EECE 3324
Summer 2022
Assignment 3

Add the pipelined ALU from Assignment 2 to the register file shown in Figure 1 below. Your final design needs to look like the one shown in Figure 3. Your final design needs to run at a 20 MHz clock rate (a full clock cycle is 50 nsec. and a half cycle is 25 nsec.).

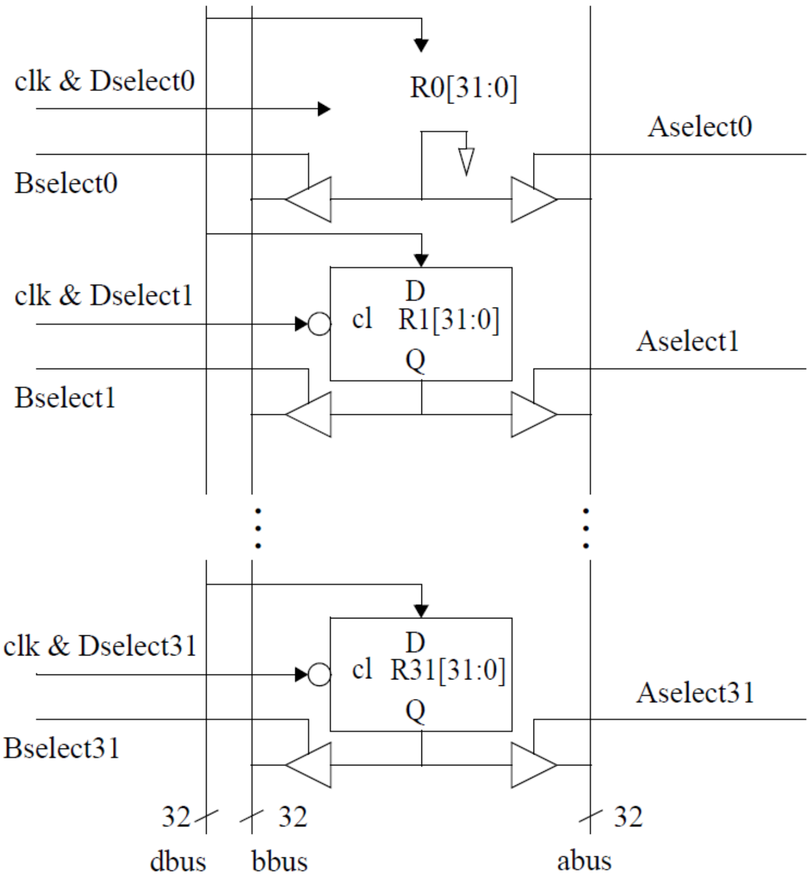

Figure 1. 32x32 Register File

A real register file is usually designed with latches as discussed in the class, but we will use falling (negative) edge sensitive flip-flops to simplify the timing.

There are no flip-flops for R0 since the R0 outputs are hardwired to ground to produce a constant logic 0 in the register as required by the MIPS instruction set. The circle on the cl inputs of the 32-bit registers indicates they are negative edge sensitive. The buffers on the abus and bbus are 32-bit parallel tri-state buffers. A 32-bit tri-state buffer can be described behaviorally as follows.

```
assign bus = select ? Q : 32’bz ;
```

Your design should have the following inputs and outputs for the 32x32 register file.

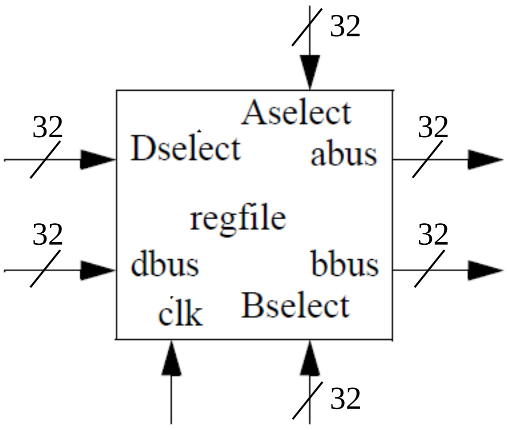

Figure 2. 32x32 Register File Top Module

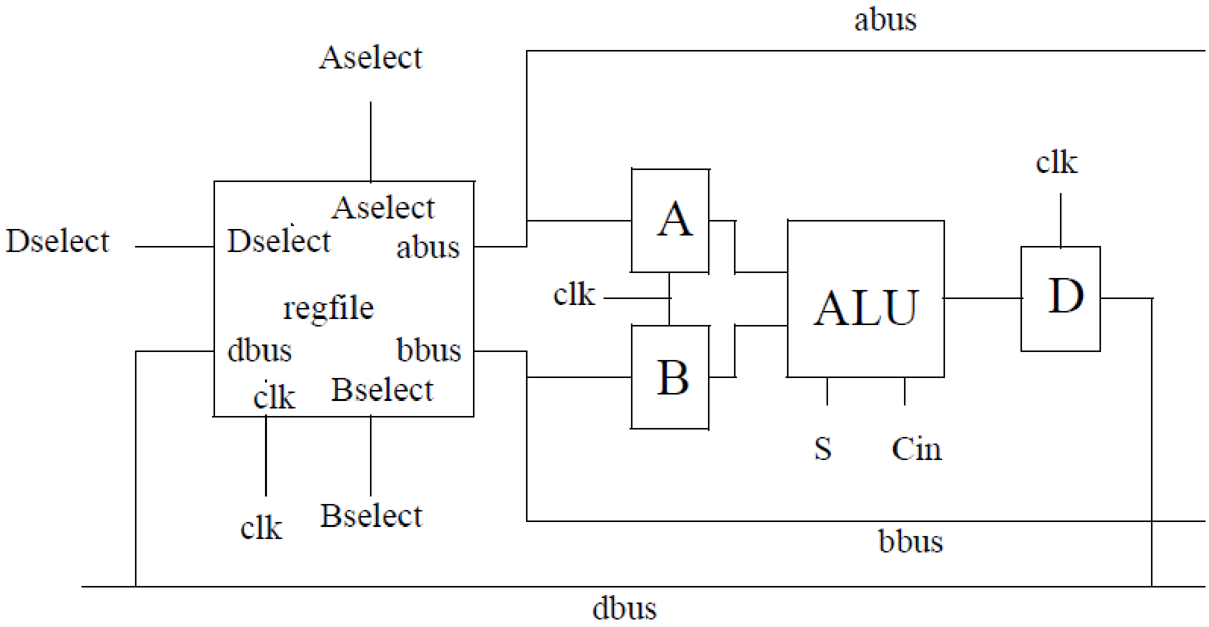

Figure 3. Register ALU overall design

The abus/bbus is driven by the register file with the contents of the register selected by the Aselect/Bselect inputs. The register file reads data off the dbus bus and writes it in the register selected by the Dselect inputs. Remember that the register file has been made with falling edge sensitive flip-flops, whereas the A, B, and D pipeline registers are made with rising edge sensitive flip-flops. This means that the data that the D pipeline register puts on the dbus at the rising clock edge gets written during the first half of the clock period into the register file at the falling clock edge, and the new data from the register file is read on the abus and bbus during the second half of the clock period and gets into the A and B pipeline registers at the next rising clock edge. This write first, then read strategy is wrong for a single cycle implementation, but just right for a pipelined implementation.

Your CPU design should have the terminals shown in Figure 4.

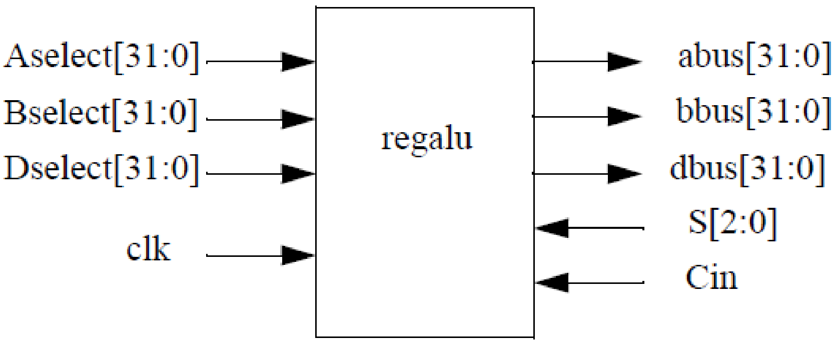

Figure 4. CPU Terminals

Before you start designing the hardware as shown in Figure 3 using Verilog, fill out the table below and find the value of the wires and output ports at each clock cycle. The purpose of filling out the table is to check your basic understanding of the hardware. If you believe the output is undefined, then use Xs to denote it, for example 32’hxxxxxxxx.

You must use a software such as Microsoft Word or Microsoft Excel to fill out the table. Using your own handwriting will not earn you any points.

Submit this table as part of your final submission to Canvas.

Clock Status	Asel	Bsel	Dsel	abus	bbus	ALUinputA	ALUinputB	ALUoutput	S	Cin	dbus
clk = 0	32'h00000001	32'h00000001	32'h00000001						3'b001	0	
↑	32'h00000001	32'h00000001	32'h00000001						3'b001	0	
clk = 1	32'h00000001	32'h00000001	32'h00000001						3'b001	0	
↓	32'h00000001	32'h00000001	32'h00000001						3'b001	0	
clk = 0	32'h00000001	32'h00000001	32'h00000001						3'b001	0	
↑	32'h00000001	32'h00000001	32'h00000001						3'b001	0	
clk = 1	32'h00000001	32'h00000001	32'h00000002						3'b110	0	
↓	32'h00000001	32'h00000001	32'h00000002						3'b110	0	
clk = 0	32'h00000001	32'h00000002	32'h00000001						3'b001	0	
↑	32'h00000001	32'h00000002	32'h00000001						3'b001	0	
clk = 1	32'h00000001	32'h00000001	32'h00000004						3'b100	0	
↓	32'h00000001	32'h00000001	32'h00000004						3'b100	0	
clk = 0	32'h00000004	32'h00000002	32'h00000001						3'b100	0	
↑	32'h00000004	32'h00000002	32'h00000001						3'b100	0	
clk = 1	32'h00000001	32'h00000001	32'h00000008						3'b000	0	
↓	32'h00000001	32'h00000001	32'h00000008						3'b000	0	

Table 1. Register ALU Table

A Verilog testbench file to provide inputs to your design will be provided by your course instructor. A similar file may be used to grade your design. Your highest-level Verilog file needs to be named **regalu.v** to be compatible with the testbench file.

<u>**Create your own the testbench file**</u> that fully tests your design with at least 32 test scenarios. You must provide comments on your testbench file so that others can follow your work. You will post your testbench to Assignment 3 Discussion on Canvas. Your classmates will review your testbench and provide feedback about it. Your classmates will also grade/rate your testbench file.

Please submit all your Verilog files including your new testbench file to Assignment 3 Link on Canvas. For your initial post to Assignment 3 Discussion, you only need to post your new testbench file.

Rules of Engagement:

1. You are only allowed to complete this assignment individually/as a group as assigned by the course instructor.

2. You are NOT allowed to work/collaborate/show your work to other people.

3. Your work must be your own. Should you be found guilty of using someone else’s work (either full or partly), an “F” will be assigned as your overall course grade for the semester.

4. Read, understand, and follow NU Honor Code.

5. After you have completed your work, you must submit all your files to Canvas. Once you have submitted all your code to Canvas, you then need to be live interviewed by your TA/course instructor to verify all your work for this assignment using Xilinx Vivado. Failure to demonstrate your work to your grader or course instructor will result in a zero as the final grade of this assignment.

6. All work must be done using Xilinx Vivado software. Failure to do this will result in a zero as the final grade of this assignment.

7. **The use of * (star symbol) in Verilog is strictly prohibited**. Using a * (star symbol) will result in a zero as the final grade of this assignment.

Lab Submission:

You must show a live **working demo** to your course instructor or grader to receive a grade. The demo must be completed individually/as a group as instructed by the course instructor. Questions will be asked during the demo. Points will be deducted from your Lab grade should you fail to answer any questions. Comments such as “I cannot remember why I did that” will not help you. Come prepared.

Upload all the Verilog files used and the improved test bench file to Canvas. Make sure that your Verilog files (including the test bench file) are heavily commented as it is part of the grading. Your work will also be judged by the complexity of your modified test bench file. The more complex and rigorous your test bench file in testing different scenarios, the more points you earn.

Do not zip all your files together.

An automatic zero will be assigned for anyone who fails to show a live working demo to your course instructor or grader.