PROBLEMAS DE UNIDAD DE CONTROL

Nuevas instrucciones:

1. Deseamos añadir la instrucción **j** a la ruta de datos de un sólo ciclo. Añadir los elementos de la ruta de datos y puntos de control necesarios para realizar esta instrucción. El código de operación de la instrucción **j** es: 000001.

j dirección /* PC=dirección, los 4 bits más significativos son los del PC */

¿Cómo habría que modificar la unidad de control?

2. Realizar el problema anterior para la ruta multiciclo optimizada.
Mostrar los añadidos a la máquina de estados finitos que se implementó en clase para realizar esta operación.

3. Deseamos añadir la instrucción **jal** a la ruta de datos de un sólo ciclo. Añadir los elementos de la ruta de datos y puntos de control necesarios para realizar esta instrucción. El código de operación de la instrucción jal es: 000011.

jal dirección /* ($31= PC+4; PC=dirección) */

¿Cómo habría que modificar la unidad de control?

4. Realizar el problema anterior para la ruta multiciclo optimizada.
Mostrar los añadidos a la máquina de estados finitos que se implementó en clase para realizar esta operación.

5. Añadir los elementos de la ruta de datos y puntos de control necesarios para realizar la instrucción **addiu**, tanto con ruta de datos monociclo como multiciclo.

addiu $i, $j, inmediato /* $i = $j + SignExt(inmediato) */

6. ¿Qué modificaciones hay que realizar en la ruta de datos y la U.C. tanto monociclo como multiciclo para añadir la instrucción **bne** (bifurcar si no igual) que se comporta de manera similar a **beq**.?. El código de operación de la instrucción bne es: 000101.

7. Se está considerando la ampliación del repertorio MIPS con la instrucción **addm Rt, Rs, despl.** que suma el contenido del registro **Rt** con el contenido de la posición de memoria **BR[Rs] + signo_extendido(despl)** y deposita el resultado en **Rt**.

a) Modifica la ruta de datos y la FSM de control multiciclo para incluir esta nueva instrucción.
b) ¿Se modificará el tiempo de ciclo? En caso afirmativo, ¿cuál será el nuevo tiempo de ciclo? ¿Cuánto tardará la instrucción en ejecutarse?

Retardos: Memoria: 2ns; BR:1,5ns; ALU:2ns; MUX:0,5ns; decod:1ns; CtrlALU:1ns; Tsetup, Thold, CLK_to_Q:0,1ns; <<2 y ExtSigno:0ns.

Rendimiento:

8. Ciertos evaluadores del rendimiento del procesador MIPS han determinado que la ruta crítica que fija la longitud del ciclo de reloj en la ruta multiciclo corresponde a los accesos a memoria para carga y almacenamiento (no para lectura de instrucciones). Esto ha llevado a replantear la implementación para que la frecuencia de reloj sea de 500 MHz en vez de 750 MHz. inicialmente propuesta. Sin embargo, uno de los ingenieros ha propuesto como solución de compromiso que los ciclos de acceso a memoria se subdividan en dos para, de este modo, admitir la frecuencia de reloj inicial (750 MHz). Usando las proporciones de instrucciones que se muestran a continuación, determina la ganancia de velocidad que se obtiene empleando la máquina de 2 ciclos de acceso a memoria con reloj de 750 MHz, frente a la máquina con 1 ciclo de acceso a memoria y reloj de 500 MHz.
Frecuencia de uso de las diferentes instrucciones para un caso típico (gcc):

- Carga: 22%
- Almacenamiento: 11%
- Aritmético-lógicas: 49%
- Salto condicional, realizando el salto: 4%
- Salto condicional, no realizando el salto: 12%
- Salto incondicional: 2%

9. Usando las mismas proporciones de instrucciones del ejercicio 8,
a) determinar cuál de las tres máquinas siguientes es la más rápida y con qué relación:
M1: ruta multiciclo no optimizada y reloj de 500 MHz
M2: ruta multiciclo en donde la actualización del banco de registros se hace en el mismo ciclo de reloj que la lectura de memoria o la operación ALU. Esto disminuye en 2 el número de estados y rebaja la frecuencia de reloj a 400 MHz al aumentar la longitud del camino crítico
M3: ruta multiciclo como la de M2 donde además el cálculo de dirección efectiva se hace en el mismo ciclo que el acceso a memoria. Esto disminuye en 4 el número de estados anterior y rebaja la frecuencia de reloj a 250 MHz, al alargar la longitud del ciclo para el camino crítico.
b) ¿Hay proporciones de instrucciones que hiciesen que fuese otra la máquina más rápida? Si es así, ¿qué proporciones son esas?

10. Suponiendo los siguientes tiempos de ejecución de los diversos elementos de una ruta de datos para el MIPS:
Lectura de memoria: 5ns; escritura de memoria: 7ns; lógica de control de la ALU: 2ns; lógica de control principal: 2 ns; lectura BR: 1ns; escritura BR: 2ns; operación ALU: 5ns; extensión signo: 1 ns; multiplicación por 4: 1 ns; multiplexores: 1 ns; Tsetup=0,1 ns; Thold=0,1 ns; CLK_to_Q= 0,1 ns.
a) Calcular cuál es el mejor tiempo de ciclo para una implementación multiciclo.
b) Si el tiempo de escritura en memoria fuese 15 ns, realizar el mismo cálculo.

Ruta de datos y cronogramas:

11. Sobre la estructura de la transparencia nº 25, que permite implementar el control multiciclo de nuestro procesador, indicar los valores de todas las líneas de datos y los registros, en los siguientes casos:

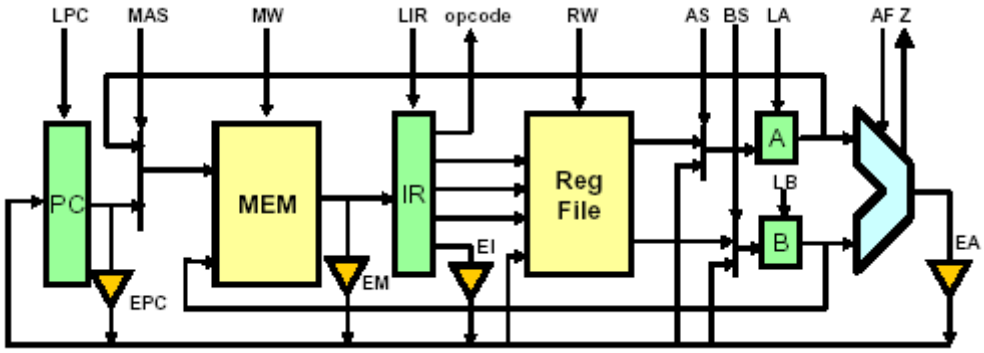
a.- Ejecución de la instrucción OR $1, $2, $3, al final de cada una de sus cuatro etapas.
Estado actual de la máquina:
$1=HEX(00000017)
$2=HEX(001100FF)
$3=HEX(FF000345)
PC=HEX(00003400)

b.- Ejecución de la instrucción JAL 1024 (1024 en decimal) al final de las tres etapas.

Unidades de control:

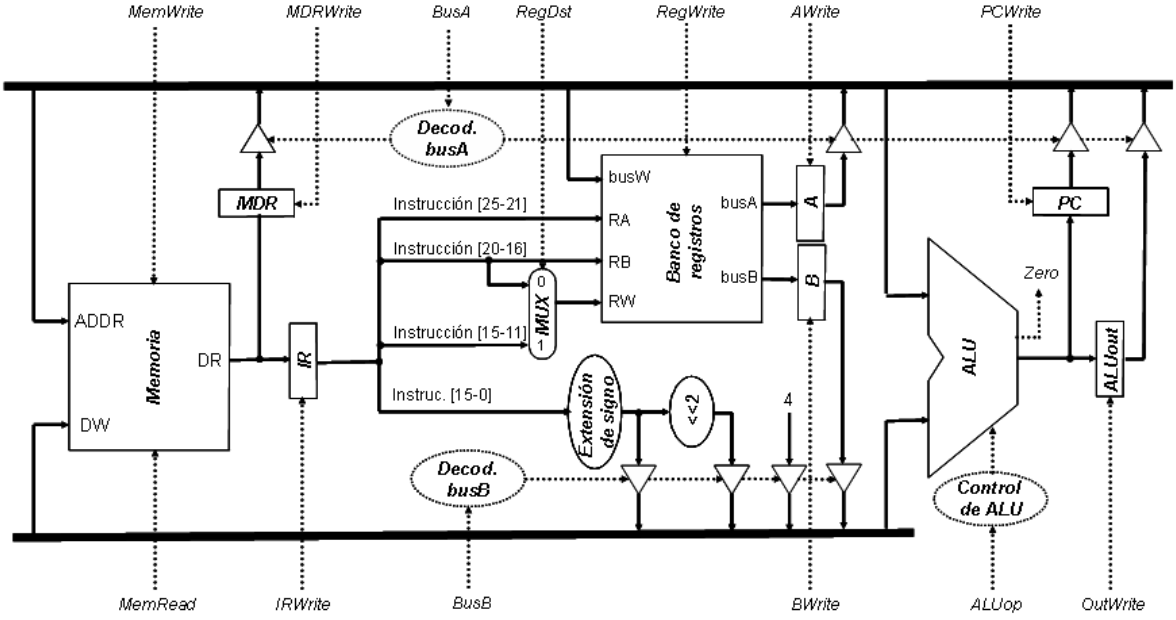
12. Diseñar la U.C. completa con lógica discreta para las instrucciones lw, sw, beq, add, jal en una implementación **multiciclo con buses** como la vista en clase.
Diseñar la U.C. microprogramada con formato horizontal, vertical y vertical por campos.

13. Dada esta ruta de datos, variante para MIPS centrada alrededor de un bus,



a. Escribir las micro-operaciones en formato de transferencia de registros de cada una de las instrucciones básicas: tipo-R, load, store y beq, indicando las señales de control activas en cada ciclo.
(Si para que funcione es imprescindible introducir alguna modificación, indicar cuál es e incorporarla)
b. ¿Cuál es el rendimiento de esta máquina expresado en MIPS para una proporción de instrucciones 50%, 20%, 10%, 2.5% (beq-salta) y 17.5%(beq-no_salta) si la frecuencia de reloj es 100 MHz?
Nota: Suponer que las operaciones de la ALU son: A+B, A-B, A&B, A | B, A+4.

14. Sobre la ruta de datos de la figura:



a) Escribir las micro-operaciones, a nivel de transferencia entre registros, necesarias para implementar la instrucción:

Rd<=Mem[rs] + Mem[rt]

Proponer las modificaciones de la ruta de datos que sean necesarias, tratando de que sean el menor número posible.
b) Escribir el microprograma de la instrucción utilizando una codificación vertical por campos, explicando qué campos existen y por qué se han elegido.

Segmentación:

15. El siguiente fragmento de código se ejecuta en un MIPS con segmentación:

```
sub   $1,$2,$3
add   $4,$5,$6
sub   $5,$4,$8
add   $7,$2,$3
add   $9,$7,$3
lw    $1,10($6)
add   $3,$1,$4
sub   $6,$7,$8
```

Suponiendo que un dato se puede escribir en un banco de registros y leer su nuevo valor en el mismo ciclo:
a) Calcular el número de ciclos necesarios para ejecutar este código si no existe la posibilidad de anticipar operandos o reordenar el código.
b) Calcular el número de ciclos si existe anticipación de operandos, pero no reordenación de código.
c) Tratar de reordenar el código para conseguir que el número de ciclos sea mínimo, si no hay anticipación de operandos. ¿Cuántos ciclos son necesarios en este caso?

16. Sobre la estructura del computador MIPS segmentado, se ejecuta la siguiente secuencia de instrucciones:

```
ADD $1, $2, $3
SUB $4, $2, $3
AND $5, $2, $3
OR  $6, $2, $3
```

Si la instrucción ADD está colocada en la dirección de memoria 00002000 (Hex), y el contenido de los registros es:

```
$1=00000005
$2=00000004
$3=00000002
PC=00002000
```

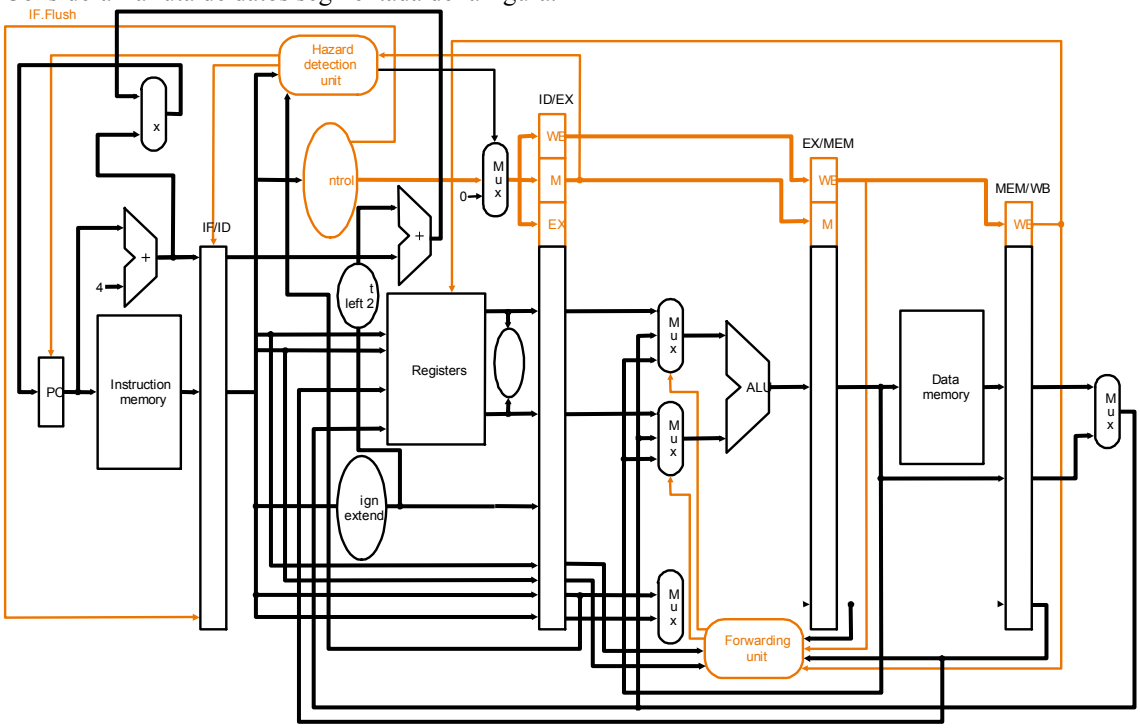
a) Señalar el contenido de los siguientes registros al cabo de 4 ciclos de reloj: IF/ID.pc, ID/EX.pc, EX/MEM.pc, ID/EX.A, ID/EX.B, EX/MEM.ALUout, MEM/WB.ALUout, ID/EX.rd, EX/MEM.rd, MEM/WB.rd.
b) Indicar también el contenido de cada uno de los registros que almacena el control.

17. Se tiene el siguiente fragmento de código del MIPS.

```
OR $4, $8, $9
(100 instrucciones con dependencias internas de datos, pero no detención de pipeline)
ADD $5, $6, $7
OR $4, $1, $6
(100 instrucciones con dependencias internas de datos, pero no detención de pipeline)
LW $10, 20($4)
ADD $8, $9, $10
SUB $6, $8, $1
OR $1, $3, $5
BEQ $1,$2,1000
ADD $1, $5, $6
```

a) Cuánto tardaría en ejecutarse, si cuando hay un conflicto de control se espera a que se solucione y se supone que el destino de salto se conoce en la fase ID y la comparación de salto también. Suponer que no hay anticipación de operandos.
b) Si se tuviesen saltos retardados, ¿cómo se podría rellenar el hueco de salto para disminuir el tiempo de ejecución? ¿Cuál sería este tiempo?
c) ¿y si además de salto retardados tuviésemos anticipación de operandos?
El destino del salto es la instrucción OR $4, $1, $6, y el salto se repite 100 veces.

18. Considerar la ruta de datos segmentada de la figura.



a). En esta ruta puede ocurrir simultáneamente un intento de anulación de instrucción (*flush*) debido a un salto y un intento de parada (bloqueo o *stall*) debido a una dependencia de datos que afecta a una instrucción LW. Mostrar una secuencia de código donde se manifieste la situación descrita.

b). Mostrar cuál es el efecto del bloqueo y de la anulación sobre el *pipeline*, indicando cómo se detectan y a qué señales y registros afectan.

c). Las acciones realizadas en el bloqueo y anulación simultáneos, ¿colaboran o tienen efectos que entran en conflicto? Si hay colaboración, ¿cómo trabajarían juntos? Si hay conflicto, ¿qué acción debe tener prioridad

Ensamblador:

19. El desenrollado de bucles es una técnica que permite elevar la velocidad de ejecución de códigos iterativos. Para el siguiente ejemplo en una arquitectura pipeline,

Código de bucle original	Código de bucle desenrollado una vez

`bucle: lw $t0, 0($s1)`
`add $t0, $t0, $s2`
`sw $t0, 0($1)`
`addi $s1, $s1, -4`
`bne $s1, $zero, bucle`
	`bucle: lw $t0, 0($s1)`
	`add $t0, $t0, $s2`
	`sw $t0, 0($1)`
	`lw $t1, -4($s1)`
	`add $t1, $t0, $s2`
	`sw $t1, -4($1)`
	`addi $s1, $s1, -8`
	`bne $s1, $zero, bucle`

comparar la diferencia de rendimiento, mediante contabilización de ciclos de ejecución, para los tres casos siguientes:

a) bucle original
b) bucle desenrollado
c) bucle desenrollado y reordenado para evitar bloqueos si es posible

(Suponer que la instrucción `bne` toma la decisión de salto en la etapa ID, y se dispone de circuitería de anticipación para resolver dependencias de datos).

20. Supongamos que los dos fragmentos de código ensamblador siguientes han sido generados por sendos compiladores (C1 y C2) para un mismo código fuente.

Compilador 1	Compilador 2

`add $t0, $zero, $zero`
`add $t1, $a1, $zero`
`loop: bge $t1, $a2, retorno`
`sll $t2, $t1, 2`
`add $t2, $t2, $a0`
`lw $t3, 0($t2)`
`add $t0, $t0, $t3`
`add $t1, $t1, $a3`
`j loop`
`retorno: add $v0, $t0, $zero`
`jal $ra`
	`add $v0, $zero, $zero`
	`sll $a2, $a2, 2`
	`sll $a3, $a3, 2`
	`add $t1, $a1, $zero`
	`bge $t1, $a2, retorno`
	`loop: add $t2, $t1, $a0`
	`lw $t3, 0($t2)`
	`add $v0, $v0, $t3`
	`add $t1, $t1, $a3`
	`blt $t1, $a2, loop`
	`retorno: jal $ra`

a. Inicialmente $a1=0, $a2=5, $a3=1. Determinar la frecuencia de cada tipo de instrucción rellenando la tabla siguiente:

Tipo de instrucción	CPI	Frecuencia en C1	Frecuencia en C2
ALU Aritmética			
ALU lógica			
Memoria (“load”)			
Memoria (“sw”)			
Bifurc. Cond. SI			
Bifurc. Cond. NO			
Bifurc. Incond.			
Total			

(Suponemos que la instrucción “sll” – shift logical left – se ejecuta en 3 ciclos. El registro $zero se corresponde con $0 y siempre vale 0)

b. Calcular el CPI de cada programa usando las frecuencias obtenidas en a)
c. Si P1 se ejecuta en una máquina de 500 MHz y P2 en una de 400 MHz ¿qué combinación h/w-s/w es más rápida? ¿Cuál es la ganancia?

21. Codificar en ensamblador RISC el siguiente trozo de código C

```
if (a==b && c==d) a=a/2;
else a=a*4;
```

utilizando instrucciones de desplazamiento para las operaciones de multiplicación y división. (Las variables a, b, c y d ocupan una palabra cada una y están almacenadas consecutivamente a partir de la dirección contenida en el registro r26. Los registros r2 a r7 están libres para operar con ellos – utilizar el menor número de instrucciones y registros posible)

¿Cuántos ciclos tardaría en ejecutarse sobre la arquitectura MIPS multiciclo? ¿Y sobre la arquitectura MIPS segmentada con hardware de anticipación y bloqueo, y salto en 3 ciclos?