Laboratorio de Estructura de Computadores
Curso 2021-2022

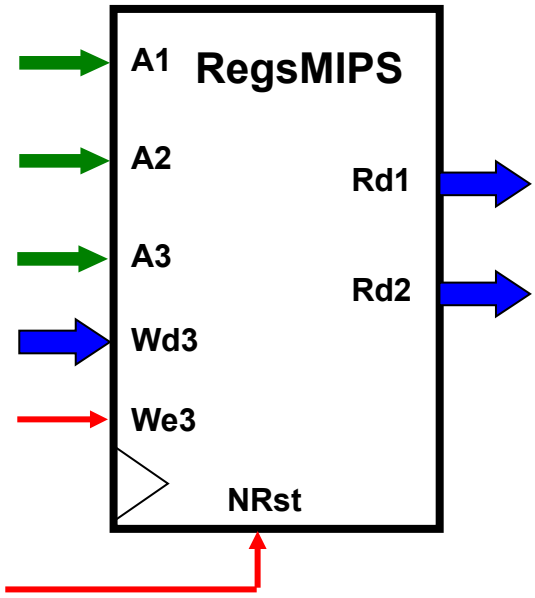
Práctica 2: Microprocesador simplificado, GPR y ALU

Ejercicio 1. Diseño del banco de registros.

Diseñar en VHDL el banco de registros de propósito general (entidad “RegsMIPS”) del microprocesador MIPS, cuya interfaz se muestra en la figura adjunta. Se proporciona el banco de pruebas completo (fichero "*RegsMIPSTb.vhd*").

Este banco consta de 32 registros de 32 bits, que están identificados del 0 al 31. El registro 0 siempre tiene el valor de 0, por lo que debe bloquearse su escritura.

El banco permite la escritura síncrona de un único registro (registro destino). La dirección de este registro viene dada por la señal de 5 bits A3, que puede tomar valores entre 0 y 31. El valor que tomará el registro se indica en la entrada de 32 bits Wd3. La escritura se hace efectiva cuando se produce un flanco ascendente del reloj y está activada la señal de habilitación We3. El banco de registros incluye una señal de NRst activa a nivel bajo que inicializará todos los registros a valor 0 cuando se active.



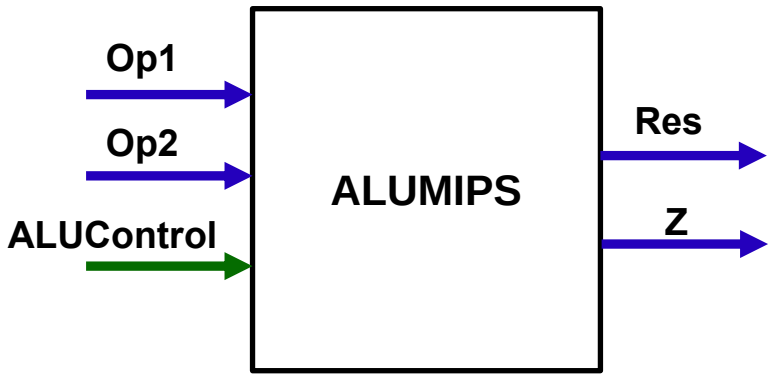
Además, este banco permite la lectura asíncrona simultánea de dos registros independientes. Las direcciones de estos registros se indican respectivamente en las señales de entrada de 5 bits: *A1* y *A2*. Los valores de los registros seleccionados son copiados a las salidas Rd1 y Rd2, respectivamente.

Objetivo

Diseñar y verificar mediante el testbench proporcionado (“*RegsMIPSTb.vhd*”) el funcionamiento del banco de registros. Este banco de registros será el que se utilice en la implementación del micro completo. Este ejercicio se valorará en la calificación de evaluación continua si se termina y muestra al profesor durante la primera sesión de prácticas.

Ejercicio 2. Diseño de la ALU.

Diseñar una unidad aritmético-lógica combinacional (entidad "*ALUMIPS*") con dos entradas de datos de 32 bits (señales *OP1* y *OP2*), una entrada de selección de operación de 3 bits (señal ALUControl), una salida de datos de 32 bits (*Res*) y una salida de estado de un bit (*Z*). Esta señal de estado corresponde a la bandera de Z, la cual se debe activar cuando el resultado sea igual a cero.



Los valores de entrada de selección codifican las siguientes operaciones:

ALUControl[2:0]	Operación
000	OP1 AND OP2
001	OP1 OR OP2
010	OP1 + OP2
011	OP1 XOR OP2
100	SIN USAR
101	OP1 NOR OP2
110	OP1 – OP2
111	OP1 SLT OP2*

* La instrucción *slt* pone a 0x00000001 el resultado cuando el operando 1 es menor que el operando 2. En caso contrario, el resultado será 0x00000000. Esta instrucción tiene en cuenta el signo de los operandos de entrada que considera en complemento a 2 (p. ej. un número negativo es menor que uno positivo).

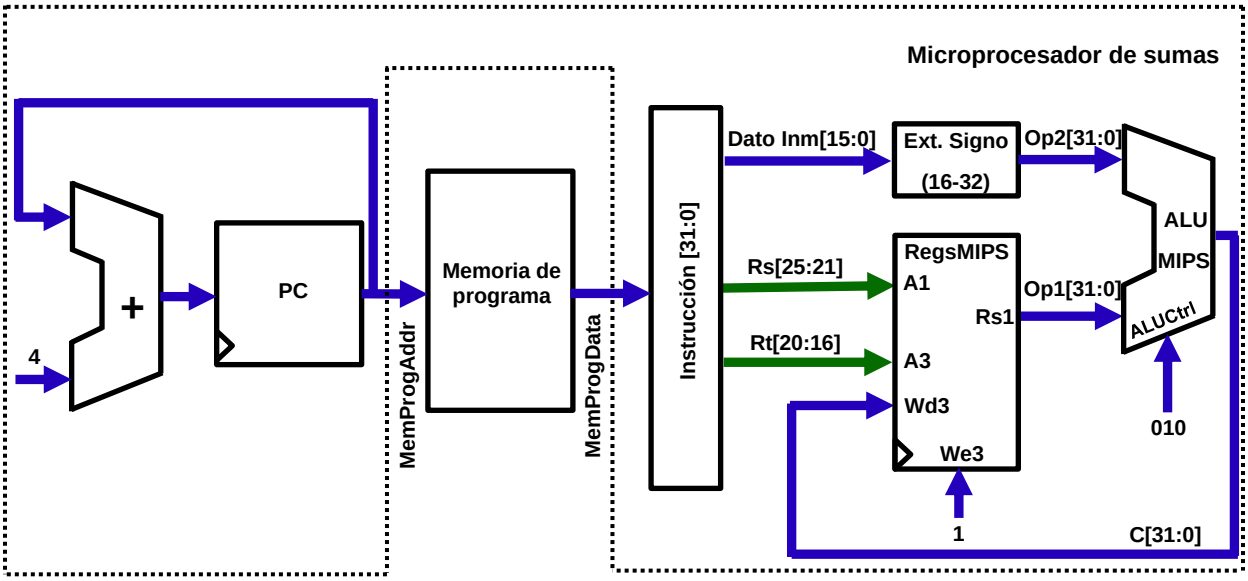
Para verificar el diseño correcto de la ALU, se proporciona un *testbench* ("*ALUMIPSTb.vhd*") que comprueba el correcto funcionamiento de la ALU. El *testbench* indicará si la ALU no realiza correctamente alguna operación.

Objetivo

Diseñar y verificar mediante un testbench el funcionamiento de una ALU. Este ejercicio se valorará en la calificación de evaluación continua si se termina y muestra al profesor durante la primera sesión de prácticas.

Ejercicio 3. Diseño del microprocesador de sumas.

En este ejercicio se trata de completar el diseño de un microprocesador que permita realizar únicamente sumas entre un registro y un dato inmediato. Para llevar a cabo este ejercicio se deben añadir al banco de registros los distintos elementos que proporcionan la funcionalidad necesaria y conectarlos tal y como se muestra en la siguiente figura.



El microprocesador a desarrollar no incluye la memoria de programa, que se suministra junto con el enunciado en el fichero "*MemProg.vhd*", que se debe incorporar al proyecto como un archivo adicional pero que tan sólo es necesario para ejecutar el testbench de prueba del circuito.

Se suministra la entidad vacía del micro a desarrollar, "*MicroSuma.vhd*". Aparte del reloj (*Clk*) y del *Reset* (*NRst*, asíncrono y activo por nivel bajo que pone los registros del banco de registros a 0 y también el contador de programa), la única entrada de esta entidad es *MemProgData*, de 32 bits, por donde llega de la memoria el código de cada instrucción. Y la única salida es *MemProgAddr*, también de 32 bits, por donde se debe suministrar el contador de programa, PC, que por tanto ha de ser también de 32 bits. Las tareas a desarrollar en el interior de esta entidad son las siguientes:

1. Instanciar el módulo del banco de registros del ejercicio 1. La dirección A2 no se usa, así que puede recibir cualquier valor. Igualmente, la salida Rs2 no se utiliza así que se puede utilizar el término *open* cuando vaya a instanciarla. Por último, la entrada We3 debe estar siempre activada, puesto que los resultados de las sumas deben ser escritos en el banco de registros.
2. Instanciar el módulo de la ALU del ejercicio 2. La ALU siempre debe sumar, por lo que el puerto ALUControl debe tener el valor 010. Por otra parte, la salida Z no se usa, así que se puede utilizar el término *open* cuando vaya a instanciarla.
3. Implementar una extensión de signo de 16 a 32 bits.
4. La ruta de datos para el registro contador de programa (PC). El PC consiste en un registro de 32 bits y la circuitería que lo incrementa en 4 cada ciclo de reloj (tipo contador).

Para verificar el correcto funcionamiento del microprocesador, se suministra un banco de pruebas, "*MicroSumaTb.vhd*". En él se instancia al micro a desarrollar y a la memoria suministrada, haciendo que se ejecute el programa de la misma, a saber:

```
R1 = R0 +10;
R2 = R1 + 5;
R3 = R2 + 25;
R0 = R0 + 5;
R4 = R3 - 5;
```

Se debe comprobar el correcto funcionamiento del microprocesador comprobando los valores de los registros R0, R1, R2, R3 y R4, que son señales internas del banco de registros.

Objetivo

Comprender el problema propuesto, plantear e implementar una solución para dicho problema. Se recomienda consultar el primer epígrafe de la Unidad 2 de teoría. Este ejercicio se valorará en la calificación de evaluación continua si se termina y muestra al profesor durante la segunda sesión de prácticas.

Otros ejercicios.

- ¿Qué cambios necesitaría el banco de registros si el reset fuera síncrono?
- ¿Qué cambios necesitaría el banco de registros si las lecturas fueran síncronas?
- ¿Qué cambios necesitaría la ALU si se implementara la bandera N (negativo), que sólo debe activarse cuando el resultado es negativo?
- ¿Qué cambios necesitaría la ALU si se implementara la bandera C (carry), que se activa (C = '1'), cuando se genera un acarreo de salida en una operación aritmética?
- ¿Qué cambios necesitaría la ALU si se implementara la bandera V (overflow), que se activa (V='1'), cuando el resultado de la operación sobrepasa la capacidad de representación con números en complemento a 2?
- ¿Cambiaría el tamaño de alguna entrada o salida si la ALU puede implementar 10 operaciones diferentes?
- ¿Qué cambios habría que hacer en el microprocesador para que realizara operaciones AND sabiendo que dicha operación necesita que el dato inmediato esté extendido en cero?
- ¿Cómo implementar en la ALU las operaciones de desplazamiento sll (*shift left logic*), srl (*shift right logic*) y sra (*shift right arithmetic*), en las que el desplazamiento máximo de 31 posiciones viene señalado por los 5 lsb del segundo operando?
- ¿Cuánto debería sumar el contador PC si cada instrucción estuviera escrita en 64 bits?