In this milestone you have to design MIPS CPU Using VHDL. Your proposed CPU should be able to perform certain instructions. These instructions are R-type instructions and they only include AND, OR, ADD, SUB, SLT and finally NOR instructions and I-type instructions: lw, sw, beq, bne and J-type instruction: j. You may refer to lab08.

Required Instructions:

Instruction	Format	op	funct
add	R	0	32
and	R	0	36
sub	R	0	34
nor	R	0	39
Or	R	0	37
Slt	R	0	42
Lw	I	35	
Sw	I	43	
Beq	I	4	
Bne	I	5	
J	J	2	

- Your Implemented module should be named as "MainModule".
- Entity should look as follows:

  START: IN STD_LOGIC;

  CLK: IN STD_LOGIC;

  RegFileOut1: OUT STD_LOGIC_VECTOR(31 downto 0);

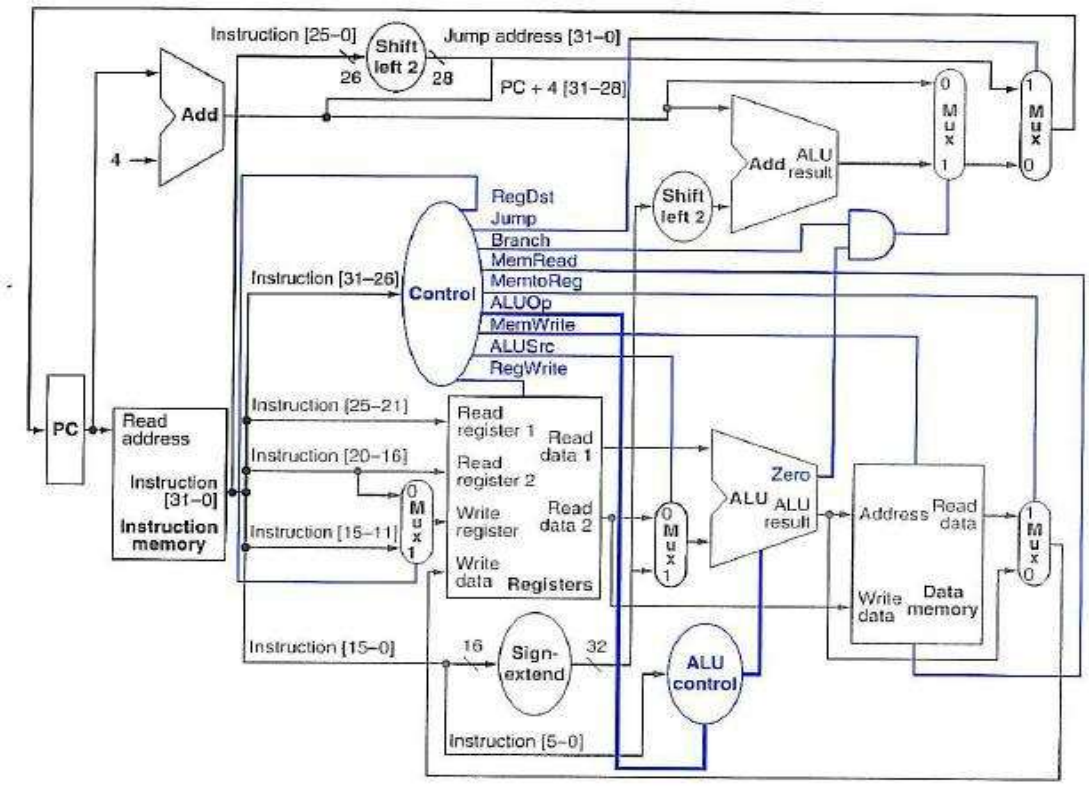
  RegFileOut2: OUT STD_LOGIC_VECTOR(31 downto 0);

  ALUOut: OUT STD_LOGIC_VECTOR(31 downto 0);

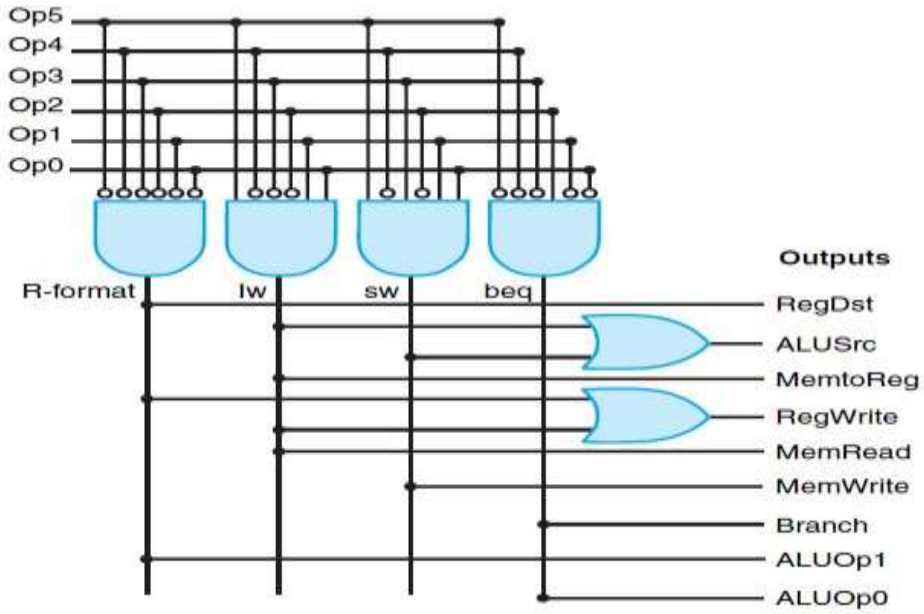
  PCOut: OUT STD_LOGIC_VECTOR(31 downto 0);

  DataMemOut: OUT STD_LOGIC_VECTOR(31 downto 0);

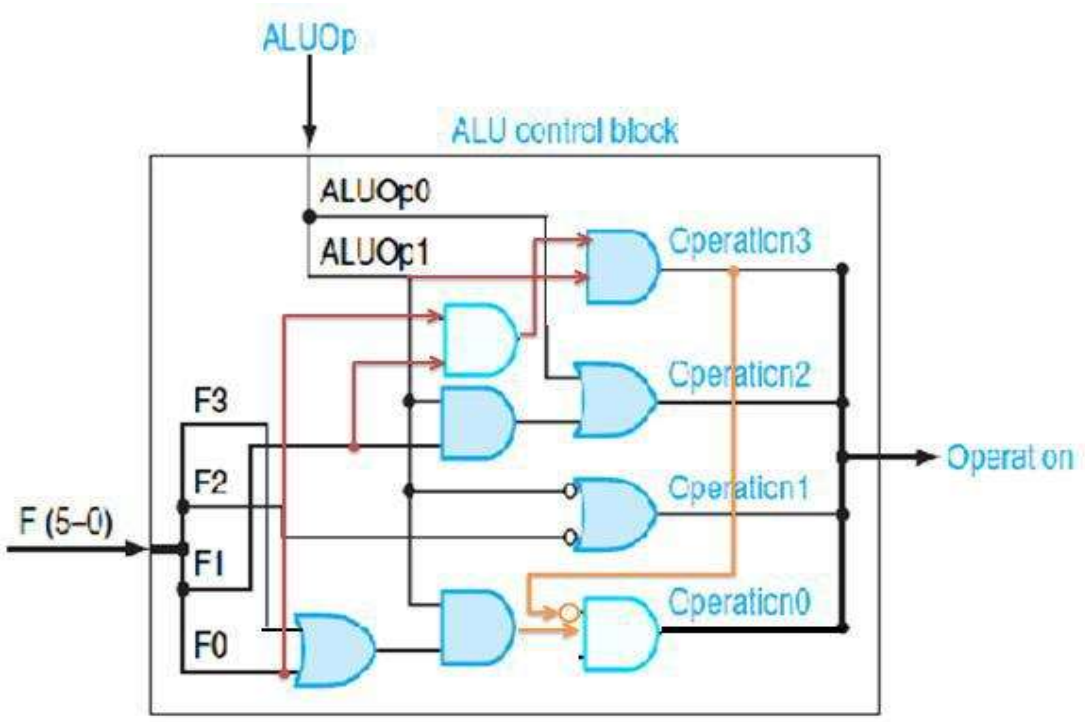
- Required design for MainModule:



- **CU** should match the **RTL** design:



- **ALUControl** should match the **RTL** design:



Given Modules:

- Data Memory
- Instruction Memory
- Generic Register (Rising Edge)

Important Notes:

- Add the implementation of the instruction **bne** to your design (you have to do two modifications, the 1st in MainModule and the 2nd in CU).
- Change back the register **$1 to be initialized by 0** as other registers.
- All registers should work on **Rising Edge.**
- **Test before submit** (attached test case)**.**

Test Case:

- Assembly program that calculate the first **twelve Fibonacci numbers** and store them in array at the beginning of the Data Memory.
- After the program finishes, you should see the following sequence in **$s0**:
  1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 0, -1
- Machine code is already loaded in the Mem's.

Deadline: **Friday, 03 May, 2019 23:59.**

Good luck ☺