



## ARQUITECTURA DE COMPUTADORAS

Trabajo Práctico N° 6  
**Pipeline - Secuenciamiento**  
Primer Cuatrimestre de 2010

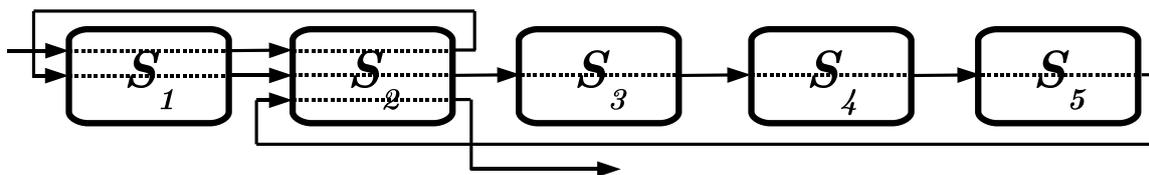
### Ejercicios

1. Suponiendo que un cierto pipeline presenta la siguiente *tabla de reservación*:

	1	2	3	4	5	6	7	8
$S_1$	×						×	
$S_2$		×	×	×				
$S_3$					×			
$S_4$						×		
$S_5$								×

- Determinar todas las *latencias críticas* así como el *vector de colisión* asociado.
- Encontrar todos los posibles *ciclos greedy*.
- Calcular la *latencia promedio mínima*.
- Analizar la evolución de los primeros cinco datos en el pipe al ser secuenciados según el régimen óptimo.
- ¿Qué sugiere la ocupación de las distintas etapas? ¿Cuál hubiese sido el límite de máximo régimen y por qué?

2. Asumiendo el siguiente esquema de secuenciamiento para un cierto pipeline:

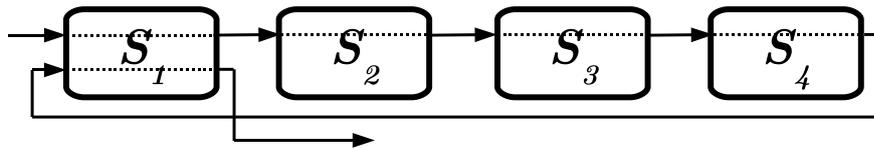


- Construir la *tabla de reservación* correspondiente.
- Determinar todas las *latencias críticas* así como el *vector de colisión* asociado.
- Encontrar todos los posibles *ciclos greedy*.
- Calcular la *latencia promedio mínima*.

3. Analizar la veracidad de la siguiente afirmación:

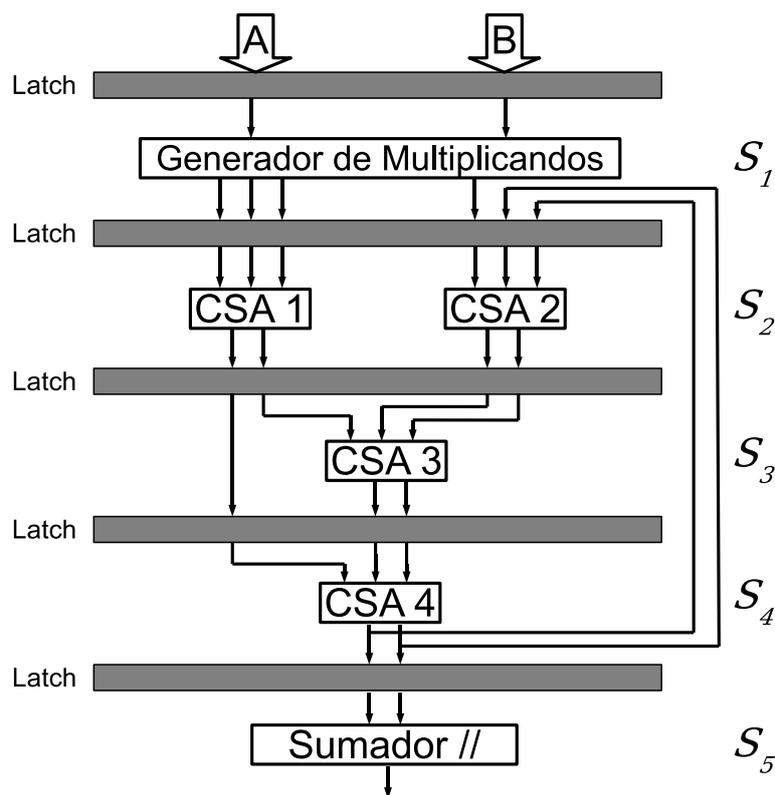
“La *latencia promedio mínima* será mayor ó igual a la cantidad de  $\times$  que figuren en la fila con mayor número de marcas.”

4. Encontrar la latencia promedio mínima y ciclos asociados a la misma para los pipelines unifuncionales descrito por el siguiente esquema de secuenciamiento y la siguiente tabla de reservación:



	1	2	3	4	5	6	7	8	9
$S_1$	×								×
$S_2$		×	×					×	
$S_3$				×					
$S_4$					×	×			
$S_5$							×	×	

5. A partir del siguiente árbol de CSA (parcial) que permite multiplicar dos operandos de 16 bits, calcular el tiempo de ejecución, las latencias críticas y el secuenciamiento óptimo a partir de su tabla de reservación. Tener en cuenta que los distintos componentes demandan un retardo de  $1T$ , salvo el sumador paralelo, el cual requiere  $2T$ .



6. Sea  $D$  un *stream de datos* sobre cuyos elementos se desean aplicar dos funciones,  $f_1$  y  $f_2$ . Es decir, para cada dato  $d$  que aparezca en  $D$ , se debe computar  $f_2(f_1(d))$ . Esta tarea será realizada mediante múltiples unidades funcionales estructuradas en sendos pipeline, uno de los cuales computa  $f_1$  y el otro computa  $f_2$ , y cuyas tablas de reservación efectivas son las siguientes:

	1	2	3	4	5	6	7	8
$S_1$	×						×	
$S_2$		×	×					
$S_3$				×				
$S_4$		×			×			×
$S_5$						×		
				$f_1$				

	1	2	3	4	5	6	7	8	9	10
$S_1$	×									×
$S_2$		×								×
$S_3$			×					×		
$S_4$				×				×		
$S_5$					×				×	
$S_6$						×				×
							$f_2$			

- a) ¿Cuál es el máximo *throughput* para  $f_1$  y  $f_2$ , asumiendo que trabajan de forma completamente independiente (digamos, tomando los datos de streams independientes  $D_1$  y  $D_2$ )?
- b) Si se estructura un pipeline en cadena, de modo que el buffer de salida de  $f_1$  sea el buffer de entrada de  $f_2$ , ¿cuál será el máximo *throughput* para las tareas sobre  $D$  bajo esta nueva configuración?
- c) ¿Qué se puede concluir sobre la efectividad de encadenar pipelines que presentan *feedbacks*? Considerar el efecto de contención de memoria y los requerimientos en ancho de banda de memoria.
7. En ocasiones algunos pipelines presentan un comportamiento un tanto llamativo, puesto que puede ser posible alcanzar ritmos de iniciaciones superiores mediante la incorporación de etapas de retardo.

Por caso, considerando la siguiente tabla de reservación:

	0	1	2	3	4	5
$S_1$	×		×			×
$S_2$		×	×		×	
$S_3$			×	×		

Determinar el mejor ritmo de iniciaciones incorporando retardos como se sugiere en la figura siguiente y verificar la posibilidad del ciclo (1, 5), el cual es óptimo.

	0	1	2	3	4	5	6	7	8	9	10
$S_1$	×		×	$d_1$			$d_2$	$d_3$	$d_4$	$d_5$	×
$S_2$		×	$d_6$	×		×					
$S_3$			×	$d_7$	×						

Notar que  $d_6$  retarda a  $S_2$  en la columna 2 y que  $d_1$  y  $d_7$  están para contemplar el hecho de que todas las etapas en una columna dada deben completarse antes de que alguna etapa de la próxima columna sea ejecutada. Para los retardos se emplean etapas sin función de cómputo, que eventualmente podrán ser compartidas por varios retardos elementales. Asumir para los retardos lo siguiente:

$S_4$ para $d_1$ y $d_2$	$S_7$ para $d_6$
$S_5$ para $d_4$	$S_8$ para $d_3$ y $d_7$
$S_6$ para $d_5$	

## Referencias

- [HB84] HWANG, K., AND BRIGGS, F. *Computer Architecture and Parallel Processing*. McGraw-Hill, 1984.