Ejercicios de Paralelismo a Nivel de Instrucción

J. Daniel García Sánchez (coordinador)
David Expósito Singh
Javier García Blas
Óscar Pérez Alonso
J. Manuel Pérez Lobato

Arquitectura de Computadores
Grupo ARCOS
Departamento de Informática
Universidad Carlos III de Madrid

1. Ejercicios recomendados

Se recomienda la realización de los siguientes ejercicios del libro:
Fuente: Computer Architecture: A Quantitative Approach. 5 Ed
Hennessy and Patterson. Morgan Kaufmann. 2012.

- Apéndice C: Ejercicios C.1, C.2, C.3, C.4 y C.5.
- Capítulo 3: Ejercicios 3.1, 3.2, 3.3, 3.4, 3.6, 3.7, 3.10, 3.12.

2. Ejercicios de examen

Ejercicio 1 *Examen de junio de 2015.*

Considere un procesador tipo MIPS con arquitectura segmentada (pipeline) que tiene dos bancos de registros separados uno para números enteros y otro para números en coma flotante. El banco de registros enteros dispone de 32 registros. El banco de registros de coma flotante dispone de 16 registros de doble precisión (**$f0**, **$f2**, . . . , **$f30**).

Se supone que se dispone de suficiente ancho de banda de captación y decodificación como para que no se generen detenciones por esta causa y que se puede iniciar la ejecución de una instrucción en cada ciclo, excepto en los casos de detenciones debidas a dependencias de datos.

La siguiente tabla muestra las latencias adicionales de algunas categorías de instrucciones, en caso de que haya dependencia de datos. Si no hay dependencia de datos la latencia no es aplicable.

La instrucción bnez usa bifurcación retrasada con una ranura de retraso (*delay slot*).

En esta máquina se desea ejecutar el siguiente código:

```
bucle:  ldc1 $f0, ($t0)
        ldc1 $f2, ($t1)
        add.d $f4, $f0, $f2
        mul.d $f4, $f4, $f6
        sdc1 $f4, ($t2)
```

Cuadro 1: Latencias adicionales por instrucción

Instrucción	Latencia adicional	Operación
ldc1	+2	Carga un valor de 64 bits en un registro de coma flotante.
sdc1	+2	Almacena un valor de 64 bits en memoria principal.
add.d	+4	Suma registros de coma flotante de doble precisión.
mul.d	+6	Multiplica registros de coma flotante de doble precisión.
addi	+0	Suma un valor a un registro entero.
subi	+0	Resta un valor a un registro entero.
bnez	+1	Salta si el valor de un registro no es cero.

```
addi $t0, $t0, 8
addi $t1, $t1, 8
subi $t3, $t3, 1
bnez $t3, bucle
addi $t2, $t2, 8
```

Inicialmente los valores de los registros son:

- **$t0**: **0x00100000**.
- **$t1**: **0x00140000**.
- **$t2**: **0x00180000**.
- **$t3**: **0x00000100**.

Se pide:

1. Enumere las dependencias de datos RAW que hay en el código anterior.
2. Indique todas las detenciones que se producen al ejecutar una iteración del código anterior, e indique el número total de ciclos por iteración.
3. Intente planificar el bucle para reducir el número de detenciones.
4. Desenrolle el bucle de manera que en cada iteración se procesen cuatro posiciones de los arrays y determine el speedup conseguido. Utilice nombres de registros reales (**$f0**, **$f2**, ..., **$f30**).

IMPORTANTE: Se considerarán incorrectas soluciones que no usen registros realmente existentes (p. ej. **$f2'** o **$f2"** son nombres no válidos).

Ejercicio 2 *Examen de enero de 2015.*

Sea el siguiente fragmento de código se encuentra almacenado a partir de la dirección de memoria **0x1000100C** en una máquina en la que todas las instrucciones ocupan 4 bytes:

```
bucle:  lw $r2, 0($r0)
        addi $r3, $r2, 20
        sw $r3, 0($r1)
        addi $r0, $r0, 4
        addi $r1, $r1, 4
        bnez $r2, bucle
```

Dicho código se ejecuta en una máquina que dispone de una caché L1 para datos asociativa por conjuntos de 2 vías y con un tamaño de 32 KB y una caché L1 para instrucciones de iguales características. También dispone de una caché L2 unificada asociativa por conjuntos de 8 vías con un tamaño de 1 MB. En ambos casos el tamaño de línea es de 32 bytes. Se asume que un acierto en la caché L1 requiere 4 ciclos, y un acierto en la caché L2 requiere 14 ciclos adicionales y la penalización por traer un bloque de memoria principal a la caché de nivel 2 es de 80 ciclos. Todas las cachés tienen una política de escritura write-back.

Inicialmente el valor de los registros es:

- **$r0**: **0x00010000**.
- **$r1**: **0x00080000**.

A partir de la posición **0x00010000** todos los valores en memoria son distintos de cero hasta la posición **0x000100FC**. En la posición de memoria **0x000100FC** hay un valor de cero.

1. Determine cuál debería ser el tiempo medio de acceso asumiendo que un programa (distinto del facilitado) realiza en promedio 2 accesos a datos por instrucción y tiene las siguientes tasas de fallo:
   - L1 instrucciones: 10 %
   - L1 datos: 5 %
   - L2: 2 %
2. Determine el número de fallos que se producen durante la ejecución del fragmento de código facilitado en el enunciado para las cachés L1 de datos, L1 de instrucciones y L2.
3. 1.3: Elabore un diagrama de tiempos para una arquitectura MIPS con un pipeline de 5 etapas para la primera iteración del bucle asumiendo que inicialmente no hay datos ni instrucciones en las cachés y con las siguientes consideraciones:
   - No hay hardware de forwarding.
   - La arquitectura permite que una instrucción escriba en un registro y otra instrucción lea ese mismo registro sin problemas.
   - Las bifurcaciones se tratan vaciando el pipeline.
   - La dirección efectiva de salto se calcula en la etapa de ejecución.

   NOTA: Tenga en cuenta en el cronograma las detenciones debidas a los fallos en toda la jerarquía de caché tanto para instrucciones (etapa IF) como para lecturas y escrituras de datos (etapa M).
4. Repita el diagrama de tiempos para la segunda iteración.

Ejercicio 3 *Examen de octubre de 2014.*

Sea el siguiente fragmento de código:

```
bucle:  lw $f0, 0($r1)
        lw $f2, 0($r2)
        mul.f $f4, $f0, $f2
        add.d $f6, $f6, $f4
        addi $r1, $r1, 4
        addi $r2, $r2, 4
        sub $r3, $r3, 1
        bnez $r3, bucle
```

1. Haga una lista con todas las posibles dependencia de datos, sin considerar una estructura específica de la arquitectura segmentada. Para cada dependencia debe indicar, registro, instrucción de origen, instrucción de destino y tipo de dependencia.
2. Elabore un diagrama de tiempos para una arquitectura MIPS con un pipeline en 5 etapas, con las siguientes consideraciones:
   - No hay hardware de forwarding.
   - La arquitectura permite que una instrucción escriba en un registro y otra instrucción lea ese mismo registro sin problemas.
   - Las bifurcaciones se tratan vaciando el pipeline.
   - Las referencias a memoria requieren un ciclo de reloj.
   - La dirección efectiva de salto se calcula en la etapa de ejecución.
3. Determine cuantos ciclos se necesitan para ejecutar N iteraciones del bucle.
4. Elabore un diagrama de tiempos para una arquitectura MIPS con un pipeline en 5 etapas con las siguientes consideraciones:
   - Hay hardware completo de forwarding.
   - Asuma que las bifurcaciones se tratan prediciendo todos los saltos como tomados.
5. Determine cuantos ciclos se necesitan para ejecutar N iteraciones del bucle en las condiciones del apartado 4.

Ejercicio 4 *Octubre de 2014.*

Sea el siguiente fragmento de código:

```
bucle:  lw $f0, 0($r1)
        lw $f2, 0($r2)
        sub.f $f4, $f0, $f2
        mul.d $f4, $f4, $f4
        add.d $f6, $f6, $f4
        addi $r1, $r1, 4
        addi $r2, $r2, 4
        sub $r3, $r3, 1
        bnez $r3, bucle
```

1. Haga una lista con toda la posible dependencia de datos, sin considerar una estructura específica de la arquitectura segmentada. Para cada dependencia debe indicar, registro, instrucción de origen, instrucción de destino y tipo de dependencia.
2. Elabore un diagrama de tiempos para una arquitectura MIPS con un pipeline en 5 etapas, con las siguientes consideraciones:

- No hay hardware de forwarding.
- La arquitectura permite que una instrucción escriba en un registro y otra instrucción lea ese mismo registro sin problemas en el mismo ciclo de reloj.
- Las bifurcaciones se tratan vaciando el pipeline.
- Las referencias a memoria requieren un ciclo de reloj.
- La dirección efectiva de salto se calcula en la etapa de ejecución.

3. Determine cuantos ciclos se necesitan para ejecutar N iteraciones del bucle.
4. Elabore un diagrama de tiempos para una arquitectura MIPS con un pipeline en 5 etapas con las siguientes consideraciones:
   - Hay hardware completo de forwarding.
   - Asuma que las bifurcaciones se tratan prediciendo todos los saltos como tomados.
5. Determine cuantos ciclos se necesitan para ejecutar N iteraciones del bucle en las condiciones del apartado 4.

Ejercicio 5 *Examen de junio de 2014.*

En un determinado procesador se dispone de la tabla 2 de latencia entre instrucciones:

Cuadro 2: Latencia entre instrucciones

Instrucción que produce el resultado	Instrucción que usa el resultado	Latencia
Operación ALU FP	Otra operación ALU FP	6
Operación ALU FP	Almacenar doble	3
Cargar doble	Operación ALU FP	2
Cargar doble	Almacenar doble	0

En esta máquina se desea ejecutar el siguiente trozo de código:

```
BUCLE:  L.D  F0, 0(R1)
        L.D  F2, 0(R2)
        ADD.D F4, F0, F2
        S.D  F4, 0(R3)
        DADDUI R1, R1, #-8
        BNE  R1, R4, BUCLE
```

Inicialmente se tienen los siguientes valores:

- **R1**: Último elemento de primer array origen.
- **R2**: Último elemento de segundo array origen.
- **R3**: Último elemento de array destino.
- **R4**: Precalculado. Tal que **8(R4)** sea el primer elemento del primer array origen.

Todos los arrays tienen un tamaño de 4000 elementos.
Se pide:

1. Determine cuántos ciclos se requiere para ejecutar todas las iteraciones del bucle si no se realiza ninguna modificación.

2. Determine cuántos ciclos se requiere para ejecutar todas las iteraciones si se realiza planificación del bucle.

3. Determine cuántos ciclos se requiere para ejecutar todas las iteraciones si se desenrolla el bucle para cada dos iteraciones.

4. Determine cuántos ciclos se requiere para ejecutar todas las iteraciones si se desenrolla el bucle para cada cuatro iteraciones.

Ejercicio 6 *Examen de enero de 2014.*

En un determinado procesador se pretende ejecutar el siguiente segmento de código:

```
i0: lw $r4, 0($r1)
i1: lw $r5, 0($r2)
i2: add $r4, $r4, $r5
i3: sw $r4, 0($r3)
i4: addi $r1, $r1, 4
i5: addi $r2, $r2, 4
i6: addi $r3, $r3, 4
i7: bne $r3, $r0, i0
```

Asuma que el procesador tiene una arquitectura segmentada de 5 etapas (captación, decodificación, ejecución, memoria y post-escritura) sin envío adelantado (forwarding). Todas las operaciones se ejecutan en un ciclo por etapa, excepto:

- Las operaciones de carga y almacenamiento, que requieren un total de dos ciclos para la etapa de memoria (un ciclo adicional).
- Las instrucciones de salto, que requieren un ciclo adicional en la etapa de ejecución. Considere que estas instrucciones no cuentan con ningún tipo de predicción de saltos.

1. Determine los riesgos de datos RAW que presenta el código que tienen impacto en la ejecución del código.

2. Muestre un diagrama de tiempos con las fases de ejecución de cada instrucción para una iteración.

3. Determine cuantos ciclos requiere la ejecución de una iteración del bucle si no hay ningún tipo de predicción de saltos.

4. Proponga un desenrollamiento (*loop unrolling*) del bucle asumiendo que bucle se ejecuta 1000 iteraciones. Desenrolle con un factor de cuatro iteraciones.

5. Determine la aceleración o speedup obtenido mediante el desenrollamiento del apartado anterior.

Ejercicio 7 *Examen de octubre de 2013.*

En un determinado procesador pretende ejecutar el siguiente segmento de código:

```
i0:     lw $r3, 0($r0)
i1:     lw $r1, 0($r3)
i2:     addi $r1, $r1, 1
i3:     sub $r4, $r3, $r2
i4:     sw $r1, 0($r3)
i5:     bnz $r4, i0
```

Asuma que el procesador tiene una arquitectura segmentada de 5 etapas (captación, decodificación, ejecución, memoria y post-escritura) sin envío adelantado. Todas las operaciones se ejecutan en un ciclo, excepto las operaciones de carga y almacenamiento que requieren dos ciclos adicionales de latencia en el acceso a memoria, y las instrucciones de salto que requieren un ciclo adicional de ejecución.

1. Determine los riesgos de datos RAW que presenta el código.
2. Muestre un diagrama tiempos con las fases de ejecución de cada instrucción para una iteración:
3. Determine cuantos ciclos requiere la ejecución de una iteración del bucle si no hay ningún tipo de predicción de saltos.
4. Determine cuantos ciclos requiere la ejecución de una iteración si se usa un predictor de saltos que predice siempre a tomado.

Ejercicio 8 *Examen de octubre de 2013.*

Se dispone del siguiente código escrito en el ensamblador de MIPS. Suponga que, antes de empezar la ejecución las instrucciones R3 y R5 contienen, respectivamente, las direcciones de memoria del primer y último elemento de un array de 9 entradas (Valor inicial de R1=0x010 y valor inicial de R5=0x018).

```
Loop:   LD      R4  0(R1)
        DIV     R2  R2  R4
        ADD     R1  R1  #1
        SUB     R5  R5  #1
        SD      R4  0(R5)
        SUB     R6  R1  R5
        BNEZ    R6  Loop
```

1. Indique todos los riesgos de datos RAW y WAW presentes en el código anterior.
2. Diagrama de temporización asumiendo que usamos un procesador segmentado de 5 etapas (*fetch*, *decode*, *execution*, *memory* y *write back*), se emite una instrucción por ciclo y **no se usa forwarding**. Asuma que existe congelación del pipe en el salto y que **hay 1 ciclo adicional por acceso a memoria en lectura (LD) (no ocurre en escrituras)**.
3. Número de ciclos que tardaría el bucle Loop (todas sus iteraciones) en ejecutarse.

Ejercicio 9 *Examen de octubre de 2013.*

Considere el siguiente fragmento de código:

```
Bucle:  LD  R4,0(R2)
        LD  R5,0(R3)
        ADD R6,R4,R5
        SD  R6,0(R3)
        BNZ R6,Bucle
```

1. Número de ciclos necesarios para ejecutar una iteración del bucle en un procesador no segmentado.Las instrucciones de acceso a memoria tienen una latencia de 3 ciclos. La instrucción de salto tiene una latencia de 1 ciclo.
2. Identifique las dependencias de datos RAW existentes en el código.
3. Calcule el número de ciclos necesarios para ejecutar una iteración del bucle en un procesador segmentado en 5 etapas. Se usa la técnica de forwarding y la estrategia de predicción de salto Congelación del Pipeline. Complete el diagrama de temporización siguiente

Ejercicio 10 *Examen de junio de 2013.*

Dado el siguiente código, donde cada instrucción como coste asociado un ciclo además de los indicados en la siguiente tabla. Suponga además que la máquina en la que ejecuta es capaz de emitir una instrucción por ciclo, salvo las esperas debidas a detenciones y que es un procesador segmentado con un único camino de datos (pipeline).

Debido a riesgos estructurales, las detenciones de una instrucción (stalls) se realizan siempre, independientemente de que haya o no dependencia de datos con las instrucciones siguientes. Inicialmente se tiene que R1=0, R2=24, R3=16.

```
Loop1:  LD F2, 0(R1)
        ADDD F2,F0,F2
Loop2:  LD F4,0(R3)
        MULTD F4,F0,F4
        DIVD F10,F4,F0
        ADDD F12,F10,F4
        ADDI R1,R1,#8
        SUB R18,R2,R1
        BNZ R18,Loop2
        SD F2,0(R3)
        ADDI R3, R3, #8
        SUB R20,R2,R3
        BNZ R20,Loop1
```

Cuadro 3: Costes adicionales por instrucción

Instrucción	Latencia
LD	3
SD	1
ADD	2
MULTD	4
DIVD	10
ADDI	0
SUB	0
BNZ	1

1. Calcular el número de ciclos necesarios para ejecutar una iteración del bucle exterior y treinta del bucle interior.
2. Desenrollar tres iteraciones del bucle interior, no desenrollar el bucle exterior y volver a realizar el cálculo anterior.
3. Comparar los resultados de tiempos (número de ciclos) del primer y segundo apartado. Justificar la respuesta.

Ejercicio 11 *Examen de enero de 2013.*

Se dispone del siguiente código escrito en el ensamblador de MIPS. En el mismo se lee y escribe la misma posición de memoria un número de veces. Inicialmente los valores de R1 y R2 son R1=1 y R2=1000.

```
        ADDI R3,R3,1
Loop:   LD R4,0(16)
        MULT R5,R4,R4
        ADD R5,R3,R5
        SD R5,0(16)
        DADDI R3,R4,1
        DADDI R1,R1,1
        BNE R1,R2,Loop
```

Se tiene un procesador segmentado tipo MIPS con las siguientes fases de ejecución: fetch, decodificación, ejecución, memoria y escritura. Dicho procesador emite una instrucción por ciclo y tiene capacidad de envío adelantado (*forwarding*). Todas las etapas del camino de datos de ejecutan en un ciclo salvo los siguientes casos: la operación SD utiliza un ciclo adicional para leer **el registro R5 del banco de registros**, la operación LD utiliza un ciclo adicional para escribir **el valor de memoria en el registro R4 del banco de registros** y las operaciones ADD y MULT utilizan un ciclo adicional para ejecutarse en ALU. Asumimos que la estrategia de predicción de salto es No Tomada.

1. Indique sólo los riesgos de datos WAW presentes en el código anterior indicando las instrucciones que lo causan y el registro. ¿En qué situación un riesgo WAW podría originar un resultado incorrecto del programa?

2. Diagrama de temporización asumiendo forwarding. Número de ciclos que tardaría el programa en ejecutarse.

Ejercicio 12 *Examen de junio de 2012.*

Dado el siguiente fragmento de código:

```
Bucle:  LD R4, 0(R2)
        LD R5, 0(R3)
        ADD R6, R4,R5
        SD R6, 0(R3)
        ADD R6, R6, R4
        ADDI R3, R3, #8
        SUB R20, R4, R3
        BNZ R20, Bucle
```

1. Indique las dependencias de datos existentes en el código. Indicando el dato que provoca cada dependencia.

2. Presente la temporización de esta secuencia para el pipeline RISC de 5 etapas sin hardware de *forwarding* o *bypassing*, pero asumiendo que una lectura y una escritura de un dato en el banco de registros se pueden realizar en el mismo ciclo de reloj (mediante el envío adelantado a través del banco de registros). Asuma que la bifurcación se trata vaciando el pipeline y que todos los accesos a memoria (incluidas el fetch) tardan dos ciclos. Justifique la respuesta.

Ejercicio 13 *Examen de mayo de 2012.*

Dada la siguiente sección de código:

```
        DADDUI R3, R1, #40      ; I1
BUCLE:  L.D F0, 0(R1)           ; I2
        L.D F2, 0(R2)           ; I3
        ADD.D F4, F0, F2        ; I4
        S.D F4, 0(R1)           ; I5
        DADDUI R1, R1, #8       ; I6
        DADDUI R2, R2, #8       ; I7
        BLE R1, R3, BUCLE       ; I8
```

Y teniendo en cuenta que se ejecuta en una máquina con las latencias adicionales entre instrucciones indicada por la tabla 4.

La instrucción de salto (*branch*) tiene una latencia de un ciclo y no tiene *delay slot*.

Suponga además que la máquina en la que ejecuta es capaz de emitir una instrucción por ciclo, salvo las esperas debidas a detenciones y que es un procesador segmentado con un único camino de datos (*pipeline*).

Cuadro 4: Latencias adicionales

Instrucción que produce el resultado (previa)	Instrucción que usa el resultado (posterior)	Latencia
Operación ALU FP	Operación ALU FP	5
Operación ALU FP	Cargar/almacenar double	4
Operación ALU FP	Instrucción de salto	4
Cargar doble	Operación ALU FP	2
Cargar doble	Cargar doble	1
Almacenar doble	Operación ALU FP	2

1. Determine todas las dependencias de datos.
2. Determine el número de ciclos total que se necesita para ejecutar la sección de código completa.
3. Modifique el código para reducir las detenciones mediante la técnica de planificación de bucle. Determine la aceleración obtenida respecto a la versión sin planificar (apartado 2).
4. Modifique el código realizando un desenrollamiento de bucle con dos iteraciones por bucle. Determine la aceleración obtenida respecto a la versión sin planificar (apartado 2).