National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

EE-421 Digital System Design

Assignment: **2** *Due Date:* ***22nd May 2022 by 10pm*** *[FIRM]* *Total Marks:* **100**

- *This is a group assignment and group size is two. Each group must work alone, without the help from any other person.*
- *Copying material from the web or from documents and submitting it for this assignment will be considered as plagiarism. Copied assignment will not be accepted and will be awarded ZERO.*
- *The Verilog Code should be synthesizable.*
- *The Verilog code should be properly commented.*
- *Final design should be demonstrated on an FPGA board.*
- *An optimized implementation will result in more credits.*
- *Please upload your Assignment on LMS by 10pm on 22nd May2022. You are required to upload your design, testbench and word report file.*

You need to write Verilog Code for a soft core microprocessor "NuCore" and its testbench named "NuCore_TB".

A. The NuCore shall have the following major blocks
1. A 64 deep Instruction Queue, addressed by a Program Counter. The Program Counter resets on the RESET signal. The Program Counter is incremented by one on each subsequent *posedge* of clock.
2. Instruction decode module: The instruction from the Instruction Queue Module is decoded according to the supported instruction format as given in Table below.
3. Two register files: REG_FILE_A (for keeping operands A) and REG_FILE_B (for keeping operands B). Both of these are 16 deep with a word size of 32
4. An Arithmetic Logic Unit that performs the arithmetic as defined by the instruction. The output of ALU is registered and goes directly as the output of NuCore. ALU also produces a registered ZERO FLAG that goes out as an output of "NuCore" module.

B. The testbench "NuCore_TB" shall perform the following:
1. Provide the necessary interfacing signal, CLK and RESET and instantiate the NuCore.
2. The testbench should then perform the testing as follows:
   a. Load the following set of instructions to the instruction queue
      - **i.** RESET
      - **ii.** Store values of 1,2,3,4,5,6,7,8,9,10 at locations 1,2,3,4,5,6,7,8,9,10 of REG FILE A
      - **iii.** Store values of 10,9,8,7,6,5,4,3,2,1 at locations 1,2,3,4,5,6,7,8,9,10 of REG FILE B
      - **iv.** Add all the consecutive locations of REG FILE A & REG FILE B
      - **v.** Bitwise-AND all the consecutive locations of REG FILE A & REG FILE B

The supported instructions are:

Instructions	Instruction Format (MAX 39 BITS)
RESET : Clear Output Register and flags and Register Files	**<3 Bit Instruction>** **Example: 000XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX**
Store Value to Reg File A at specified address	**<3 Bit Instruction><4 bit Address><32 bit Value>** **Example : 00100100000111100001111000011110000 1111**
Store Value to Reg File B at specified address	**<3 Bit Instruction><4 bit Address><32 bit Value>** **Example: 01000100000111100001111000011110000 1111**
Add values Op-A, Op-B using the supplied addresses	**<3 Bit Instruction><4 bit Address to Register File A><4 bit Address to Register File B> Example: 01100100011**

Subtract Op-A, Op-B	**<3 Bit Instruction><4 bit Address to Register File A><4 bit Address to Register File B> Example: 10000100011**
Bitwise-OR Op-A, Op-B	**<3 Bit Instruction><4 bit Address to Register File A><4 bit Address to Register File B> Example: 10100100011**
Bitwise-AND Op-A, Op-B	**<3 Bit Instruction><4 bit Address to Register File A><4 bit Address to Register File B> Example: 11000100011**
Shift Left Op-A by Op-B times	**<3 Bit Instruction><4 bit Address to Register File A><4 bit Address to Register File B> Example: 11100100011**

C. The final demo should be demonstrated on any FPGA board available in the lab.

Marks Distribution

	Description	**Marks**
1.	**Correct Simulation.**	**40 Marks (You will get zero marks in this section if a snapshot of testbench is not provided in the word document)**
2.	**Pipelined Datapath Implementation**	**10 Marks**
3.	**FPGA Demo**	**50 Marks**

A WORD file with following:

a) Your final block level diagram showing all the interfacing signals.
b) A snapshot of testbench waveform for first 32 cycles.

Happy Learning ☺