ESTRUCTURA DE COMPUTADORES

15 de enero de 2016

APELLIDOS: _____ **NOMBRE:**_____

1. **[0.5p]** Supón que el ordenador A tiene una tasa SPECfp2000 de 1200 y el ordenador B una tasa SPECfp2000 de 1800 para el mismo programa de prueba, entonces ¿cuál es más rápido y por cuánto? Razona la respuesta basándote en la definición de tasa SPEC.

   Como la tasa SPEC es la relación entre el tiempo de ejecución en la máquina evaluada con respecto al tiempo de ejecución en una máquina de referencia ($SPEC_x = T_{ref}/T_x$) entonces: $SPEC_B/SPEC_A = T_A/T_B = R_B/R_A$.

   La máquina B es 1800/1200=1,5 veces más rápida que la máquina A.

2. Los siguientes fragmentos de código se ejecutarán en un procesador segmentado de 5 etapas como el MIPS estudiado en clase: IF, ID, EX, MEM e WB. En la etapa de ejecución, una instrucción puede utilizar una ALU para sumas/restas con enteros y operaciones lógicas, o una unidad de multiplicación en punto flotante con latencia 4, o una de suma en punto flotante de latencia 2. Las unidades en punto flotante son segmentadas. El salto se decide en la etapa ID y el procesador usa la técnica de salto fijo no efectivo y tiene una unidad de anticipación en la etapa EX.

   a) **[0.5p]** ¿Existe algún riesgo en la siguiente secuencia? ¿De qué tipo? Dibuja el diagrama en múltiples ciclos para la ejecución de estas instrucciones, señalando explícitamente (si existen) las anticipaciones y los bloqueos.

   ```
   add $1, $2, $3
   bne $1, $2, loop
   ```

   El diagrama multiciclo sería de la siguiente manera:

	C1	C2	C3	C4	C5	C6	C7	C8
add $1, $2, $3	IF	ID	EX	MEM	WB			
bne $1, $2, Loop		IF	ID	♡	♡	EX	MEM	WB

   Como el salto se decide en la etapa ID, es necesario bloquear la bne hasta que el dato necesario ($1) se haya escrito en la etapa WB de la instrucción anterior. No hay anticipación. Esto es un riesgo de tipo RAW.

   b) **[0.5p]** ¿Existe algún riesgo en la siguiente secuencia? ¿De qué tipo? Dibuja el diagrama en múltiples ciclos para la ejecución de estas instrucciones, señalando explícitamente (si existen) las anticipaciones y los bloqueos.

   ```
   lwc1 $f0, 0($2)
   add.s $f2, $f4, $f0
   ```

   El diagrama multiciclo sería de la siguiente manera:

	C1	C2	C3	C4	C5	C6	C7	C8
lwc1 $f0, 0($2)	IF	ID	EX	MEM	WB $\downarrow \$f0$			
add.s $f2, $f4, $f0		IF	ID	♡	EX1	EX2	MEM	WB

Como la instrucción lwc1 no obtiene el dato hasta finalizar la etapa MEM, es necesario bloquear la add.s hasta que el dato necesario ($f0) se le pueda pasar a la etapa EX por anticipación.
Esto es un riesgo de tipo RAW.

c) [**0.75p**] ¿Existe algún riesgo en la siguiente secuencia? ¿De qué tipo? Dibuja el diagrama en múltiples ciclos para la ejecución de estas instrucciones, señalando explícitamente (si existen) las anticipaciones y los bloqueos.

```
add.d $f4, $f6, $f8
mul.d $f6, $f4, $f0
add.d $f6, $f4, $f0
```

El diagrama multiciclo sería de la siguiente manera:

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
add.d $f4, $f6, $f8	IF	ID	EX1	EX2	MEM $\downarrow \$f4$	WB					
mul.d $f6, $f4, $f0		IF	ID	♡	EX1	EX2	EX3	EX4	MEM	WB	
add.d $f6, $f4, $f0			IF	♡	ID	♡	♡	EX1	EX2	MEM	WB

Como la primera add.d no obtiene el dato hasta finalizar las etapas EX, es necesario bloquear la mul.d hasta que el dato necesario ($f4) se le pueda pasar a la etapa EX por anticipación. Este riesgo es de tipo RAW.
Como la segunda add.d acabaría antes que la mul.d, y ambas comparten el mismo registro destino, es necesario bloquear esta add.d para retrasar su etapa de WB y acabar en el orden correcto. Como la unidad de detección de riesgos está en la etapa ID (igual que en el camino de datos visto en clase) los bloqueos se producen en la etapa ID. Esto es un riesgo de tipo WAW.

3. [**0.5p**] Si el siguiente código se ejecutase en un procesador que implementase salto retardado, indica <u>todas</u> las instrucciones que podrías mover al hueco de retardo (indicando qué técnica usas en cada caso). Además, de todas esas opciones, indica cuál sería la que finalmente escogerías y por qué.

```
      add $t2, $0, $0
loop: lw $t0, 0($s2)
      add $t1, $t0, $t0
      add $t2, $t2, $t1
      addi $s2, $s2, 4
      bne $s2, $s3, loop
      [hueco]
      sw $t2, 0($s3)
```

Técnica "desde antes": se podría mover la add $t2,$t2,$t1 pues no tiene dependencias con la instrucción de salto
Técnica "desde después": se podrían mover la lw $t0,0($s2), si se replica antes de la etiqueta loop; o la addi $s2, $s2, 4, si se replica también antes de la etiqueta loop y se cambia el desplazamiento de la lw por un -4.
Técnica "desde instrucciones siguientes": se puede mover la sw $t2,0($s3) porque no tiene dependencias con las instrucciones del bucle, y siempre se escribe en la misma posición de memoria (la última vez quedando el valor correcto).

La mejor solución es la primera, pues al ser una instrucción que tenía que ejecutarse antes del salto, hace que el hueco de retardo sea siempre útil, con lo que nunca se perderá ese ciclo.

4. [**0.5p**] En una memoria física de 4GB entrelazada la dirección física 0xFFFFFF30 se encuentra ubicada en en el módulo 7 (los módulos se numeran empezando en el 0). Indica cuántos módulos hay y si el entrelazamiento es de orden superior o inferior.

   La única forma de que se puede determinar que el dato está en el módulo 7 es cogiendo los 3 bits más significativos de la dirección. Ello implica que hay $2^3 = 8$ módulos y que el entrelazamiento es de orden superior.

5. [**1.5p**] Un sistema tiene 4GB de memoria física y una memoria caché de 64KB de correspondencia directa, dividida en líneas de 16 bytes. Considera que las variables escalares están permanentemente almacenadas en registros del procesador y que por lo tanto no producen accesos a memoria caché. El array a está almacenado a partir de la dirección física 0x3F200000 y cada entero ocupa 4 bytes. Dado el siguiente código:

```
int a[2];
for(i=0;i<2;i++)
     x+=a[i];
```

   a) Mostrar la evolución de las etiquetas de las líneas almacenadas en la caché durante la ejecución del bucle iteración a iteración.

   La dirección física a la hora de acceder a la caché se divide en Etiqueta, Índice y Desplazamiento.

   El campo desplazamiento es de 4 bits porque cada línea ocupa $2^4 = 16$ bytes

   El campo índice tiene 12 bits porque la caché es de correspondencia directa y tiene $64KB/16B = 4K = 2^{12}$ líneas

   Los restantes (32-12-4)=16 bits son la etiqueta.

   ===

   Iteración i=0

   ===

   Dirección accedida 0x3F20 000 0 (Produce un Fallo)

   Etiqueta: 3F20

   Índice: 000

   Desplazamiento 0

   Contenido de la etiqueta de la línea 0 de la caché: 3F20

   ===

   Iteración i=1

   ===

   Dirección accedida 0x3F20 000 4 (Produce un Acierto)

   Etiqueta: 3F20

   Índice: 000

   Desplazamiento 4

   Contenido de la etiqueta de la línea 0 de la caché: 3F20

   b) Calcula el tiempo medio de acceso a la caché (en segundos) considerando que el tiempo de acierto es de 1 ciclo, la penalización por fallo es de 100 ciclos y la velocidad de reloj del procesador es de 1Ghz. Considera para el cálculo la tasa de fallos derivada de los accesos del apartado anterior.

$tiempo\ de\ ciclo = 1/frecuencia = 1/1\ Ghz = 1\ ns$
La tasa de fallos del apartado (a) es 1/2
tiempo medio de acceso = tiempo de acierto + tasa de fallos x penalización por fallo
$Tiempo\ medio\ de\ acceso\ = (1 + (1/2) \times 100) \times 1\ ns = 51\ ns$

6. **[0.5p]** En un sistema de memoria virtual paginado en 2 niveles, la tabla de páginas del primer nivel y cada una de las tablas de página del segundo nivel tienen 32 entradas de 4 bytes cada una. Teniendo en cuenta que el tamaño de una tabla de páginas es exactamente el de una página del sistema, calcula el tamaño de la memoria virtual.

   Como hay 32 tablas de páginas de nivel 2 y cada una tiene 32 entradas, tenemos 32×32 páginas virtuales.

   Cada tabla de páginas ocupa $32 \times 4 = 128$ bytes y este es también el tamaño de página. Por tanto el tamaño del espacio virtual es $32 \times 32 \times 32 \times 4 = 128KB$

7. **[0.5p]** En sistema de memoria virtual paginado, el contenido actual de la TLB (expresado en binario) es el siguiente:

Página virtual	Página física
110	10
101	01
111	00

   Muestra en qué dirección física se encuentra la dirección virtual 0x1F.

   Nota: En el examen se aclaró que la dirección virtual tenía una longitud de 5 bits.

   La dirección física en binario es la 11111.

   Como vemos en la TLB el número de página virtual son 3 bits. En este caso la página virtual es la 111 y según la TLB está en la página física 00.

   Finalmente le concatenamos el desplazamient (11) a la página física, para obtener la dirección física completa 0011 (0x3)

8. **[1.5p]** ¿Qué esquemas de RAID se ajustan mejor a cada uno de estos servicios? Justifica cada respuesta.

   a) Un servidor que normalmente no debe atender muchos usuarios concurrentes. Debemos atender cada petición en el menor tiempo posible, y se necesita una buena fiabilidad a un precio ajustado.
   Valdría RAID 3 porque hace stripping a nivel de byte, lo cual le permite atender cada petición individual lo más rápido posible, y proporcionan fiabilidad con pocos discos adicionales.

   b) Una servidor que debe atender muchas peticiones concurrentes. Debemos atender el mayor número de peticiones posible por unidad de tiempo, y también se necesita una buena fiabilidad a un precio ajustado.
   Valdría RAID 5 o 6 porque hacen stripping a nivel de bloque, , lo cual favorece atender cada petición individual lo más rápido posible, y proporcionan fiabilidad con pocos discos adicionales.

9. **[0.5p]** Sea un sistema con las siguientes características:

   - Un sistema de memoria y de bus que soporta acceso a bloques de 16 palabras de 32 bits.

- Un bus síncrono de 64 bits a 200 MHz, en el que tanto una transferencia de 64 bits como el envío de la dirección a memoria requieren 1 ciclo de reloj
- El tiempo de lectura de memoria para cada dos palabras es de 5 ns

Calcular el ancho de banda y la latencia para la lectura de 256 palabras.

Tciclo =1/200MHz = 5ns

Bloques de 16 palabras de 4 bytes (32 bits) cada una.

Se requieren 256 palabras / 16 palabras por transacción = 16 transacciones.

Una transacción de un bloque requiere:

- 1 ciclo para enviar la dirección
- 5 ns / 5 ns /ciclo = 1 ciclo para leer las dos primeras palabras de memoria
- 1 ciclo para enviar los datos del grupo de 2 palabras leído (solapados con el inicio de la lectura siguiente)

Cada grupo restante de 2 palabras en el bloque de 16 palabras necesita el último paso (puesto que se solapa la lectura de un grupo con el envío del grupo anterior, ya que ambas operaciones tardan exactamente lo mismo, un ciclo.

Total para el bloque de 16 palabras: 1+1+8*(1)=10 ciclos por transacción.

Tiempo total de transferencia = 10 ciclos * 16 transacciones = 160 ciclos * 5ns = 800 ns

Ancho de banda del bus = (256 * 4) / 800ns= 1,28 MB/s