1) Simular el siguiente programa de suma de números en punto flotante y analizar minuciosamente la ejecución paso a paso. Inhabilitar Delay Slot y mantener habilitado Forwarding.

```
        .data
n1:     .double 9.13
n2:     .double 6.58
res1:   .double 0.0
res2:   .double 0.0
        .code
        l.d f1, n1(r0)
        l.d f2, n2(r0)
        add.d f3, f2, f1
        mul.d f4, f2, f1
        s.d f3, res1(r0)
        s.d f4, res2(r0)
        halt
```

a) Tomar nota de la cantidad de ciclos, instrucciones y CPI luego de la ejecución del programa.

b) ¿Cuántos atascos por dependencia de datos se generan? Observar en cada caso cuál es el dato en conflicto y las instrucciones involucradas.

c) ¿Por qué se producen los atascos estructurales? Observar cuales son las instrucciones que los generan y en qué etapas del pipeline aparecen.

d) Modificar el programa agregando la instrucción mul.d f1, f2, f1 entre las instrucciones add.d y mul.d. Repetir la ejecución y observar los resultados. ¿Por qué aparece un atasco tipo WAR?

e) Explicar por qué colocando un NOP antes de la suma, se soluciona el RAW de la instrucción ADD y como consecuencia se elimina el WAR.

Respuestas:

1)

a) La cantidad de ciclos de reloj es 16, la cantidad de instrucciones es 7by el CPI es 2,286.

b) La cantidad de atascos por dependencia de datos es 4.

- RAW en add.d f3, f2, f1: El dato en conflicto es f2 (el cual todavia no ha sido cargado desde memoria).

- 2 RAW en s.d f3, res1(r0): El dato en conflicto es f3 (el cual todavia no ha sido cargado con el valor de la suma en coma flotante).

- RAW en s.d f4, res2(r0): El dato en conflicto es f4 (el cual todavia no ha sido cargado con el valor de la multiplicacion en coma flotante).

c) Los atascos estructurales son provocados por conflictos por los recursos, es decir, cuando dos etapas del cauce requieren utilizar un mismo hardware. Las instrucciones que los generan son:

- s.d f3, res1(r0): Requiere acceder a memoria, pero en el mismo ciclo de reloj la instruccion de suma en punto flotante esta accediendo a la misma.

- s.d f4, res2(r0): Requiere acceder a memoria, pero en el mismo ciclo de reloj la instruccion de multiplicacion en punto flotante esta accediendo a la misma.

d) Aparece un atasco de tipo WAR el cual se da cuando una instrucción modifica un valor antes de que otra anterior que lo tiene que leer, lo lea. En este caso mul.d f1, f2, f1 precisa modificar f1 el cual la instruccion anterior (add.d f3, f2, f1) precisa leer. Se produce en la etapa ID ya que ante este potencial atasco el simulador genera un atasco de manera apresurada.

e) Al agregar la instruccion NOP hace que el cumplimiento de la etapa del cauce el cual tiene conflicto se retrase en un ciclo de reloj lo cual evita el atasco RAW y el WAR.