Arquitectura del Procesador II

Predicción de Saltos

Ejercicio 1: Para el siguiente fragmento de código:

```
        addi    $1, $0, 10
loop:   sw      $1, 0($2)
        addi    $2, $2, #4
        addi    $1, $1, -1
        bne     $1, $0, loop
        addi    $1, $0, 10
```

Muestre el diagrama de ciclos de reloj para el MIPS segmentado con unidad de adelantamiento para saltos, donde la dirección efectiva del salto se resuelve en la etapa ID, suponiendo:

1. Que se utiliza la política: Detener/Atascar la segmentación.
2. Que se utiliza la política: Salto no tomado.
3. Que se utiliza la política: salto demorado (planifique el ***slot*** de forma adecuada).

Compare estas mismas alternativas contra un procesador MIPS segmentado que no posea adelantamiento para saltos.

Ejercicio 2: Supongamos la arquitectura básica del MIPS segmentado de 5 etapas, sobre la cual se implementa una política de saltos demorado (delayed branch). Los saltos se resuelven en la etapa ID. Indique si las siguientes afirmaciones son ciertas o falsas:

□ a) La siguiente transformación es legal:

```
        sub r1,r2,r3                sub r1,r2,r3
        beqz  r1, Lab               beqz  r1, Lab1
        nop            ⟹           xor  r4,r2,r2
        add  r4,r1,r1               add  r4,r1,r1
        …                             …
Lab:    xor  r4,r2,r2       Lab:    xor  r4,r2,r2
        ld  r1,0(r4)        Lab1:   ld  r1,0(r4)
```

□ b) La siguiente transformación es legal :

```
add  r6,r2,r2          …
sub  r1,r2,r6          sub  r1,r2,r6
beqz  r1, Lab   ⟹      beqz  r1, Lab
nop                    add  r6,r2,r2
add  r5,r4,r6          add  r5,r4,r6
…                        …
```

□ c) Cuando el **delay slot** se rellena con una instrucción del destino del salto y el salto no se toma, entonces se produce una penalización de un ciclo de reloj.

□ d) La siguiente transformación es legal :

```
       sub r1,r2,r3            sub r1,r2,r3
       beqz  r1, Lab           beqz  r1, Lab
       nop                     add  r6,r1,r1
       add  r6,r1,r1    ⟹     xor  r8,r6,r5
       xor  r8,r6,r5           …
       …                         …
Lab:   sub  r5,r6,r7     Lab:   sub  r5,r6,r7
```

□ e) Si se cambia a una política de predicción de saltos como no tomado (predict-not-taken), entonces la instrucción que está a continuación del salto se lee igualmente de la memoria de instrucciones, pero su ejecución se cancela en caso de que el salto no se tome.

Ejercicio 3: Para el siguiente fragmento de código MIPS:

```
        addi    $1, $0, 10
loop:   sw      $1, 0($2)
        addi    $2, $2, #4
        addi    $1, $1, -1
        bnez    $1, loop
        NOP
        add     $29,$30,$31
```

4. Muestre el diagrama de ciclos de reloj de esta secuencia de instrucciones para el MIPS segmentado donde la dirección efectiva del salto se resuelve en la etapa ID y se utiliza un predictor de saltos dinámico de 2 bits y unidades de adelantamiento para instrucciones de salto.

5. Luego, suponiendo que se utiliza la política *salto demorado*, planifique la franja de instrucciones (slot) indicado por NOP y muestre el diagrama de ciclos de reloj donde la dirección efectiva del salto se resuelve en la etapa ID.