UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA

ORGANIZACIÓN DE COMPUTADORES

LABORATORIO 2

PARTE 1

Profesores: Alfonso Guzmán & Daniel Wladdimiro

Ayudantes: Mario Álvarez, Sebastián Pinto-Agüero & Ariel Undurraga

Santiago - Chile

29 de septiembre de 2017

CAPÍTULO 1. CONTEXTO

Como lo confirma la teoría vista en cátedra, en la organización de computadores lo esencial es el camino de datos (*datapath* en inglés), el cual consiste en un conjunto de unidades funcionales (unidad de control, ALU, multiplexores, buses, registros, etc), donde sin la existencia de estos elementos, no sería posible ejecutar las instrucciones que se les dan a un computador, es decir, no existirían los computadores como los conocemos hoy en día.

Este camino de datos, como su nombre lo sugiere, representa el camino que recorren las señales eléctricas a través de las distintas unidades funcionales, con el fin de realizar una instrucción específica. Es por esto que cada instrucción básica, como las del lenguaje MIPS, utiliza un conjunto acotado de unidades del camino de datos, que se pueden evidenciar en las señales activas de la unidad de control. Ante esto surge la siguiente interrogante: ¿Qué unidades funcionales se activan al ejecutar un determinado conjunto de instrucciones?

CAPÍTULO 2. INSTRUCCIONES

El programa a construir debe leer dos archivos de entrada, que contendrán instrucciones de un código en MIPS y los valores de cada uno de los registros. Además debe ser capaz de generar dos archivos de salida, el primero entrega la traza de los etapas del pipeline por cada ciclo de reloj, y el segundo corresponde a los hazard de datos y control detectados por cada ciclo de reloj. Para efectos de este laboratorio, debe considerar que el camino de datos está siendo ejecutado en un procesador optimizado con *forwarding*, con una predicción del branch `not taken` (siempre se considera el branch como no tomado) y que utiliza los componentes presentes en la imagen del Anexo.

Etapas	CC_1	CC_2	CC_3	...	CC_n
IF	add $t0, $t1, $t2	sub $s0, $t1, $t2	lw $t0, 0($t2)	...	beq $s0, $s1, `LABEL`
ID	`NOP`	add $t0, $t1, $t2	sub $s0, $t1, $t2	...	add $t4, $t8, $t7
EX	`NOP`	`NOP`	add $t1, $t1, $t2	...	add $t9, $s3, $t7
MEM	`NOP`	`NOP`	`NOP`	...	`NOP`
WEB	`NOP`	`NOP`	`NOP`	...	lw $s3, 40($t2)

Cuadro 2.1: Ejemplo del primer archivo de salida.

Hazard	CC_1	CC_2	CC_3	...	CC_n
$Control$	-	-	-	...	-
$Datos$	-	-	$t1	...	-

Cuadro 2.2: Ejemplo del segundo archivo de salida.

Cabe recordar que en el pipeline de 5 etapas que se ha utilizado, en los buffers `EX/MEM` y `MEM/WB` es donde debe aplicarse la técnica de *forwarding*, de tal manera de modificar los valores de los registros leídos. Las condiciones que deben cumplirse para que haya un hazard de dato son:

```
if (EX/MEM.RegWrite=1)
and (EX/MEM.Mux_RegDst != 0)
and (EX/MEM.Mux_RegDst = ID/EX.Rs)

if (EX/MEM.RegWrite=1)
and (EX/MEM.Mux_RegDst != 0)
and (EX/MEM.Mux_RegDst = ID/EX.Rt)
```

```
if (MEM/WB.RegWrite=1)
and (MEM/WB.Mux_RegDst != 0)
and (MEM/WB.Mux_RegDst = ID/EX.Rs)

if (MEM/WB.RegWrite=1)
and (MEM/WB.Mux_RegDst != 0)
and (MEM/WB.Mux_RegDst = ID/EX.Rt)
```

En el caso, que se necesite agregar un `NOP` (utilizando el forwarding), se deberá cumplir la siguiente condición:

```
if (ID/EX.MemRead=1)
and ((ID/EX.Rt=IF/ID.Rs)
or (ID/EX.Rt=IF/ID.Rt))
    stall the pipeline
```

De esta manera, cuando se realice la espera, todos los valores de la línea de control deberán valer 0, para que no se realice ninguna acción y se espere un ciclo para realizar la acción.

Supongamos que poseemos la siguientes dos instrucciones y realicemos la traza correspondiente:

```
lw $t0, 0($t1)
add $t3, $t0, $t2
```

Por lo tanto, en el ejemplo anterior, se analiza que los registros sean iguales y de esta manera se establezcan las líneas de control en 0 para el siguiente ciclo, es decir, que la instrucción sea un `NOP`.

Por otra parte, en el caso de existir un hazard de control, se debe realizar un `flush` a los buffer para aquellas instrucciones que no debieron ser planificadas, estableciendo todos sus valores en 0.

Cabe destacar que para las salidas, deben realizarse en un archivo `.csv` como está estipulado en los archivos de salida de ejemplo.

Para efectos de pruebas, se entregarán archivos de entrada distintos, los cuales estarán disponibles en Moodle.

Junto con el programa, usted debe entregar un informe que cumpla con el formato tesis del Departamento de Ingeniería Informática de la Universidad de Santiago de Chile. Para efectos prácticos, se dejará disponible una plantilla en LaTeX en el Moodle, la cual posee las secciones a evaluar en la entrega del laboratorio.

En el desarrollo del informe se evaluará la forma en cómo usted dio solución al enunciado de este laboratorio, es decir, cómo decodificó el archivo de entrada, cómo realizó el seguimiento de las trazas, y así mismo, cómo presenta los resultados en el archivo de salida.

2.1 EXIGENCIAS

- El programa debe estar escrito en C o C++ (estándar ANSI C). En caso de usar C++ se exigirá orientación a objetos, de caso contrario no será revisado.
- El programa y el informe deben ser entregados como una carpeta comprimida en formato `zip` o `tar.gz`-
- El nombre de la carpeta comprimida debe ser `ApellidoPaterno_RutEstudiante.zip`.
- El programa debe ser entregado en su código fuente junto a un archivo Makefile para su compilación.
- El programa debe tener usabilidad. El usuario debe ser capaz de ingresar el nombre del archivo a leer y los nombres de los archivos de salida por una interfaz.
- El programa debe cumplir con un mínimo de calidad de software (funciones separadas y nombres tanto de funciones como de variables, de fácil comprensión).
- El informe escrito no debe exceder 10 páginas de texto escrito, sin considerar portada ni índice, en caso contrario, por cada página extra, se descontará 5 décimas.
- El informe escrito debe ser entregado en formato PDF, por lo que puede ser desarrollado en LaTeX, Microsoft Word, OpenOffice Writer, etc.

2.2 RECOMENDACIONES

- Pueden usar estructuras de datos para almacenar la información de los registros y señales de control en caso de utilizar C, sino, debe utilizar un objeto para esto.
- Utilizar la plantilla de LaTeX disponible en el Moodle del curso.
- Consultar a los ayudantes y en Moodle del curso.

2.3 DESCUENTOS

- Por cada exigencia no cumplida, se descontarán dos décimas a la nota, a excepción las que ya mencionan su descuento.
- Por cada día de atraso, se descontará un punto a la nota.
- Por cada tres faltas ortográficas o gramaticales en el informe, se descontará una décima a la nota.

- Por cada falta de formato en el informe, se descontará una décima a la nota.

2.4 EVALUACIÓN

- La nota del laboratorio será el promedio aritmético del código fuente con el informe, y en caso de obtener una nota inferior a 4 en alguno de las dos calificaciones, se evaluará con la menor nota.
- En caso que no se entregue alguno de los dos, se evaluará con la nota mínima.

Este laboratorio debe ser entregado el día 13 de Octubre del año 2017, hasta las 23:59 hrs. Los descuentos por atraso corren a contar de las 00:59 hrs del día 14 de Octubre del año 2017

En caso de dudas o problemas en el desarrollo, comunicarse con su ayudante de laboratorio.

CAPÍTULO 3. ANEXOS

Camino de datos de ejemplo: