1-point Homework 4

Use the last 3 digits of your personal number (PN) to select which question you have to answer. Modulo divide your 3 digits with the number of questions, in this case, 2, plus 1. The resulting number is the question that you should answer.

!!! (PN mod 2) + 1 !!!

For example, if your PN ends in 730, then (730 mod 2) + 1 = 1, meaning you must answer question 1.

Make sure you comment on the top of each file you submit, your PN, name and the question you are answering.

!!! IMPORTANT !!!

If you answer the wrong question, your submission will be invalid.

We can call you to explain your solution. If you cannot explain your answer, the homework will be invalid!

KTH has a zero-tolerance policy against cheating.

https://www.kth.se/en/student/stod/studier/fusk-1.997287

If you have questions or need clarifications, you can ask in the discussion forum in Canvas.

QUESTIONS

1.1 QUESTION 1

Consider a digital circuit that implements a simple 4-bit binary unsigned adder. Design a set of immediate assertions to verify the correctness of the adder's output. Assume that the design also outputs the carry signal and that you have to input A and B that are corresponding to the results. The assertions should cover the following properties:

1. Addition Correctness:

   Check if the sum output is equal to the addition of the input operands.

2. Propagation of Carry:

   Verify that the carry-out from each bit is propagated to the next bit correctly.

```
module RCA (
  input  logic [3:0] A  ,
  input  logic [3:0] B  ,
  output logic [4:0] carry_chain,
  output logic [4:0] Sum);
```

1.2 QUESTION 2

Given the code below, explain what conditions we want to cover using the cross coverage.

```
logic [3:0] signal1;
logic [3:0] signal2;
covergroup my_covergroup;
cross signal1, signal2 {
    bins both_low = {binsof S1 intersect [0:9] && binsof S2 intersect [0:9]};
    bins signal1_low_signal2_high = {!binsof S1 intersect [0:9] && binsof S2 intersect [0:9]};
    bins signal1_high_signal2_low = {binsof S1 intersect [0:9] && !binsof S2 intersect [0:9]};
    bins both_high = {!binsof S1 intersect [0:9] && !binsof S2 intersect [0:9]};
}
endgroup
```