “virus con rebote”, las pelotas rebotan, además de con los bordes de la pantalla, con todo carácter distinto del espacio, sin destruirlo. En caso de que el usuario haya seleccionado más de un “virus pelota”, las diferentes pelotas rebotan entre sí en caso de choque. Tras el rebote sobre un carácter, borde u otra pelota, una pelota no puede nunca posicionarse sobre un carácter distinto del espacio.

Si se conmuta entre los modos F1 y F2, los “virus pelota” no comienzan con posiciones nuevas aleatorias sino que continúan con la posición, dirección y sentido que lleven.

Se acepta que en modo rebote (F2) una pelota pueda quedar encerrada entre caracteres sin movimiento, bien al establecer condiciones iniciales (por ejemplo al principio o al conmutar de F3 a F2) o bien al conmutar de modo (por ejemplo de modo F1 a F2).



2.2.3. Frontón (F3)

Al comenzar el modo “frontón”, deben desaparecer todos los caracteres de pantalla y el programa mostrará por pantalla un frontón para un jugador. Las paredes del frontón serán la fila 1, y los bordes derecho e izquierdo de la pantalla (columnas 79 y 0, respectivamente). La raqueta para el juego se encontrará en la parte inferior de la pantalla (fila 24). A continuación se mostrará el cursor de ratón.

En este modo la posición inicial de la pelota se seleccionará mediante un “doble clic” del botón izquierdo del ratón, quedando situada la pelota en el carácter correspondiente al cursor del ratón en el momento del “doble clic”. La dirección de movimiento será aleatoria. Se

considerará que se ha producido un “doble clic” de ratón si entre una pulsación y la siguiente transcurren menos de 440 ms. (8 “ticks” del reloj del sistema).

Cada vez que la pelota escape del recinto frontón, el programa ha de colocar la pelota para un nuevo juego en una posición aleatoria con dirección y sentido de salida también aleatorios.

Como se ha comentado, en mitad del borde inferior aparecerá la raqueta que estará formada por cuatro caracteres ‘—’ colocados horizontalmente y con atributo verde intenso. La pelota rebotará sobre la raqueta igual que sobre una pared. El movimiento horizontal de esta raqueta se controlará con los cursores  y .

En este juego cada rebote se acompañará de un sonido diferente. Dependiendo de si la pelota rebota en los bordes superior, derecho o izquierdo, o en la raqueta, o si se pierde por el borde inferior, en la tabla siguiente se indican las características del sonido que se han de producir.

Rebote	Frecuencia	Duración
Borde superior Borde derecho Borde izquierdo	659 Hz (mi5)	10 ms
Raqueta	784 Hz (sol5)	20 ms
Salida inferior	523 Hz (do5)	50 ms

En el modo “frontón”, además del breve menú con la funcionalidad de las teclas deben aparecer, en el borde superior de la pantalla, y en todo momento dos contadores de al menos dos dígitos. Uno de ellos, que será el contador de puntos de la partida, debe contabilizar el número de rebotes que se consiguen con la raqueta. El otro contador, se inicializará al número máximo de vidas permitidas en una partida y cada alumno puede decidir su valor. Este segundo contador se ha de decrementar con cada pelota que se escapa por el borde inferior sin rebotar en la raqueta. Cuando este segundo contador alcance el valor 0, el modo frontón finalizará volviendo al modo inicial (F1).



Cuando se commuta del modo F3 a uno de los otros 2 modos, estos comienzan con las condiciones iniciales descritas en el punto 3, restaurando la pantalla con el contenido original.

2.3. Especificaciones adicionales.

Además de las especificaciones básicas, se deberá implementar una de las siguientes especificaciones propuestas. También serán evaluadas cuantas mejoras sean realizadas por el alumno, especialmente aquellas mejoras que requieran una ampliación de los conocimientos..

- Durante el juego puede generarse una melodía. Se usará el canal 2 del timer como generador de sonidos.
- Construir una rutina propia de manejo de teclado. Para ello hay que elaborar la rutina de tratamiento de interrupción de la interrupción 09H correspondiente a la IRQ 1. Dicha rutina leerá del controlador de teclado (puertos 60H y 61H) el código correspondiente a la tecla pulsada.

Además de controlar el movimiento de la raqueta, o el cambio a otro modo de juego, se añadirán las siguientes nuevas funciones controladas desde el teclado:

- Se puede modificar la velocidad de las pelotas, bien incrementándola o decrementándola usando las teclas de cursor  y .
 - Además en el modo “frontón” el usuario podrá deshabilitar/habilitar los sonidos con la tecla TAB. Habrá un indicativo en pantalla que muestre si el sonido está activo o no.
 - También el usuario podrá alargar o acortar la longitud de la raqueta pulsando respectivamente las teclas ‘>’, ‘<’.
 - Se implementarán velocidades de movimiento diferentes para cada una de las pelotas (en caso de que haya más de una). Las velocidades de cada pelota se podrán incrementar o decrementar con pares de teclas (‘a’-‘b’, ‘c’-‘d’ ...)
 - También se puede modificar la velocidad de la raqueta, incrementándola o decrementándola con las teclas ‘+’, ‘-’.
 - En el frontón se puede añadir un modo de disparo que permita disparar balas desde la raqueta con la barra espaciadora, de manera que puedan ser destruidos los ladrillos.
- En la pared del frontón se pueden dibujar un conjunto de ladrillos que pueden ser destruidos por la pelota cuando rebota con ellos. Cuando la pelota choca con uno de los ladrillos, también ha de rebotar. Cuanto más arriba está el ladrillo que se destruye, más puntos conseguirá el jugador.

Los ladrillos pueden esconder objetos, que aparecerán moviéndose si el ladrillo que los escondía es destruido. Si la raqueta toca los objetos antes de que salgan por la derecha, el jugador puede conseguir puntos o vidas extra.

Serán evaluadas cuantas mejoras sean realizadas por el alumno a los anteriores requerimientos. En este sentido, el alumno podrá añadir a las funcionalidades descritas todas aquellas que considere que pueden enriquecer y completar el funcionamiento del frontón.

3. Pistas, sugerencias y detalles de realización.

A continuación incluimos algunas sugerencias que sería conveniente tener en cuenta al realizar la práctica.

- Puesto que es necesario restaurar la pantalla cuando salimos del modo “virus borrador” o “frontón”, tenemos que guardar la pantalla original. Para ello se pueden utilizar dos páginas de vídeo (al menos). El programa primero pregunta por la página activa en la que se encuentra. Supongamos que es la página P. A continuación se puede volcar dicha página en la siguiente página, es decir la P+1. Restaurar la pantalla original no supone más que volver a volcar la página P+1 en la página P. El modo texto 80x25 define 4 páginas, donde la página 0 comienza en la posición de memoria B800:0000h, la página 1 en la posición de memoria B800:1000h, la página 2 en la posición de memoria B800:2000h, y la página 3 en la posición de memoria B800:3000h, siendo el tamaño útil de cada página de 4000 bytes. El 8086 dispone de instrucciones específicas (denominadas de cadena) para mover bloques continuos de datos (MOVSB, MOVSW, ...) que son útiles en este caso.

Otra opción es que al salir del programa no es necesario restaurar el contenido de la pantalla, basta hacer un cambio de página activa.

- No olvidar restaurar la posición del cursor cuando restauramos la pantalla, para lo cual habrá que almacenarla previamente al comenzar la ejecución del programa.
- La forma más rápida de escribir por pantalla es hacerlo directamente en el banco de memoria asociado a la página de vídeo activa. En el modo de vídeo exigido, para escribir un carácter en la posición (X, Y) de la pantalla, hay que escribirlo en la dirección lógica:

$$B800H : (P \cdot 1000H + 160 \cdot X + 2 \cdot Y + B)$$

donde, P = número de página; Y = columna; X = fila; $B = 0$ es el byte de carácter (ascii); $B = 1$ es el byte de atributo (colores).

- Los números pseudoaleatorios se pueden simular a partir del contador de “ticks” del sistema. Este contador es una doble palabra que está en la posición 0040:006Ch del área de datos de la BIOS y que se incrementa por cada interrupción del canal 0 del temporizador (por defecto cada 55 ms.). Si queremos generar un número aleatorio entero comprendido en el intervalo de enteros $[0, Rango - 1]$, dicho número se obtiene calculando el resto de dividir el número de “ticks” entre $Rango$.

A veces, como ocurre durante la generación de las coordenadas iniciales de los virus, podemos requerir varios números aleatorios consecutivos y, como el contador de “ticks” puede no haberse modificado entre lecturas consecutivas, los números a efectos prácticos no serán aleatorios. Se propone para ello un algoritmo mejorado basado en la introducción de un valor-semilla, que será una nueva variable S . Esta variable S se controla de la siguiente manera:

1. En la rutina de inicialización del programa, se obtiene la semilla a partir del contador de “ticks”:

$$S \leftarrow \text{contador_de_ticks}$$

2. Cuando queramos generar un número aleatorio en el rango $[0, Rango - 1]$, lo haremos a partir de la semilla que es actualizada sumándole el valor del contador de “ticks” y realizando alguna transformación binaria del resultado:

$$S \leftarrow \text{transformar}(S + \text{contador_de_ticks})$$

$$\text{Aleatorio} \leftarrow S \text{ módulo } Rango$$

donde *transformar* puede ser, por ejemplo, un desplazamiento hacia la izquierda de 6 bits.

- La información de esta misma posición de memoria donde se guarda la cuenta de “ticks” puede ser empleada para determinar cuando se ha producido un “doble clic” de ratón en aquellas circunstancias que haga falta.
- La dirección y sentido de la pelota estarán controlados por uno de los cuatro vectores indicados en la figura 1.

La pelota por lo tanto siempre formará un ángulo de 45° con cualquier borde de la pantalla. Ello es debido a que en cada movimiento de la pelota sus coordenadas X e Y se ven afectadas por incrementos de 1 ó -1. Si se quisieran simular bordes de la pantalla o de la raqueta rugosos, se podría elegir como ángulos de salida, ángulos aleatorios por lo que se tendría que utilizar varios tipos de incrementos ($\dots, -2, -1, 1, 2, \dots$) para modificar las coordenadas de la pelota.

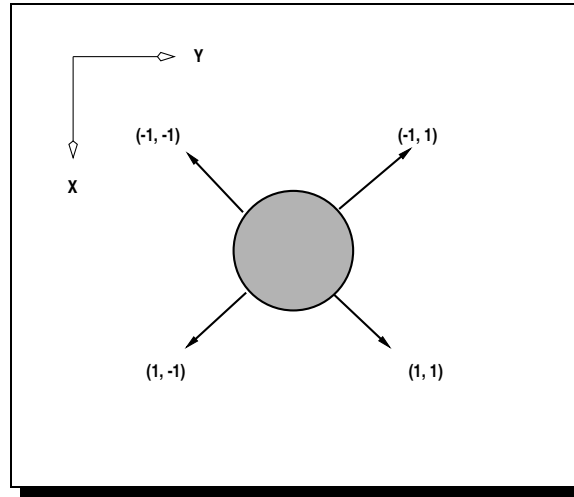
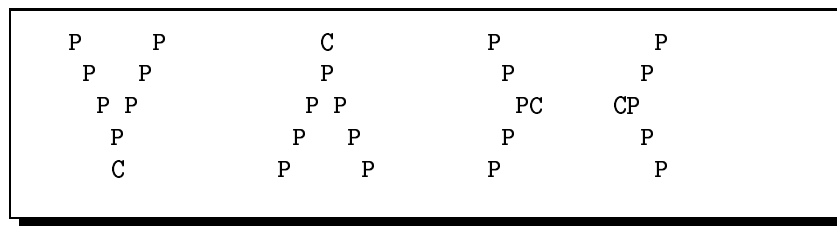
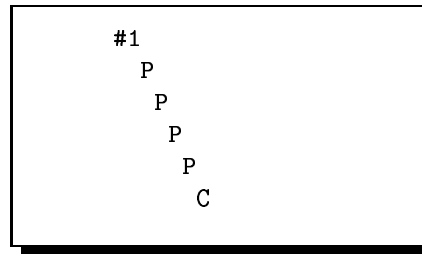


Figura 1: Vectores de dirección de la pelota.

- Para que la pelota rebote con un carácter ambos deben de estar colocados en la misma fila o columna. Gráficamente, si P representa a la pelota y C al carácter, el rebote se produce:



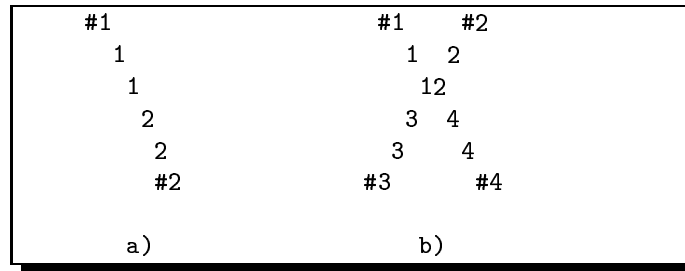
Otra forma de rebote con un carácter es cuando la pelota incide en una *esquina* del carácter, es decir, cuando carácter y pelota coinciden en una diagonal. En este caso la pelota invierte su sentido de movimiento.



En este ejemplo, si la pelota entra por #1, con esta dirección y sentido al chocar con el carácter invertirá su sentido volviendo a salir por #1.

- Además en el modo “virus múltiple con rebote”, una vez que se ha producido un rebote, hay que comprobar si la nueva posición de la pelota no ocupa una posición donde ya hay un carácter. Si así ocurriera la pelota no se movería sino que volvería a computar el rebote. Es posible que la pelota quedara ‘encerrada’ computando el rebote sin moverse.

- En caso de que existan varios “virus pelota” en modo F2, éstas pueden colisionar entre sí. Se pueden dar dos casos: si llevan la misma dirección pero sentido contrario ambas invierten su sentido y no cambian de dirección. Si llevan diferente dirección invertirán aquella componente del vector dirección que se vea afectado por el choque. En la figura siguiente se muestra ambas situaciones: a) si el “virus” #1 chocara con el #2 ambos invierten su sentido; b) si la pelota #1 chocara con la #2 se invierte la componente de dirección en que difieren ambas: #1 continúa en dirección #3 y #2 continúa en dirección #4.



4. Entrega de las prácticas.

Los trabajos han de ser entregados al profesor responsable de la asignatura antes del viernes 21 de Mayo de 2004.

Cada alumno entregará un diskette de $3\frac{1}{2}$ pulgadas, formateado a 1.44 M., no de sistema y libre de virus. En la etiqueta adhesiva deberá aparecer: el nombre y apellidos del alumno, titulación y grupo al que pertenece.

En la etiqueta adhesiva deberá aparecer: los apellidos, el nombre y DNI del alumno y la titulación y grupo al que pertenece. Por favor, ponga estos datos en el orden y orientación que se muestra en la figura.



Dicho diskette deberá contener:

- Los fuentes en ensamblador que constituyen la implementación de la práctica. El nombre del programa principal será **FRONTON.ASM**. Si el alumno usara otros módulos adicionales estos se nombrarían siguiendo la siguiente notación: **FR_####.ASM**, donde **####** representa un mnemónico relacionado con el contenido del módulo en cuestión (por ejemplo, **FR_rat.ASM**, **FR_tim.ASM**, ...). Se recomienda que las funciones diferentes (ratón, temporización, lectura de teclado, rutinas de pantalla, ...) se distribuyan en módulos diferen-

tes ¹. En los fuentes se incluirá una cabecera de comentario con el nombre del alumno, apellidos, DNI, su titulación y grupo.

Es importante que en la cabecera de cada módulo fuente además se comente la función de ese módulo, así como cada una de las rutinas que define.

Sólo se aceptarán ficheros fuente que se puedan ensamblar con `tasm 3.1` y enlazar (montar) con `tlink 5.1`, ambos disponibles en el laboratorio.

- Un fichero ASCII con nombre `LEEME.TXT` en el cual figurará: el nombre, apellidos, DNI, titulación, grupo del alumno y una descripción de los diferentes módulos en los que se ha dividido la práctica. Aquí se hará constar con la mayor claridad posible qué hace cada uno de los módulos fuente.
- Otro fichero ASCII con nombre `MEJORAS.TXT`, donde se hará constar las mejoras y cómo se activan.
- Un fichero de procesado por lotes con nombre `ENSAMBLA.BAT`, que se encargará de ensamblar y enlazar los módulos fuentes. El contenido de este fichero sería, por ejemplo, para un conjunto de ficheros fuentes `FRONTON.ASM`, `FR_AAA.ASM`, `FR_BBB.ASM`:

```
echo off
echo -----
echo Nombre, Apellidos:
echo DNI:
echo Titulación:
echo Grupo:
echo -----
echo Ensamblando: FRONTON.ASM
tasm /zi FRONTON
pause
echo Ensamblando: FR_AAA.ASM
tasm /zi FR_AAA
pause
echo Ensamblando: FR_BBB.ASM
tasm /zi FR_BBB
pause
echo Enlazando: FRONTON.exe
tlink /v FRONTON+FR_AAA+FR_BBB
```

Los objetos y ejecutables NO se incluirán en el diskette.

Con respecto a la evaluación, se recuerda que la entrega de la práctica es opcional para las personas que deseen optar a más de aprobado. Debe, por tanto, incluir además de las especificaciones mínimas, alguna de las opcionales y no debe generar ningún tipo de error en el proceso de ensamblado y enlazado y el ejecutable debe funcionar correctamente (no puede quedarse ‘colgado’). Tampoco se debe olvidar que los códigos fuente han de escribirse *lo más claramente posible sin reparo en el uso de comentarios*.

¹No usar más de 8 módulos, ya que pueden surgir problemas al estar limitados el número de caracteres de una línea de comandos MS-DOS

En el laboratorio están a disposición de los alumnos varios códigos fuente donde se emplean rutinas de manejo de ratón, control del temporizador, del puerto paralelo y salida de datos por pantalla.

5. Referencias

Los datos necesarios en la elaboración de la práctica se pueden encontrar en:

- *Prácticas de Estructura de Computadores*. G. Bandera, M.A. González, E.D. Gutiérrez, J. Ramos, M.A. Trenas, J. Villalba. Servicio de Publicaciones e Intercambio Científico de la Universidad de Málaga, 2003.
- *8088-8086/87 Programación Ensamblador en MSDOS*. M. Angel R. Rosselló. Anaya Multimedia.
- *The 8086/88 family: desing, programming and interfacing*. Jonh Offenback. Prentice-Hall international editions.
- *El Universo Digital del IBM PC, AT y PS/2-v4.0*. Ciriaco García de Celis. Grupo Universitario de Informática, 1997. Apto. Correos 6062, Valladolid.
<http://www.ac.uma.es/educacion/cursos/informatica/LabEstComp-Plan94/udigital/>
- *The Norton Guides*, Peter Norton Computing. Guía 'on-line'.
- *HelpPC*, David Jurgens. Guía 'on-line'.

Vía internet encontraréis la información del curso en:

<http://www.ac.uma.es/educacion/cursos/informatica/EstComp/>