Faculty of Engineering and Technology
Department of Electrical and Computer Engineering
ENCS 211 Digital Electronics and Computer Organization Lab
Second Semester 2022-2023
Practical Final Exam Question Sets

1. Introduction:

Each student will need to implement the problem by him/herself and show understanding of what is being done, isolate errors and introduce correction when needed. Each student will be given a set of problem that includes problems from **Area1 (Basic Digital Design) including breadboarding and Area2 (Verilog/Quartus).**

The result needs to be demonstrated to the instructor/teaching assistant and the student will be evaluated on his/her ability to understand, change and diagnose errors in the design.

The task you get in the exam may be a modification of one of the variants so please be ready.

Please bring in your laptop with quartus installed and functioning.

2. Evaluation Points:

Students will be evaluated on the following items (out of 100):

1. Clear sketch of the circuit to be implemented (**15%**).
2. Using the correct equipment (kit) and selecting inputs/outputs (**15%**).
3. Implementation, testing and locating errors and fixing them (**30%**).
4. Ability to change the circuit to achieve a related functionality (**30%**).
5. Time management (**10%**)

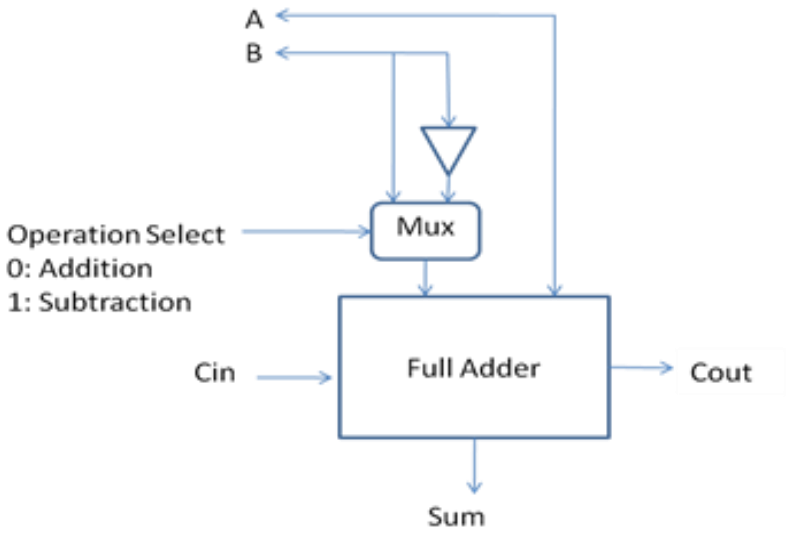
Good luck

Problem Set 0:

a) Use JK Flip-Flops to construct a 2-bit counter. Sketch your design then implement it.

Sketch the changes needed to make the counter 3 bit.

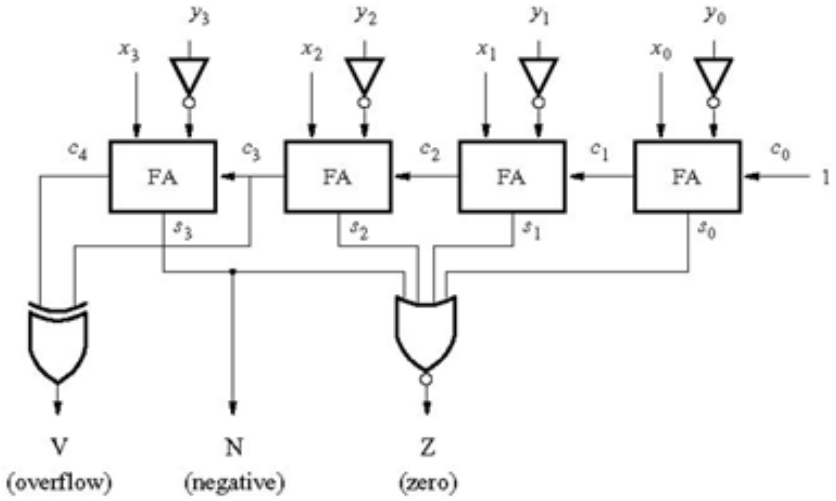
b) Use Verilog HDL to implement a 2-to-1 MUX. Use Verilog HDL to implement a Full Adder. Create schematic symbols for both the MUX and the Full adder, then connect them as shown in the figure. Run a meaningful simulation for this circuit.



Problem Set 1:

a) **Design** a 2-to-4 decoder using logic gates. Then use a chip based 3-to-8 decoder to **implement** a basic function, say a full adder (3 inputs). Sketch the design before the implementation and show the designs to your instructor/TA.

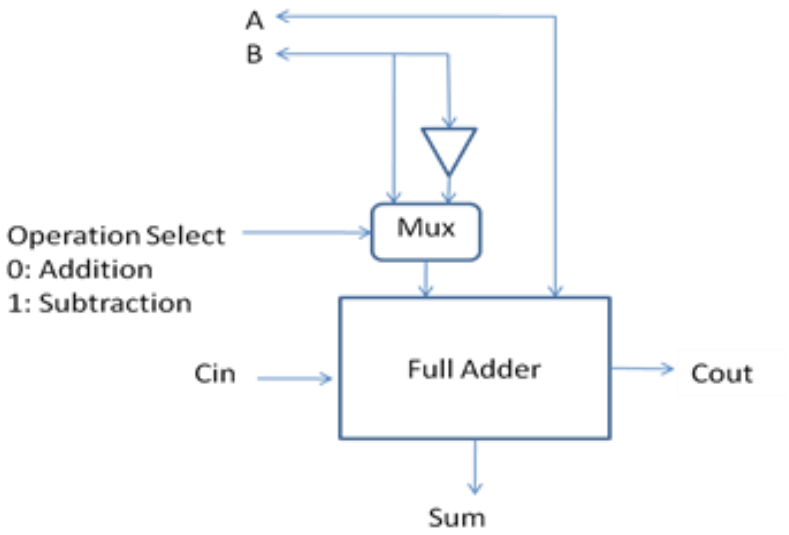
b) Create schematic symbols as shown in the figure, Use: $Y_3Y_2Y_1Y_0 = 0111$. What does this circuit do? Use Verilog to implement the circuit and Run a meaningful simulation for this circuit.



Problem Set 2:

a) **Design** a basic 2-to-1 MUX using gates. Then use a chip based 4-to-1 MUX to **implement** a 3-input basic function, say a full adder, or something similar.

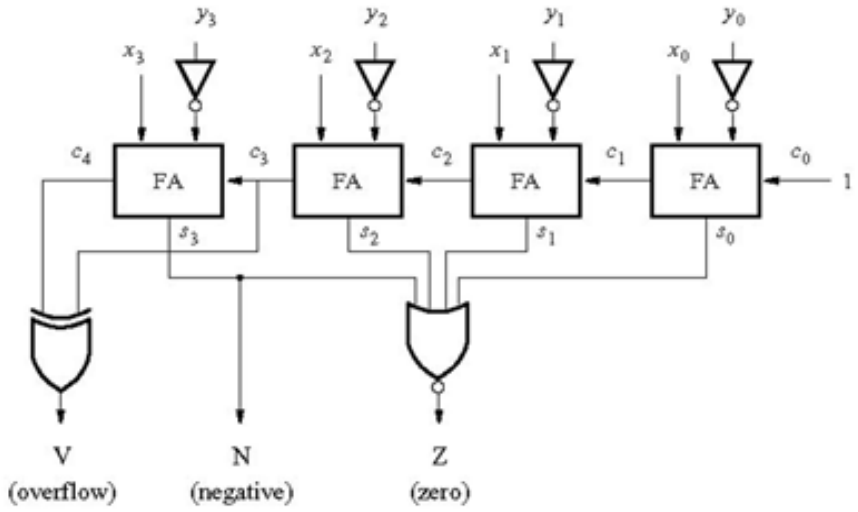
b) Use Verilog HDL to implement a 2-to-1 MUX. Use Verilog HDL to implement a Full Adder. Create schematic symbols for both the MUX and the Full adder, then connect them as shown in the figure. Run a meaningful simulation for this circuit.



Problem Set 3:

a) **Design** a basic 2-bit parity checker using gates. Then use a chip based (8-bit) parity checker and connect it so that it checks parity for 5 inputs only. Sketch the design before the implementation and show the designs to your instructor/TA.

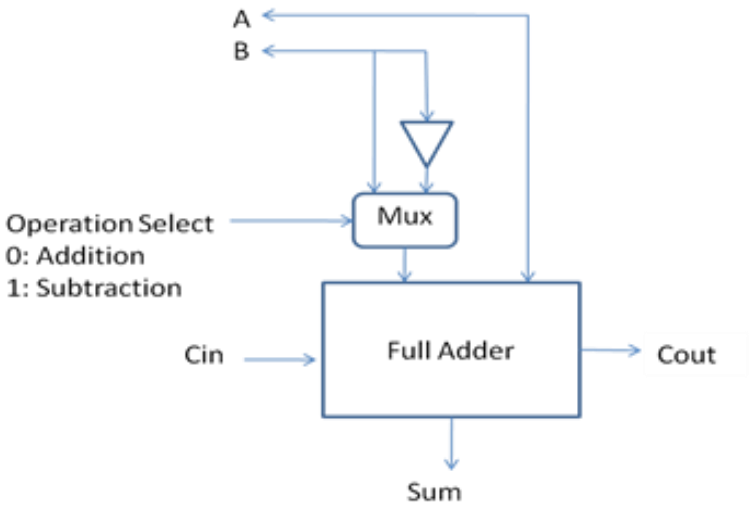
b) Create schematic symbols as shown in the figure, Use : $Y_3Y_2Y_1Y_0 = 0111$. What does this circuit do? Use Verilog to implement the circuit and Run a meaningful simulation for this circuit.



Problem Set 4:

a) **Design and Build** a basic 2-bit full adder from gates. Use the result and 2 shift registers to design a serial adder. Sketch the design before the implementation and show the design to your instructor/TA.

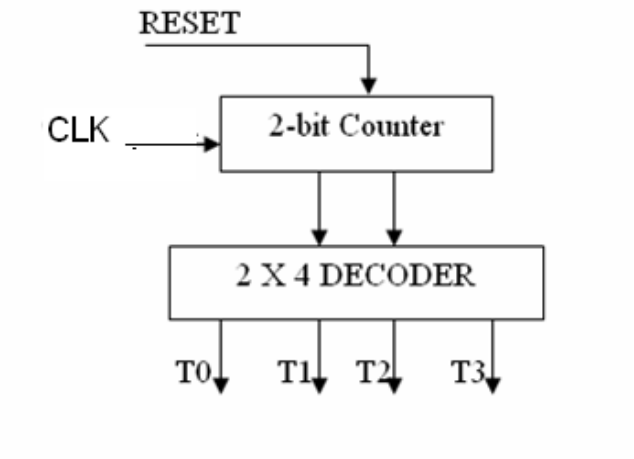
b) Use Verilog HDL to implement a 2-to-1 MUX. Use Verilog HDL to implement a Full Adder. Create schematic symbols for both the MUX and the Full adder, then connect them as shown in the figure. Run a meaningful simulation for this circuit.



Problem Set 5:

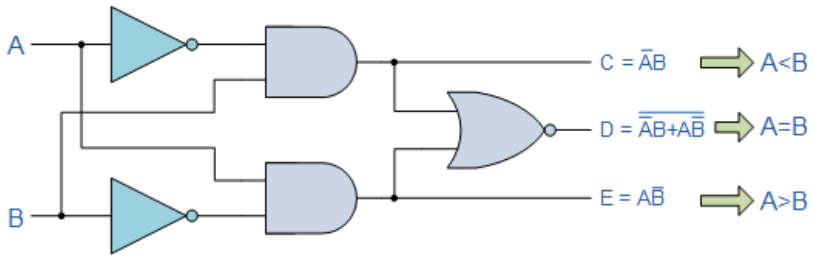
a) Build a basic 2-bit parity checker from gates. Use the result and a shift registers to design and implement a serial parity checker for 4 bits. Sketch the design before the implementation and show the designs to your instructor/TA.

b) Use Verilog HDL to implement a 2-bit counter with direct reset input (RESET). Use Verilog HDL to implement a 2-to-4 Decoder. Create schematic symbols for both the counter and the decoder, then connect them as shown in figure below. Run a meaningful simulation for this circuit.

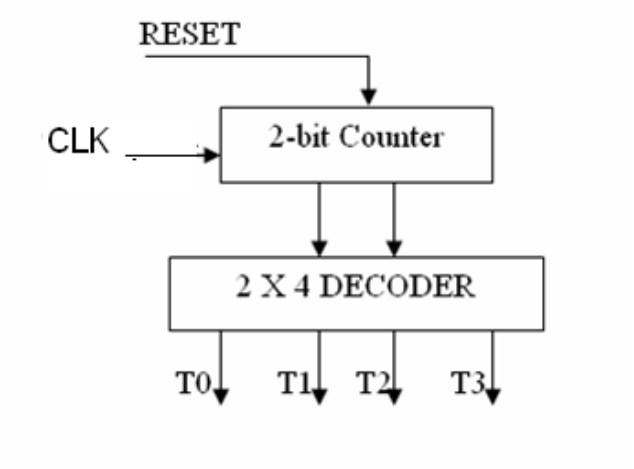


Problem Set 6:

a) Use the breadboard to implement the circuit below (2 inputs, 3 outputs). We will give you the chip and its datasheet and you have to do the rest. The circuit could be different for different students/sections. Sketch the design before the implementation and show the designs to your instructor/TA.

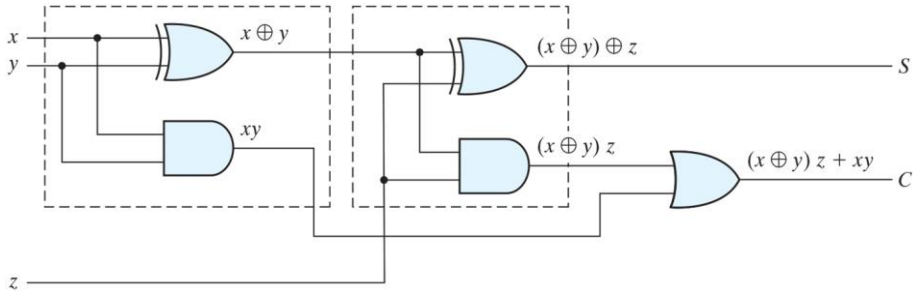


b) Use Verilog HDL to implement a 2-bit counter with direct reset input (RESET). Use Verilog HDL to implement a 2-to-4 Decoder. Create schematic symbols for both the counter and the decoder, then connect them as shown in figure below. Run a meaningful simulation for this circuit.

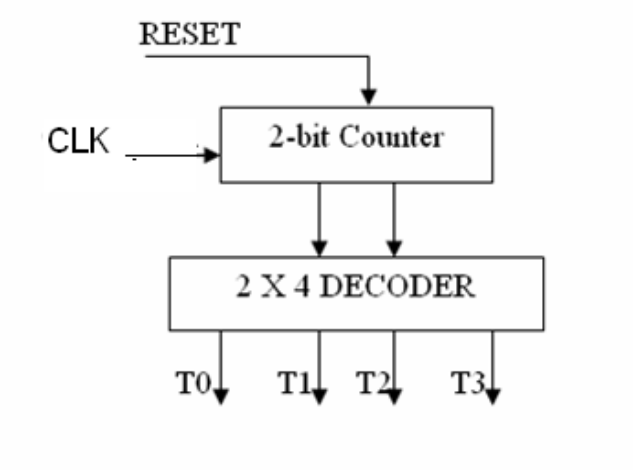


Problem Set 7:

a) Use the breadboard to implement the circuit below (full adder). We will give you the chip and its data and you have to do the rest. The circuit could be different for different students/sections.



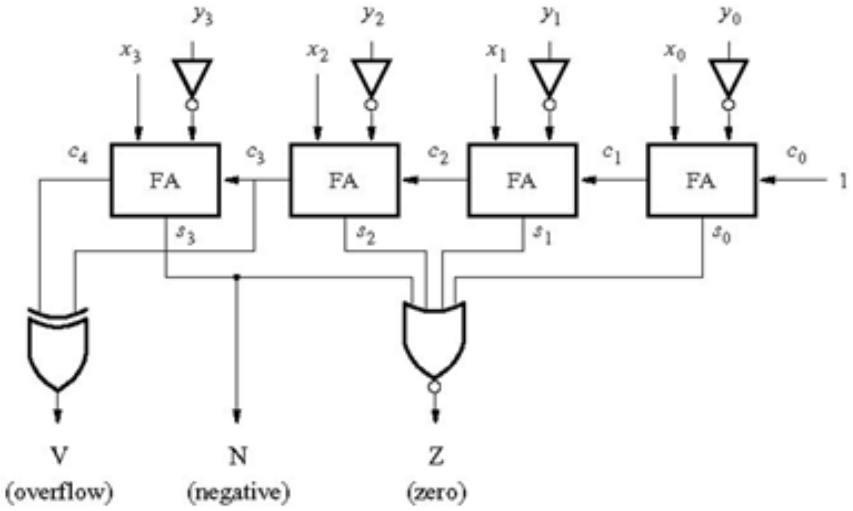
b) Use Verilog HDL to implement a 2-bit counter with direct reset input (RESET). Use Verilog HDL to implement a 2-to-4 Decoder. Create schematic symbols for both the counter and the decoder, then connect them as shown in figure below. Run a meaningful simulation for this circuit.



Problem Set 8:

a) Given F(A, B, C) = ∑m(0,1,6,7), construct the truth table for the F. Then draw the logic circuit and implement F in the breadboard using basic gates. You have to define the ICs needed based on your design and the given datasheets.

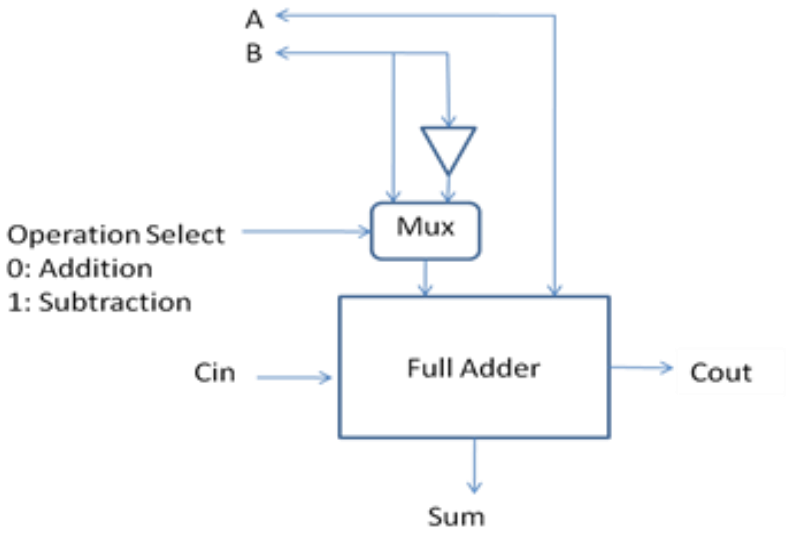
b) Create schematic symbols as shown in the figure, Use : $Y_3Y_2Y_1Y_0$ = 0111. What does this circuit do? Use Verilog to implement the circuit and Run a meaningful simulation for this circuit.



Problem Set 9:

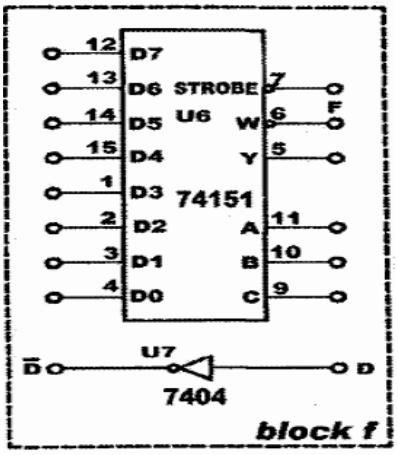
a) Connect the BCD counter and show its operation on a 7-segment display. Modify the circuit so that it counts only to 3 then back to 0.

b) Use Verilog HDL to implement a 2-to-1 MUX. Use Verilog HDL to implement a Full Adder. Create schematic symbols for both the MUX and the Full adder, then connect them as shown in the figure. Run a meaningful simulation for this circuit.



Problem Set 10:

a) Construct a 4-bit **odd** parity generator using an 8x1 multiplexer and any basic gates if needed. Connect the four input bits to switches and connect the output parity bit to Led. Show that your designs works correctly.



8-to-1 MUX (KL-26004 block f)

b) Create schematic symbols as shown in the figure, Use : $Y_3Y_2Y_1Y_0 = 0111$. What does this circuit do? Use Verilog to implement the circuit and Run a meaningful simulation for this circuit.