Electrónica Digital
Ingeniería de Sistemas – Curso 2022

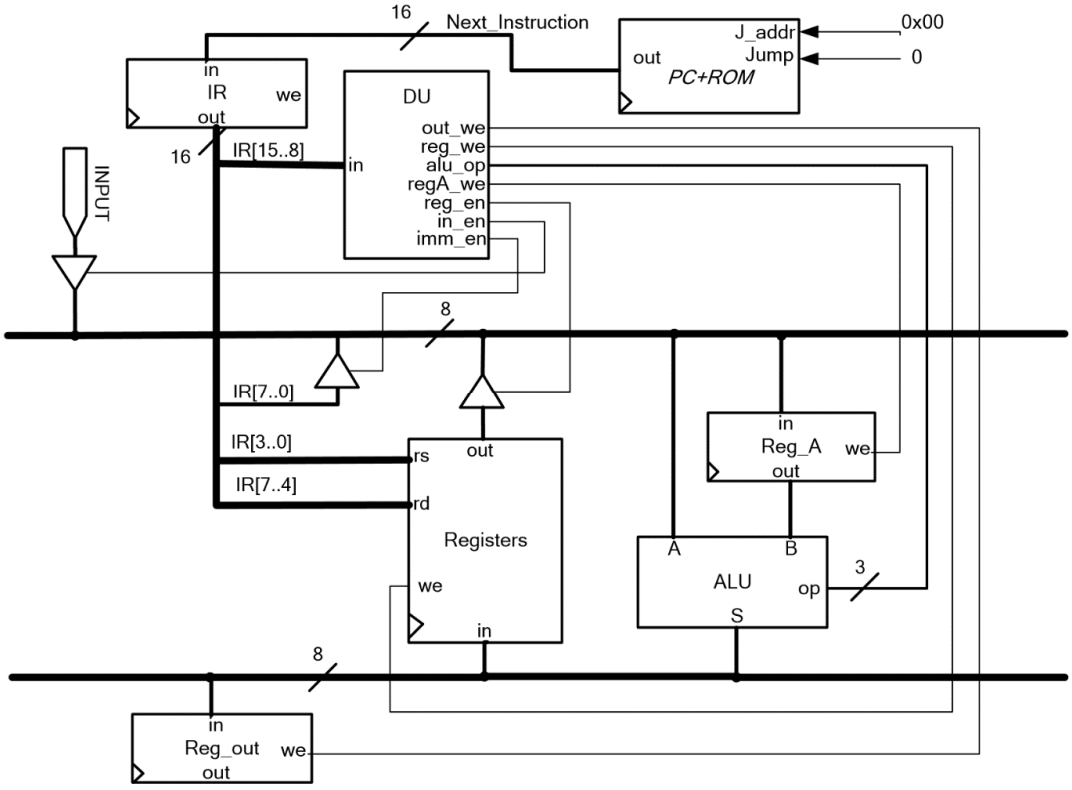
Laboratorio – Práctica 4
Procesador

Implemente un procesador a partir del diseño esquemático de sus componentes.

Este procesador se encuentra compuesto por:

- Memoria de instrucciones (*ROM*): almacena el programa instrucción por instrucción
- Unidad Aritmético Lógica (*ALU*)
- Registro de instrucción (*IR*): almacena la instrucción actual a ejecutar
- Unidad de decodificación (*DU*): determina las señales de control que deben activarse para ejecutar la instrucción.
- Banco de registros (*Registers*): compuesto por 16 registros de 8 bits
- Registro acumulador (*Reg_A*): permite almacenar temporalmente el operando *B* de la *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.

Nota: Por simplificación del esquema, se omiten las interconexiones de reloj y *reset*.

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 soporta las siguientes instrucciones con sus correspondientes códigos de operación.

IN *rd*

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

OUT *rs*

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

MOV *rd*, *rs*

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

LDA *rs*

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

LDI *immediate*

Codigo de instrucción: 0x05
Descripción: *Reg_A* = *immediate*

ADD *rd*, *rs*

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

SUB *rd*, *rs*

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

SHL *rd*, *rs*

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

SHR *rd*, *rs*

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

AND *rd*, *rs*

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

OR *rd*, *rs*

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

XOR *rd*, *rs*

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

Realice:

1. Complete la tabla e implemente la *DU* (*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	*in_en*	*reg_en*	*alu_op*	*regA_we*	*out_we*	*reg_we*	*Inm_en*
IN (0x01)	1	0	000	0	0	1	0
OUT (0x02)							
MOV (0x03)							
LDA (0x04)	0	1	000	1	0	0	0
LDI (0x05)							
ADD (0x10)							
SUB (0x11)	0	1	011	0	0	1	0
AND (0x12)							
OR (0x13)							
XOR (0x14)							
SHL(0x20)							
SHR(0x21)							

2. Utilizando la unidad de decodificación del ejercicio 1, junto con componentes desarrollados en los prácticos anteriores (ALU, banco de registros, registros), realice la arquitectura del procesador. Se permite el uso de *buffers triestado* con anchos de datos de 8 bits. Los mecanismos de entrada/salida (*clock*, *reset*, *INPUT*, *OUTPUT*) deben ser implementados de manera que facilite el ingreso/visualización de datos durante la simulación.

3. Realice un programa que posea todas las instrucciones soportadas por el procesador. Simule el comportamiento del procesador implementado en LOGISIM ejecutando el programa realizado.