AIC Exàmens

Exàmens

UT2(2.1;2.2;2.3) Seguimiento

Pàgina d'índex

Temps restant: 0:35:56

Amaga/mostra el temps restant

Part 1 de 6 -

Pregunta 1 de 10

1 Punts

En el procesador MIPS segmentado, las instrucciones de carga insertan 1 ciclo de parada si la instrucción siguiente consume el dato leído de la memoria. Si el compilador, en esos casos, coloca instrucciones NOP entre ambas instrucciones, el valor del CPI obtenido por el programa (aumenta/disminuye/no cambia)

disminuye

Pregunta 2 de 10

1 Punts

Important: utilitza únicament el punt com a separador decimal.

Si un procesador segmentado calcula la dirección y condición de salto así como la escritura del PC en la fase 2 del ciclo de instrucción, cuando se emplea el salto retardado para resolver los riesgos del control, el tamaño del delay slot es de 1 instrucciones.

Pregunta 3 de 10

1 Punts. Punts descomptats en cas de resposta incorrecta: 0.3

Teniendo en cuenta la ruta de datos segmentada del procesador MIPS en las etapas IF, ID, EX, M y WB, y teniendo en cuenta que el ciclo de reloj es de 10 ns, indica qué respuesta es CIERTA:

- **El tiempo de lectura o escritura en el banco de registros no puede ser inferior a 10 ns**
- **La suma de retardos del registro de segmentación y la etapa mas lenta no puede superar los 10 ns**
- **La aceleración que se obtendría, en comparación a la ruta de datos sin segmentar, es de 5, independientemente de la duración de cada etapa**
- **Todas las etapas deben tener un retardo idéntico a 10 ns**

Esborra la selecció

Pregunta 4 de 10

1 Punts. Punts descomptats en cas de resposta incorrecta: 0.3

Una unidad segmentada se compone de 4 etapas de retardos 20ns,20ns,11ns,13ns. Los registros intermedios tienen un retardo de 2ns y el desfase del reloj es como máximo de 1ns. El periodo de reloj mínimo debería ser:

- ◯ **64 ns**
- ◯ **20 ns**
- ◯ **21 ns**
- ◯ **23 ns**

Esborra la selecció

Part 2 de 6 -

Pregunta 5 de 10

1 Punts. Punts descomptats en cas de resposta incorrecta: 0.3

Indica el número de ciclos de parada que aplicaría el procesador MIPS segmentado para resolver el riesgo generado por la secuencia de instrucciones mostrada. Considera que las latencias del multiplicador y del sumador son 4 y 3, respectivamente:

```
mul.d f0,f1,f2

add.d f0,f3,f4
```

- ◯ **4 stalls**
- ◯ **3 stalls**
- ◯ **1 stalls**
- ◯ **0 stalls**

Esborra la selecció

Pregunta 6 de 10

1 Punts

Dado el fragmento de código MIPS que se muestra a continuación:

```
1 bnez r1,loop
2 l.d f0,100(r10)
3 add.d f4,f0,f2
4 s.d f4,100(r10)
5 l.d f0,200(r10)
6 sub.d f4,f0,f3
7 s.d f4,200(r10)
```

relaciona cada par de instrucciones con un tipo de dependencia:

A. i1 y i6
B. i3 y i6
C. i2 y i3
D. i4 y i6

C	1. Dependencia de datos
D	2. Antidependencia
B	3. Dependencia de salida
A	4. Dependencia de control

Part 3 de 6 -

Pregunta 7 de 10

1 Punts

Sea un procesador con un predictor de saltos de 2 bits con saturación, donde se ejecuta un código con una instrucción de salto que implementa un bucle que realiza 100 iteraciones. Ten en cuenta que el procesador realiza una predicción predict-not-taken ante la ausencia de historia de la instrucción de salto, que el procesador no tiene historia reciente de la instrucción de salto, y que la primera vez que se inserta un salto en la tabla su estado será *strongly-taken* o *strongly-not taken*, según proceda. Se generarán en total 2 **fallos de predicción.**

Part 4 de 6 -

Pregunta 8 de 10

1 Punts

Feu clic per veure instruccions addicionals

Teniendo en cuenta la ruta de datos del procesador MIPS segmentada en cinco etapas (IF, ID, EX, M, WB), que aplica todos los cortocircuitos posibles para resolver conflictos de datos, que resuelve los conflictos de control mediante la técnica predict not-taken, que calcula la dirección de salto y modifica el PC en la etapa ID, y que no tiene ningún conflicto estructural, calcula el CPI para una iteración del siguiente código (redondea a dos decimales):

```
loop: ld r3, 0(r2)
      ld r4, 0(r3)
      dadd r1, r1, r4
      daddi r2, r2, 8
      daddi r10, r10, -1
      bnez r10, loop
      sd r1, 0(r11)
```

CPI = 1,86

Part 5 de 6 -

Pregunta 9 de 10

1 Punts

Dado el siguiente bucle:

```
loop:
      l.d f0, X(r10)
      mul.d f1,f0,f10
      add.d f2,f1,f11
      s.d f2,X(r20)
      ...
      <salto a loop>
```

Indica cómo sería el cuerpo del bucle de la versión en la que se ha aplicado software pipelining:

l.d f0, X(r10)
mul.d f1,f0,f10
l.d f0, X(r10)
add.d f2,f1,f11

Part 6 de 6 -

Pregunta 10 de 10

1 Punts

Sea el siguiente código en ensamblador:

```
nozero:     li $t0, 8          # Número de elementos del vector
            li $v0, 0          # contador inicial = 0
            li $t1, V          # dirección vector V
loop:       lw $t2, 0(t1)      # lectura V[i]
            addi $t0, $t0, -1 # Decrementa elementos vector
            bnez $t2, sigue    # Si V[i] es distinto de cero salta
            addi $v0, $v0, 1   # Incrementa contador
sigue:      addi $t1, $t1, 4   # Incrementa dirección vector V
            bnez $t0, loop     # Siguiente iteración
```

Dicho código implementa la función *nozero* que calcula el número de elementos de un vector de 8 elementos con valor distinto a cero.

El código se ejecuta en un procesador segmentado de 5 etapas el cual resuelve todos los conflictos de datos con cortocircuitos. El procesador implementa un BTB con un predictor de dos bits con histéresis. La tabla tiene 16 entradas e inicialmente está vacía. En ausencia de historia del salto se utiliza predict-not-taken. Un fallo de predicción ocasiona la inserción de dos ciclos de parada.

Indica cuantos ciclos de penalización introducirá cada instrucción de salto en la ejecución del código anterior para el caso de un vector que contenga elementos con los valores "0 1 0 1 0 1 0 1"

a) Ciclos de penalización `bnez $t2, sigue`: 8 **ciclos**

b) Ciclos de penalización `bnez $t0, loop`: 4 **ciclos**

Desa Lliura per a l'avaluació

- PoliformaT
- UPV
- Powered by Sakai