Complemento de Arquitectura de Computadoras

Solución Examen 21 de Julio de 2017

(ref: scac20170721.odt)

- **Indique su nombre completo y número de cédula en cada hoja.**
- **Numere todas las hojas e indique el total de hojas en la primera. Escriba las hojas de un solo lado.**
- **Solo se responderán dudas de letra. No se responderán preguntas en los últimos 30 minutos de la prueba.**
- **La prueba es individual y sin material. Duración de la prueba : 3 horas.**
- **El puntaje mínimo de aprobación es de 60 puntos.**
- **Justifique todas sus respuestas.**

Pregunta 1 (10 puntos)

Explique las políticas de escritura write-back y write-through.

Pregunta 2 (10 puntos)

1. ¿Tiempo de CPU, es una buena medida para determinar el rendimiento de un programa? ¿Por qué?
2. ¿MIPS, es una buena medida para determinar el rendimiento de un programa? ¿Por qué?

Pregunta 3 (10 puntos)

Explique qué es una antidependencia y comente en qué tipo de pipelines se produce. Aporte un ejemplo y comente como se puede resolver.

Pregunta 4 (10 puntos)

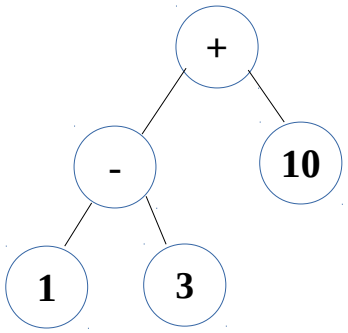
Explique qué es un DMA y cómo lo programa la CPU para su uso.

Ejercicio 1 (30 puntos)

Se almacena una expresión que involucra sumas y restas de enteros como un árbol binario. Los nodos hoja contienen los valores de los operandos y los nodos que no son hoja contienen la operación codificada (suma = 0, resta = 1). A continuación se muestra dicha estructura y una función que dado un árbol de este tipo devuelve el resultado de evaluar dicha expresión.

```
typedef struct{
    short val;
    nodo* izq, der;
} nodo
```

```
short evaluar(nodo* nod){
  if (nodo→izq != NULL){
    short res_a = evaluar(nod→izq);
    short res_b = evaluar(nod→der);
    if (nod→val == 0){
      return res_a + res_b
    }else{
      return res_a - res_b
    }
  }else{
    return nodo→val;
  }
}
```



Árbol de ejemplo. Codifica la expresión ((1-3)+10) y por lo tanto el resultado de evaluarlo es: 8

Se pide:

A) Compilar la función en 8086. Los punteros son desplazamientos en el segmento ES. Los parámetros se pasan y devuelven por stack. Se debe preservar el valor de todos los registros.

B) Dado un árbol de N nodos, calcular cuál es el consumo del stack en el peor caso.

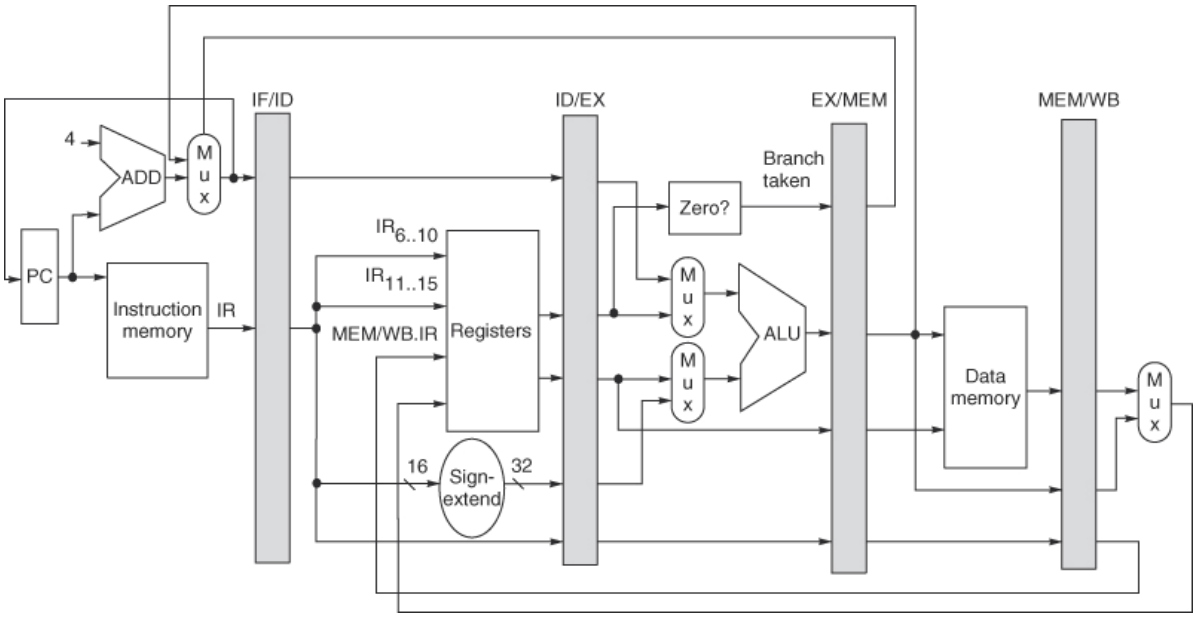
Solución Problema 1

A)

```
evaluar proc near

      push bp
      mov bp, sp
      push ax,
      push bx,
      push cx,

      mov bx, [bp + 4] // *nodo
      cmp word ptr es:[bx + 2], 0 // if (nodo → izq != NULL)
      je ELSE
      push es:[bx + 2]
      call evaluar
      pop ax // short res_a = evaluar(nodo → izq);
      push es:[bx + 4]
      call evaluar
      pop cx // short res_b = evaluar(nodo → der);
      cmp word ptr es:[bx], 0
      jne ELSE2
      add ax, cx
      jmp FIN
ELSE2:
      sub ax, cx
      jmp FIN

ELSE:
      mov ax, es:[bx]

FIN:
      mov [bp+4], ax
      pop cx
      pop bx
      pop ax
      pop bp
      ret

evaluar endp
B) Llamada = 2 bytes parámetro + 2 bytes IP
   Preservo -> 8 bytes * llamada
   => consumo stack = 12 bytes * llamadas

El peor caso se da cuando ningún hijo derecho tiene hijos. Puede pensarse como
dos listas paralelas una de largo L y otra de largo L-1.La cantidad de nodos es
N = L + L – 1. L coincide con la cantidad total de llamadas anidadas (incluyendo
la inicial), por lo que se tiene que el consumo del stack es: 12 * ((N + 1) / 2)
bytes
```

Ejercicio 2 (30 puntos)

Considere un computador con un procesador MIPS como el de la figura, y las siguientes características:



- Las memoria de instrucciones tiene una latencia menor que un ciclo de reloj. La memoria de datos consta de una cache con hit time menor que un ciclo de reloj. La latencia en memoria es tal que un miss en la cache provoca una detención completa del pipeline durante un ciclo.

- La cache de datos tiene una organización de correspondencia directa. Cuenta con 128 líneas y un tamaño de bloque de 128 bytes.

Considere la ejecución del siguiente fragmento de código MIPS con la memoria cache inicialmente vacía.

```
li $9, 0x1000
lw $10, 0($9)
lw $11, 4($9)
addu $12, $11, $10
addiu $9, $9, 0x4000
sw $12, 0($9)
```

a) Explique, para cada acceso a memoria, si ocurre un hit o un miss en la cache.

b) Realice un diagrama mostrando, para cada ciclo de reloj, en qué etapa del pipeline se encuentra cada instrucción. Recuerde que el hardware utilizado es el de la figura (no hay forwarding implementado). Suponga que un valor escrito en el banco de registros por una instrucción en la etapa WB puede ser leído durante el mismo ciclo de reloj por otra instrucción en la etapa ID.

Solución Ejercicio 2

a) Como la cache de datos tiene 128 líneas y 128 bytes por línea, los 7 bits menos significativos de la dirección indican el byte dentro del bloque a obtener, los siguientes 7 bits seleccionan la línea con la que se corresponde el bloque a acceder, y los restantes bits forman la etiqueta.

El primer acceso a memoria se realiza a la dirección 0x1000 (= 1000000000000_b). La misma causará un compulsory miss (obligatorio) por estar la cache 'vacía', y provocará que se traiga el bloque correspondiente y se ubique en la línea nro 32 (line = 100000_b).

El segundo acceso a memoria se realiza a la dirección 0x1004 (= 1000000000100_b). El bloque a buscar se encuentra en la línea nro 32, y como el tag coincide con el del bloque que se encuentra ahí (tag = 0), el acceso es un hit.

El último acceso a memoria se realiza a la dirección 0x5000 (= 1010000000000000_b). Este acceso es a la misma línea, pero el tag varía (en este caso vale 1), por lo tanto provocará un miss por colisión.

b)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
li	IF	ID	EX	Mem	WB													
lw1		IF	ID	ID	ID	EX	Mem	Mem	WB									
lw2			IF	IF	IF	ID	EX	EX	Mem	WB								
addu						IF	ID	ID	ID	ID	EX	Mem	WB					
addiu							IF	IF	IF	IF	ID	EX	Mem	WB				
sw											IF	ID	ID	ID	EX	Mem	Mem	WB