CME341 - Midterm #1
Material up to end of Assignment 5

```
Date: Wednesday, October 13, 2020, 5:30PM
Time = 1.00 hours
```

Instructions

Exam-Dependent Seed Setup

Before starting the exam modify the second "initial" procedure in the testbench so that `exam_dependent_seed` is assigned to the last two digits in your student number, treated as a DECIMAL (base-10) number. For example, if your student number is 11263654, you should set the `exam_dependent_seed` to `8'd54`. You will use this same seed for all of the questions on the exam.

Academic Honesty

In answering the following questions, you may refer to any of the materials on the course website (notes, slides, lectures, etc.). You MAY NOT communicate, discuss, or otherwise share any information or data with anyone during the exam or access any websites aside from the CME 341 canvas page. Use of personal laptops, tablets, cell phones, etc. is strictly prohibited. By submitting your exam, you certify that you have not discussed the questions with anyone or used any other resources (aside from those on the course website) to assist with your solution.

The consequences of violating this policy can be severe. The instructors and the college may use various methods including in-person observation and electronic monitoring of network activity during the exam to detect academic dishonesty. Any detected cases of academic dishonesty will be taken very seriously and referred to the appropriate college or university disciplinary committees.

Exam Submission

There are a total of five questions, all having equal value. Please indicate your answer to each question on the provided answer slip. To be clear, the answer to be provided is the 16-bit hexadecimal value of `accumulator_output` at the moment when `counter_full_bar` is low for your exam-dependent seed.

After you have completed the exam and handed in the answer sheet, you must archive and upload to Canvas the verilog source files used to answer each question. This must be completed, but it is not part of the timed exam. The instructors may refer to to this code in the event of a marking discrepancy or dispute.

Good luck!

(2)

1. Re-design the student circuit to implement an 8-bit synchronous counter which is connected to `cct_output`. The operational specifications of the counter are as follows:

   - When the clear input is high, the counter should be asynchronously reset to `8'd0`.
   - When `cct_input` is equal to `8'd31`, the counter should be synchronously loaded with `8'd100` on the positive edge of the clock.
   - Otherwise, the counter should synchronously increment on each positive edge of the clock.

   The accumulator output, i.e. `accumulator_output`, when `counter_full_bar` is low for `seed = 8'HAA` should be `16'H4669`.
   Once `accumulator_output` is correct for `seed = 8'HAA` report it for the seed specified in the instructions.

(2)

2. Re-design prototype `student_circuit` to describe a combinational logic circuit that detects whether the value of the input (`cct_input`) is a two-digit number when viewed in decimal (ie. whether it is between 10 and 99, inclusive). When the input is a two-digit number, the output (`cct_output`) should be equal to the input. When the input is not a two-digit number, the output should be equal to the input value XNOR'ed with the constant 8'HC6. The clear input should be ignored for this question.

   The accumulator output, i.e. `accumulator_output`, when `counter_full_bar` is low for `seed = 8'HAA` should be `16'H9696`.
   Once `accumulator_output` is correct for `seed = 8'HAA` report it for the seed specified in the instructions.

(2) 3. Re-design prototype `student_circuit` to describe a combinational logic circuit with the following behavior:

- If the most significant bit (bit [7]) of `cct_input` is high, `cct_output` should be four copies of bits [1:0] of the input. For example, if the input was `8'b1000_0010`, the output should be `8'b1010_1010`.
- If the most significant bit (bit [7] of `cct_input` is low, `cct_output` should be the one's complement (all bits inverted) of the input.
- `cct_output` should be 0 when the clear input is high. This should take priority over the two conditions listed above.

The accumulator output, i.e. `accumulator_output`, when `counter_full_bar` is low for `seed = 8'HAA` should be `16'H4567`.
Once `accumulator_output` is correct for `seed = 8'HAA` report it for the seed specified in the instructions.

(2) 4. Re-design prototype `student_circuit` to implement a synchronous circuit according to the following specifications:

- On each positive edge of the clock input, the circuit loads the value of `cct_input` into an 8-bit register called `delay`.
- On each negative edge of the clock input, `cct_output` is set to the result of a bitwise NAND operation between `delay` and `cct_input`.
- The clear input is used in an active-high asynchronous fashion to reset both `delay` and `cct_output` to 0.

The accumulator output, i.e. `accumulator_output`, when `counter_full_bar` is low for `seed = 8'HAA` should be `16'HEB46`.
Once `accumulator_output` is correct for `seed = 8'HAA` report it for the seed specified in the instructions.

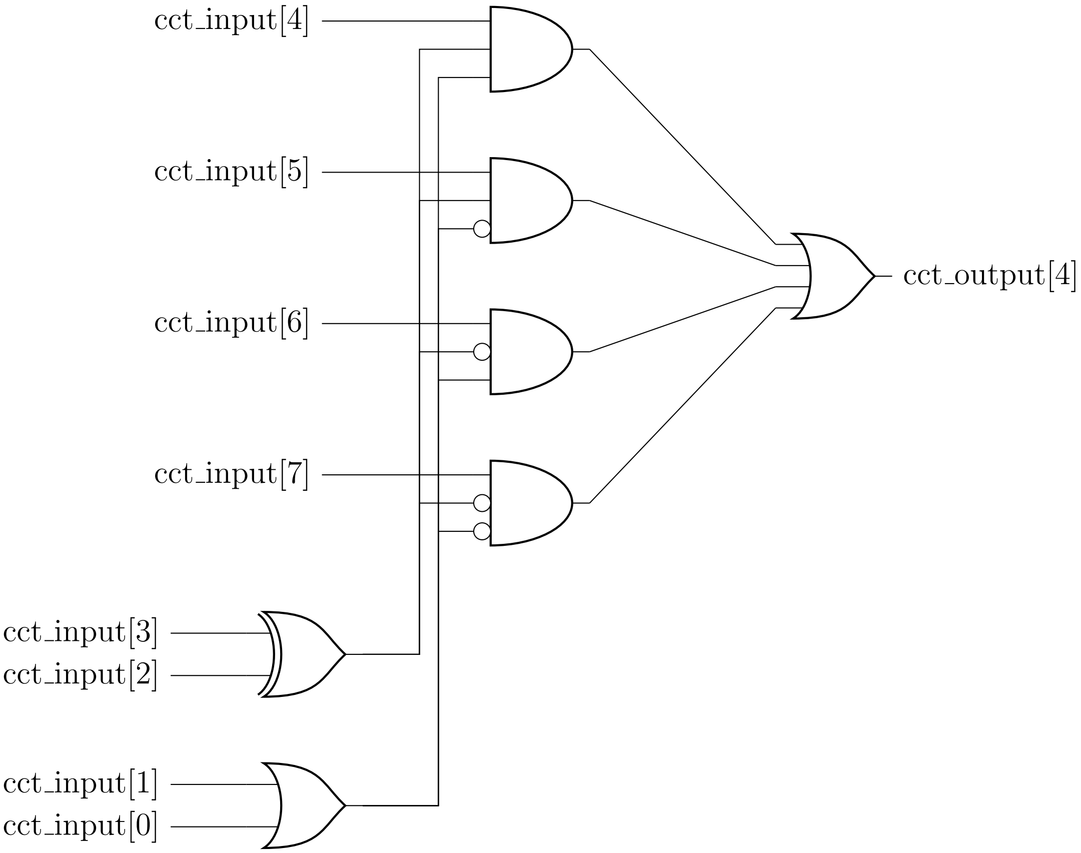

Figure 1: Circuit for Question 5

(2) 5. Re-design `student_circuit` to implement the circuit shown in Figure 1. Any bits of the circuit output vector (`cct_output`) that are not shown in the diagram should be tied low (connected to 0).

The accumulator output, i.e. `accumulator_output`, when `counter_full_bar` is low for `seed = 8'HAA` should be `16'HF37B`.
Once `accumulator_output` is correct for `seed = 8'HAA` report it for the seed specified in the instructions.