Universidad Católica San Pablo | Departamento de Ciencia de la Computación

Arquitectura de Computadores 2020-01 (CCOMP3-1.x)
Laboratorio 06c: Arquitectura en Pipelining

Yván Jesús Túpac Valdivia
Universidad Católica San Pablo, Arequipa – Perú

17 de junio de 2020

---

1 Objetivos

- Practicar con la arquitectura en *pipelining* para la ejecución de instrucciones MicroMIPS

2 Contenido base

- Diapositivas Ruta de datos y control en *Pipelining*
- Libro guía [Parhami, 2005]

Todos los alumnos antes de la sesión de laboratorio deberán revisar las clases "Ruta de datos y control en Pipelining" en el material de clases y en [Parhami, 2005]

3 Actividades

3.1 Dependencias de datos y control

Sea el mismo código MicroMIPS que fue analizado en la práctica 06b, a ejecutarse en una arquitectura pipelining:

```
# Rutina para hallar el mayor entero de la lista values y
# colocarlo en $t0.
      la   $s1,values      # remember how change this pseudo-instruction
                           # in real MIPS instructions
      lw   $t0,0($s1)      # initialize maximum to A[0]
      la   $s1,size
      lw   $s2,0($s1)
      addi $t1,$zero,0     # initialize index i to 0
loop: add  $t1,$t1,1       # increment index i by 1
      beq  $t1,$s2,done    # if all elements examined, quit
      add  $t2,$t1,$t1     # compute 2i in $t2
      add  $t2,$t2,$t2     # compute 4i in $t2
      add  $t2,$t2,$s1     # form address of A[i] in $t2
      lw   $t3,0($t2)      # load value of A[i] into $t3
      slt  $t4,$t0,$t3     # maximum < A[i]?
      beq  $t4,$zero,loop  # if not, repeat with no change
      addi $t0,$t3,0       # if so, A[i] is the new maximum
      j loop               # change completed; now repeat
done: ...                        # continuation of the program

.data
values: .word 3,10,15,9,7,12,9,20
size:   .word 6
```

a) Identifique e indique qué instrucciones generan dependencias de datos (tipo I y tipo II) y de control.

b) Si este código se ejecuta en un pipeline con *data forwarding*, determine dónde se debería insertar burbujas (cuantas), explicando el motivo para los siguientes casos:

   i) Sin posibilidad de reordenar instrucciones

   ii) Habiendo posibilidad de reordenar instrucciones

3.2 Calculo del CPI en procesadores en pipelining

Asumiendo que las dependencias de datos pueden resolverse mediante *data forwarding* o reordenamiento de instrucciones, y cada bifurcación debe esperar un ciclo de reloj para resolverse y poder colocar las instrucciones correctas.

a) Calcule la cantidad de ciclos de reloj que se requieren para la ejecución del código anteriormente analizado.

b) Calcule la CPI promedio (Ciclos de reloj por instrucción) alcanzada.

c) Considere que si en vez de esperar un ciclo de reloj por bifurcacion, se apuesta a que ésta nunca se cumple y si hay error (la bifurcación sí se cumple), se pagan tres ciclos de reloj en la reparación del *pipeline* (reposición de instrucciones correctas). Calcule la cantidad de ciclos de reloj y CPI para este caso

d) Compare con el CPI promedio obtenido si siempre se espera 1 ciclo de reloj ante cada bifurcación.

References

[Parhami, 2005] Parhami, B. (2005). *Computer Architecture: From Microprocessors to Supercomputers*. The Oxford Series in Electrical and Computer Engineering. OUP USA.