Diseño Lógico 2016

Ingeniería en Computación

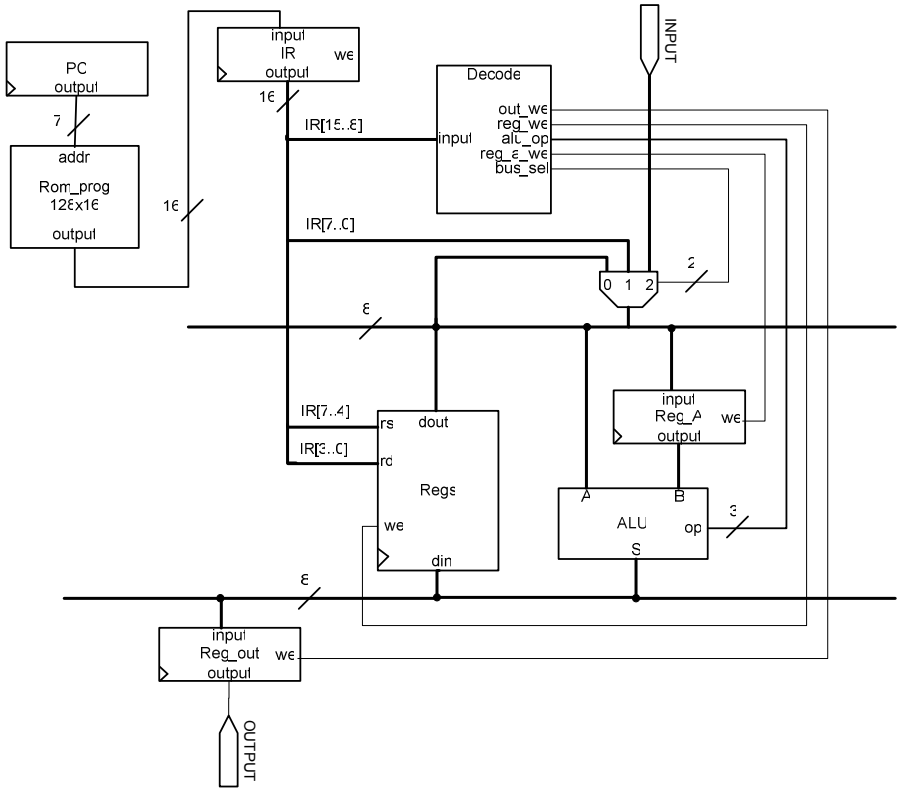
Práctica de Laboratorio II: Procesador

El objetivo de esta práctica es implementar un procesador uniciclo (todas las instrucciones se realizan en un único ciclo de reloj), a partir de la descripción VHDL de sus componentes.

Este procesador se encuentra compuesto por:

- **Memoria de instrucciones** (ROM): almacena el programa instrucción por instrucción
- **Contador de programa** (PC): dirección de instrucción a ejecutar
- **Registro de instrucción** (IR): almacena la instrucción actual a ejecutar
- **Unidad de decodificación** (Decode): determina las señales de control que deben activarse para ejecutar la instrucción.
- **Banco de registros** (Regs): compuesto por 16 registros de 8 bits
- **Registro acumulador** (Reg_a): permite almacenar temporalmente el operando B de la ALU
- **Unidad aritmético lógica** (ALU)
- **Registro de salida** (Reg_out)

La arquitectura general se define en la siguiente figura,



Buses

Los buses de datos tienen un ancho de 8 bits, mientras que el de instrucción (IR) posee un ancho de 16 bits. En el esquema presentado se omiten las interconexiones de reloj y reset, para simplificar el diseño.

Instrucciones:

El registro de instrucción (IR) contiene la instrucción a ejecutar en el ciclo actual de operación y se divide en los siguientes campos:

Código de instrucción	rd	rs
8 bits	4 bits	4 bits

Código de instrucción	immediate
8 bits	8 bits

El procesador debe soportar las siguientes instrucciones con sus correspondientes códigos de operación.

IN rd

Codigo de instrucción: 0x01
Descripción: Reg[rd] = IN

OUT rs

Codigo de instrucción: 0x02
Descripción: Reg_out = Reg[rs]

MOV rd, rs

Codigo de instrucción: 0x03
Descripción: Reg[rd] = Reg[rs]

LDA rs

Codigo de instrucción: 0x04
Descripción: Reg_A = Reg[rs]

LDI immediate

Codigo de instrucción: 0x05
Descripción: Reg_A = immediate

ADD rd, rs

Codigo de instrucción: 0x10
Descripción: Reg[rd] = Reg[rs] + Reg_A

SUB rd, rs

Codigo de instrucción: 0x11
Descripción: Reg[rd] = Reg[rs] - Reg_A

SHL rd, rs

Codigo de instrucción: 0x20
Descripción: Reg[rd] = Reg[rs] << 1

SHR rd, rs

Codigo de instrucción: 0x21
Descripción Reg[rd] = Reg[rs] >> 1

AND rd, rs

Codigo de instrucción: 0x12
Descripción: Reg[rd] = Reg[rs] and Reg_A

OR rd, rs

Codigo de instrucción: 0x13
Descripción: Reg[rd] = Reg[rs] or Reg_A

XOR rd, rs

Codigo de instrucción: 0x14
Descripción: Reg[rd] = Reg[rs] xor Reg_A

Ejercicios:

1. Complete la tabla e implemente la *Decode* (Decoding Unit). Esta unidad permite la activación de las señales de control a partir del valor del código de instrucción almacenado en IR. Se recomienda realizar la minimización de las funciones.

Instrucción	bus_sel	alu_op	reg_a_we	out_we	reg_we
IN (0x01)	10	000	0	0	1
OUT (0x02)					
MOV (0x03)					
LDA (0x04)	00	000	1	0	0
LDI (0x05)					
ADD (0x10)					
SUB (0x11)	00	011	0	0	1
AND (0x12)					
OR (0x13)					
XOR (0x14)					
SHL (0x20)					
SHR (0x21)					

2. Utilizando la unidad de decodificación del ejercicio 1, junto con los componentes desarrollados realice la arquitectura del procesador.

3. Implemente el testbench del procesador y realice la simulación del procesador con el siguiente conjunto de instrucciones, verificando el correcto funcionamiento:

```
0: in r3
1: lda r3
2: add r4, r3
3: sub r5,r4
4: or r6, r4
5: and r7, r0
6: mov r14, r4
7: out r3
8: out r4
9: out r5
10: out r6
11: out r7
12: out r8
13: out r13
14: out r14
```

Entrega:

Se debe realizar la entrega del archivo “proc.vhd” completo, el archivo “proc.vhd” y el resto de los archivos que se realizaron para lograr la implementación del procesador, junto con un informe en donde se describa el desarrollo del trabajo y las conclusiones.
Los trabajos prácticos desarrollados deben ser enviados por mail **antes** del 27 de noviembre.
Todos los desarrollos adicionales que se realicen serán tenidos en cuenta en la evaluación.