Facultad de Informática
Universidad Complutense de Madrid

FUNDAMENTOS DE COMPUTADORES II

MICROARQUITECTURA SEGMENTADA: CONFLICTOS

Resumen de conflictos

- No tiene conflictos estructurales.
- Tiene un **conflicto de datos** al ejecutar una instrucción que tiene como registro fuente el mismo registro que tiene como destino cualquiera de las 3 instrucciones anteriores.

	1	2	3	4	5	6	7	8
`addi t1,t2,5`	IF	ID	EX	M	**WB**			
`add t3,t1,t2`		IF	**ID**	EX	M	WB		
`addi t4,t1,15`			IF	**ID**	EX	M	WB	
`add t5,t1,t1`				IF	**ID**	EX	M	WB

- Tiene un conflicto de control al ejecutar una instrucción **`jal`** o **`beq`**.

	1	2	3	4	5	6
`beq t1,t2,L1`	IF	ID	**EX**	M	WB	
`???`		**IF**	ID	EX	M	WB

Procesador Segmentado sin gestión de conflictos

(escritura del BR a final de ciclo)

- Todos los conflictos deben resolverse por software.
- **Conflictos de datos**: según el caso, insertar 1, 2 ó 3 instrucciones **nop** entre la instrucción que escribe el registro y la que lo lee.

	1	2	3	4	5	6	7	8	9
`addi t1,t2,5`	IF	ID	EX	M	**WB**				
`nop`		IF	ID	EX	M	WB			
`nop`			IF	ID	EX	M	WB		
`nop`				IF	ID	EX	M	WB	
`add t3,t1,t2`					IF	**ID**	EX	M	WB

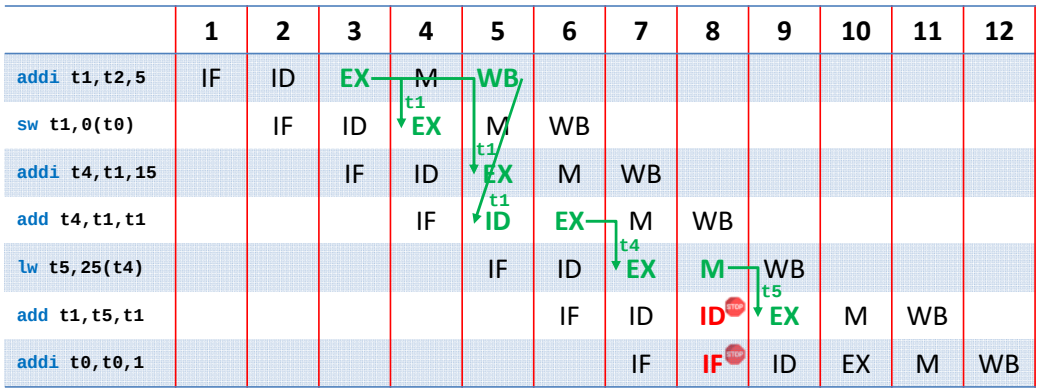
- **Conflictos de datos**: insertar 2 instrucciones **nop** después de cada instrucción de salto.

	1	2	3	4	5	6	7	8
`beq t1,t2,L1`	IF	ID	**EX**	M	WB			
`nop`		IF	ID	EX	M	WB		
`nop`			IF	ID	EX	M	WB	
`L1:addi t3,t1,1`				**IF**	ID	EX	M	WB

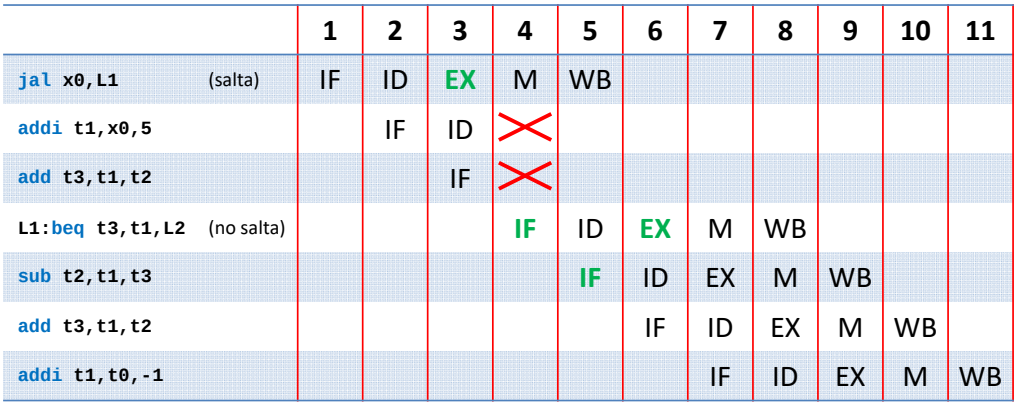
Procesador Segmentado con gestión de conflictos

(escritura del BR a mitad de ciclo, unidad de anticipación y unidad de conflictos con predicción de salto no tomado)

- Todos los conflictos se resuelven por hardware.
- Reglas para resolver **conflictos de datos**:
  1. Si la instrucción en EX tiene como fuente un registro que la instrucción en MEM tiene como destino: el dato se anticipa desde MEM a EX.
  2. Si la instrucción en EX tiene como fuente un registro que la instrucción en WB tiene como destino: el dato se anticipa desde WB a EX.
  3. Si la instrucción en EX tiene como fuente un registro que tanto la instrucción en MEM como la en WB tienen como destino: el dato se anticipa desde MEM a EX.
  4. Si la instrucción en ID tiene como fuente un registro que la instrucción en WB tiene como destino: el dato correcto se lee del BR porque se ha escrito a mitad de ciclo.
  5. Si la instrucción en EX es **`lw`** y tiene como destino un registro que la instrucción en ID tiene como fuente: las instrucciones en IF e ID se paran (las restantes avanzan normalmente). Penaliza 1 ciclo la ejecución.



- Reglas para resolver **conflictos de control**:
  1. Si la instrucción en EX es **`jal`** o es **`beq`** y toma el salto: las instrucciones en IF e ID dejan de ejecutarse. Penaliza 2 ciclos la ejecución.
  2. Si la instrucción en EX es **`beq`** y no toma el salto: las instrucciones avanzan normalmente.



OTRAS MICROARQUITECTURAS SEGMENTADAS ALTERNATIVAS

Procesador Segmentado con anticipación optimizada

(tratamiento específico **`lw`→`sw`***)*

- Añade al anterior, las siguientes reglas para resolver **conflictos de datos**:
  6. Si la instrucción en EX es **`lw`** y tiene como destino el registro que una instrucción **`sw`** en ID tiene como fuente para escribir en memoria: ninguna instrucción se para.
  7. Si la instrucción en MEM es **`sw`** y tiene como registro fuente para escribir en memoria el que una instrucción **`lw`** en WB tiene como destino: el dato se anticipa desde WB a MEM.

	1	2	3	4	5	6
`lw s3,16(s0)`	IF	ID	EX	M	WB	
`sw s3,20(s0)`		IF	ID	EX	M (s3)	WB

Procesador Segmentado con gestión de conflictos por parada

(escritura del BR a mitad de ciclo y unidad de conflictos por parada)

- Todos los conflictos se resuelven por hardware.
- Reglas para resolver **conflictos de datos**:
  1. Si la instrucción en ID tiene como fuente un registro que la instrucción en EX o MEM tiene como destino: las instrucciones en IF e ID se paran (las restantes avanzan normalmente). Según el caso, penaliza 1 ó 2 ciclos la ejecución.
  2. Si la instrucción en ID tiene como fuente un registro que la instrucción en WB tiene como destino: el dato correcto se lee del BR porque se ha escrito a mitad de ciclo.
- Reglas para resolver **conflictos de control**:
  1. Si la instrucción en ID o EX es **`jal`** o **`beq`**: la instrucción en IF se para (las restantes avanzan normalmente). Penaliza 2 ciclos la ejecución.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
`addi t1,t2,5`	IF	ID	EX	M	WB (t1)									
`add t3,t1,t2`		IF	ID	ID	ID	EX	M	WB						
`addi t4,t1,15`			IF	IF	IF	ID	EX	M	WB					
`beq t1,t2,L2`						IF	ID	EX	M	WB				
`lw t5,25(t4)`							IF	IF	IF	ID	EX	M	WB	
`add t1,t5,t1`										IF	ID	EX	M	WB

Procesador Segmentado sin gestión de conflictos

(escritura del BR a mitad de ciclo)

- Todos los conflictos deben resolverse por software.
- **Conflictos de datos**: según el caso, insertar 1 ó 2 instrucciones **nop** entre la instrucción que escribe el registro y la que lo lee.

	1	2	3	4	5	6	7	8
`addi t1,t2,5`	IF	ID	EX	M	**WB**			
`nop`		IF	ID	EX	M	WB		
`nop`			IF	ID	EX	M	WB	
`add t3,t1,t2`				IF	**ID** (t1)	EX	M	WB

- **Conflictos de datos**: insertar 2 instrucciones **nop** después de cada instrucción de salto.

	1	2	3	4	5	6	7	8
`beq t1,t2,L1`	IF	ID	**EX**	M	WB			
`nop`		IF	ID	EX	M	WB		
`nop`			IF	ID	EX	M	WB	
`L1:addi t3,t1,1`				**IF**	ID	EX	M	WB