Examen final Fundamentos de Computadores 1ª convocatoria

Grado en Ingeniería Informática – Curso 2020/21

Ejercicio 1. (2 puntos) Consideremos un formato similar a IEEE 754 para punto flotante de 32 bits y para el cual el primer bit es de signo, a continuación, tenemos 7 bits de exponente y 24 bits de mantisa normalizada. Se mantiene el esquema del IEEE 754 para representar el $\pm\infty$, los NaN y los números no normalizados. Indica para esta representación:

a) (0.5 puntos) El valor en base 10 del número positivo más pequeño mayor que 0 que se puede representar con esta representación incluyendo a los números no normalizados. Indica también su representación en 32 bits indicando la división en campos.

b) (0.5 puntos) Valor mínimo positivo (mayor que 0) normalizado en decimal y en representación de 32 bits.

c) (0.5 puntos) Indicar el valor decimal del número que se representa como 0xFC200000.

d) (0.5 puntos) La representación en 32 bits del número decimal $-2500{,}4$ e indicar su valor en hexadecimal.

Ejercicio 2. (2 puntos) Responde brevemente a las siguientes cuestiones, razonando la respuesta:

a) (0.5 puntos) Calcula el AMAT (tiempo medio de acceso por instrucción) para un procesador con un tiempo de ciclo de reloj de 1 ns, una penalización por fallo de lectura o de escritura de 20 ciclos de reloj, una tasa de fallos de $0{,}05$ fallos por instrucción y un tiempo de acceso a la caché (sea el acceso un acierto o un fallo) de 1 ciclo de reloj.

b) (0.5 puntos) ¿Es cierto que la potencia consumida por un procesador es directamente proporcional al voltaje de alimentación e inversamente proporcional a la frecuencia de trabajo del procesador?

c) (0.5 puntos) Cuando un programa se compila para el computador A requiere 1000 millones de instrucciones y el CPI en dicho computador es de 1. Calcular el valor de MIPS para este computador considerando que tiene una frecuencia de reloj de 4 GHz.

d) (0.5 puntos) Indica muy brevemente las principales ventajas que puede tener una gestión del sistema de entrada/salida por interrupciones frente a otras opciones.

Ejercicio 3. (2 puntos) Considerando el repertorio de instrucciones del MIPS de 32 bits y dado el siguiente fragmento de código:

```
lui $8, 0x1001
ori $8, $8, 0x0040
instrucción_b
jal resta_vector
add $16, $17, $18
```

a) (0.25 puntos) ¿Qué funciones realizan las 2 primeras instrucciones?

b) (0.25 puntos) ¿Por qué pseudo-instrucción podemos sustituir las 2 primeras instrucciones? Indica su codificación equivalente en lenguaje ensamblador suponiendo que la dirección referenciada equivale a la etiqueta vector.

c) (0.5 puntos) Queremos cargar en el registro $16 la palabra almacenada en la dirección de memoria 0x10010034. Sustituye *instrucción_b* por la instrucción que nos permite realizar esa operación. Indica también su codificación en binario indicando cada uno de sus campos sabiendo que su código de operación es 100011.

d) (0.5 puntos) ¿Qué acciones realiza la instrucción *jal resta_vector*?

e) (0.5 puntos) La dirección de memoria en la que está almacenada la instrucción jal resta_vector es 0x00400090 y su codificación en hexadecimal es 0x0c10003e. ¿A qué dirección de memoria realiza un salto? ¿De qué tipo de direccionamiento se trata?

Ejercicio 4. (2 puntos) Consideremos una caché de asignación directa de un sistema BigEndian, con ISA MIPS de 32 bits, es decir, con direcciones y palabras de 32 bits, y direccionamiento de memoria a nivel de byte. Tenemos una caché de asignación directa de 32 bytes y líneas de una palabra. Consideramos que la caché está vacía cuando la CPU emite la siguiente secuencia de direcciones de acceso a memoria:

$$0x00000003, 0x00000002, 0x000000E2, 0x000000A2, 0x000000BE, 0x000000BA$$

a) (1 punto) Para cada una de las direcciones anteriores indicar la división en campos de la dirección y el significado de cada campo justificando si se trata de un acierto o de un fallo. Indicar el número de reemplazos que se producen. Indicar cual es el contenido final de la caché tras producirse la lista de accesos anterior.

b) (0.5 puntos) Si la caché fuese asociativa por conjuntos de 2 vías, ¿cómo se dividiría cada dirección en campos indicando su significado? Mostrar como ejemplo la división en campos de la dirección 0x000000BA.

c) (0.5 puntos) Indicar el número y tamaño de los comparadores que harían falta para cada una de las dos cachés consideradas en el ejercicio justificando la respuesta.

Ejercicio 5. (2 puntos) Se desea añadir la instrucción de tipo de acceso a memoria addsw al procesador tipo MIPS de la figura (implementación vista en clase) que realiza la siguiente función que se ejecuta en un solo ciclo de reloj:

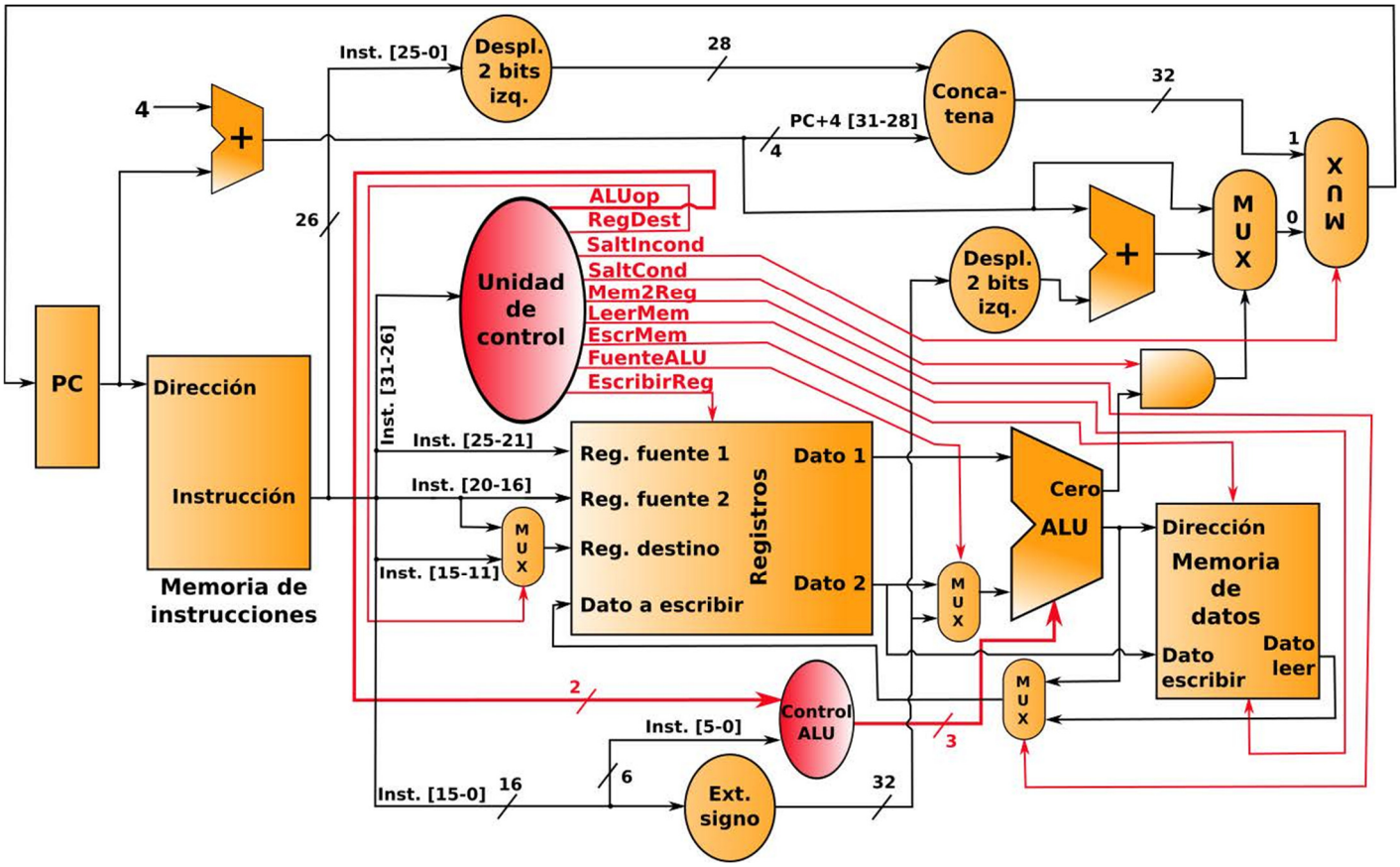
```
addsw $t1, cte($t2);   $t1 → Mem[($t2+$t1) + ext.signo.cte]
```

Esta instrucción almacena el contenido de un registro ($t1 en el ejemplo) en memoria, en la posición obtenida como la suma del contenido de dos registros ($t2 y $t1 en el ejemplo) y una constante a la que se le extiende el signo.
Indica, **razonando la respuesta**:

a) (0.5 puntos) Formato de la nueva instrucción detallando la longitud y el significado de cada campo en que se divide. Indicar de qué tipo es la instrucción atendiendo a su división en campos.

b) (0.75 puntos) Especifica la ruta de datos sobre el dibujo indicando, en caso de ser necesario, qué hardware adicional se requeriría, dónde sería necesario colocarlo y cómo se conectaría en la figura y para qué.

c) (0.75 puntos) Indica en la tabla que se muestra a continuación los valores que deben tomar todas las señales de control para que se pueda ejecutar la nueva instrucción, añadiendo nuevas filas si es necesario añadir nuevas señales.



ALUop	Funct	Acción de la ALU	Control de la ALU
00	XXXXXX	suma	010
01	XXXXXX	resta	110
10	100000	suma	010
10	100010	resta	110
10	100100	and	000
10	100101	or	001

Señal de control	Valor
ALUop	
RegDest	
SaltIncond	
SaltCond	
Mem2Reg	
LeerMem	
EscrMem	
FuenteALU	
EscribirReg	