Department of Computer Engineering
Faculty of Engineering, University of Peradeniya

CO221 : Digital Design
Project Specification for Phase 2

Deadline for Verilog design: 25th April
Deadline for FPGA implementation: 2nd May

Phase 2 of the project is the Verilog implementation and FPGA execution of the simple ALU.

Important!

There are some differences in the operation assignment, as it was decided that you all groups should implement one operation using sequential logic. Accordingly, the operation assignment is given in Table 1 at the end of the document. Changed operations are highlighted. All groups should implement the operation given as **Operation 1 using sequential logic.**

Operation Codes (Opcodes) of all the operations are given in Table 2. All groups must select the Opcodes corresponding the two operations assigned to them same as before.

As the first step, you are required to develop the Verilog design and verify its functionality by simulating it. As the next step, the Verilog design must be executed on an FPGA.

Verilog design

Verilog design should be done using gate-level modelling. That is, the whole system must be implemented using logic gates.

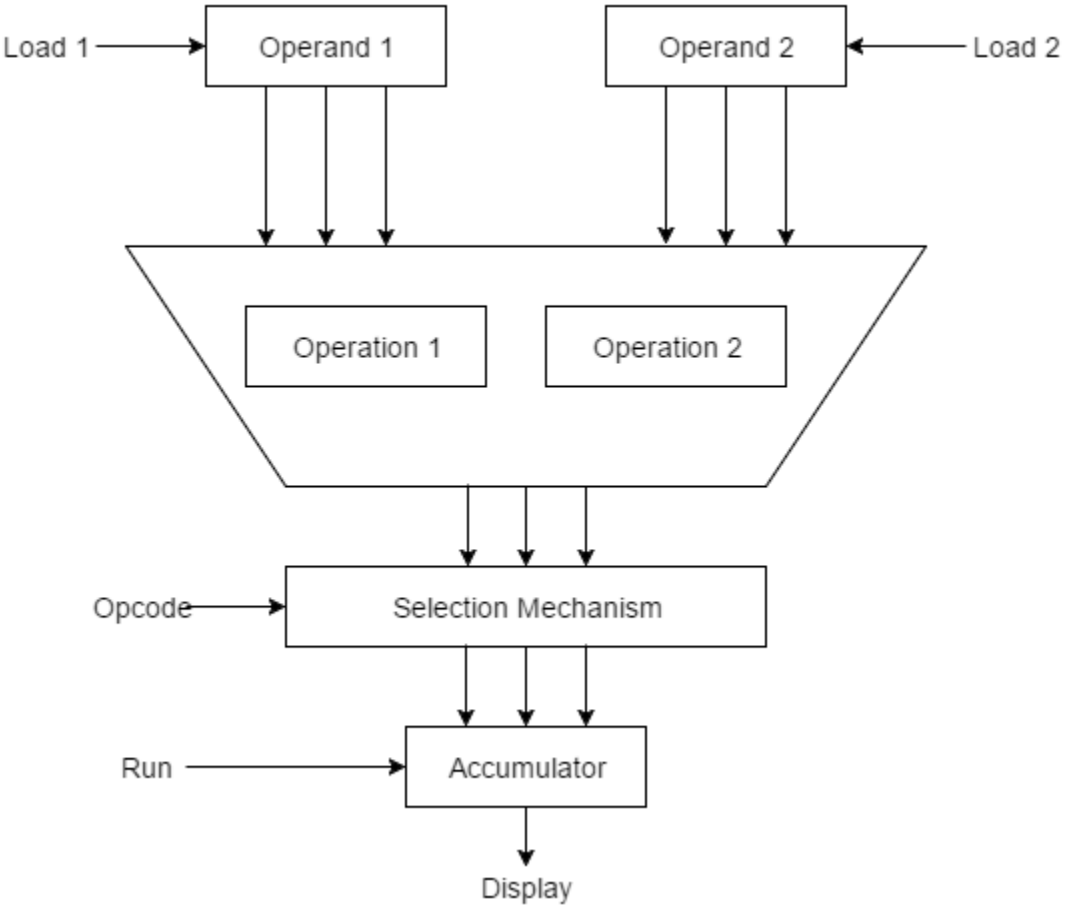
Requirements

- There is no limitation on the number of design modules that the implementation may contain. But, it is required that the two operations of your ALU be implemented as two separate modules.

- The file **alutestbed.v** contains the **testbed** required to test your implementation. You are required to implement the module called **ALU** according to the **testbed**. This is the top-most design module of your implementation.

- As you are aware by now, register is a sequential component which can be implemented using flip-flops. You are required to implement the register module for your design using positive-edge triggered D-flip-flops. In each place where you are required to use a register, you must instantiate the module you created.

The block diagram of the design is given in figure below.



The top most module named **ALU** must take *Operand 1*, *Operand 2*, *Load 1*, *Load 2*, *Operation Code (Opcode)* and *Run* signals as inputs.

When *Load 1* and *Load 2* are set to high, values given as *Operand 1* and *Operand 2* must be loaded to two registers. Note that, the values of *Operand 1* and *Operand 2* must change only when *Load 1* and *Load 2* signals are high.

Then, based on the *Opcode*, ALU must calculate the corresponding operation. Operation selection mechanism can be any mechanism of your choice.

Finally, when ***Run*** signal is high, calculated value should be put to the ***Accumulator***, which will be subsequently displayed via the stimulus module. ***Accumulator*** is yet another register which stores the result of the ALU until it is stored in the memory of the CPU.

Note that, once the output is displayed it should remain unchanged even when input operands and opcode are changed. Displayed value should change only when *Run* signal is set to high.

Note: Verilog implementation must be submitted as a single .v file to the submission link in FEeLS on or before the deadline. All files should be renamed as GroupYXX_Phase2.v, where Y is either A or B according to your group and XX is your 2 digit group number.

One submission from a group is sufficient.

Execution on the FPGA

Final step of Phase 2 is to execute the Verilog design on actual hardware. The hardware you will be using is an FPGA.

FPGA will allow you to give inputs to the system using built-in switches and view the outputs via built-in LEDs.

You are not required to do any modifications to the Verilog implementation.

Note: Evaluation of the FPGA implementation will be done as a viva session.

Notes

- It is strongly advised to start the project early rather than waiting till the deadline, as you may face problems on the way.
- For anything that is not explicitly stated, you may do reasonable assumptions. But, you should be able to explain them during the viva session.
- If there are any ambiguities, you are welcome to use the discussion forum in FEeLS.
- Plagiarism of any kind will result in zero marks for the phase.

Table 1: Allocation of operations to groups

Group	Operation 1	Operation 2
A1	One bit Shift left	Addition
A2	One bit Shift right	Bitwise XOR
A3	Multiplication	Bitwise NAND
A4	Shift Left	Bitwise NOR
A5	Shift Right	Bitwise AND
A6	One bit Shift right	Addition

A7	One bit Shift left	Bitwise XOR
A8	Multiplication	Bitwise NOR
A9	Shift Left	Bitwise AND
A10	Shift Right	Bitwise OR
B1	One bit Shift left	Addition
B2	One bit Shift right	Bitwise XOR
B3	Multiplication	Bitwise AND
B4	Shift Left	Bitwise OR
B5	Shift Right	Bitwise NAND
B6	One bit Shift right	Addition
B7	One bit Shift left	Bitwise XOR
B8	Multiplication	Bitwise OR
B9	Shift Left	Bitwise NAND
B10	Shift Right	Bitwise NOR

Table 2: Operation codes

Operation	Operation code
Addition	0000
Bitwise XOR	0001
Multiplication	0010
Shift Left	0011
Shift Right	0100
Bitwise AND	0101
Bitwise OR	0110
Bitwise NAND	0111
Bitwise NOR	1000
One bit shift left	1001
One bit shift right	1010

- One bit shift left and one bit shift right will shift Operand 1 by one bit to the left or to the right respectively.