

Estructura y Tecnología de Computadores

Examen final

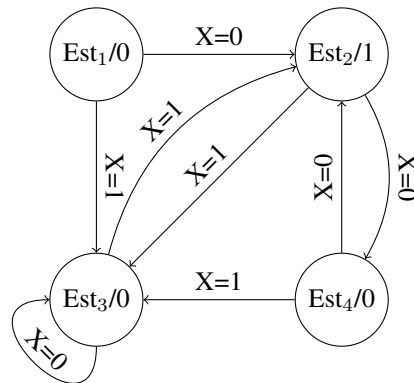
3 de junio de 2011

Grupo:	DNI:
Nombre:	Apellidos:

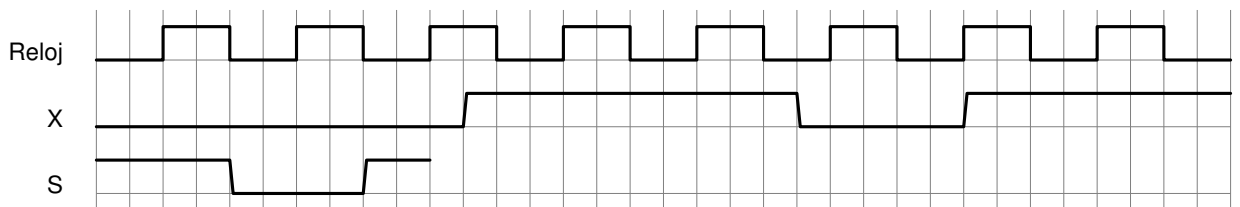
Instrucciones:

- Cada pregunta se calificará con una nota entre 0 y 10. **Sólo en dos de las cuatro preguntas se podrá obtener una nota inferior a 3.** Si no se cumple esta condición, el examen estará suspenso.
- Las preguntas tienen pesos distintos en la nota final de teoría, la cuál se calculará como la media ponderada de las notas obtenidas. Estos pesos, para las preguntas 1, 2, 3 y 4, son, respectivamente, 30 %, 25 %, 30 % y 15 %.
- La nota obtenida en el examen representará el 50 % de la nota final de la asignatura, siendo necesario obtener una calificación mayor o igual que 5 para eliminar esta parte.
- No se dará por válida ninguna respuesta que no contenga el desarrollo de la solución paso a paso.
- Responda a las preguntas en bolígrafo.

1. (10 puntos) El siguiente autómata especifica un sistema secuencial síncrono con una entrada (X) y un bit de salida (S):



- (7 puntos) Diseñe y dibuje un circuito secuencial síncrono correspondiente al autómata anterior utilizando flip-flops T disparados por flanco descendente.
- (3 puntos) Complete el cronograma siguiente basándose en la especificación del circuito.



2. (10 puntos) Utilizando la metodología descrita en los apuntes para la inclusión de nuevas instrucciones en el esquema de implementación multiciclo, realice las fases de análisis y diseño de la siguiente instrucción:

pop \$a: realiza la operación contraria a push: almacena el elemento que se encuentra en la cima de la pila (apuntado por \$sp) en \$x e incrementa \$sp en 4.

Para que la solución del ejercicio se considere correcta, es importante que incluya claramente todos los pasos correspondientes a las fases de análisis y diseño. Las modificaciones al camino de datos y la unidad de control deberán estar explicadas por escrito. De forma complementaria, se pueden realizar modificaciones y anotaciones sobre los diagramas originales para facilitar la explicación.

```

##### SEGMENTO DE DATOS #####
        .data
almacen: .space 96
        . . .
##### SEGMENTO DE TEXTO #####
        .text
        li    $s0, 2
        li    $s1, 0
        la    $s2, almacen
        lw    $t0, 0($s2)
        add   $s1, $s1, $t0

```

Figura 1: Programa MIPS para el ejercicio 3

3. (10 puntos) La máquina M posee un sistema de memoria con las siguientes características:

- Tabla de páginas de 512 entradas.
- Memoria física de 256 bytes.
- Tamaño de página virtual/página física de 8 bytes.
- TLB asociativo de 4 conjuntos de 2 vías (cada uno) con una estrategia de reemplazo LRU y una política de postescritura.
- Memoria caché de correspondencia directa combinada para datos e instrucciones con capacidad para 8 palabras (palabras de 32 bits), bloques de 2 palabras y política de postescritura.

Se pide:

- a) (1.5 puntos) Especificar detalladamente el formato de la dirección física y de la dirección virtual. Dibujar un esquema de la tabla de páginas y calcular su tamaño, incluyendo los bits de control necesarios (justificando su inclusión).
 - b) (1.5 puntos) Dibujar un esquema detallado del TLB y de la caché. Calcular sus tamaños incluyendo todos los bits de control necesarios (justificando su inclusión).
 - c) (6 puntos) Hacer un seguimiento del estado del TLB, la tabla de páginas y la caché en la ejecución del programa mostrado en la figura 1, teniendo en cuenta que:
 - La dirección virtual de comienzo del segmento de datos es 0_{10} .
 - La dirección virtual de comienzo del segmento de código es 96_{10} .
 - Los bits de validez de la tabla de páginas están puestos a 1 en las posiciones 0 y 12, siendo los números de página física que ocupan esas posiciones 16_{10} y 20_{10} respectivamente. La página física 21 y siguientes están todas libres.
 - Inicialmente, la caché y el TLB están vacíos.
 - d) (1 punto) Suponiendo que el procesador está dedicado en exclusiva a la ejecución de nuestra aplicación y que son 20 los ciclos necesarios para acceder a memoria principal y 2000 los necesarios para acceder a memoria secundaria, ¿cuántos ciclos estará parado el procesador sin realizar ninguna tarea debido a los distintos fallos de la caché, el TLB y la tabla de páginas? (En los 2000 ciclos de acceso a memoria secundaria se incluye el tiempo necesario para llevar la página desde memoria secundaria a memoria principal y la posterior actualización de la tabla de páginas y del TLB).
4. (10 puntos) Un sistema tiene una tarjeta de sonido capaz de generar sonido con calidad de CD, es decir, 44.100 muestras por segundo, donde en cada muestra se envían 32 bits (16 bits por cada uno de los dos canales de sonido que forman una comunicación en estéreo). Disponemos de un Pentium I a 133 Mhz. Se pretende usar este sistema para reproducir ficheros de sonido en formato WAV. Para ello, se leen muestras de sonido del disco duro y se envían a la tarjeta de sonido:
- ETAPA 1: Se leen muestras de sonido en bloques de 4KB del disco duro. Para ello, se usa un controlador DMA que requiere 2000 ciclos de inicialización y 1000 ciclos al terminar la transferencia.
 - ETAPA 2: Procesamiento de los datos en la CPU. Para el formato WAV el procesamiento requiere 0 ciclos de CPU.
 - ETAPA 3: Para reproducir el sonido, el procedimiento para enviar las muestras a la tarjeta de sonido puede ser por uno de estos dos métodos diferentes:
 - Por interrupciones (44.100 interrupciones por segundo), donde en cada interrupción, que requiere 100 ciclos de CPU, se envía a la tarjeta una muestra de 32 bits.
 - Por DMA, donde cada transferencia de 16KB requiere un procesamiento de 2000 ciclos de inicialización más 1000 ciclos al terminar.
- a) (3 puntos) Calcular el % de CPU consumido en esta tarea (las 3 etapas), usando interrupciones en la ETAPA 3.
 - b) (3 puntos) Calcular el % de CPU consumido en esta tarea (las 3 etapas), usando DMA en la ETAPA 3.
 - c) (4 puntos) Calcular el % de CPU consumido en esta tarea (las 3 etapas), si se reproducen ficheros MP3 (formato que comprime WAV a un 10 % del tamaño original), sabiendo que el procesamiento para descomprimir los datos en la CPU de un bloque de 4KB de MP3 requiere 20.000.000 ciclos en la etapa 2 (calcular el % considerando los dos métodos posibles en la ETAPA 3).

Soluciones

Ejercicio 1

Apartado a

Dado que tenemos 4 estados, necesitaremos dos flip-flops para codificarlos (Q_0 y Q_1). Utilizaremos la siguiente codificación para los estados:

Estado	Q_0	Q_1
Est ₁	0	0
Est ₂	0	1
Est ₃	1	0
Est ₄	1	1

Por tanto, la tabla de verdad de la función de salida (S) será:

Q_0	Q_1	S
0	0	0
0	1	1
1	0	0
1	1	0

Y la tabla de la función de transición (Q_0^* y Q_1^*) y las entradas de los dos biestables (T_0 y T_1) será:

Q_0	Q_1	X	Q_0^*	Q_1^*	T_0	T_1
0	0	0	0	1	0	1
0	0	1	1	0	1	0
0	1	0	1	1	1	0
0	1	1	1	0	1	1
1	0	0	1	0	0	0
1	0	1	0	1	1	1
1	1	0	0	1	1	0
1	1	1	1	0	0	1

La expresión simplificada de la función de salida será:

$$S = \overline{Q_0}Q_1$$

La simplificación de la función T_0 es (otras soluciones también son posibles):

		Q_0Q_1			
		00	01	11	10
X	0	0	1	1	0
	1	1	1	0	1

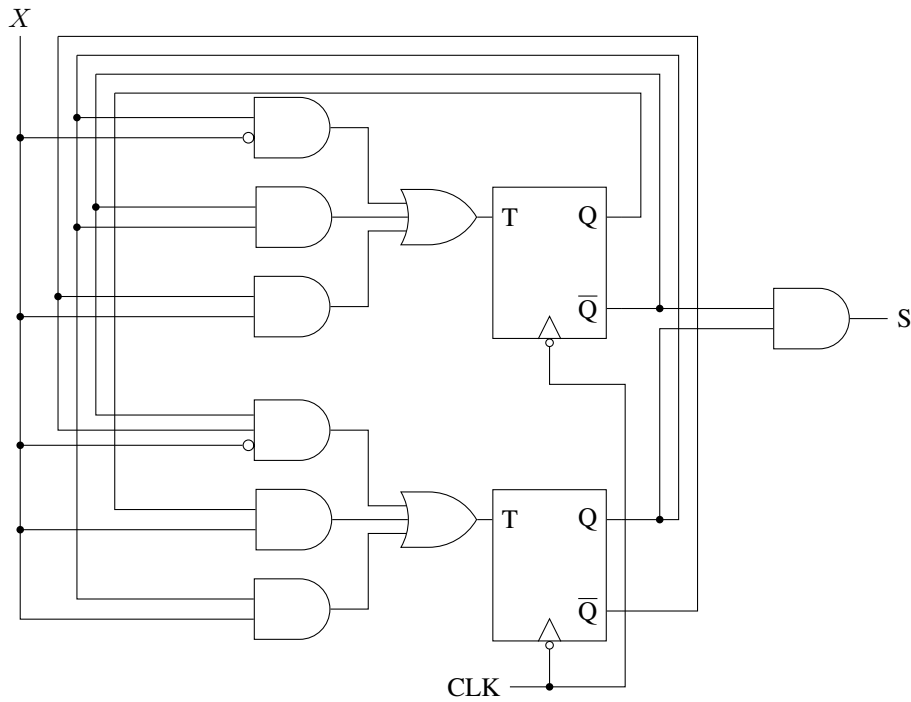
$$T_0 = Q_1\overline{X} + \overline{Q_0}Q_1 + \overline{Q_1}X$$

Y la de T_1 :

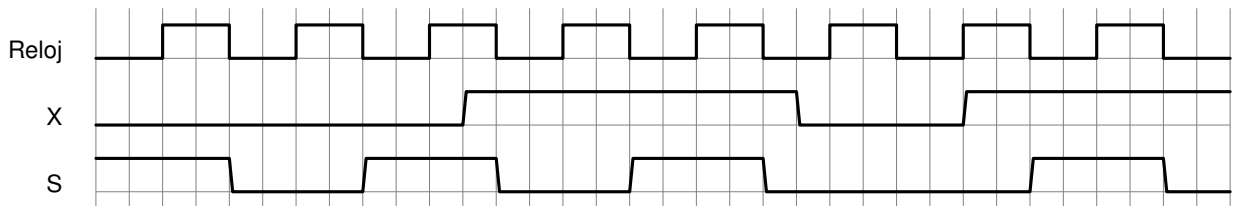
		Q_0Q_1			
		00	01	11	10
X	0	1 ₀	0 ₂	0 ₆	0 ₄
	1	0 ₁	1 ₃	1 ₇	1 ₅

$$T_1 = \overline{Q_0} \overline{Q_1} \overline{X} + Q_0 X + Q_1 X$$

Con lo que el circuito resultante usando puertas AND, OR y NOT será:



Apartado b



Ejercicio 2

■ Análisis:

1. Especificación precisa de la semántica de la instrucción:

La semántica de “pop \$a” es:

$$\begin{aligned} \$a &\leftarrow \text{Mem}[\$29] \\ \$29 &\leftarrow \$29 + 4 \end{aligned}$$

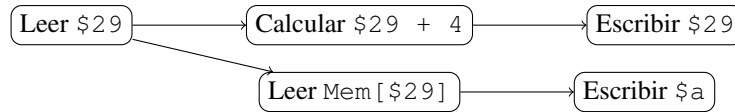
(suponiendo que ambas operaciones se realizan secuencialmente)

2. Identificación del trabajo a realizar por cada unidad funcional principal:

- Banco de registros:
 - Leer \$29.
 - Escribir \$29.
 - Escribir \$a.
- ALU:

- Calcular $\$29 + 4$.
- Memoria:
 - Leer Mem[$\$29$].

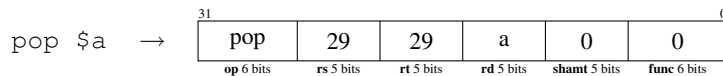
3. Establecimiento del orden de precedencia entre las distintas tareas a realizar:



■ Diseño:

1. **Definición de la codificación de la instrucción:** lo mejor es usar el formato R, ya que nos permite indicar, usando los multiplexores de manera adecuada, un registro para leer (campo **rs**), un registro para escribir (campo **rd**) y un registro que se puede leer y escribir, según nos convenga (campo **rt**). En nuestro caso, necesitamos leer un registro, $\$29$, y escribir dos registros, $\$29$ y $\$a$. También nos interesa poder calcular cómodamente $\$29 + 4$.

Usaremos, por tanto, la siguiente distribución de campos:



2. División del trabajo en ciclos:

Ciclo 1	Ciclo 2	Ciclo 3	Ciclo 4	Ciclo 5
IR ← Memoria[PC]	A ← Reg[29]	MDR ← Memoria[A]	ALUOut ← A + 4	Reg[29] ← ALUOut
PC ← PC + 4			Reg[a] ← MDR	

3. Extensión del camino de datos:

Para realizar algunas de las acciones especificadas en la tabla anterior, necesitamos modificar el camino de datos de la siguiente manera:

MDR ← Memoria[A]: añadir una nueva entrada al multiplexor controlado por la señal de control IoD que esté conectada al registro A (en el que tendremos el contenido de $\$29$). La señal IoD pasará a ser una señal de dos bits en lugar de uno, y la combinación “10” seleccionará la nueva entrada.

4. Extensión del control:

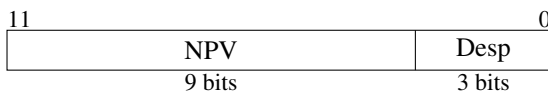
Habrá que añadir 3 nuevos estados al autómata de control original. El resultado se muestra en la figura 2.

Ejercicio 3

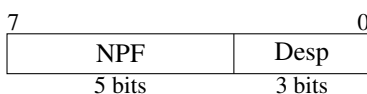
Apartado a

Como la tabla de páginas tiene 512 entradas, eso es porque hay 512 páginas virtuales, luego el campo NPV de la dirección virtual necesitará $\log_2 512 = 9$ bits.

Como las páginas tienen un tamaño de 8 bytes, eso significa que el campo de desplazamiento de la dirección virtual necesita $\log_2 8 = 3$ bits. La dirección virtual, por lo tanto, tiene un tamaño total de $9 + 3 = 12$ bits y queda formada de la siguiente manera:



En el caso de la dirección física, sabemos que la memoria principal tiene un tamaño total de 256 bytes, por lo que la dirección física tiene un tamaño total de $\log_2 256 = 8$ bits. De esos, 3 bits constituyen el desplazamiento y el resto el NPF. La dirección física queda, pues, de la siguiente manera:



Cada entrada de la tabla de páginas necesitará un bit de modificación (M) para permitir la política de postescritura y un bit de uso (U) para permitir la política de reemplazo LRU, además del bit de validez (V). El NPF necesita 5 bits, por lo que cada entrada de la tabla de páginas tiene 8 bits en total. La tabla de páginas queda así:

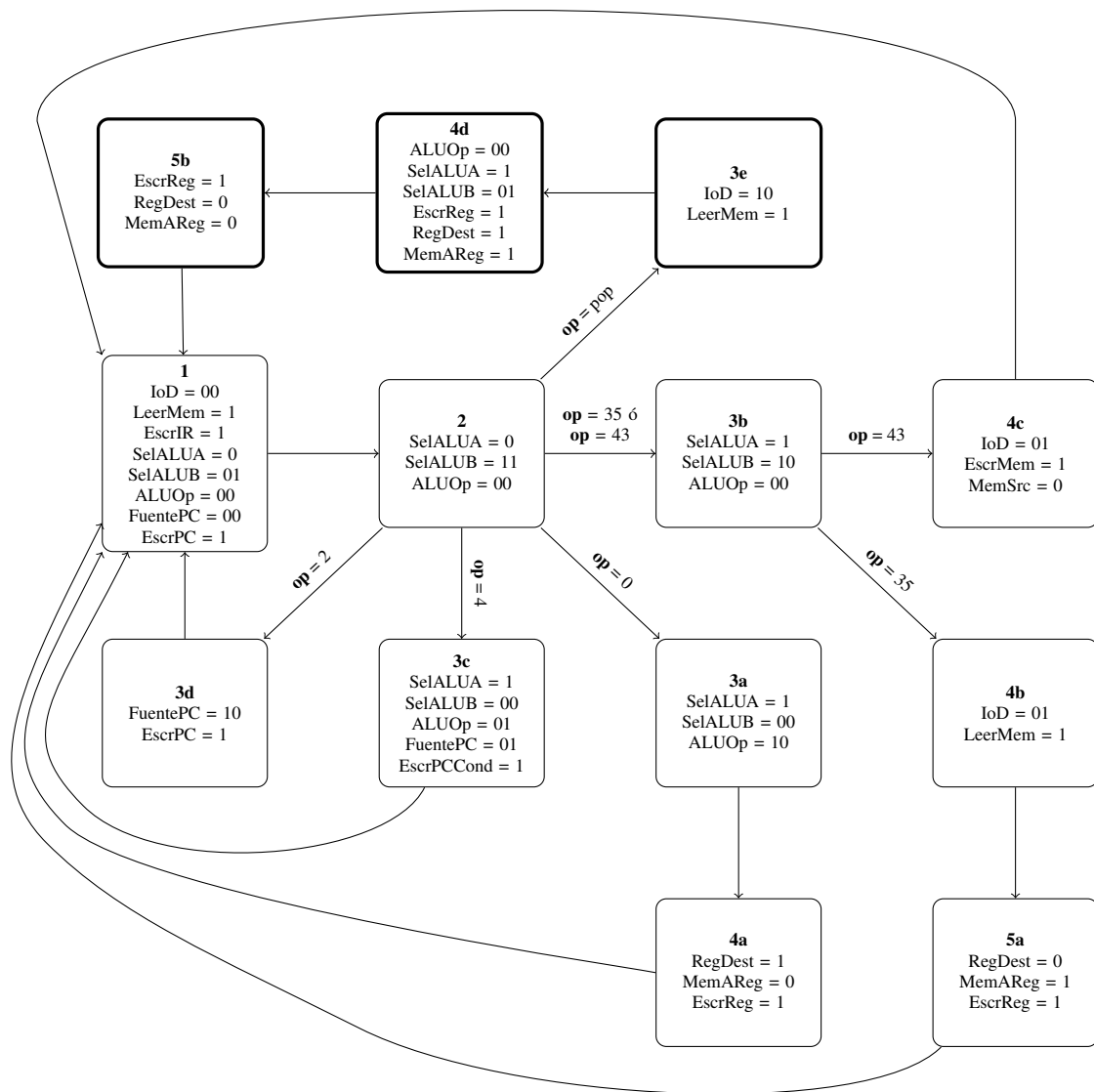


Figura 2: Modificaciones al autómata de control para incluir la instrucción pushm.

	V	M	U	NPF (5 bits)
0	1	0	0	16
1	0	0	-	-
2	0	0	-	-
3	0	0	-	-
4	0	0	-	-
5	0	0	-	-
6	0	0	-	-
7	0	0	-	-
8	0	-	-	-
9	0	-	-	-
10	0	-	-	-
11	0	-	-	-
12	1	0	0	20
13	0	-	-	-
...
511	0	-	-	-

La tabla de páginas ocuparía 512 bytes en memoria (no cabría en la memoria del sistema propuesto).

Apartado b

Debido a que el TLB tiene 4 conjuntos, los 2 bits de menor peso del NPV se utilizarán para elegir el conjunto en el que almacenar la información de una entrada de la tabla de páginas. Los 7 bits restantes del NPV formarán la etiqueta.

Cada conjunto tiene 2 vías. Cada vía necesitará un bit de modificación (M) para permitir la política de postescritura, un bit de uso (U) porque el TLB sigue una política de reemplazo LRU, y un bit de validez (V) para saber cuándo la vía contiene información útil.

Además de esos bits, cada vía contendrá también los bits M y U de la entrada correspondiente de la tabla de páginas (que mostraremos como M' y U', respectivamente) y el campo NPF con el correspondiente número de página física. El TLB queda, pues, de la siguiente forma:

Vía 0							Vía 1							
	V	M	U	Etiqueta (7 bits)	M'	U'	NPF (5 bits)	V	M	U	Etiqueta (7 bits)	M'	U'	NPF (5 bits)
0	0	-	-	-	-	-	-	0	-	-	-	-	-	-
1	0	-	-	-	-	-	-	0	-	-	-	-	-	-
2	0	-	-	-	-	-	-	0	-	-	-	-	-	-
3	0	-	-	-	-	-	-	0	-	-	-	-	-	-

Por tanto, el tamaño total del TLB será de $4 \times 2 \times (1 + 1 + 1 + 7 + 1 + 1 + 5) = 136$ bits.

Cada bloque de caché almacena 8 bytes (2 palabras de 4 bytes), por lo que la caché almacena un total de 4 bloques (32 bytes/8 bytes/bloque). Cada conjunto almacena un bloque (por ser de correspondencia directa), por lo que hay 4 conjuntos en la caché.

Por tanto, son necesarios $\log_2 4 = 2$ bits para el campo del índice. Estos bits serán los cuarto y quinto de menor peso, porque los 3 de menor peso indicarán el desplazamiento de palabra ($\log_2 2 = 1$) y el desplazamiento de byte ($\log_2 4 = 2$) dentro de un bloque. Los 3 bits restantes de la dirección física se utilizarán para la etiqueta.

En resumen, todo esto hace que una dirección física se descomponga de la siguiente manera para acceder a la caché:

3 bits	2 bits	1 bit	2 bits
Etiqueta	Índice	Desp. palabra	Desp. byte

La caché queda de la siguiente manera:

	V	M	Etiqueta (3 bits)	Bloque (8 bytes)
0	0	-	-	-
1	0	-	-	-
2	0	-	-	-
3	0	-	-	-

El bit de validez (V) indica si la entrada contiene datos válidos o no, mientras que el bit de modificación (M) se ha incluido al seguir la caché una política de postescritura para saber si una determinada entrada ha sido modificada o no.

El tamaño total de la caché es, pues, de $4 \times (1 + 1 + 3 + 8 \times 8) = 276$ bits.

Apartado c

Los accesos que realiza el programa de la figura 1 son los siguientes:

- Lectura de instrucción de la dirección 96.

Instrucción: li \$s0, 2.

NPV: 12.

Desplazamiento: 0.

Conjunto de TLB: 0.

Etiqueta de TLB: 3.

Acierto/fallo de TLB: Fallo (el TLB está inicialmente vacío).

Acierto/fallo de página: Acierto.

NPF: 20.

Dirección física resultante: 160.

Índice del conjunto de caché: 0.

Etiqueta de caché: 5.

Acierto/fallo de caché: Fallo.

Vía 0						
V	M	U	Etiqueta (7 bits)	M'	U'	NPF (5 bits)
0	1	1	3	0	1	20

V	M	U	NPF (5 bits)
1	0	0	20

V	M	Etiqueta (3 bits)	Bloque (8 bytes)
0	1	5	li... li...

- Lectura de instrucción de la dirección 100.

Instrucción: li \$s1, 0.

NPV: 12.

Desplazamiento: 4.

Conjunto de TLB: 0.

Etiqueta de TLB: 3.

Acierto/fallo de TLB: Acierto.

Acierto/fallo de página: No se accede.

NPF: 20.

Dirección física resultante: 164.

Índice del conjunto de caché: 0.

Etiqueta de caché: 5.

Acierto/fallo de caché: Acierto.

Vía 0						
V	M	U	Etiqueta (7 bits)	M'	U'	NPF (5 bits)
0	1	1	3	0	1	20

V	M	U	NPF (5 bits)
1	0	0	20

V	M	Etiqueta (3 bits)	Bloque (8 bytes)
0	1	5	li... li...

- Lectura de instrucción de la dirección 104.

Instrucción: la \$s2, almacen.

NPV: 13.

Desplazamiento: 0.

Conjunto de TLB: 1.

Etiqueta de TLB: 3.

Acierto/fallo de TLB: Fallo.

Acierto/fallo de página: Fallo.

NPF: 21 (primera página física libre).

Dirección física resultante: 168.

Índice del conjunto de caché: 1.

Etiqueta de caché: 5.

Acierto/fallo de caché: Fallo.

Vía 0						
V	M	U	Etiqueta (7 bits)	M'	U'	NPF (5 bits)
1	1	1	3	0	1	21

V	M	U	NPF (5 bits)
1	0	0	21

V	M	Etiqueta (3 bits)	Bloque (8 bytes)
1	0	5	1a... 1w...

- Lectura de instrucción de la dirección 108.

Instrucción: lw \$t0, 0(\$s2).

NPV: 13.

Desplazamiento: 4.

Conjunto de TLB: 1.

Etiqueta de TLB: 3.

Acierto/fallo de TLB: Acierto.

Acierto/fallo de página: No se accede.

NPF: 21.

Dirección física resultante: 172.

Índice del conjunto de caché: 1.

Etiqueta de caché: 5.

Acierto/fallo de caché: Acierto.

Vía 0						
V	M	U	Etiqueta (7 bits)	M'	U'	NPF (5 bits)
1	1	1	3	0	1	21

V	M	U	NPF (5 bits)
1	0	0	21

V	M	Etiqueta (3 bits)	Bloque (8 bytes)
1	0	5	1a... 1w...

- Lectura del dato de la dirección 0

Instrucción: lw \$t0, 0(\$s2).

NPV: 0.

Desplazamiento: 0.

Conjunto de TLB: 0.

Etiqueta de TLB: 0.

Acierto/fallo de TLB: Fallo.

Acierto/fallo de página: Acierto.

NPF: 16.

Dirección física resultante: 128.

Índice del conjunto de caché: 0.

Etiqueta de caché: 4.

Acierto/fallo de caché: Fallo.

Vía 1						
V	M	U	Etiqueta (7 bits)	M'	U'	NPF (5 bits)
0	1	1	0	0	1	16

Además, en la vía 0 del mismo conjunto, se pone el bit U a 0.

V	M	U	NPF (5 bits)
0	1	0	16

Entrada de caché: 0	V	M	Etiqueta (3 bits)	Bloque (8 bytes)
	1	0	4	Mem[0] Mem[4]

- Lectura de instrucción de la dirección 112.

Instrucción: add \$s1, \$s1, \$t0.

NPV: 14.

Desplazamiento: 0.

Conjunto de TLB: 2.

Etiqueta de TLB: 3.

Acierto/fallo de TLB: Fallo.

Acierto/fallo de página: Fallo.

NPF: 22 (segunda página libre).

Dirección física resultante: 176.

Índice del conjunto de caché: 2.

Etiqueta de caché: 5.

Acierto/fallo de caché: Fallo.

Entrada de TLB: 2	Vía 0						
	V	M	U	Etiqueta (7 bits)	M'	U'	NPF (5 bits)
	1	1	1	3	0	1	22

Entrada de tabla de páginas: 14	V	M	U	NPF (5 bits)
	1	0	0	22

Entrada de caché: 2	V	M	Etiqueta (3 bits)	Bloque (8 bytes)
	1	0	5	add... ..

Nota: aunque las instrucciones `li` y `la` son pseudoinstrucciones, las hemos considerado como instrucciones reales a la hora de indicar el contenido de la caché para que así quede más claro qué es lo que hay en cada bloque de cache.

Apartado d

Hay 5 instrucciones, que producen 4 fallos de TLB, 2 fallos de página y 4 fallos de caché.

Cada fallo de TLB requerirá un acceso a la memoria para comprobar la tabla de páginas (20 ciclos), cada fallo de páginas requerirá 2000 ciclos para leer la página del disco y llevarla a memoria principal y cada fallo de caché requerirá 20 ciclos (soponiendo siempre que no es necesario realizar la postescritura).

Por tanto, el procesador permanecerá parado $4 \times 20 + 2 \times 2000 + 4 \times 20 = 4160$ ciclos.

Ejercicio 4

Apartado a

En todo momento es preciso ir leyendo el fichero WAV desde el disco duro a un ritmo de 44100 muestras/seg, es decir, $44100 \times 4 = 176400$ bytes/seg. El número de bloques de 4 KB (etapa 1) es $\frac{176400}{4096} = 43.07$ bloques/seg.

Por tanto,

- Etapa 1: $43.07 \times (2000 + 1000) = 129210$ ciclos/seg.
- Etapa 2: 0 ciclos.
- Etapa 3: 44100 interrupciones/seg, esto es, $44100 \times 100 = 4410000$ ciclos/seg.

En total, entre las 3 etapas en cada segundo la tarea consume $129210 + 0 + 4410000 = 4539210$ ciclos. Como la CPU trabaja a un ritmo de 133×10^6 ciclos/seg, el % de CPU consumido en las 3 etapas es $\frac{4539210}{133 \times 10^6} = 3.41 \%$.

Apartado b

- Etapa 1: 129210 ciclos (como en el apartado anterior; esto no cambia).
- Etapa 2: 0 ciclos (tampoco cambia).
- Etapa 3: Hay que enviar 176400 bytes/seg; esto, medido en bloques de 16 KB, son $\frac{176400}{16 \times 1024} = 10.77$ bloques. En ciclos: $10.77 \times (2000 + 1000) = 32310$ ciclos.

En total, $\frac{129210+0+32310}{133 \times 10^6} = 0.12 \%$.

Apartado c

- Etapa 1: En el MP3 la compresión es de un 10 % en relación al tamaño del WAV; por tanto, ahora el ritmo de lectura del disco ya no debe adaptarse a las 44100 muestras/seg, sino a $44100 \times 0.10 = 4410$ muestras/seg, que son $4410 \times 4 = 17640$ bytes/seg, es decir, $\frac{17640}{4096} = 4.31$ bloques de 4 KB/seg. Esto, en ciclos, es: $4.31 \times (2000 + 1000) = 12930$ ciclos/seg.
- Etapa 2: Cada bloque de 4 KB de MP3 son 20000000 ciclos. Como a la CPU le llegan 4.31 bloques/seg, el número de ciclos consumido en esta segunda etapa es $20000000 \times 4.31 = 86200000$.
- Etapa 3:
 - Con interrupciones: 4410000 ciclos/seg (ver apartado a). % CPU = $\frac{12930+86200000+4410000}{133 \times 10^6} = 68.14 \%$.
 - Con DMA: 32310 ciclos/seg (ver apartado b). % CPU = $\frac{12930+86200000+32310}{133 \times 10^6} = 64.85 \%$.