Laboratorio 1

Arquitectura de Computadoras 2014

Objetivos

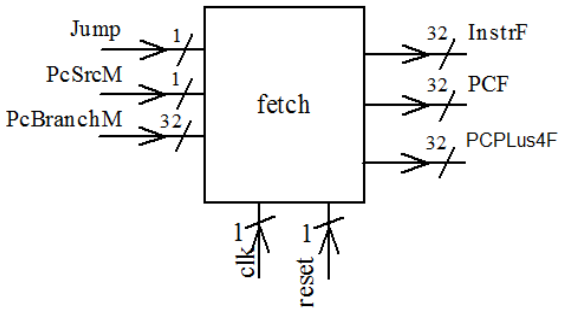
- Desarrollar código a base de componentes simples en lenguaje VHDL, que describan pequeños circuitos secuenciales y combinacionales vistos en el teórico y el práctico.
- Utilizar la herramienta GHDL para analizar y "compilar" el código VHDL.
- Desarrollar código *VHDL* (*testbench*) que testee el código del circuito.
- Mediante el uso de *gtkwave*, analizar las formas de onda y testear el resultado.
- Aprender a reutilizar código VHDL mediante componentes.

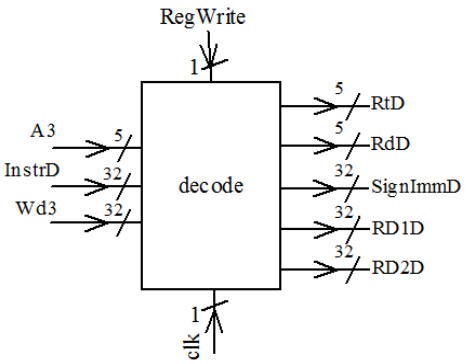
Tarea

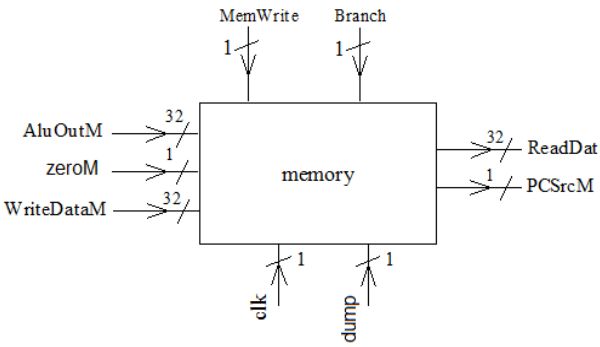
La tarea a desarrollar en este laboratorio, consta de la implementación del microprocesador MIPS con *Pipeline*, en una versión simple sin control de *hazards*. Deberán reutilizar la mayoría del código a partir de su versión *single cycle* vista en los prácticos 1 y 2. Se pide que modifiquen el diseño del módulo *datapath* del diseño original de tal modo que se implementen las etapas *fetch*, *decode*, *execute*, *memory* y *writeback*.

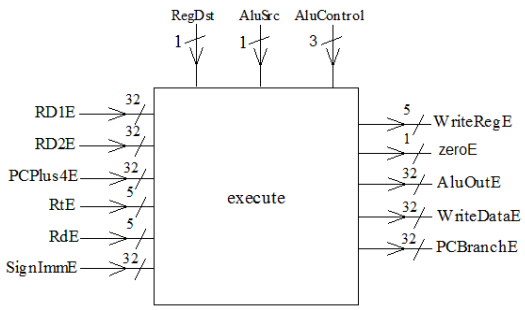
Recomendaciones:

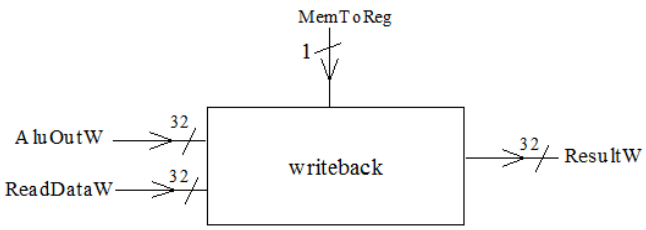
Se recomienda fuertemente que realicen pequeñas modificaciones (incrementalmente) al diseño del módulo *datapath.* No deberían agregar código sino redistribuir los componentes que ya están realizando en el *datapath*. Para lograrlo, se recomienda que se dividan las etapas en nuevas entidades según los siguientes diagramas:







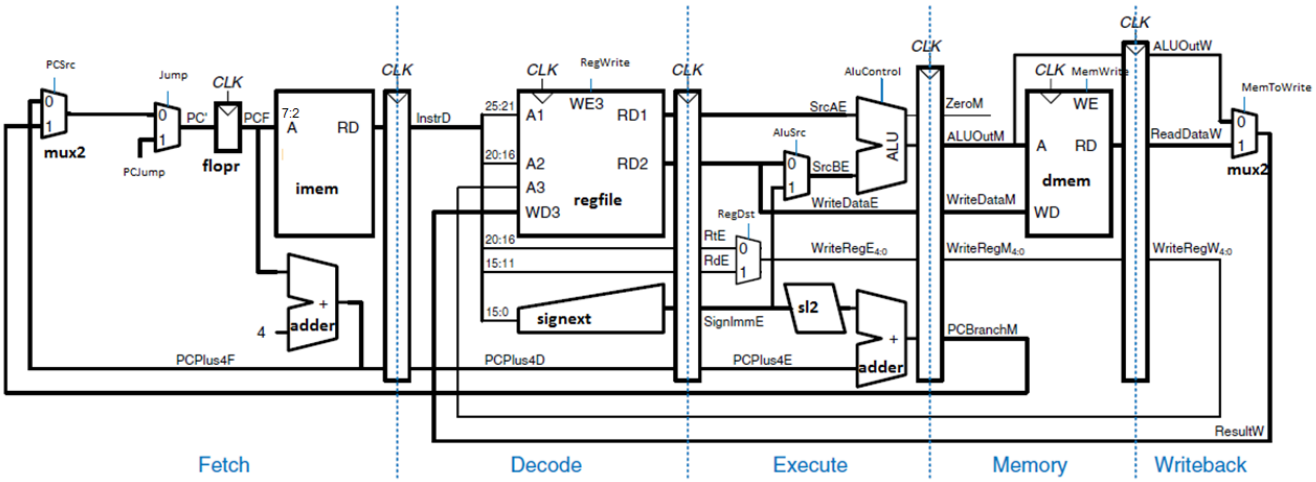




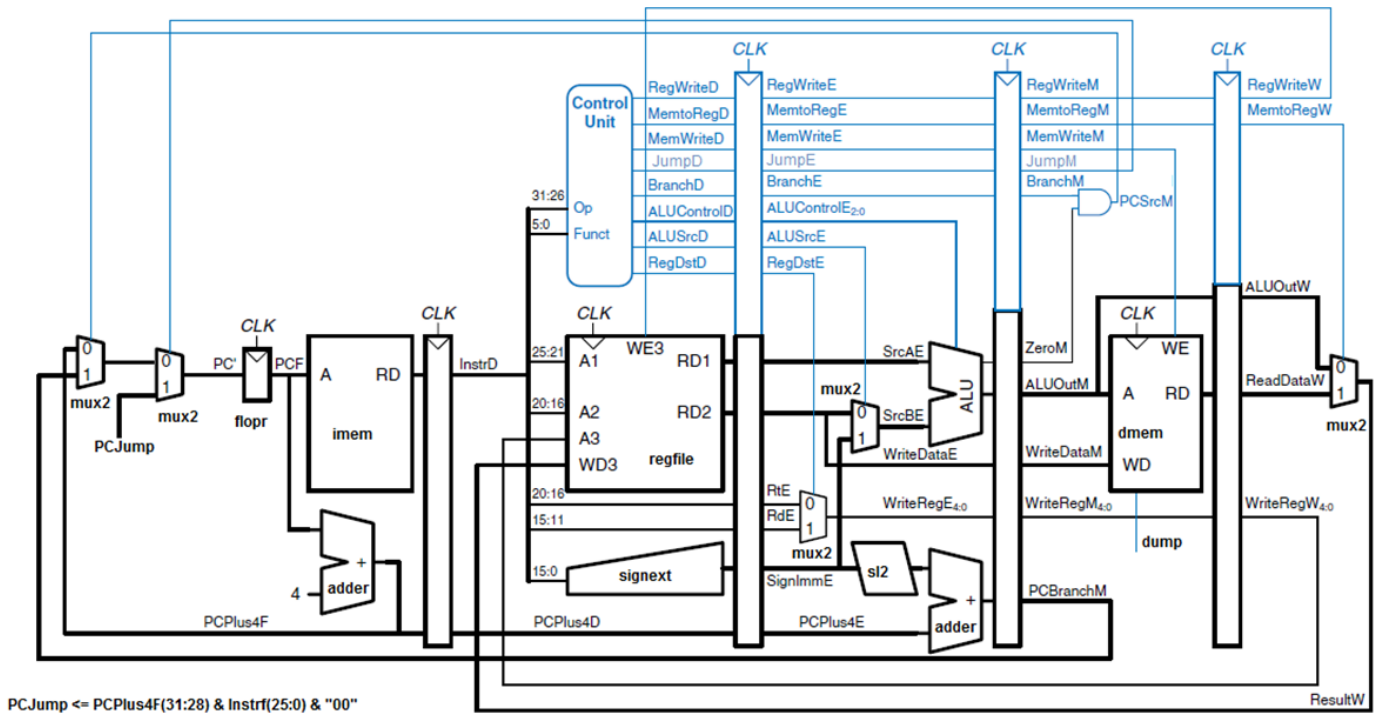
Laboratorio 1

Arquitectura de Computadoras 2014

Antes de agregar *Pipeline*, deben probar si el diseño de los módulos funciona correctamente (es decir, conectarlos mediante señales sin pasar por "clock"). Una vez que estén seguros y confiados en que el diseño se comporta como la versión original, deben agregarle la división en etapas DENTRO del componente "*datapath*" según el diagrama:



Recuerden que deben agregar las señales de control al *pipeline:*



Laboratorio 1

Arquitectura de Computadoras 2014

Código ejemplo

Su microprocesador MIPS encausado debe ser capaz de correr el siguiente programa sin ningun problema (ya que no tiene *hazards* de ningún tipo).

Dirección	Instrucción de MIPS	Código para *imem*
0x00	*addi $t0, $0, 0*	20080000
0x01	*addi $t1, $0, 1*	20090001
0x02	*addi $t2, $0, 2*	200a0002
0x03	*addi $t3, $0, 3*	200b0003
0x04	*addi $t4, $0, 4*	200c0004
0x05	*addi $t5, $0, 5*	200d0005
0x06	*addi $t6, $0, 6*	200e0006
0x07	*addi $t7, $0, 7*	200f0007
0x08	*sw $t0, 0($0)*	ac080000
0x09	*sw $t1, 4($0)*	ac090004
0x0A	*sw $t2, 8($0)*	ac0a0008
0x0B	*sw $t3, 12($0)*	ac0b000c
0x0C	*sw $t4, 16($0)*	ac0c0010
0x0D	*sw $t5, 20($0)*	ac0d0014
0x0E	*sw $t6, 24($0)*	ac0e0018
0x0F	*sw $t7, 28($0)*	ac0f001c

Recuerde colocar en su testbench la señal reset -> 1 por dos ciclos; correr el programa anterior por 17 ciclos, y luego colocar la señal dump -> 1 por tres ciclos, para obtener el archivo con el dump de la memoria. Verificar en el archivo de salida que el contenido de las siete primeras posiciones de la memoria contiene su índice (es decir, el contenido de la posición 0 es 0; el contenido de la posición 1 es 1, y así sucesivamente hasta la posición 7).

Elaboración 1

Suponga el siguiente programa,

```
addi $t0, $zero, 1
addi $t1, $t0, 2
addi $t2, $t1, 2
addi $t3, $t2, 2

sw $t0, 0($zero)
sw $t1, 4($zero)
sw $t2, 8($zero)
sw $t3, 12($zero)

exit: beq $zero $zero exit
```

Una técnica común para evitar *pipeline stalls* es el reordenamiento estático. Puede utilizar esta técnica para evitar los stalls? puede evitarlos con instrucciones “nops”? justifique su respuesta.

Elaboración 2

GHDL es solo una herramienta de simulación muy simple pero existen otras herramientas más complejas que le permiten realizar análisis de tiempos más específicos y rigurosos. Suponga que puede utilizar esas herramientas que le permiten observar que el retardo del datapath sin encausar es de 280ns, mientras que el retardo de la etapa fetch

es de 50ns, el de la etapa decode es de 45ns, el de la etapa execute es de 75ns, el de la etapa memory es de 90ns, el de la etapa writeback es de 20ns. ¿Cuál es el periodo del clock de la versión encausada? ¿cual es la latencia de las instrucciones? ¿Cuál es la ganancia de velocidad teórica con respecto al micro no encauzado?

Requisitos del código a entregar

- Las entregas serán a través del mail: arq.famaf@gmail.com. Deberán crear un archivo comprimido (tar, zip, rar, etc.) explicitando el numero de laboratorio y sus integrantes (ApellidoNombre): ej: "lab1_FraireJuan_SanchezEduardo.tar.gz"
- Se deben utilizar los mismos nombres de los componentes de alto nivel pedidos.
- Junto con el código, se deberá entregar un pequeño informe en el cual se explique la estructuración del código (entidades de mayor nivel), decisiones de diseño tomadas (si las hubiere), dificultades con las que se encontraron y cómo las resolvieron, *testbenchs*, etc.
- El formato de dicho informe queda a elección de ustedes, siempre y cuando sea un formato libre.
- El trabajo es grupal (Max. 3 personas). Todos los integrantes del grupo deberán ser capaces de explicar el código presentado en la fecha de entrega.
- **No** está permitido compartir código entre grupos.

Entrega de Laboratorio: **1 de octubre.**