EECE 3324
Summer 2022
Assignment 4

For this assignment, you will design a controller for a simplified 32-bit pipelined CPU (which will be designed in Assignment 5) as shown in Fig. 1.

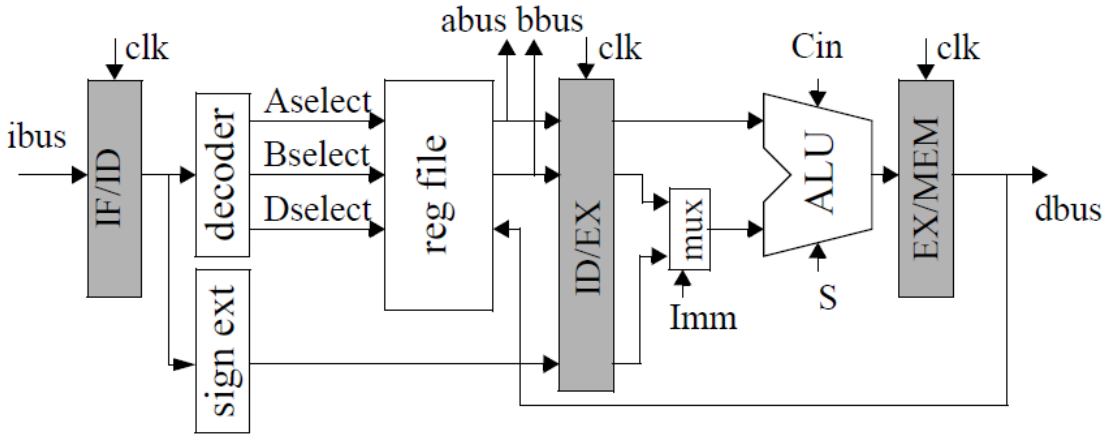

Figure 1. Pipeline Block Diagram

so that the following subset of instructions is implemented.

Code	Func	Name	Format
000011		ADDI	I
000010		SUBI	I
000001		XORI	I
001111		ANDI	I
001100		ORI	I
000000	000011	ADD	R
000000	000010	SUB	R
000000	000001	XOR	R
000000	000111	AND	R
000000	000100	OR	R

Table 1. Subset of instruction codes to be implemented.

The instruction formats follow MIPS except that only the I and R formats are implemented and the logic operations are slightly different. All other op-codes should be treated as NOPs.

The controller outputs are the control point inputs shown in Fig. 1. The control points are the Aselect, Bselect and Dselect inputs to the register file, the Imm input to the MUX and the S and Cin inputs of the ALU as shown in Fig. 1. The contents of the instruction register must be decoded to provide all of the control inputs. Binary decoders can be used to produce the Aselect, Bselect and Dselect inputs for the register file from the operand fields in the instruction register. Write the logic equations for the opcode decoder logic so that the control lines are correct for each instruction. It is probably a good idea to assign a default inactive value for each control line since you do not want to leave control lines undefined at any time. Make sure that the control lines go through the appropriate pipeline registers before connecting to the control points. The controller design should be similar to Fig. 2.

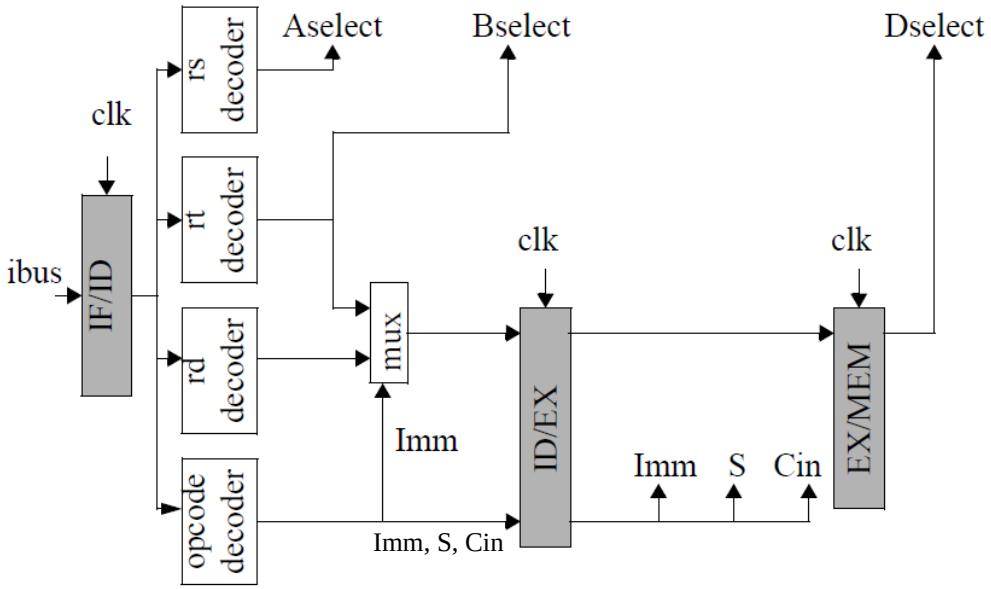

Figure 2. Controller Block Diagram

The mux is needed because the destination register is in the rd field for R format, but in the rt field for I format. The mux control is the same Imm as in the data path.

To facilitate testing of your design, the IF/ID register inputs will be connected to the instruction bus which will be made available as an external terminal. Your design should have the terminals shown in Figure 3.

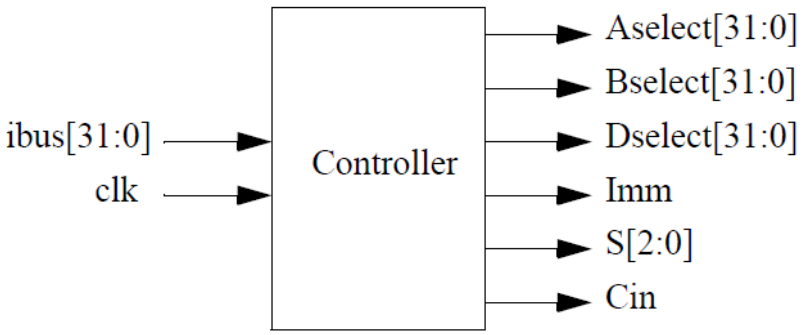

Figure 3. Controller Terminals

Before you start designing the hardware as shown in Figure 2 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	ibus	Asel	Bsel	Dsel	Imm	S	Cin
clk = 0	32'b00000000000000000110100000000010						
↑	32'b00000000000000000110100000000010						
clk = 1	32'b00000000000000000110100000000010						
clk = 0	32'b00000000000000000110100000000010						
↑	32'b00000000000000000110100000000010						
clk = 1	32'b00000000000000000110100000000010						
clk = 0	32'b00001100000000001111111111111111						
↑	32'b00001100000000001111111111111111						
clk = 1	32'b00001100000000001111111111111111						
clk = 0	32'b00001100001111101010111111000000						
↑	32'b00001100001111101010111111000000						
clk = 1	32'b00001100001111101010111111000000						
clk = 0	32'b00000000000000000000000000000010						
↑	32'b00000000000000000000000000000010						
clk = 1	32'b00000000000000000000000000000010						

Table 1. Controller Table

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

Create your own the testbench file 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 4 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 4 Link on Canvas. For your initial post to Assignment 4 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.