

## Parte 2: Atascos RAW

### 1. Dependencia de datos ↗

Los **atascos RAW** son los más comunes en un programa. Para determinar si hay un atasco RAW entre dos instrucciones, primero hay que determinar si hay una **dependencia de datos de lectura** entre las mismas.

Las dependencias de datos de lectura ocurren cuando una instrucción B requiere el valor de un registro, pero debe esperar a que otra instrucción A escriba el valor de ese registro. En este caso, B es una instrucción que comienza a ejecutarse luego de A.

La dependencia de datos no siempre implica un atasco, porque si las instrucciones están muy alejadas en el tiempo la primera termina antes de que la segunda necesite el dato.

a) Los siguientes programas cortos contienen instrucciones que utilizan registros similares. Indicar en cada caso qué instrucciones tienen dependencias de datos de lectura entre ellas.

	1	2	3	4
1	daddi \$t1,\$0,5	ld \$t1, A(\$0)	daddi \$t1,\$0,4	daddi \$t1,\$0,0
2	daddi \$t2,\$0, 7	ld \$t2, B(\$0)	daddi \$t2,\$0,3	daddi \$t2,\$0,0
3	slt \$t3, \$t1, \$t2	bne \$t1, \$t2, no	daddi \$t3,\$0,0	loop: ld \$t3,A(\$t1)
4	daddi \$t1,\$0,1	daddi \$t3,\$0,1	loop: dadd \$t3,\$t3,\$t2	dadd \$t2,\$t2,\$t3
5	and \$t4, \$t3, \$t1	j fin	daddi \$t1,\$t1,-1	daddi \$t1,\$t1,8
6	daddi \$t1, \$0, 8	no: daddi \$t3,\$0, 0	bnez \$t1, loop	bnez \$t3, loop
7	sd \$t4, A(\$t1)	fin: sd \$t3, C(\$0)	sd \$t3, res(\$0)	sd \$t2, RES(\$0)

#### Ejemplo con el programa 1:

3 depende de 1 (por \$t1) y 2 (por \$t2)

5 depende de 3 (por \$t3) y 4 (por \$t1)

7 depende de 5 (por \$t4) y 6 (por \$t1)

#### Programa 2:

3 depende de 1 (por \$t1) y 2 (por \$t2)

7 depende de 5 (si son iguales) y/o 6 (si no son iguales)

#### Programa 3:

4 depende de 2 (por \$t2) y 3 (por \$t3) y 4 (por \$t3) en cada vuelta del loop

6 depende de 5 (por \$t1)

7 depende 4 por (\$t3)

#### Programa 4:

3 depende de 1 (por \$t1)

4 depende de 2 (por \$t2) y 3 (por \$t3)

5 depende de 5 (por \$t1) en cada vuelta del loop

6 depende de 3 (por \$t3)

7 depende 4 por (\$t2)

b) ¿Cuáles de las dependencias de datos por lectura te parece que causarán atascos RAW? Probar los programas en el simulador y anotar la cantidad de atascos y CPI de cada uno. **Nota:** Ignorar los atascos por “Branch Taken Stall” (los veremos más adelante).

- Programa 1: 0 atascos | 1.5 CPI ( 12 ciclos / 8 instrucciones)

- Programa 2: 2 atascos | 2.333 CPI ( 14 ciclos / 6 instrucciones)

- Programa 3: 4 atascos | 1.706 CPI (29 ciclos / 17 instrucciones)
- Programa 4: 7 atascos | 1.813 CPI (29 ciclos / 16 instrucciones)

c) Modificar los programas para que se reduzca la cantidad de atascos RAW, reordenando las instrucciones de forma que el resultado final del programa sea el mismo. Comparar la cantidad de atascos y CPI de cada uno con el caso anterior.

## 2) Atascos RAW y forwarding

En el ejercicio previo, vimos que reordenando las instrucciones se puede obtener una mejora en el CPI del programa. Otra forma de solucionar los atascos por dependencia de datos es utilizando el Adelantamiento de Operandos o Forwarding. Un procesador con Forwarding tiene un hardware modificado que permite que menos dependencias de datos se conviertan en atascos RAW. Para lograrlo, utiliza dos estrategias complementarias:

1) El valor que se calcula en las etapas EX o MEM está disponible para que otras instrucciones lo accedan ni bien se calcula, y no se requiere esperar a la etapa WB. Por ejemplo, la instrucción `daddi $t1,$t2,5` tendrá disponible el resultado de la suma al finalizar la etapa EX.

2) Las instrucciones no requieren todos sus operandos en la etapa ID. En lugar de eso, si necesitan un operando y no está disponible, se atascarán en la etapa en que lo necesiten realmente. Por ejemplo, si la instrucción `sd $t1, A($0)` llega a la etapa ID pero el valor de \$t1 todavía está siendo calculado por otra

instrucción, avanzará igual y se atascará recién en la etapa MEM.

Para analizar el efecto del forwarding, veamos el siguiente programa que intercambia el contenido de dos palabras de la memoria de datos, etiquetadas A y B.

- a) Ejecutarlo en el simulador con la opción Configure/Enable Forwarding deshabilitada.  
 ¿Cuántos atascos RAW hay?  
 ¿Cuál es el CPI?
- Hay 2 atascos RAW. 2.2CPI**
- b) Ejecutarlo en el simulador con la opción Configure/Enable Forwarding habilitada.  
 ¿Por qué no se presenta ningún atasco en este caso? Explicar la mejora.  
 ¿Cuál es el CPI?  
 ¿Qué indica el color de los registros en la ventana Register durante la ejecución?

1.8 CPI. No hay atascos ya que se adelanta la lectura del valor de \$t2 que es leido en memoria luego de la etapa MEM y disponible para la instrucción sd en la etapa de MEM que es donde se lo necesita. Que ese registro tiene un valor actualizado y puede ser utilizado de forma adelantada. El color depende de la etapa donde se generó el valor (MEM o EX)

### 3) Atascos RAW con lazos

Ignorando por ahora los atascos por salto (Branch Taken Stall), analizar el siguiente programa:

```
.data
A: .word 1
B: .word 3
C: .word 0
.code
ld $t1, A($0) ld $t2,
B($0)
loop: dsll $t1, $t1, 1
daddi$t2, $t2, -1
bnez$t2, loop sd $t1, C($0)
halt
```

- a) **Loop sin forwarding** Ejecutar el programa deshabilitando el Forwarding y responder:

- ¿Qué instrucciones generan los atascos tipo RAW y por qué? ¿En qué etapa del cauce se produce el atasco en cada caso y durante cuántos ciclos?
- ¿Cuántos CPI tiene la ejecución del programa en este caso?
- Cambiar el valor de B a 1000 y ejecutar. ¿Cómo cambió la cantidad de CPIs?

Los atascos lo generan los saltos condicionales y son 2 ciclos por loop

CPI = 26/13 = 2

Se mantuvo igual

- b) **Loop con forwarding** Ejecutar el programa con Forwarding habilitado y responder:

- ¿Cuántos CPI tiene la ejecución de este programa? Tomar nota del número de ciclos, cantidad de instrucciones y CPI. Comparar con el caso anterior.
- Cambiar el valor de B a 1000 y ejecutar. ¿Cómo cambió la cantidad de CPIs?

$$\text{CPI} = 22/13 = 1.692$$

$\text{CPI} = 5007/3004 = 1.667$ . Cambió la cantidad ya que hay menos atascos por RAW pero aparecen atascos por los saltos tomados.

c) Reordenamiento de instrucciones para optimización de CPI:

- Reordenar las instrucciones para que la cantidad de RAW sea 0 en la ejecución del programa, ejecutando con Forwarding **habilitado** y B=3.