Assignment 3

Instructions

- For this homework, you may only use:
    a) All of the components under Wiring
    b) All of the components under Gates **EXCEPT** for **Controlled Buffer**, **Controlled Inverter**, **PLA**
    c) All of the components under Plexers
    d) All of the components under Arithmetic
    e) All of the components under Memory **EXCEPT** for **ROM,** and **Random Generator**
    f) You can use in-built Logisim components like **Adder, Subtractor, MUX, etc**
- Please label all inputs and outputs.
- Please form groups of two for this assignment.
- Write down your and your teammate's names and OS used in the comment section of your Canvas submission.
- Download your *.circ* file once done and upload that file on Canvas.
- Assignment 3 will be a total of 100 points.
- Good luck!

You have been provided a Register File circuit in the starting circuit. The **only outputs** of the register file that you are allowed to use are A_Out and B_Out.

Starter file: (Download it first, then open it on Logisim)
Mac: Link
Windows: Link

CPU.circ (100 points)

Build a 16-bit single cycle CPU (with a 4-bit opcode) that can implement the given instructions.

Instruction Format

Our CPU will be using fixed length instructions. Our CPU will also have two types of instruction formats: R-type and I-type. In R-type instructions both operands come from registers. In I-type instructions, the first operand comes from a register and the second will be contained within the instruction.

R-Type

Name	Bits	Description
OpCode	15 - 12	Determines what operation should be performed
C	11 - 8	The destination register. The C in $Reg_C = Reg_A$ OP Reg_B
A	7 - 4	The first source register. The A in $Reg_C = Reg_A$ OP Reg_B
B	3 - 0	The second source register. The B in $Reg_C = Reg_A$ OP Reg_B

I-Type

Name	Bits	Description
OpCode	15 - 12	Determines what operation should be performed
C	11 - 8	The destination register. The C in $Reg_C = Reg_A$ OP Imm
A	7 - 4	The first source register. The A in $Reg_C = Reg_A$ OP Imm
Immediate	3 - 0	The second source register. The Imm in $Reg_C = Reg_A$ OP Imm

Instructions

Operation	Encoding (The value in the OpCodeField)	Description
NOPx	0000	Do nothing
STOP	0001	The CPU ceases execution
ADDRx	0010	$Reg_C = Reg_A + Reg_B$
SUBRx	0011	$Reg_C = Reg_A - Reg_B$

ANDRx	0100	$Reg_C = Reg_A$ AND Reg_B
ORRx	0101	$Reg_C = Reg_A$ OR Reg_B
XORRx	0110	$Reg_C = Reg_A$ XOR Reg_B
ADDIx	0111	$Reg_C = Reg_A$ +Immediate
SUBIx	1000	$Reg_C = Reg_A$ - Immediate
ANDIx	1001	$Reg_C = Reg_A$ AND Immediate
ORIx	1010	$Reg_C = Reg_A$ OR Immediate
XORI	1011	$Reg_C = Reg_A$ XOR Immediate
NEGATEx	1100	$Reg_C = -Reg_A$
NOTx	1101	Reg_C = NOT Reg_A
MOVEx	1110	$Reg_C = Reg_A$
LOADx	1111	Reg_C = Immediate

Inputs

Pin	Size (in bits)	Explanation
Instruction	16	The instruction located at Instruction_Address
ClkIn	1	The Clock. Connect this to the clock ports of your registers/flip-flops. Do nothing else with this.

Outputs

Pin	Size (in bits)	Explanation
Instruction_Address_Out	5	The address of the instruction you want to execute
Reg0-15	4	The values in the register file. This has already been connected for you

CPU Components

Your CPU should have

- A Program Counter (PC)
    - This stores and keeps track of what instruction you are on
    - Your PC will increment by 2 for each clock cycle.
- Instruction Decoder
    - This is a bunch of combinational logic that sets the control signals inside of your CPU
- Register File
    - A bunch of registers as well as ways to specify which ones you want. This has already been created for you.
    - The **only outputs** of the register file that you are allowed to use are A_Out and B_Out.

The Example Program

Instruction	Meaning	Result
LOAD REG_0, 3 x	Reg_0 = 3	Reg_0 = **3**
LOAD REG_1,6 x	Reg_1 = 6	Reg_1 = **6**
NOPx	Do Nothing	No Change
MOVE REG_2, Reg1	Reg_2 = Reg_1	Reg_2 = **6**
ANDR REG_3, REG_0, REG_1	REG_3 = REG_0 AND REG_1	REG_3 = 3 & 6 = **2**
ANDI REG_4, REG_3,3	REG_4 = REG_3 AND 3	REG_4 = 2 & 3= **2**
ORR REG_5, Reg_2, REG_0	REG_5 = Reg_2 OR REG_0	REG_5 = 6 |3 = **7**
ORI REG_6, Reg_3, 12	REG_6 = Reg_3 OR 12	REG_6 = 2 OR 12 = **14**
XORR REG_7, Reg_2, REG_0 x	REG_7 = Reg_2 XOR REG_0	REG_7 = 6 ^ 3 = **5**
XORI REG_8, Reg_6, 15	REG_8 = Reg_6 XOR 15	REG_8 =14 ^15 = **1**
NEG REG_9, REG_3	REG_9 = -REG_3	REG_9 = **-2**
ADDR REG_{10}, Reg_7, REG_7	REG_{10} = Reg_7 + REG_7	REG_{10} = 5 + 5 = **10**

The test program is just an **example program.** Your CPU should function on **any program** given to it.