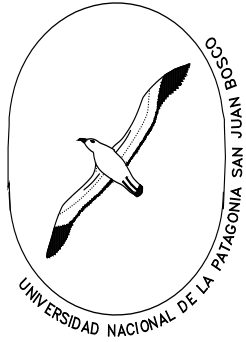


Universidad Nacional de la Patagonia San Juan Bosco

Facultad de Ingeniería. Sede Puerto Madryn



ARQUITECTURA DE COMPUTADORAS


Dependencias

Prof. Jorge P. Dignani

Dependencias de Datos

- Existen trabas que impiden que una instrucción se ejecute en el ciclo que tiene asignado.

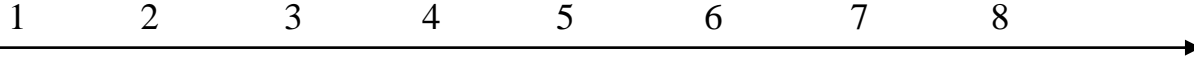
sub	\$2, \$1, \$3
and	\$12, \$2, \$5
or	\$13, \$6, \$2
add	\$14, \$2, \$2
sw	\$15, 100(\$2)



- Se debe respetar el orden de las lecturas y escrituras para mantener la semántica del programa.
- Las instrucciones *and*, *or*, *add* y *sw* deben leer el registro \$2 **después** que éste haya sido escrito por la instrucción *sub*. **Riesgo RAW**

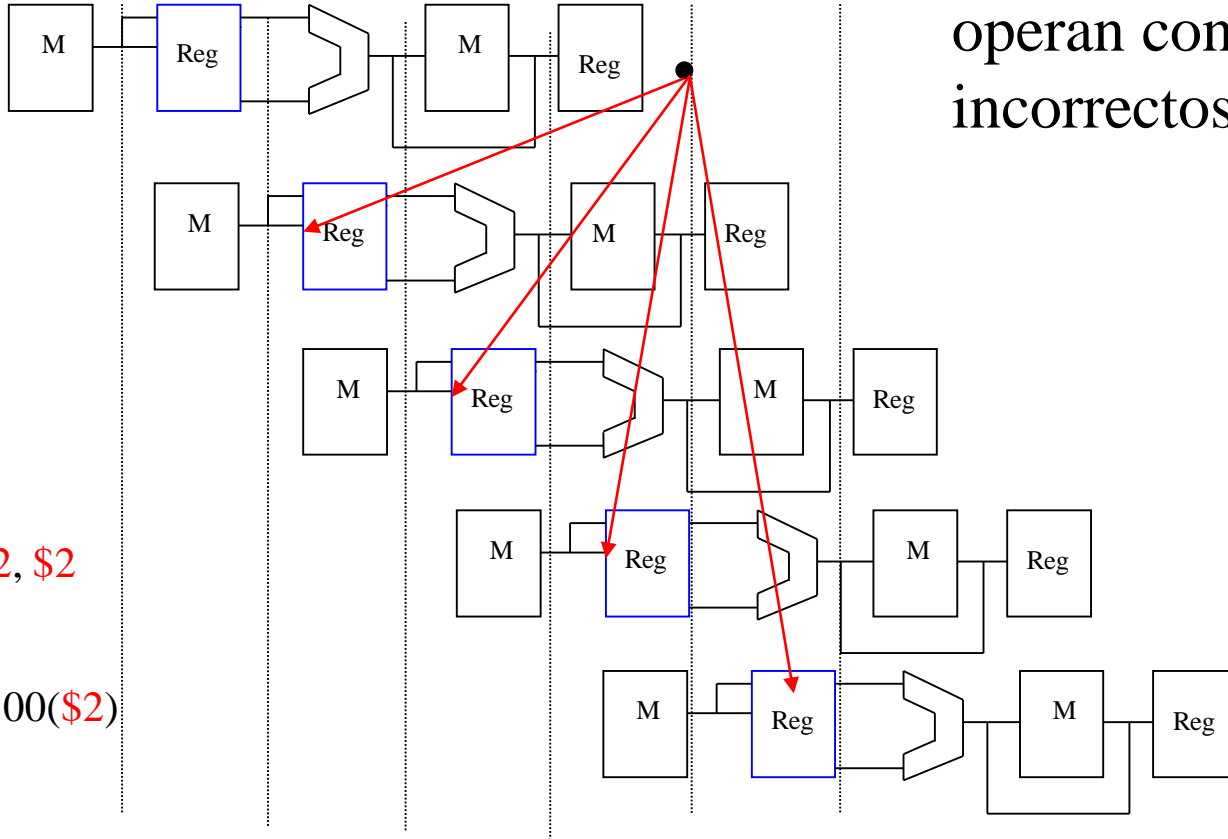
En el pipeline...

ciclos



Las instrucciones operan con valores incorrectos!

Sub \$2, \$1, \$3



And \$12, \$2, \$5

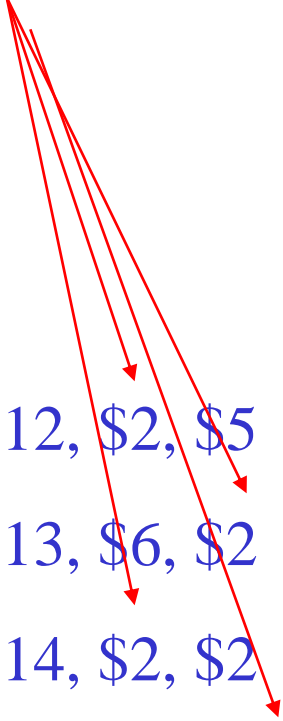
Or \$13, \$6, \$2

Add \$14, \$2, \$2

Sw \$15, 100(\$2)

Solución Estática

sub	\$2, \$1, \$3
nop	
nop	
nop	
and	\$12, \$2, \$5
or	\$13, \$6, \$2
add	\$14, \$2, \$2
sw	\$15, 100(\$2)

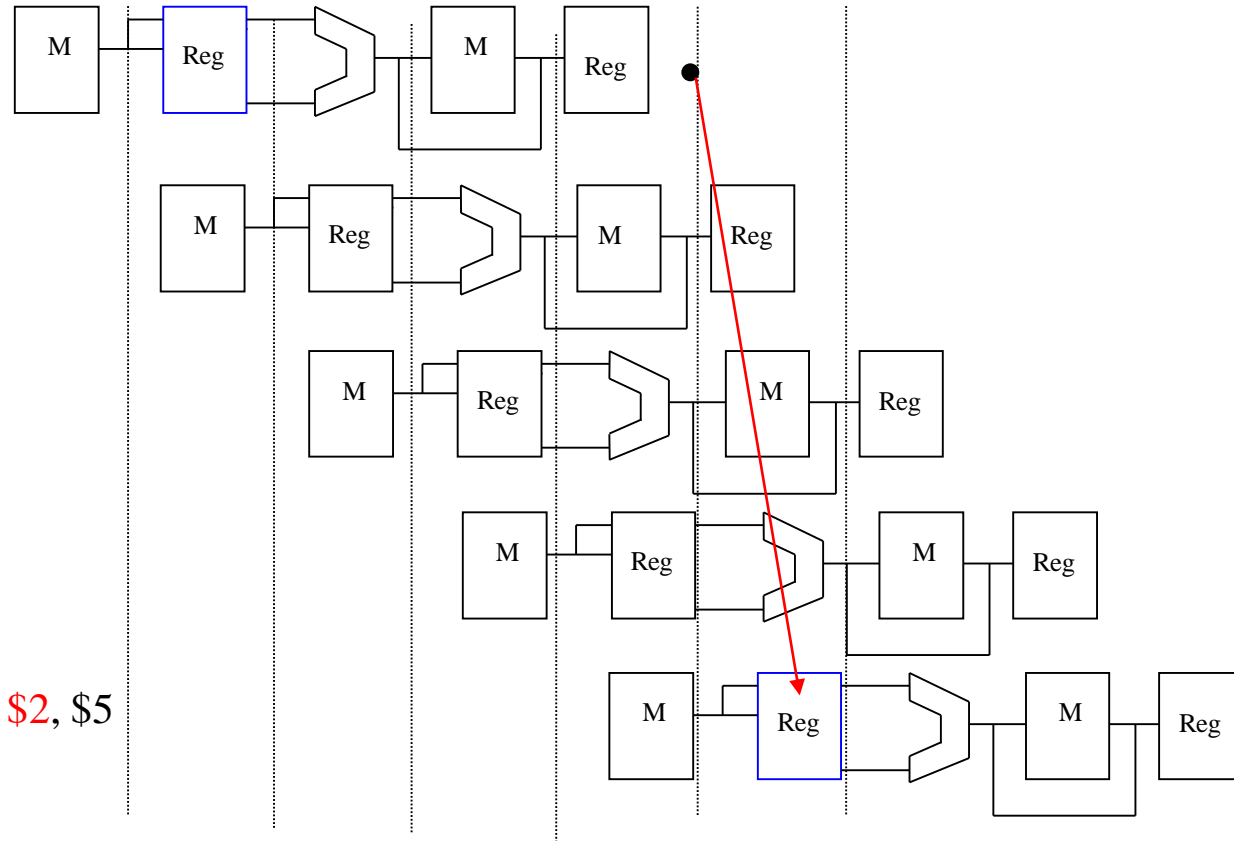


- El compilador podría ocuparse de no generar esas secuencias de código: por ejemplo insertando instrucciones entre sub y add.
- Si no encuentra ninguna instrucción útil, inserta NOP.

En el pipeline...

ciclos

1 2 3 4 5 6 7 8



Solución Dinámica: Detener

- La solución más sencilla es ESPERAR, o sea, detener las instrucciones en la segmentación.
 - Detectar el riesgo
 - Detener la segmentación
- Cuando hay riesgo? Cuando una instrucción que está en la etapa Decode intenta leer un registro que alguna(s) instrucción(es) anterior planea escribirlo en WB.
- Como se detiene la segmentación? Insertando “burbujas”

Condiciones de Riesgos

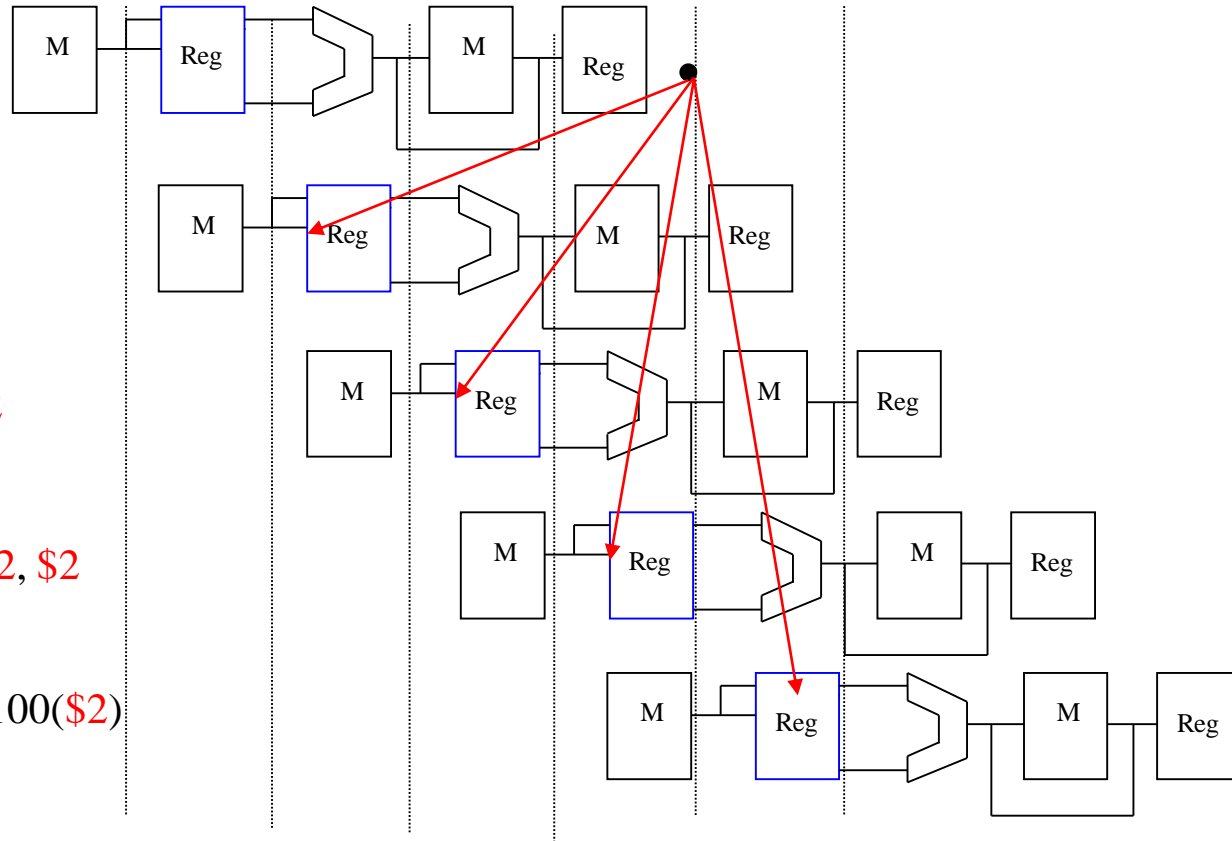
- Entre la instrucción que está en ID y la que está en EX:
 - $ID/EX.IR_{RD} == IF/ID.IR_{RS}$
 - $ID/EX.IR_{RD} == IF/ID.IR_{RT}$
 - $ID/EX.IR_{RT} == IF/ID.IR_{RS}$
 - $ID/EX.IR_{RT} == IF/ID.IR_{RT}$

Tanto RS como RT pueden ser registro destino!
- Entre la instrucción que está en ID y la que está en MEM:
 - $EX/MEM.REG == IF/ID.IR_{RS}$
 - $EX/MEM.REG == IF/ID.IR_{RT}$
- Entre la instrucción que está en ID y la que está en WB:
 - $MEM/WB.REG == IF/ID.IR_{RS}$
 - $MEM/WB.REG == IF/ID.IR_{RT}$

Riesgo RAW

ciclos

1 2 3 4 5 6 7 8



Sub \$2, \$1, \$3

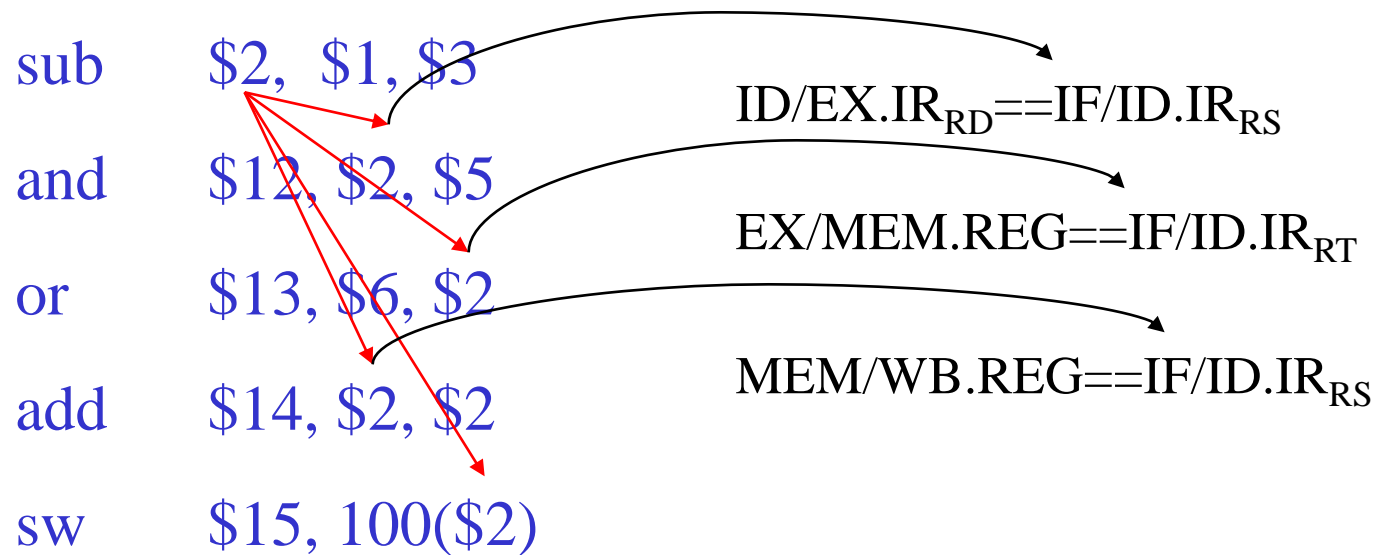
And \$12, \$2, \$5

Or \$13, \$6, \$2

Add \$14, \$2, \$2

Sw \$15, 100(\$2)

Condiciones de Riesgos



Detección de Riesgos

- Hay instrucciones que NO escriben en los registros: Sw, Beq.==> comprobarlo con la señal RegW

- ID/EX.IR_{RT}==IF/ID.IR_{RS} sw \$2, 100(\$1)
and \$12, \$2, \$5

- Hay instrucciones que escriben o en rt o en rd : tipo R o Lw ==> comprobarlo con la señal RegDst

- ID/EX.IR_{RT}==IF/ID.IR_{RT} lw \$2, 100(\$1)
and \$12, \$5, \$2

Detección de Riesgos

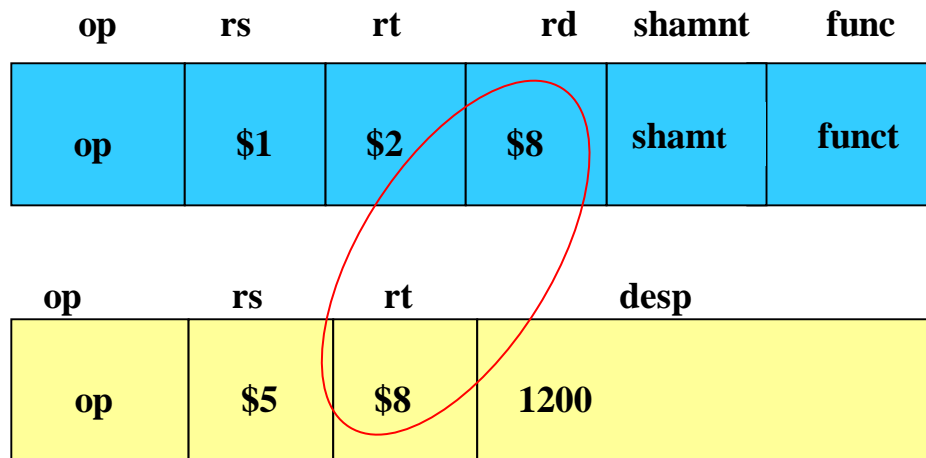
- Riesgo EX:
 - $ID/EX.WBRegW == 1$ y
 - $((ID/EX.EXRegDst == 1 \text{ y } ID/EX.IR_{RD} == IF/ID.IR_{RS}) \text{ o } (ID/EX.EXRegDst == 0 \text{ y } ID/EX.IR_{RT} == IF/ID.IR_{RS}) \text{ o } (ID/EX.EXRegDst == 1 \text{ y } ID/EX.IR_{RD} == IF/ID.IR_{RT}) \text{ o } (ID/EX.EXRegDst == 0 \text{ y } ID/EX.IR_{RT} == IF/ID.IR_{RT}))$
- Riesgo MEM:
 - $EX/MEM.WBRegW == 1$ y
 - $((EX/MEM.REG == IF/ID.IR_{RS}) \text{ o } (EX/MEM.REG == IF/ID.IR_{RT}))$
- Riesgo WB:
 - $MEM/WB.WBRegW == 1$ y
 - $((MEM/WB.REG == IF/ID.IR_{RS}) \text{ o } (MEM/WB.REG == IF/ID.IR_{RT}))$

Detección de Riesgos

- Que ocurre con este código?

add \$8, \$1, \$2

lw \$8, 1200(\$5)



- ID/EX.WBRegW==1 y
 ((ID/EX.EXRegDst==1 y ID/EX.IR_{RD}==IF/ID.IR_{RT})

Detección de Riesgos

- Sucede que la instrucción add, que es previa, escribirá sobre el registro identificado por el campo rd, cuyo valor coincide con el del campo rt de la siguiente instrucción que usualmente sirve para identificar a los registros fuente SÓLO si no se trata de una instrucción de load.

ID/EX.WBRegW==1 y

(ID/EX.EXRegDst==1 y ID/EX.IR_{RD}==IF/ID.IR_{RT} y IF/ID.Irop<>35)

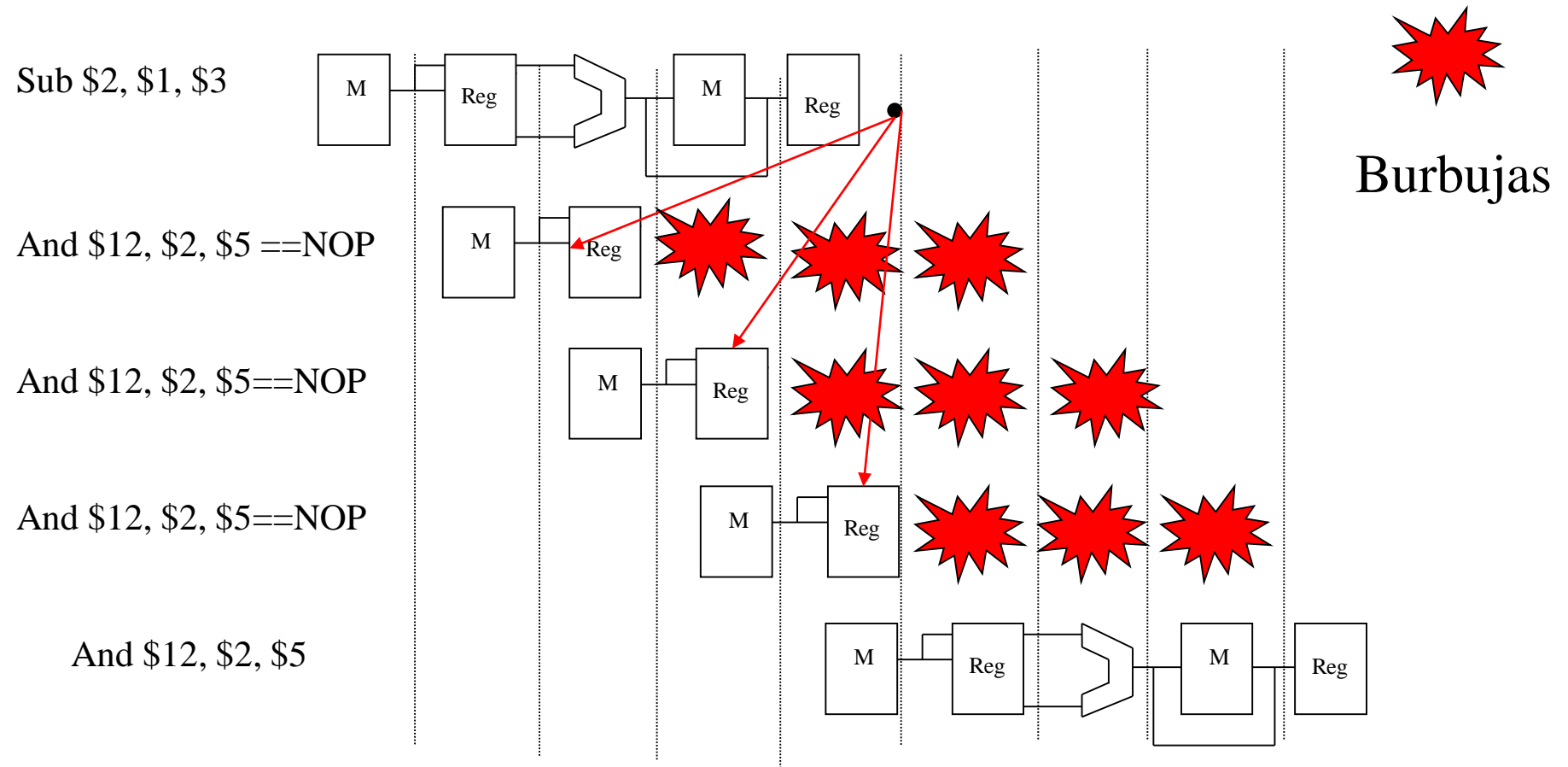
- ID/EX.WBRegW==1 y

((ID/EX.EXRegDst==1 y ID/EX.IR_{RD}==IF/ID.IR_{RT})

Detenciones

ciclos

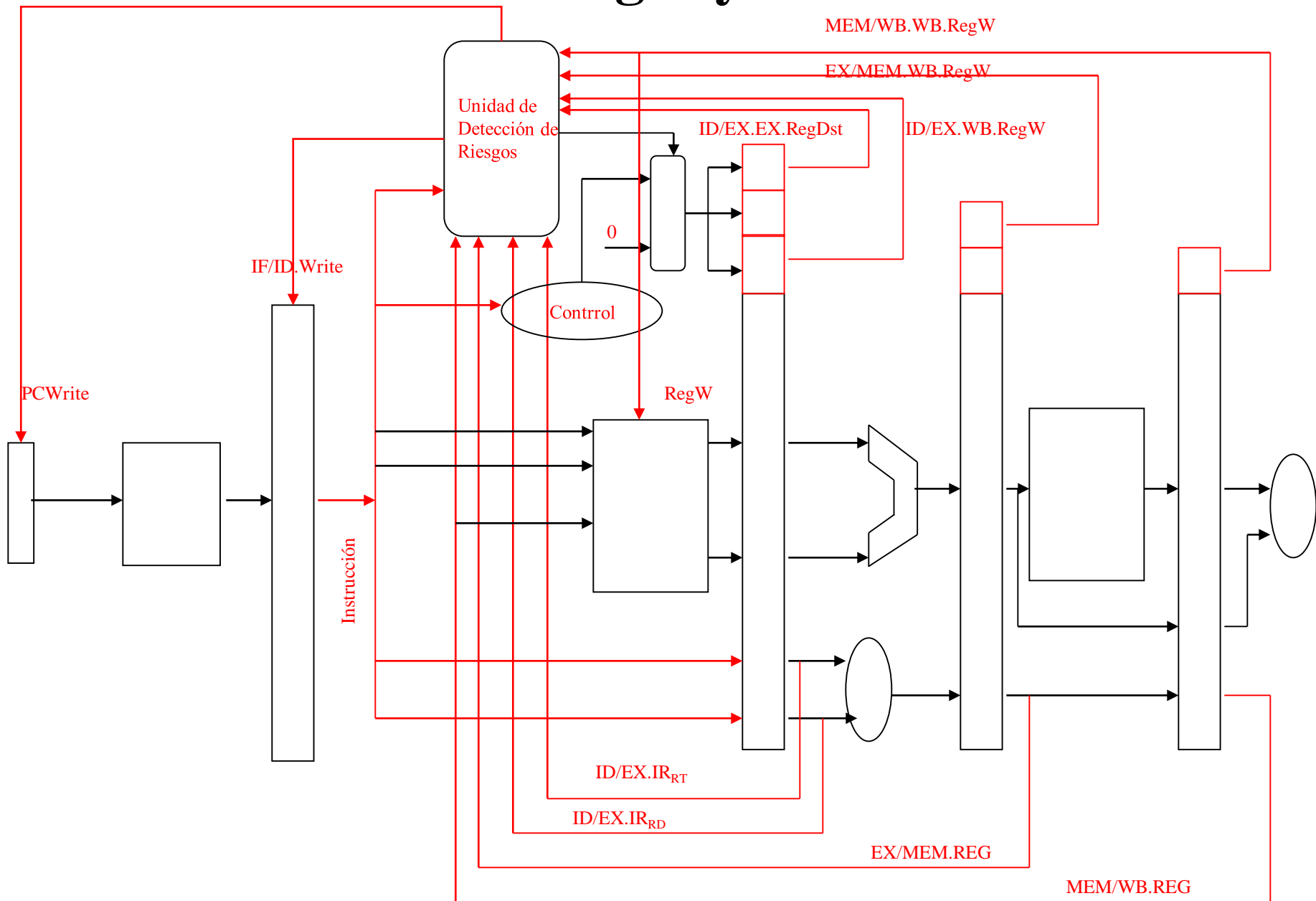
1 2 3 4 5 6 7 8



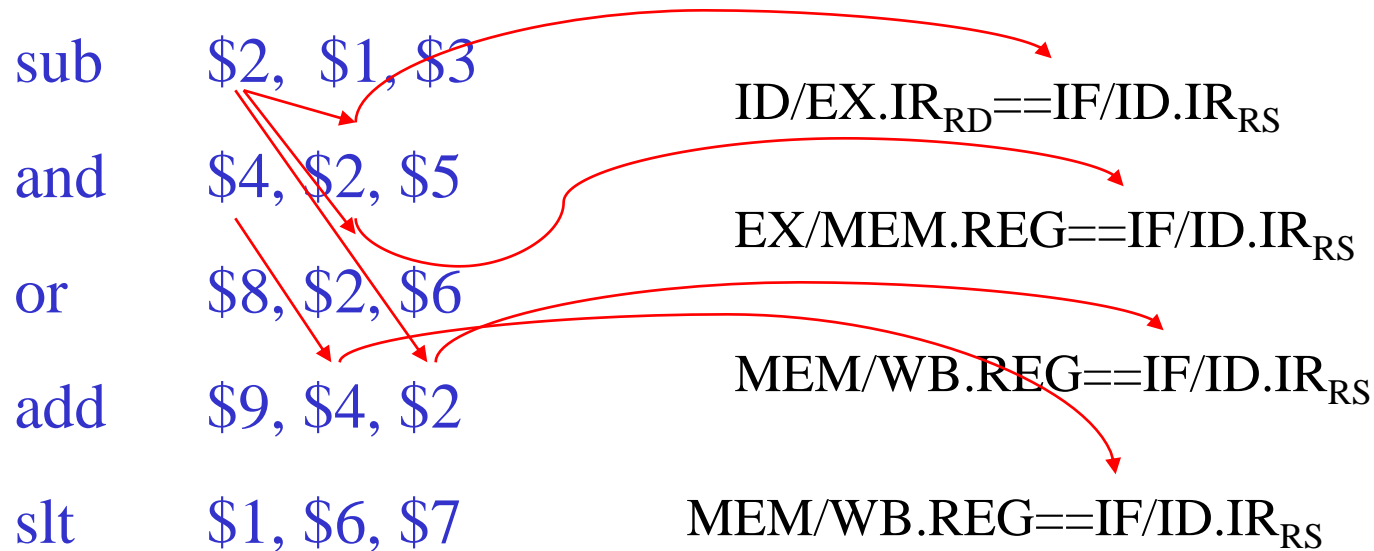
Detenciones

- Detener la instrucción que está en ID
 - No cambiar el PC
 - No cambiar IF/ID
- Insertar NOP's : tantos como ciclos tenga que durar la detención
 - 3 si el riesgo es EX
 - 2 si el riesgo es MEM
 - 1 si el riesgo es WB
- Como se insertan NOP's?
 - Generando ceros en las señales de control (EX-MEM-WB)

Control de Riesgos y Detenciones



Cuantos ciclos necesitamos para ejecutar la secuencia?

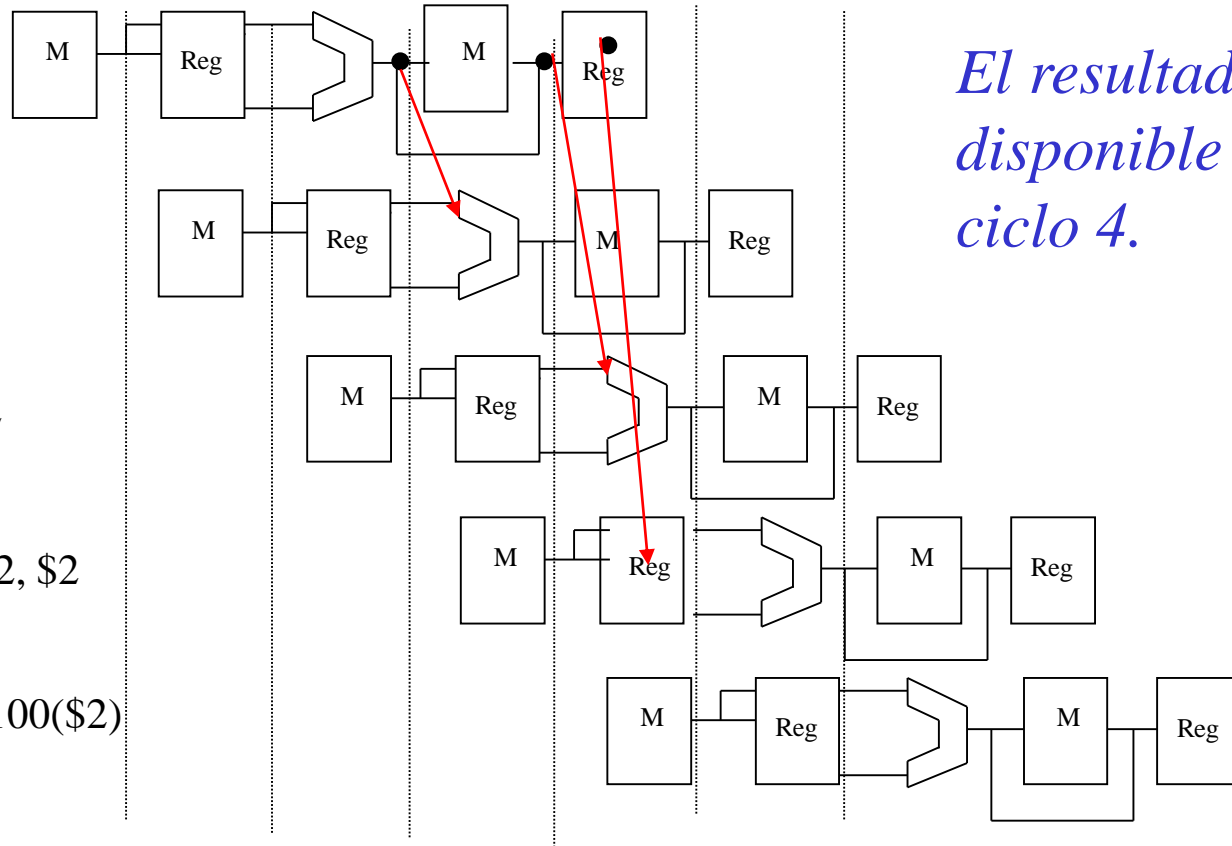


- Cual es el CPI medio?
- Indicar qué instrucciones se están ejecutando durante el ciclo 9.

Cortocircuitos: R-R

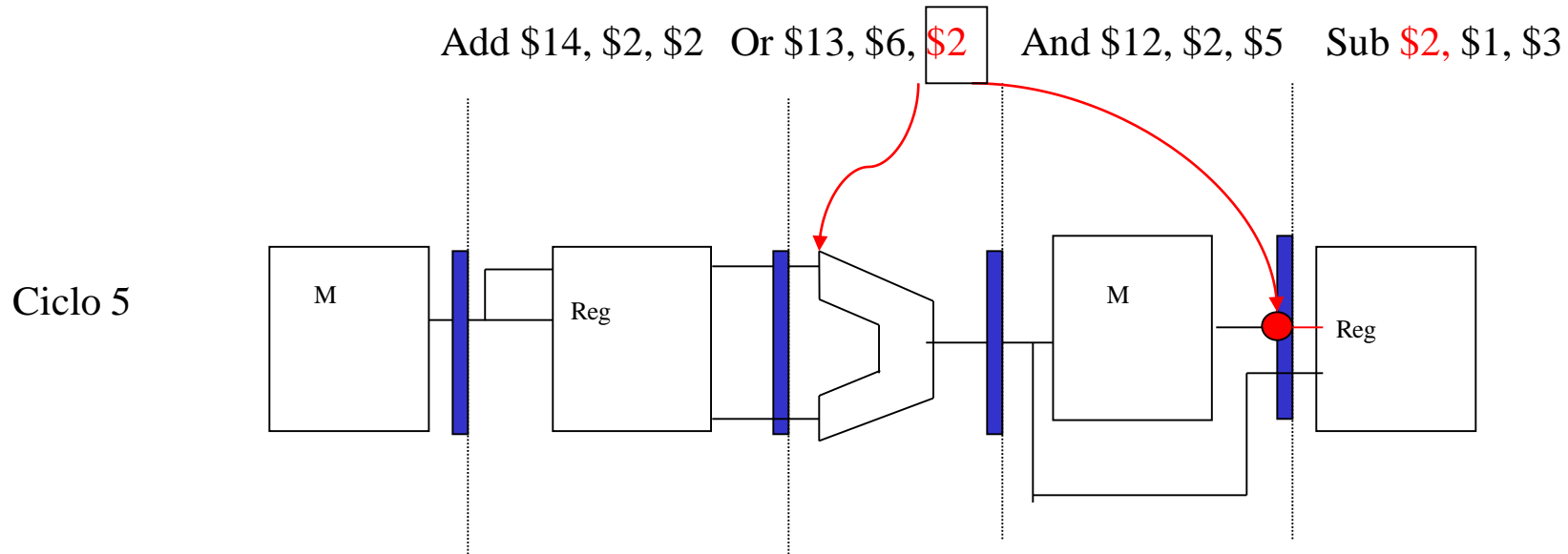
ciclos

1 2 3 4 5 6 7 8



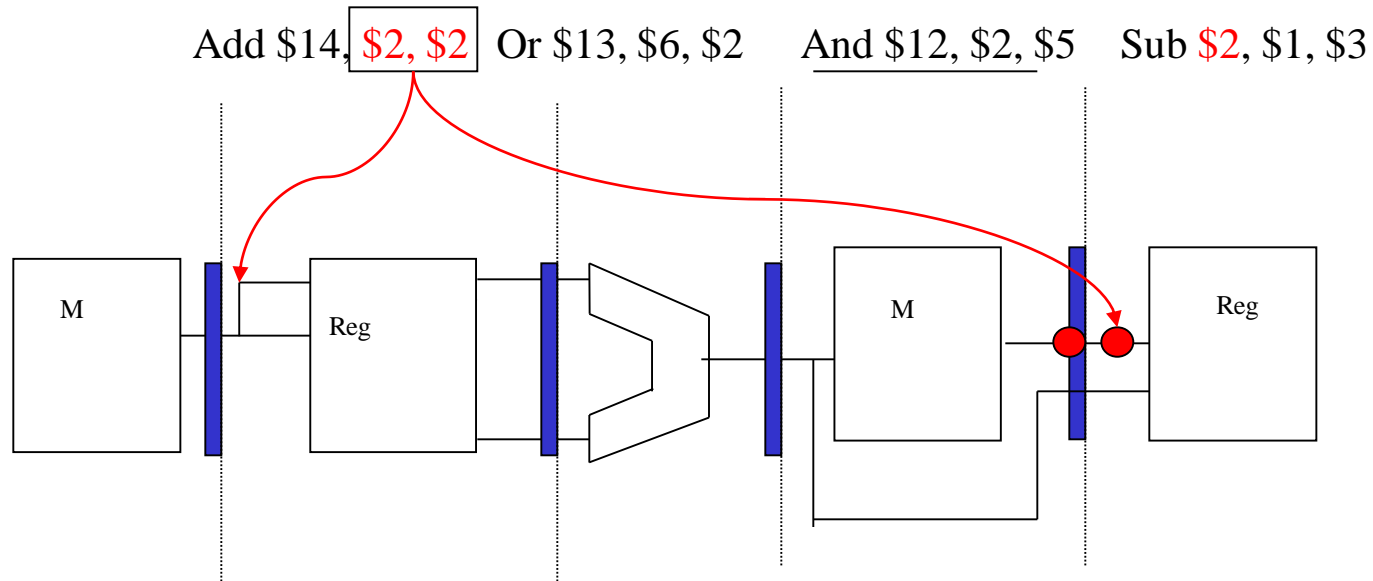
El resultado está disponible a partir del ciclo 4.

Cortocircuitos: Riesgo MEM



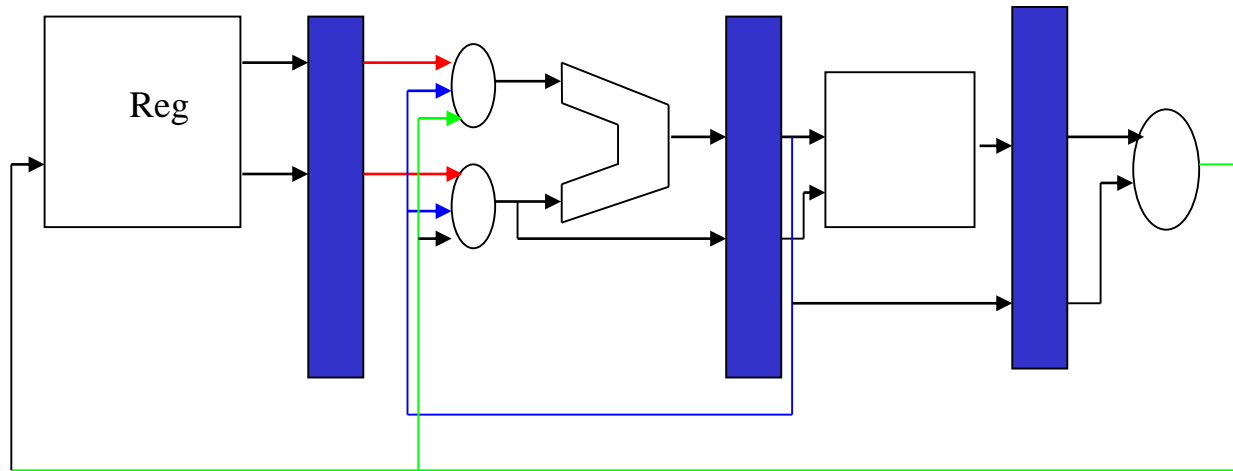
- El valor a la entrada de la ALU debe provenir del registro de segmentación M/WB.S

Cortocircuitos: Riesgo WB



- Como la UF es la misma: el Banco de Registros tiene la capacidad de suministrar el dato que se va a escribir por el BusA o el BusB.

Implementación



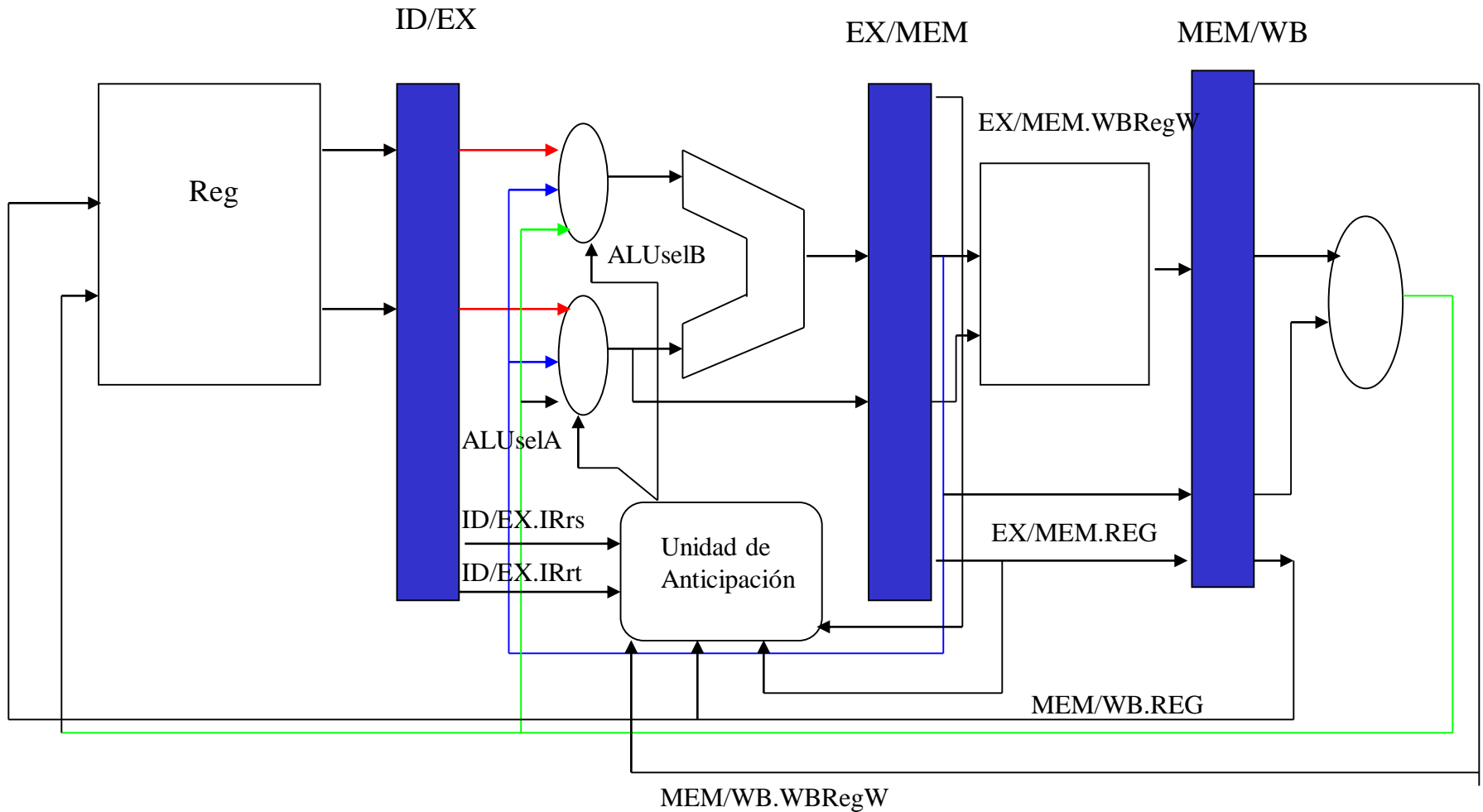
- La entrada a la ALU puede provenir de:
 - registro de segmentación ID/EX
 - registro de segmentación EX/MEM
 - registro de segmentación MEM/WB

Control de Anticipación:R-R

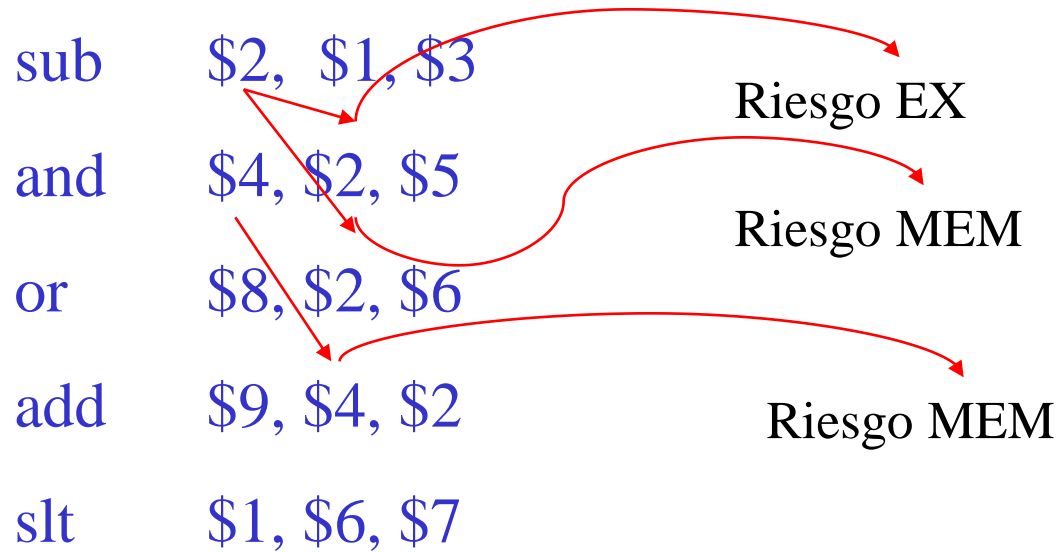
Reemplazamos la lógica de detección de riesgo por la lógica de anticipación de resultados.

- Riesgo EX:
 - $(EX/M.WBRegW==1 \text{ y } EX/M.REG==ID/EX.IR_{RS}) \text{ ALUselA}=01$
 - $(EX/M.WBRegW==1 \text{ y } EX/M.REG==ID/EX.IR_{RT}) \text{ ALUselB}=01$
- Riesgo MEM:
 - $(MEM/WB.WBRegW==1 \text{ y } MEM/WB.REG==ID/EX.IR_{RS})$
 $ALUselA=10$
 - $(MEM/WB.WBRegW==1 \text{ y } MEM/WB.REG==ID/EX.IR_{RT})$
 $ALUselB=10$

Lógica de Control R-R

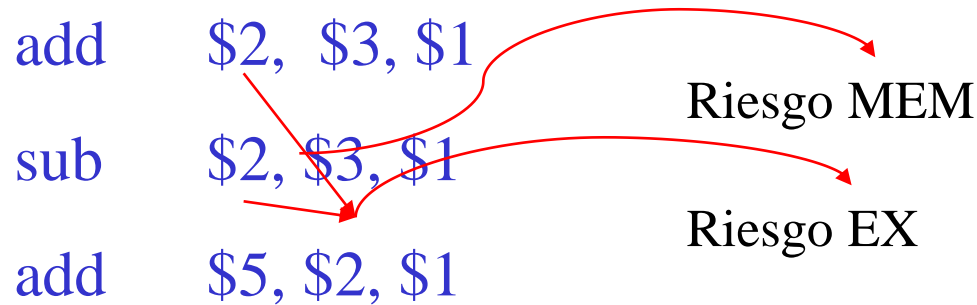


Cuantos ciclos necesitamos para ejecutar la secuencia con anticipación?



- Cual es el CPI medio? Comparar con detenciones.

Consideremos

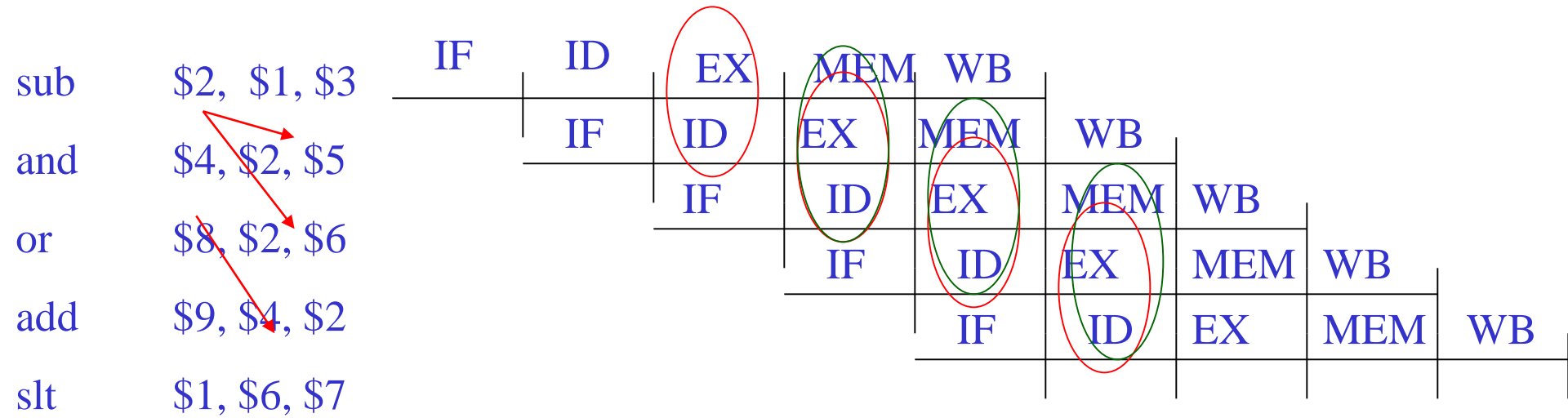


- Son dos riesgos simultáneos: uno EX y otro MEM
- Cuando la instrucción **add** llega a la etapa de ejecución, la lógica de anticipación comprueba que se cumplen las dos condiciones a la vez:
 - $(EX/M.WBRegW == 1 \text{ y } EX/M.REG == ID/EX.IR_{RS})$
 - $(MEM/WB.WBRegW == 1 \text{ y } MEM/WB.REG == ID/EX.IR_{RS})$

Control de Anticipación: R-R

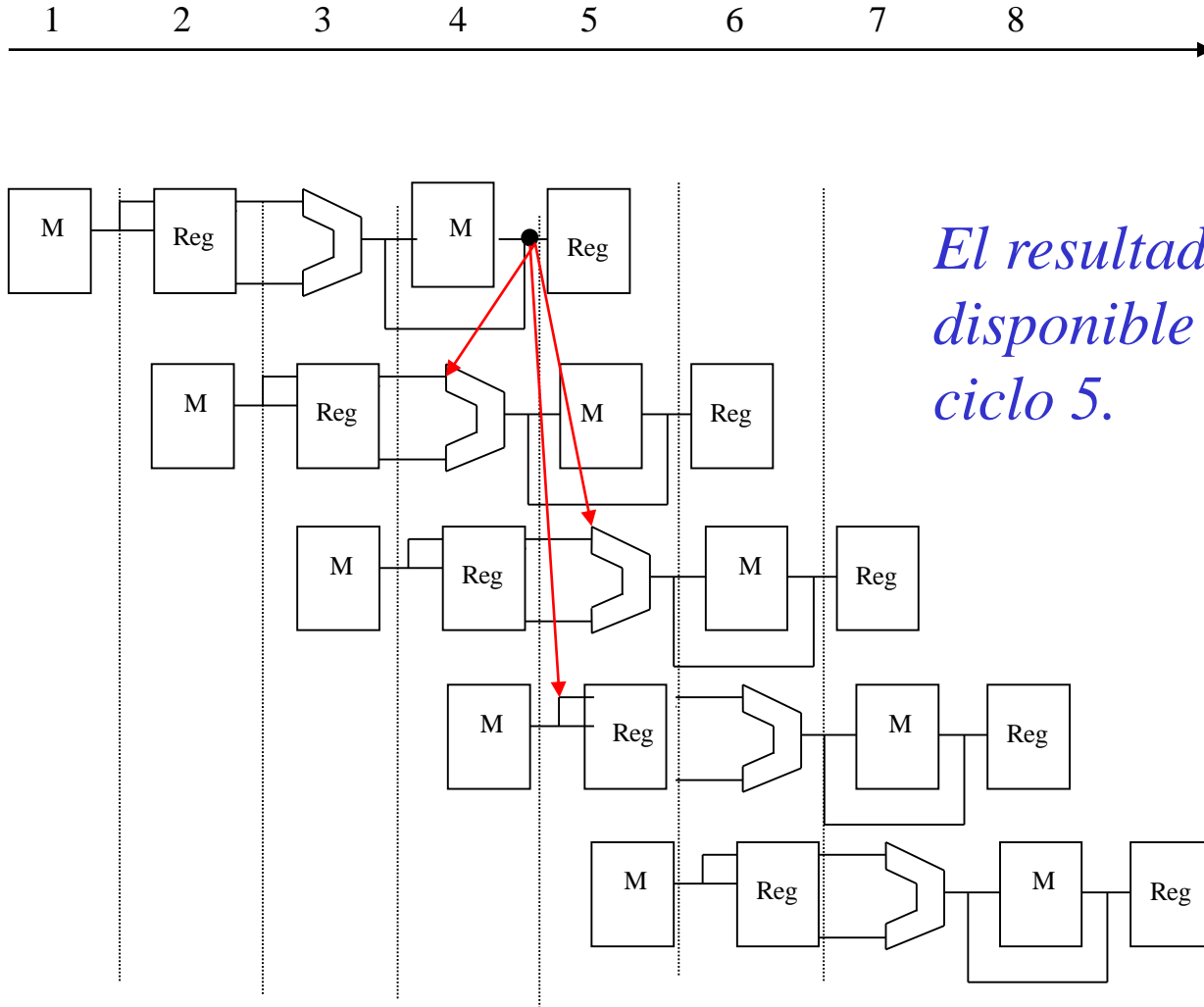
- Se debe dar prioridad a la instrucción más reciente
- Riesgo EX:
 - $(EX/M.WBRegW==1 \text{ y } EX/M.REG==ID/EX.IR_{RS}) \text{ ALUselA}=01;$
 $X=1$
 - $(EX/M.WBRegW==1 \text{ y } EX/M.REG==ID/EX.IR_{RT}) \text{ ALUselB}=01; X=1$
- Riesgo MEM:
 - $(MEM/WB.WBRegW==1) \text{ y } (MEM/WB.REG==ID/EX.IR_{RS})$
si $(X==0) \text{ ALUselA}=10$
 - $(MEM/WB.WBRegW==1) \text{ y } (MEM/WB.REG==ID/EX.IR_{RT})$
si $(X==0) \text{ ALUselB}=10$

Lógica de Anticipación en la etapa Decode



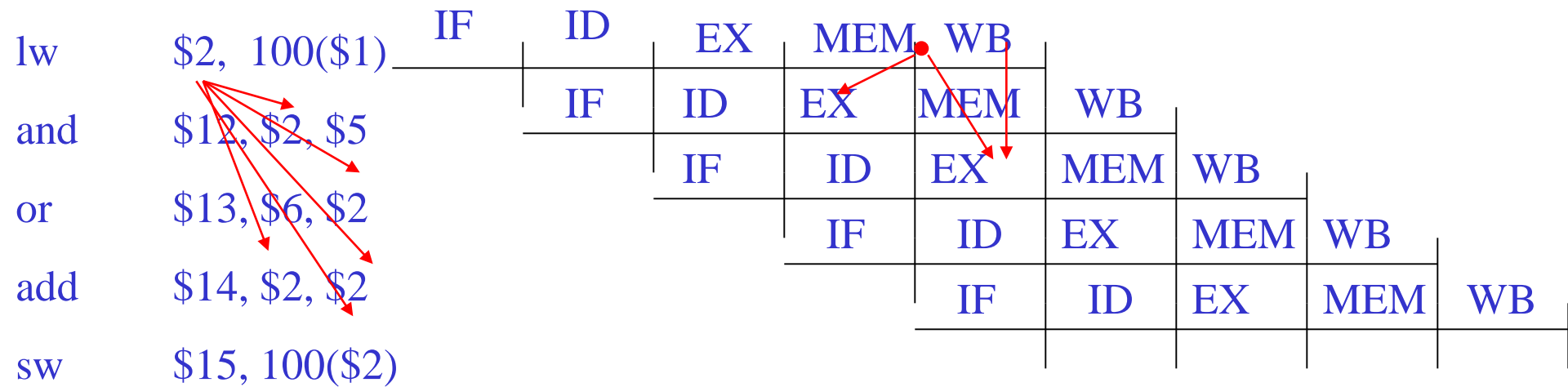
Cortocircuitos: Lw-R

ciclos



El resultado está disponible a partir del ciclo 5.

Lógica de Anticipación para cargas



- En el cuarto ciclo el dato aún no ha sido leído==> hay que detener la segmentación.
- Se necesita la unidad de detención de forma conjunta con la unidad de anticipación.

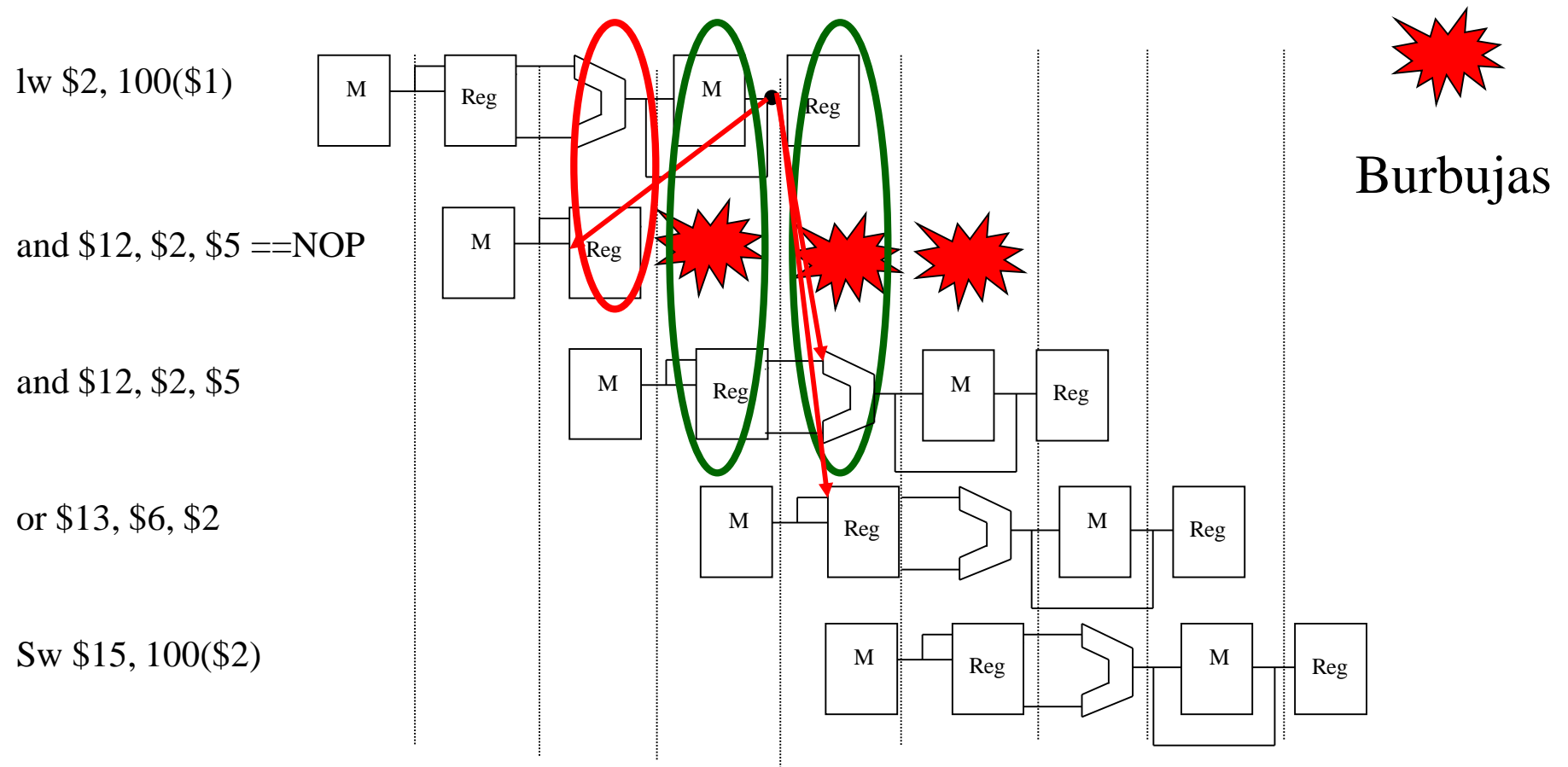
Control de Riesgos Ex para Detención

- La dependencia entre un load y una siguiente lectura del operando NO puede ser resuelta==>>>hay que detener
- Riego EX:
 - SI (ID/EX.WBRegW==1)y (ID/EX.EXRegDst==0)
y ((ID/EX.IR_{RT}==IF/ID.IR_{RS}) o (ID/EX.IR_{RT}==IF/ID.IR_{RT}))
entonces DETENER la segmentación

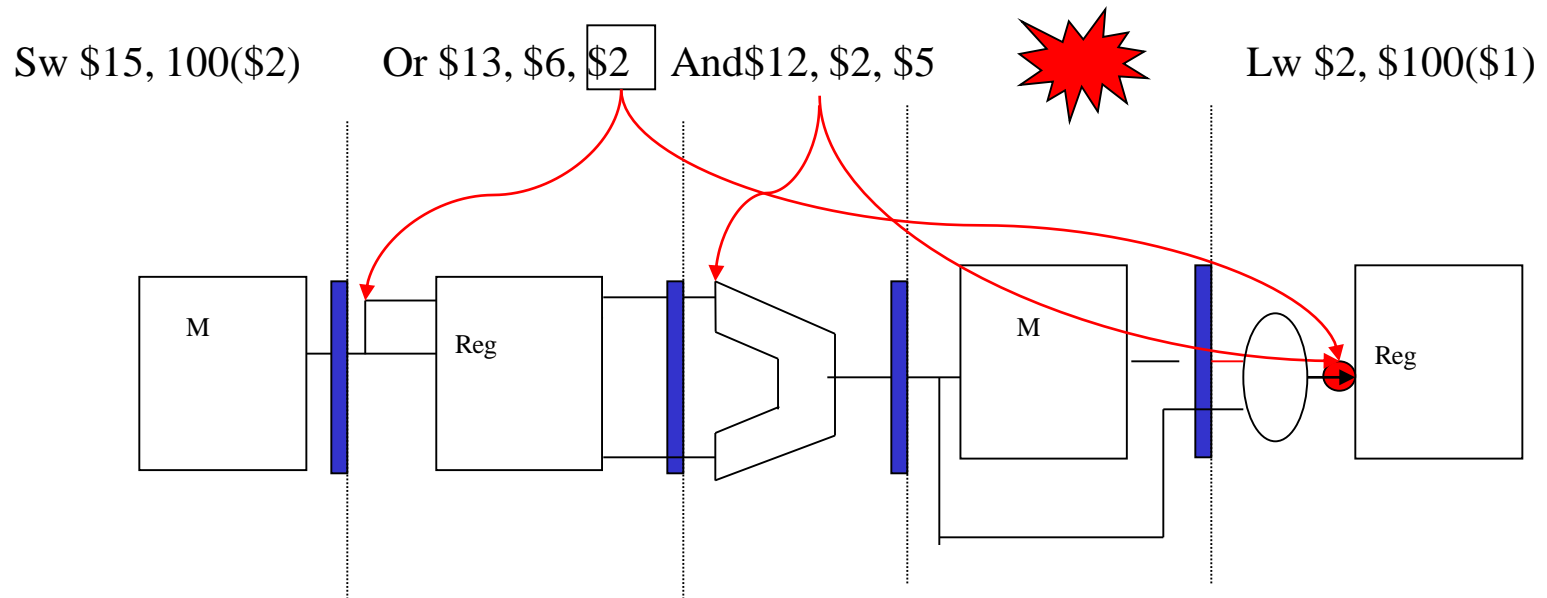
Detenciones y Anticipación para la carga

ciclos

1 2 3 4 5 6 7 8



Cortocircuitos: Riesgo MEM y WB



- El riesgo WB queda eliminado porque el Banco de Registros permite leer el dato que se está por escribir en el mismo ciclo

Solución Estática: Carga Retardada

- Problema: el resultado de la instrucción de carga no está disponible en el pipeline cuando la siguiente instrucción lo requiere.
- Se define como *load use delay* (retardo de uso de la instrucción de carga) a la diferencia de tiempo (número de ciclos) entre la disponibilidad del dato y su requerimiento de uso por una instrucción posterior
- El valor del retardo depende del tipo de memoria accedida y de la estructura del pipeline.
- Asumiendo que la memoria tiene una latencia de un único ciclo, la estructura del pipeline MIPS impone **un ciclo de retardo**.

Solución Estática: Carga Retardada

- La solución puede provenir del hardware o bien del compilador.
- La solución estática consiste en insertar una instrucción independiente de la carga en el slot de retardo (delay slot)
- En los tempranos R2000 y R3000, el hardware ASUMÍA que el compilador llenaba el delay slot o bien con una instrucción independiente, o bien con una NOP.

Solución Estática: Carga Retardada

– Ejemplo: $A=B+C$

$D=E-F$

Lw RB, B

Lw RC, C

Add RA, RB, RC

Sw RA, A

Lw RE, E

Lw RF, F

Sub RD, RE, RF

Sw RD, D

Lw RB, B

Lw RC, C

Lw RE, E

Add RA, RB, RC

Lw RF, F

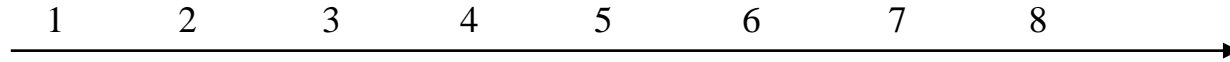
Sw RA, A

Sub RD, RE, RF

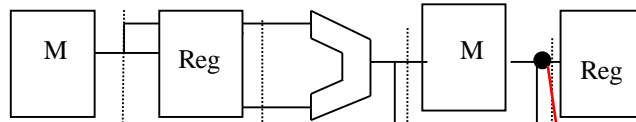
Sw RD, D

Cortocircuitos: Lw-Sw

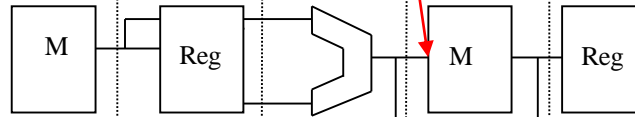
ciclos



Lw \$2, \$100(\$1)



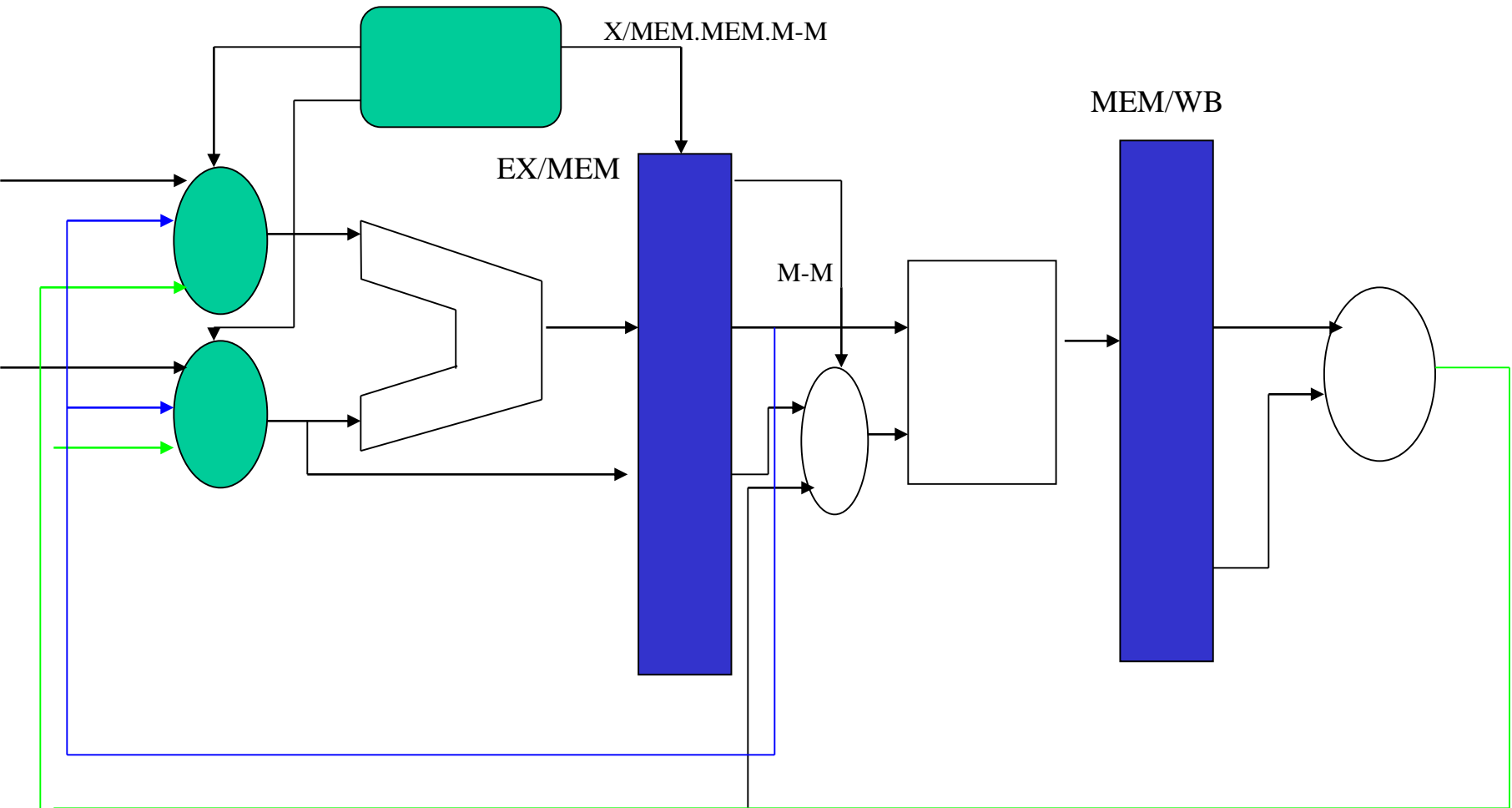
Sw \$2, \$100(\$5)



El resultado está disponible a partir del ciclo 5.

El store lo necesita en el mismo ciclo 5, pero como entrada a la memoria.

Cortocircuitos: M-M



Control de Anticipación: M-M

- La lógica de anticipación está puesta en X

- Riesgo MEM:

((EX/M.WBRegW==1 y EX/M.WBRegDst==0) /*identifica lw*/
y (EX/M.REG==ID/EX.IR_{RT}) /*identifica dependencia*/
y (ID/EX.WB.WE==1)) /*identifica sw*/
M-M=1 else 0

Resumen

- Lógica de Anticipación (EX)
 - Anticipa el resultado de una operación que está en M.
 - Ejemplo: R-R
 - Anticipa el resultado de una operación que está en WB.
 - Ejemplo: R-R; Lw-R
 - Anticipa el resultado de una operación de carga que está en WB a una instrucción de almacenamiento que está en M.
 - Ejemplo: Lw-Sw
- Lógica de Detención (DECODE)
 - Detiene el pipe cuando el riesgo es lw-R

Resumen

- Por construcción del pipeline:
 - No existen riesgos entre instrucciones de acceso a memoria
 - Ejemplo: Lw-Lw; Sw-Sw
 - No existen otros riesgos de datos salvo los debidos a dependencias de flujo.
 - Hay un ciclo de retardo debido a dependencias de datos en las instrucciones de carga
 - Ejemplo: Lw-R
 - Esto obliga a detener el pipe y entonces el CPI de la instrucción detenida AUMENTA.
 - Los riesgos se comprueban en la etapa de decodificación. Si no existen, la instrucción se EMITE. Si existen, se DETIENE.
 - Cuando una instrucción pasa de ID a X, se dice que ha sido emitida. (ISSUE).

Problema

- El 20% de las instrucciones son Lw.
 - El 50 % de las veces, la siguiente instrucción usa el resultado del Lw.
 - Cuantas veces es más rápida la máquina ideal sin detenciones (CPI=1) respecto a ésta?
-

- CPI inst.que sigue al load=1*0,5+2*0,5=1,5
- CPImedio=0,8*1+0,2*1,5=0,8+0,3=1,1

$$\frac{R_{ideal}}{R_{det}} = \frac{T_{det}}{T_{ideal}} = \frac{1,1}{1} = 1,1$$