Laboratorio de Estructura de Computadores
Curso 2021-2022

Práctica 4: Diseño del Microprocesador MIPS

Ejercicio 1. Preparación de la memoria de programa y datos

En la práctica 3 se escribieron varios programas en ensamblador para MIPS. Se vio que el simulador MARS, entre otras muchas funciones, permite ver el código máquina de las instrucciones ensamblador escritas en un programa. En este ejercicio se ofrece un programa ensamblador y unos ficheros VHDL que modelan las memorias de Programa ("*MemProgMIPS.vhd*") y de Datos ("*MemDataMIPS.vhd*") del microprocesador que se está diseñando. Estas memorias tienen un fragmento de los segmentos de código y datos del programa, pero falta por completar algunas instrucciones y datos.

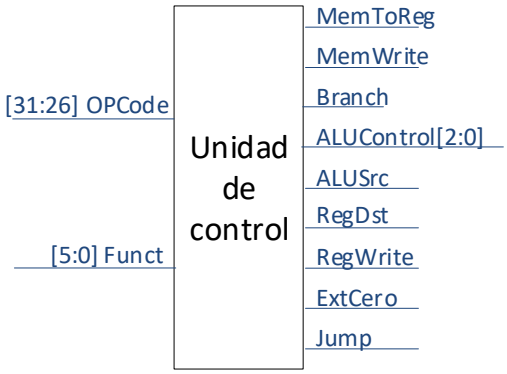
Objetivo

Utilizando la traducción a código máquina que ofrece el simulador MARS, se pide completar las memorias de Programa y Datos.

Este ejercicio se valorará en la calificación de evaluación continua si se termina y muestra al profesor durante la primera sesión de prácticas.

Ejercicio 2a. Diseño de la unidad de control

Diseñar la unidad de control del microprocesador MIPS uniciclo. La interfaz de este módulo se presenta en la imagen de la derecha.



Este módulo genera 9 señales de control. Estas señales de control definen el comportamiento del resto de elementos que componen el microprocesador.

Como entradas recibe los campos *OPCode* y *Funct* de la instrucción.

En la tabla adjunta se muestran las instrucciones que debe ejecutar el microprocesador:

Tabla I

OPCode	Nombre	Descripción	Operación
000000	R-Type	Instrucciones R-Type	Se describen en la siguiente tabla
000010	j	Salto incondicional	PC = JTA
000100	beq	Bifurca si igual (Z = 1)	Si ([rs] == [rt]); PC =BTA
001000	addi	Suma con dato inmediato	[rt] = [rs] + SigImm
001100	andi	AND con dato inmediato	[rt] = [rs] & ZeroImm
001101	ori	OR con dato inmediato	[rt] = [rs] | ZeroImm
001110	xori	Función XOR	[rd] = [rs] ⊕ ZeroImm
100011	lw	Lee una palabra de memoria	MEM ([rs]+SigImm) => [rt]
101011	sw	Escribe una palabra en memoria	[rt] => MEM ([rs]+SigImm)
001010	slti	Set on less than (inmediato)	[rs]<[SigImm] ? [rt]=1 : [rt]=0

R-TYPE

Funct	Nombre	Descripción	Operación
100100	and	Función AND	[rd] = [rs] & [rt]
100000	add	Sumar	[rd] = [rs] + [rt]
100010	sub	Restar	[rd] = [rs] - [rt]
100110	xor	Función XOR	[rd] = [rs] ⊕ [rt]
100111	nor	Función NOR	[rd] = ~ ([rs] | [rt])

100101	or	Función OR	[rd] = [rs] | [rt]
101010	slt	Set on less than	[rs]<[rt] ? [rd]=1 : [rd]=0

Para facilitar la implementación de la unidad de control, se recomienda rellenar la siguiente tabla con los valores que debe tener cada señal de control para cada una de las instrucciones:

Tabla II

INSTRUCCIÓN	MemToReg	MemWrite	Branch	ALUControl	ALUSrc	RegDst	RegWrite	ExtCero	Jump
R-Type									
Lw									
Sw									
Beq									
Lógicas Inm									
Aritméticas Inm									
J									

Objetivo

Comprender el problema propuesto, y completar la tabla de control utilizando el esquemático que se encuentra en la última página del enunciado.

Este ejercicio se valorará en la calificación de evaluación continua si se termina y muestra al profesor durante la primera sesión de prácticas.

Ejercicio 2b. Implementación en VHDL de la unidad de control

Utilizando la tabla del anterior ejercicio, diseñar en VHDL la unidad de control del microprocesador MIPS uniciclo.

Objetivo

Comprender el problema propuesto. Plantear e implementar una solución para dicho problema, para ello debe crear desde cero, un archivo denominado UnidadControl.vhd, es decir debe diseñar las librerías, la entidad y la arquitectura. Como apoyo en el diseño del archivo de la unidad de control y para para probar su funcionalidad se le facilita el archivo con el banco de pruebas "UnidadControlTb.vhd".

Este ejercicio se valorará en la calificación de evaluación continua si se termina y muestra al profesor durante la segunda sesión de prácticas.

Ejercicio 3. Diseño completo del microprocesador

En este ejercicio se deberá realizar el diseño completo del microprocesador. Para llevar a cabo este ejercicio se deberán incluir los módulos de la ALU (ALUMIPS.vhd) y del banco de registros (RegsMIPS.vhd) de la práctica 2.

El sistema que se está desarrollando sigue la arquitectura Harvard, con una memoria para el código y otra para los datos. Las entradas del microprocesador "MicroMIPS" son una señal de *Reset* activa a nivel bajo (NRst), el reloj (Clk), la entrada de memoria de programa (MemProgData) y la entrada de memoria de datos (MemDataDataRead). Como salidas, la señal que habilita la escritura en memoria de datos (MemDataWe), la dirección de memoria de programa (MemProgAddr), la dirección de memoria de datos (MemDataAddr) y el bus de escritura en memoria (MemDataDataWrite).

Al final de este enunciado se puede encontrar un esquemático correspondiente al microprocesador que debe implementarse. Este esquemático corresponde a las transparencias de la unidad 4, aunque ya están añadidos los elementos lógicos necesarios para implementar las instrucciones que no figuran en dicha unidad.

Validación

Para realizar las pruebas del microprocesador se deben usar las memorias de programa y de datos del ejercicio 1. La memoria de programa ("*MemProgMIPS.vhd"*) describe una memoria ROM que recoge las instrucciones del programa. La memoria de datos ("*MemDataMIPS.vhd"*) es una memoria de lectura asíncrona y escritura síncrona en flanco de subida. El testbench que se suministra ("*MicroMIPSTb.vhd*") instancia la CPU a diseñar por el alumno en el ejercicio 3 y las memorias completadas en el ejercicio 1.

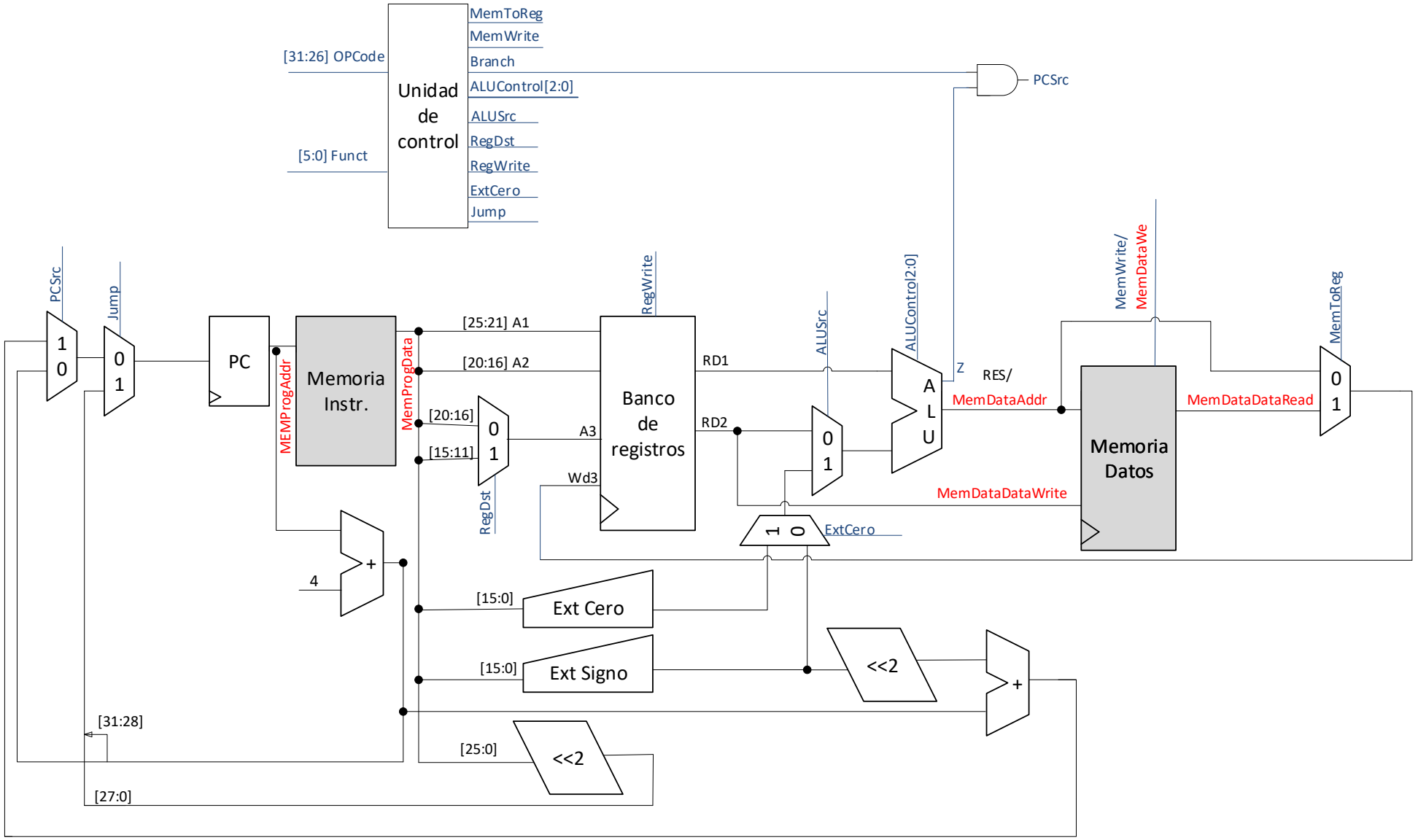
Nota: En la corrección de la práctica, para poder utilizar el archivo **.do** facilitado, al instanciar el Baco de Registros en el archivo MicroMIPS.vhd, es necesario que el nombre de dicha instancia sea GPR [GPR: RegsMIPS port map (…].

En este ejercicio se debe ejecutar el programa y comprobar que el resultado obtenido en el banco de registros del microprocesador diseñado, es el mismo que el que se obtiene utilizando el simulador MARS cuando se ejecuta el fichero en ensamblador *Ejercicio1.asm*. Para comprobar el correcto funcionamiento del micro se sugiere utilizar las formas de onda incluidas en "Wave_Ejercicio3.do" (script para la visualización de señales y el banco de registros en *ModelSim*), añadiendo las señales que se considere oportuno.

Objetivo

Comprender el problema propuesto. Plantear e implementar una solución para dicho problema. Ejecutar el banco de pruebas con las memorias entregadas para corregir los errores cometidos durante la fase de diseño.

Este ejercicio se valorará en la calificación de evaluación continua si se termina y muestra al profesor durante la tercera sesión de prácticas.



Señales de control **Entradas/Salidas** **Datos**