AMPLIACIÓN DE ESTRUCTURA DE COMPUTADORES

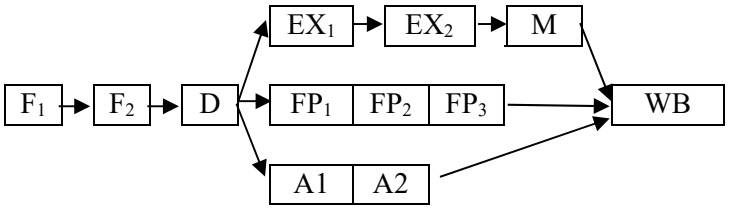
Grado en Ingeniería Informática

Examen Final – Problemas – 15 de mayo de 2023

- ❑ Grupo 1
- ❑ Grupo 2
- ❑ Grupo 3
- ❑ Grupo 9 (PCEO)

Apellidos, Nombre: _____

1. (*2 puntos*) Disponemos de un procesador con la siguiente estructura de *pipeline*:



Todas las etapas están totalmente segmentadas. Las etapas F1 y F2 realizan la carga de la instrucción de la caché de instrucciones (lo cual requiere dos ciclos), la etapa D realiza la decodificación y lectura de registros, EX1 y EX2 son las etapas de ejecución de las instrucciones enteras, FP1 a FP3 son las etapas que realizan la multiplicación en coma flotante, A1 y A2 realizan la suma o resta en coma flotante, M realiza el acceso a la caché de datos y WB realiza la escritura en el banco de registros. Se puede realizar una escritura y lectura del mismo registro en un solo ciclo. Los adelantamientos no se han implementado.

Se ha comprobado que por término medio los programas ejecutados tienen la siguiente distribución de instrucciones: 30% de ALU, 20% de cargas, 5% de almacenamientos, 10% de instrucciones de multiplicación en coma flotante, 5% de instrucciones de suma en coma flotante y 30% de saltos condicionales, de los cuales el 70% son saltos tomados. Los saltos se manejan deteniendo el cauce y se resuelven en la etapa EX2. Se ha visto, además, que:

- Un tercio de las cargas van seguidas inmediatamente por una instrucción (ej. ALU) que necesita el valor de la carga.
- El 60% de las instrucciones ALU van seguidas inmediatamente por otra instrucción que necesita el dato.
- El 30% de las instrucciones ALU tiene inmediatamente después en el programa una instrucción de salto condicional que requiere el dato calculado por ellas.
- El 35% de las instrucciones de multiplicación en coma flotante van seguidas inmediatamente por una instrucción de suma en coma flotante que necesita el dato.

a) *(1 punto)* Considerando *únicamente* las situaciones descritas en los ítems de arriba, describe los riesgos que podrán aparecer en el cauce poniendo un ejemplo de instrucción/secuencia de instrucciones, y cuantifica la incidencia que cada uno de estos riesgos identificados tendría sobre el rendimiento final (cuánto aumenta el CPI).

b) *(1 punto)* Repite el apartado *a)* pero ahora suponiendo que al procesador se le añaden todos los adelantamientos posibles y, además, los saltos se predicen estáticamente como *no-tomados*.

2. (*2 puntos*) Supongamos un procesador con cachés de instrucciones y datos separadas con tiempos de acierto de 2 y 3 ciclos respectivamente, y tasas de fallos de 3% y 6% respectivamente. La caché de datos es de post-escritura con búsqueda de bloque. La penalización por fallo en ambas cachés es de 50 ciclos y hay un 30% de bloques sucios en la de datos. La proporción de instrucciones que ejecutamos es: 25% cargas, 10% almacenamientos, 50% operaciones ALU y 15% saltos condicionales. Los saltos condicionales se resuelven en la etapa ID del *pipeline* y se aplica la estrategia de predecir estáticamente como "no tomado". Además, se sabe que sólo el 25% de los saltos son tomados. Se pide:

a) *(0.75 puntos)* Calcular el tiempo medio de acceso a memoria.

b) *(0.75 puntos)* Calcular el CPI considerando la jerarquía de memoria.

c) *(0.5 puntos)* No hemos tenido en cuenta el hecho de que ambas cachés son indexadas y etiquetadas físicamente. Supongamos que tenemos un TLB de instrucciones y otro de datos con tasas de fallo de 2% y 3% respectivamente, un tiempo de servicio de 2 ciclos en ambos casos y una penalización por fallo de 150 ciclos. ¿Cuál sería ahora el tiempo medio de acceso a memoria?

SOLUCIONES

SOLUCIÓN EJERCICIO 1

a) Los riesgos que pueden surgir serían los siguientes:

1. Riesgos de control: causados por las instrucciones de salto condicional. En concreto, puesto que los saltos paran el cauce hasta que se resuelven, cada salto condicional introduce 4 ciclos de parada al resolverse en EX2. Esto supone incrementar el CPI en 0,3×4 ciclos.

2. Riesgos de datos (enteros): surgen como consecuencia de que determinadas instrucciones vayan inmediatamente seguidas de instrucciones que consumen el dato. Dado que el procesador no implementa adelantamientos, los riesgos de datos se manejan introduciendo 3 ciclos de parada cada vez que aparecen. En concreto, tenemos tres situaciones en los que aparecen:

- Carga→ALU: ejemplo: LW R10,0(R2) // ADD R1, R10, R1. Dado que únicamente para el 33% de las instrucciones de carga se observaría esta situación, el incremento que sufre el CPI es 0,2×0,33×3
- ALU→ALU o Memoria: ejemplo ADD R1, R10, R1 // ADD R2, R1, R2. Dado que para el 60% de las instrucciones de ALU se observaría esta situación, el incremento que sufre el CPI es 0,3×0,6×3
- ALU→Salto: ejemplo ADD R1, R10, R1 // BNEZ R1, Etiq. El 30% de instrucciones ALU se encuentran con esta situación. De esta forma, el CPI se incrementaría 0,3×0,3×3

3. Riesgos de datos (coma flotante): surgen como consecuencia de que determinadas instrucciones vayan inmediatamente seguidas de instrucciones que consumen el dato en coma flotante. Tenemos el siguiente caso:

- Multiplicación→Suma: ejemplo: MULFP R1, R2, R3 // ADDFP R4, R1, R5. Dado que únicamente para el 35% de las instrucciones de multiplicación en coma flotante se observaría esta situación, el incremento que sufre el CPI es 0,1×0,35×3

Considerando lo anterior, el CPI sería:

CPI = 1 + 0,3x4 + 0,2x0,33x3 + 0,3x0,6x3 + 0,3x0,3x3 + 0,1x0,35x3 =
CPI = 1 + 1,2 + 0,198 + 0,54 + 0,27 + 0,105 = **3,313 ciclos**

b) Consideramos ahora la inclusión de los adelantamientos estudiados en clase en el procesador y el manejo de los saltos prediciéndolos estáticamente como no tomados. Los riesgos que pueden surgir ahora serían los siguientes:

1) Riesgos de control: causados por las instrucciones de salto condicional. En concreto, cuando el salto condicional se toma hay 4 ciclos de parada al resolverse en EX2. Esto supone incrementar el CPI en 0,3×0,7×4 ciclos.

2) Riesgos de datos: el uso de la técnica de adelantamiento elimina todos los riesgos de datos mostrados con anterioridad salvo cuando el siguiente caso:

a. Carga→ALU: ejemplo: LW R10,0(R2) // ADD R1, R10, R1. Dado que el procesador implementa ahora adelantamientos, el riesgo se maneja introduciendo 2 ciclos de parada. También hay que notar que únicamente para el 33% de las instrucciones de carga se observaría esta situación. De esta forma el incremento que sufre el CPI es 0,2×0,33×2.

b. ALU→ALU o Memoria: ejemplo ADD R1, R10, R1 // ADD R2, R1, R2. Dado que el procesador implementa ahora adelantamientos, el riesgo se maneja introduciendo 1 ciclo de parada. Dado que para el 60% de las instrucciones de ALU se observaría esta situación, el incremento que sufre el CPI es 0,3×0,6×1

c. ALU→Salto: ejemplo ADD R1, R10, R1 // BNEZ R1, Etiq. Dado que el procesador implementa ahora adelantamientos, el riesgo se maneja introduciendo 1 ciclo de parada, siempre que se considere que la unidad detectora de ceros se encuentra segmentada en las etapas EX1 y EX2. Además, el 30% de las instrucciones de ALU se encuentran con esta situación. De esta forma, el CPI se incrementaría 0,3×0,3×1. Si se considera que la unidad detectora de ceros sólo se encuentra en la etapa EX2, entonces se puede hacer un adelantamiento a EX2 y no habría ningún ciclo de parada.

3) Riesgos de datos (coma flotante): surgen como consecuencia de que determinadas instrucciones vayan inmediatamente seguidas de instrucciones que consumen el dato en coma flotante. Tenemos el siguiente caso:

- Multiplicación→Suma: ejemplo: MULFP R1, R2, R3 // ADDFP R4, R1, R5. Dado que únicamente para el 35% de las instrucciones de multiplicación en coma flotante se observaría esta situación, el incremento que sufre el CPI es 0,1×0,35×2

Considerando lo anterior, el CPI sería:

CPI = 1 + 0,3x0,7x4 + 0,2x0,33x2 + 0,3x0,6x1 + 0,3x0,3x1 + 0,1x0,35x2 =
CPI = 1 + 0,84 + 0,132 + 0,18 + 0,09 + 0,07 = **2,312 ciclos**

SOLUCIÓN EJERCICIO 2

a) El tiempo medio de acceso a memoria para la configuración de cachés dada en el ejercicio vendría dado por la siguiente fórmula:

$$T_{AM} = 1/1{,}35\times\underbrace{(2 + 0{,}03\times 50)}_{\text{t. acceso i-cache}} + 0{,}35/1{,}35\times\underbrace{[3 + 0{,}06\times(50 + 0{,}3\times 50)]}_{\text{t. acceso d-cache}} = \mathbf{4{,}37\ ciclos}.$$

b) Para calcular el CPI hay que tener en cuenta que los saltos se resuelven en la etapa ID y se aplica la técnica de predecir el salto como no tomado. Dados los tiempos de acceso de las cachés, el *pipeline* es de la forma:
F1 F2 ID X M1 M2 M3 W
con lo que para los saltos condicionales que sean tomados sufriremos 2 ciclos de penalización.

$$CPI_{instrucciones} = CPI_{ideal} + \underbrace{0{,}15\times 0{,}25\times 2}_{\text{detenciones saltos}} = 1{,}075 \text{ ciclos}$$

Añadimos ahora los ciclos que paramos como consecuencia de que no todos los accesos a memoria aciertan en las cachés:

$$CPI_{global} = 1{,}075 + \underbrace{0{,}03\times 50}_{\text{deten. i-cache}} + \underbrace{0{,}35\times 0{,}06\times[50 + 0{,}3\times 50]}_{\text{detenciones d-cache}} = \mathbf{3{,}935\ ciclos}.$$

c) Nos dicen que el acceso al TLB se realiza antes de poder ir a cualquiera de las dos cachés, con lo que estamos añadiendo en caso de acierto 2 ciclos (el tiempo de servicio del TLB). Con esto, el tiempo medio de acceso a memoria queda de la siguiente forma:

$$T_{AM} = 1/1{,}35\times(\underbrace{2 + 0{,}02\times 150}_{\text{t. acceso i-TLB}} + \underbrace{2 + 0{,}03\times 50}_{\text{t. acceso i-cache}}) + 0{,}35/1{,}35\times[\underbrace{2 + 0{,}03\times 150}_{\text{t. acceso d-TLB}} + \underbrace{3 + 0{,}06\times(50 + 0{,}3\times 50)}_{\text{t. acceso d-cache}}]$$

$= \mathbf{9{,}77\ ciclos}$.