**Práctica 7 La ruta de datos monociclo**

# Objetivos

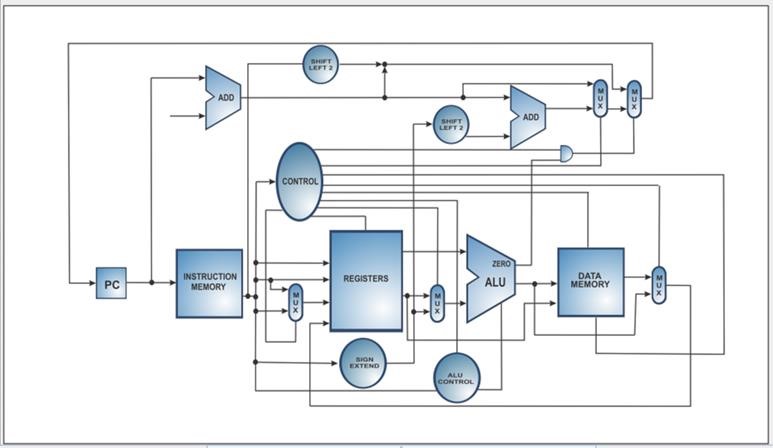
* Comprender mejor el funcionamiento de la ruta de datos monociclo utilizando animaciones gráficas.
* Obtener una mejor comprensión de los modelos de flujo de datos y el funcionamiento combinado de las unidades funcionales en la ejecución de determinadas instrucciones por el procesador.

**Materiales**

Simulador MARS y códigos fuente de partida.

# Introducción

El simulador MARS dispone de una herramienta llamada MIPS X-RAY (Tools > MIPS X RAY) que permite visualizar la ruta de datos monociclo mientras se está ejecutando una instrucción. Al acceder a la herramienta se abrirá una ventana parecida a la que puedes observar en la figura 1.



*Figura 1. Herramienta MIPS X-Ray*

La ruta de datos representada es monociclo y centrada en los datos y señales de control. Las unidades funcionales se representan como cajas negras por lo que sólo se consideran las entradas, salidas y las señales de control como relevantes. Como se puede observar, se trata de una visión estilizada de la ruta de datos que puede ejecutar todo el repertorio de instrucciones del MIPS, en ella se pueden distinguir las siguientes unidades funcionales:

* PC (Contador de Programa): es un registro simple con la función de almacenar la dirección de la siguiente instrucción a ejecutar. Este registro envía el valor de esta dirección hacia la memoria de instrucciones y recibe un nuevo valor, que puede ser generado por el sumador al incrementar el valor de la dirección a la siguiente instrucción a ejecutar o, en caso de saltos, el valor de la siguiente dirección de memoria.
* ALU (Unidad Aritmética Lógica): esta unidad es responsable de realizar cálculos aritméticos y lógicos. Los datos de entrada consisten en los valores contenidos en dos registros y una señal de control que define la operación a realizar con los datos. Esta unidad genera como salida el valor resultante de la operación aplicada a los datos y una señal binaria que indica el resultado de las operaciones de comparación.
* Control de la ALU: esta unidad establece qué operación será ejecutada por la ALU. Como entrada recibe los bits directamente del campo función de la instrucción y una señal de la unidad de control principal indicando el tipo de operación y transmite la operación definida a la ALU.
* Memoria de Instrucciones: esta unidad funciona como un banco de memoria, almacenando las instrucciones procedentes del código fuente. Por tanto, almacena los bits que se enviarán a todas las unidades funcionales y de control para que puedan realizar las operaciones necesarias. La memoria de instrucciones tiene una entrada de datos que recibe del PC la posición de memoria a la que se va a acceder y una salida con la instrucción de 32 bits que será ejecutada;
* Banco de Registros: esta unidad se encarga de almacenar los 32 registros utilizados por la arquitectura MIPS. El banco de registros presenta tres entradas, la dirección del registro a utilizar, una entrada de datos a almacenar en los registros y una entrada de control para indicar si se leerá desde el registro o se escribirá en los registros. Esta unidad dispone también de dos salidas con los datos contenidos en los registros y que han sido direccionados por los datos de entrada;
* Unidad de control: la unidad de control recibe los bits del código de operación, identifica qué instrucción se ejecutará y envía señales de control a cada unidad funcional, informando de lo que debe hacer. Por ello, la unidad de control está conectada a todas las unidades funcionales bajo su control.
* Memoria de datos: la memoria de datos es la unidad responsable de almacenar los datos que deben persistir tras la ejecución de las instrucciones. Los datos de la memoria de datos permanecen en la memoria hasta el momento en que se sobrescriban o cuando se desconecta la alimentación;
* Unidad de Signo Extendido: en algunas situaciones, el valor a considerar tiene menos de los 32 bits del registro que lo contiene, en ese caso debe extenderse el bit de signo hasta el último bit reservado para su almacenamiento. En este contexto la función de la unidad de extensión de signo es rellenar los bits restantes para obtener un dato de 32 bits;
* Desplazamiento a la izquierda: la función de esta unidad es desplazar los bits de una dirección a la izquierda y llenar con ceros los bits restantes que han quedado vacíos; se obtiene un valor múltiplo de 4.
* Multiplexores: estas unidades se utilizan para determinar qué dato se transmitirá. Los multiplexores, al igual que las demás unidades funcionales, son controlados por la unidad de control.
* Sumadores: los sumadores son unidades aritméticas con la función de sumar los datos de entrada, recibiendo dos valores y obteniendo la suma resultante como salida.

Al ejecutar una instrucción la herramienta presenta una secuencia de colores para definir cada tipo de señal, en la que cada color representa un conjunto de bits generados por una unidad funcional determinada. En la parte inferior izquierda aparecerá una leyenda basada en la instrucción actual, mostrando la manera en que se dividen los bits y se envían a cada unidad funcional, el mnemónico específico de la instrucción y una descripción de las señales de control que llegan a cada unidad funcional.

Además, se puede hacer clic en algunas unidades funcionales mientras se produce la ejecución lo que le permitirá visualizar su funcionamiento. En particular se permite observar la unidad de control, el banco de registros y la unidad de control de la ALU.

# Desarrollo de la práctica

Para conocer la herramienta partiremos del siguiente programa:

|  |
| --- |
| ####################################  # #  # Código ejemplo de la actividad 1 #  # #  # #  ####################################      .data  n: .word 7    .text la $t1, n add $s1, $t1, $zero  lw $t5, 0($t1) addi $s1, $s1, 4 subi $t5, $t5, 1  bucle: beq $t5, $zero, exit  sw $t5, 0($s1) exit: li $v0, 10  syscall |

**Actividad 1**

* Abre la herramienta MIPS X-RAY y ensambla el código de la actividad 1. A continuación, puedes avanzar por el código utilizando el botón verde "Step" situado en la propia herramienta.
* Observa para cada instrucción el recorrido del flujo de datos y señales de control. Los cables rojos indican que la señal de control está activada
* Haz clic en cada una de las unidades funcionales que se permita: Unidad de control, Banco de registros y unidad de control de la ALU.

**Cuestión 1.**

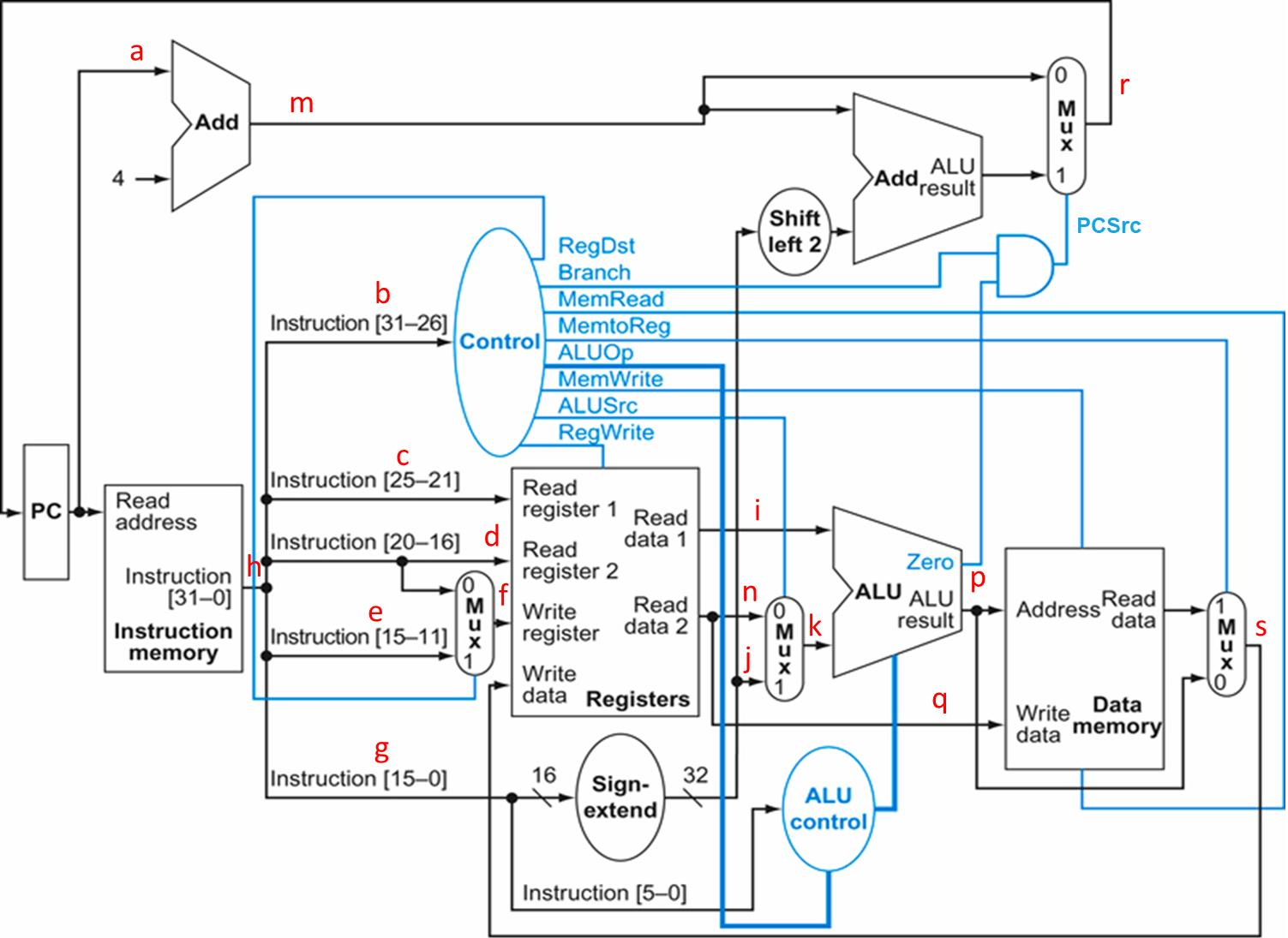
* ¿Qué señales de control se activan durante la ejecución de la instrucción lw y sw?
* ¿Cuál es el valor de los bits de operación de la ALU (Op1, Op2 y Op3) para cada instrucción del programa? ¿Cuál es el valor de opALU1 y opALU0 para cada instrucción? ¿Por qué la señal opALU1 y opALU0 es la misma para las instrucciones lw y sw?

**Actividad 2**

* Ejecuta el código de la actividad 2 en el simulador MARS. Si lo deseas puedes utilizar la herramienta MIPS X Ray para ver el recorrido realizado por las instrucciones en la ruta de datos.
* Identifica el ciclo de reloj en el que se está ejecutando cada instrucción.

|  |
| --- |
| ####################################  # #  # Código ejemplo de la actividad 2 #  # #  # #  ####################################      .data  num: .word 7  .word 3, 8    .text  addi $t0, $zero, 5 addi $t1, $zero, 3 la $t4, num lw $t3, 4($t4) beq $t3, $t1, salto  add $t1, $t1,$t0 j final  salto: add $t1, $t3, $t1  final: addi $a0, $t1, 0  li $v0, 1 syscall li $v0, 10  syscall |

En la figura 2 se muestra la ruta de datos monociclo del MIPS en la que se han etiquetado con las letras *a,b c, d, e, f g, h i, j k m, p q, r, s* distintos partes de la ruta.



*Figura 2. Ruta de datos MIPS monociclo*

**Cuestión 2.**

* Identifica los valores que contendrán las etiquetas de la figura 2 cuando se ejecute la instrucción en el ciclo especificado. Rellena la tabla 1 con el valor de las etiquetas en formato decimal (dec), hexadecimal (hex) o en cualquiera de los dos, según se indique. Si la instrucción no atraviesa la etiqueta o no se conoce el valor, márcala como **X**.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Núm** | | **Etiqueta** | | **Valor durante**  **ciclo 5** | | **Valor durante**  **ciclo 6** | | **Valor durante**  **ciclo 7** | |
| 1 | | **a** (en hex) | |  | |  | |  | |
| 2 | | **b** (hex o dec) | |  | |  | |  | |
| 3 | | **c** (en dec) | |  | |  | |  | |
| 4 | | **d** (en dec) | |  | |  | |  | |
| 5 | | **e** (en dec) | |  | |  | |  | |
| 6 | | **f** (en dec) | |  | |  | |  | |
| 7 | | **g** (en dec) | |  | |  | |  | |
| 8 | | **h** (en hex) | |  | |  | |  | |
| 9 | | **i** (hex o dec) | |  | |  | |  | |
| 10 | | **j** (en dec) | |  | |  | |  | |
| 11 | | **k** (en dec) | |  | |  | |  | |
| 12 | | **m** (en hex) | |  | |  | |  | |
| 13 | | **n** (en dec) | |  | |  | |  | |
| 14 | | **p** (hex o dec) | |  | |  | |  | |
| 15 | | **q** (en dec) | |  | |  | |  | |
| 15 | | **r** (en hex) | |  | |  | |  | |
| 17 | | **s** (hex o dec) | |  | |  | |  | |

*Tabla 1. Valor de los datos en la ruta de datos monociclo durante distintos ciclos de reloj*

**Cuestión 3.**

* Identifica el valor de las señales de control de la figura 2 que se activarán cuando se ejecute la instrucción en el ciclo especificado. Rellena la tabla 2 con el valor de la señal de control.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Núm** | **Señales de control** | **Valor durante**  **ciclo 4** | **Valor durante**  **ciclo 5** | **Valor durante**  **ciclo 6** |
| 18 | **RegDst** |  |  |  |
| 19 | **Branch** |  |  |  |
| 20 | **MemRead** |  |  |  |
| 21 | **MemtoReg** |  |  |  |
| 22 | **ALUop** |  |  |  |
| 23 | **MemWrite** |  |  |  |
| 24 | **ALUSrc** |  |  |  |
| 25 | **RegWrite** |  |  |  |
| 26 | **PCSrc** |  |  |  |
| 27 | **Zero** |  |  |  |

*Tabla 2. Valor de las señales de control en la ruta de datos monociclo durante distintos ciclos de reloj*

**Actividad 3**

* Suponer que el código de la actividad 3 se ejecuta en la ruta de datos monociclo, eres libre si quieres de utilizar la herramienta MIPS X Ray.

####################################

# #

# Código ejemplo de la actividad 3 #

# #

# #

####################################

.data

A: .word 2, 4, 6, 8, 10, 12, 14, 16, 18

n: .word 9

.text main: add $t0, $0, $zero addi $t1, $zero, 1 la $t3, n lw $t3, 0($t3)

fib: beq $t3, $0, final add $t2,$t1,$t0 move $t0, $t1 move $t1, $t2

subi $t3, $t3, 1

j fib

final: addi $a0, $t0, 0

li $v0, 1

syscall li $v0, 10 syscall

**Cuestión 4.**

* ¿Cuántos ciclos de reloj tarda en ejecutarse el código completo?
* Si la frecuencia de reloj es de 3GHz, ¿Cuánto tiempo tarda en ejecutarse?

**Cuestión 5.**

* Modifica el programa de la actividad 4 para que almacene en la memoria los distintitos valores de Fibonacci, para ello sobrescribe el vector A. ¿Cuántos ciclos de reloj tarda ahora en ejecutarse el código completo?
* Si la frecuencia de reloj es la misma que en la cuestión 4, ¿Cuánto tiempo tarda ahora en ejecutarse el programa?

**Cuestión 6.**

* Modifica de nuevo el programa original de la actividad 4 para que muestre por pantalla el doceavo valor de la sucesión de Fibonacci. ¿Cuántos ciclos tarda ahora en ejecutarse el programa? ¿Cuánto tiempo necesitaría para ejecutarse si la frecuencia de reloj fuera de 500MHz?

# Resumen

* En una ruta de datos monociclo todas las instrucciones se ejecutan en un ciclo de reloj.
* La duración del ciclo de reloj viene determinada por la instrucción más lenta.
* Las unidades funcionales en una ruta de datos monociclo solo pueden utilizarse una vez durante la ejecución de una instrucción.