Arquitectura de Computadores I

Ejercicios de Autoevaluación - 2

1.- [1 punto] Responde a las siguientes preguntas.

a) ¿Por qué se utiliza la unidad de control cableada sobre todo en las máquinas RISC? ¿Por qué se utiliza la unidad de control microprogramada sobre todo en las máquinas CISC? ¿Cuál es el motivo por el que se dice que la cableada es más rápida que la microprogramada?

b) Teniendo en cuenta la máquina BIRD explicada en la asignatura, se quiere ampliar el conjunto de instrucciones de ese procesador para incluir la instrucción "`change rd, VARIABLE`". Esa instrucción intercambia los contenidos del registro `rd` y `VARIABLE`, es decir:

```
rd := MEM[variable];
MEM[variable] := rd;
```

Su formato es el siguiente:

6	5	5	16
C.O.	rd		variable

Escribe la parte de microprograma que corresponde a la fase de ejecución de dicha instrucción. Si hay que modificar la Unidad de Proceso, indícalo claramente esquemáticamente en la figura que se facilita en el cuadernillo de respuestas del examen.

2.- [1,5 puntos] Contesta a las siguientes preguntas:

a) Realiza la multiplicación **23*71** con el algoritmo de ***suma y desplazamiento*** y la codificación ***Booth en base dos*** con ***8 bits***, haciendo las sumas parciales con un sumador CSA y la suma final con un RCA. ¿Cuál será el retardo necesario para este caso? ¿Y para el caso peor?

b) Dibuja cómo serán los circuitos ***carry-skip*** y ***carry select*** para el caso que haya que sumar números de 20 bits utilizando sumadores de 4 bits. ¿Cuáles son las diferencias desde el punto de vista de los circuitos a añadir? ¿Y cuáles son las diferencias desde el punto de vista de los retardos?

c) Expresa el número **-10,98** en coma flotante utilizando un bit para el signo, 7 bits para la mantisa y 4 bits para el exponente. Representa la mantisa de manera normalizada, usando la técnica del bit oculto. El exponente se representa en Exceso 7. Representa el número por medio del inmediato superior, el inmediato inferior, truncamiento y redondeo, calculando el error que se produce en cada uno de los casos.

3.- [1,5 puntos] Contesta las siguientes preguntas:

a) ¿Para qué se utilizan las funciones `Eoi()` e `Iret()`? ¿Cuál de ellas no se utiliza cuando programamos la rutina de atención al reloj? ¿Por qué?

b) Haz un esquema del vector de interrupciones del PC. ¿Para qué se utiliza? ¿Qué se guarda en cada posición? ¿En qué direcciones de memoria está la entrada correspondiente a la rutina de atención al teclado? Escribe el código de la función que cambia el vector de interrupción para atender a un periférico que interrumpe por la línea IRQ4.

*void **CambiarVI4** (unsigned short IPNuevo, unsigned short CSNuevo,*
*unsigned short * IPOld, unsigned short * CSOld);*

c) Explica brevemente para qué sirve hacer *strobe* en el registro de control del teclado. Programa la función **strobe()** que realiza una secuencia de strobe sobre el bit 6 del registro de control de un periférico que interrumpe por la línea IRQ3. El registro de control está en el puerto 0x80.

NOTA: Para hacer este ejercicio, dispones de las siguientes definiciones y rutinas ya implementadas:

*unsigned char **InPort** (puerto);* *void **EnableInts**();*
*void **OutPort** (puerto, valor);* *void **DisableInts**();*

*unsigned char **LeeByteFis** (unsigned short Seg, unsigned short Desp);*
*unsigned short **LeePalFis** (unsigned short Seg, unsigned short Desp);*
*void **EscribeByteFis** (unsigned short Seg, unsigned short Desp, unsigned char valor);*
*void **EscribePalFis** (unsigned short Seg, unsigned short Desp, unsigned short valor);*

4.- [2 puntos] El sistema de memoria de un procesador tiene las siguientes características:

- Direccionamiento al byte con palabras de 4 bytes.
- Memoria virtual paginada de 2 MB, con páginas de 1024 bytes.
- Memoria principal de 128 kB, formada por 2 bancos consecutivos, cada uno de ellos con 4 módulos entrelazados.
- Tiempo de acceso a memoria principal: 10 ciclos (1 ciclo desde el buffer de entrelazado).
- Se utiliza un TLB para la traducción de direcciones, inicialmente vacío, con tiempos de acceso 1 ciclo en caso de acierto y 20 ciclos en caso de fallo. La información de la tabla de páginas es:

Página lógica:	2	4	5	3	6
Pagina física:	20	80	100	15	90

Suponemos que el procesador quiere ejecutar el siguiente programa (en C y ensamblador):

```
for (i=0;i<256;i++)
   C[i]=(A[i]*B[i+1])+(A[i+1]*B[0]);
```

```
        movi r1, #0
        movi r3,#255
        load r21, B
bucle:  load r10, A[r1]
        load r20, B[r1+4]
        mul r10, r10, r20
        load r11,A[r1+4]
        mul r11, r11, r21
        add r5, r10, r11
        store r5, C[r1]
        addi r1, r1, #4
        subi r3, r3, #1
        bge r3, bucle
```

Los datos del programa están cargados de forma que el vector A se almacena a partir de la dirección lógica 4096, B a partir de la 5200 y C a partir de la dirección lógica 6228. Por su parte, las instrucciones están almacenadas a partir de la dirección 3048. Cada instrucción y cada elemento de los vectores ocupan 1 palabra.

Se pide:

a) Haz un esquema de la estructura del sistema de memoria, tanto de la memoria virtual como de la memoria física, indicando claramente la división en bits de las direcciones. ¿Cuántas páginas puede tener un programa? ¿Cuántas páginas tiene la memoria principal? ¿De qué tamaño (en bytes) son los módulos de memoria?

b) Teniendo en cuenta la información de la tabla de páginas, traduce las referencias que genera el programa en la primera pasada por el bucle (incluyendo las 3 instrucciones fuera del bucle) y calcula el tiempo de acceso al sistema de memoria para cada una de esas referencias. Para ello, indica claramente, mediante una tabla, los diferentes pasos que sigues para el cálculo del tiempo de acceso.

c) Calcula el tiempo necesario para traducir las direcciones de todos los elementos de los vectores B y C.

5.- [0,5 puntos] Tenemos un computador cuyo procesador trabaja con un reloj de 500 MHz y queremos calcular la sobrecarga que sufre el procesador al tener que realizar una operación de Entrada/Salida con un disco duro que transfiere los datos en bloques de 128 palabras (siendo cada palabra de 32 bits) y cuya velocidad de funcionamiento es de 32 MB/s. ¿Cuál es la sobrecarga del sistema si la transferencia se realiza por encuesta temporizada, sincronizada a la velocidad de funcionamiento del disco? La rutina de encuesta requiere 300 ciclos para acceder al registro de estado y 200 ciclos para la transferencia del dato

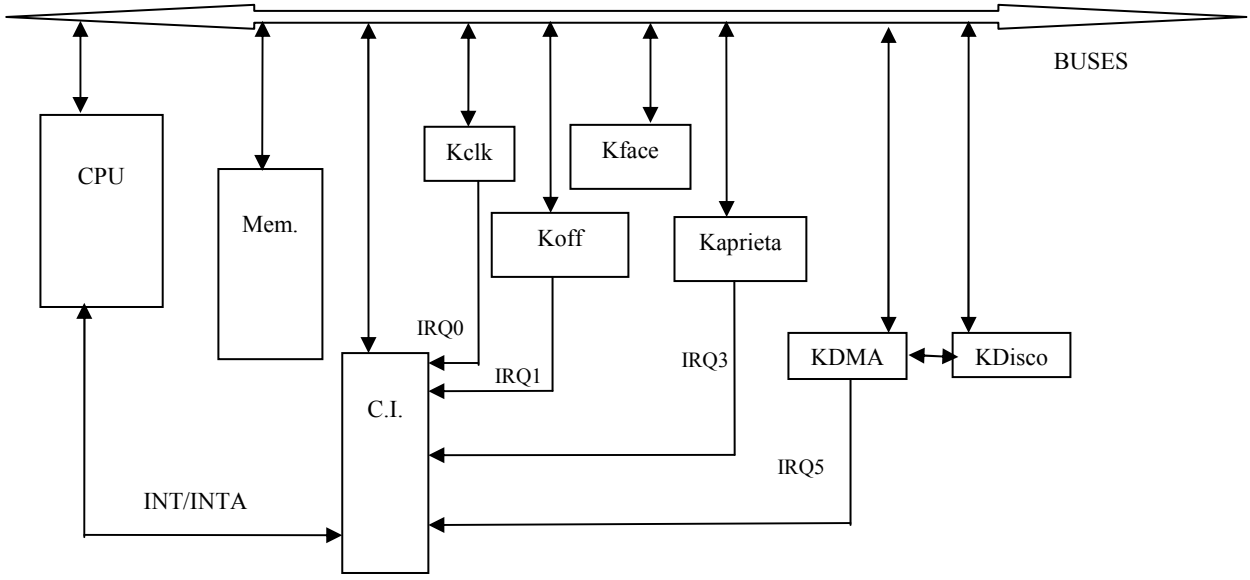
Conviene recordar que la sobrecarga que sufre el procesador en una operación de E/S es el porcentaje de tiempo que le dedica a la operación de E/S frente al tiempo total que tiene.

6.- [1 punto] Responde a las siguientes preguntas sobre buses:

(a) Indica las principales características de un bus de ciclo partido. Haz un esquema de este tipo de protocolo. Si para una transferencia entre dos dispositivos concretos, el ancho de banda alcanzado es de 300 MB/s, ¿cuánto tiempo se necesita para transferir 2 GB?

(b) En un determinado bus de memoria, las líneas de dirección están multiplexadas. Las líneas de dirección son de 32 bits, mientras que las líneas de datos son de 16 bits. Para realizar una transferencia hacen falta 6 ciclos de reloj y la frecuencia de reloj es de 1 GHz. ¿Cuál es el ancho de banda alcanzado?

7.- [2,5 puntos] Se quiere diseñar un **muñeco de peluche emocional**. Estando en marcha, dará respuestas según las personas que estén en su entorno y las acciones que hagan las personas. El hardware del sistema se muestra en la figura.



Los controladores del reloj y de interrupciones son los vistos en clase. A excepción del controlador del **sensor face**, que se trata **por encuesta**, los demás controladores se tratan por interrupción. Las características de los periféricos son los siguientes:

Koff: es el botón para apagar el sistema. Pide interrumpir al apretar el botón o cuando la batería es escasa, para que se apague el sistema tras guardar cierta información (ver comportamiento del sistema). Hay que realizar una secuencia de strobe en su registro de control RCON_KOFF en cada petición de interrupción.

Kface: es el controlador que obtiene las características de la cara que se ve más cercana a los ojos del peluche. El registro de estado REST_KFACE tendrá un 1 cada vez que haya un valor nuevo en el registro de datos. Cuando reciba en el registro de datos RDAT_KFACE un valor impar supondrá que esa cara está enfadada o, en caso contrario, que está contenta. En su registro de control RCON_KFACE hay que hacer una secuencia de strobe cada vez que se lea del registro de datos de este controlador. Como ya se ha indicado, la **sincronización con este periférico es por encuesta**.

Kaprieta: cada vez que alguien achuche el peluche solicita interrumpir. Hay que hacer una secuencia de strobe en su registro de control RCON_KAPRIETA cuando se admita una interrupción.

KDMA: Es el controlador de DMA y se utiliza para hacer transferencias de memoria a disco. Posee los siguientes registros:

- **registro de control** (RCON_KDMA): cuando se escribe un 1 en él se inicia la transferencia (se pone a 0 automáticamente).
- **registro de dirección** (RDIR_KDMA): dirección de comienzo del bloque a transferir.
- **registro de longitud** (RLON_KDMA): contiene la longitud en bytes del bloque a transferir.
- **registro de estado** (REST_KDMA): cuando su contenido es un 3 indica que la última operación realizada ha sido errónea.

Kdisco: Es el controlador de disco. Para realizar una transferencia hay que programar este controlador, para lo que tenemos la rutina *ProgramarKDisco()*.

El comportamiento del peluche es el siguiente cuando el sistema está en marcha. Si alguien lo achucha, estará vibrando durante 10 segundos. Para ello existen las rutinas ***vibrando()***, que provoca las vibraciones, y ***parar()***, para parar las vibraciones. Si a lo largo de esos 10 segundos, se vuelve a achuchar el peluche, hay que contar nuevamente desde entonces los 10 segundos. Para facilitar la resolución del problema, supondremos que no ocurre nada al volver a llamar a ***vibrando()*** mientras esté vibrando ni tampoco pasará nada si se llama a ***parar()*** y el peluche está parado. Mientras esté vibrando, no se pueden detectar las caras del entorno.

Para detectar las caras, por tanto, el sistema tiene que estar en marcha pero sin vibrar. Los valores que se reciben por medio del controlador *Kface* se guardan en memoria a partir de la dirección DIRINI. Cada valor ocupa 2 bytes y se guardan en memoria por medio de la rutina ***guarda(dirección,valor)*** en la dirección que se indica como primer parámetro. Además de eso, puede suceder que se interprete que la cara detectada esté enfadada (cuando el valor recibido sea impar), o que esté alegre (cuando el valor recibido sea par). Al detectar rostros enfadados, pondrá en marcha una pieza relajante de música clásica por medio de la rutina ***clasica()***; en caso contrario, si detecta un rostro alegre, cantará una canción alegre por medio de la rutina ***canta()***. En cualquier caso, el sonido se emite a lo sumo durante un minuto tras el cual se silencia por medio de la rutina ***calla()***. A lo largo de ese minuto no se detectarán nuevas caras. Si se le achucha mientras emite sonidos, habrá que parar el sonido y se pondrá a vibrar el peluche a lo sumo 10 segundos.

De cualquier manera, si se pulsa el botón de apagado o el sistema detecta que tiene poca batería (controlador *Koff*) hay que copiar a disco por medio de DMA la información guardada en memoria y no se detectarán ni las caras ni los achuchones al peluche. Si ocurre algún error al copiar a disco, se reintentará una y otra vez. Al final, cuando la transferencia finalice de forma correcta, el sistema se apagará completamente de forma automática.

Se pide: Escribe en lenguaje algorítmico todas las rutinas de atención que consideres necesarias, así como el programa principal. Comenta cualquier supuesto que hagas para la resolución del problema. Se valorará representar el comportamiento del sistema mediante un autómata de estados y transiciones.