# Pass/Fail Homework 2

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, 4, plus 1. The resulting number is the question that you should answer.

**!!! (PN mod 4) + 1 !!!**

For example, if your PN ends in 730, then (730 mod 4) + 1 = 3, meaning you must answer question 2.

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**

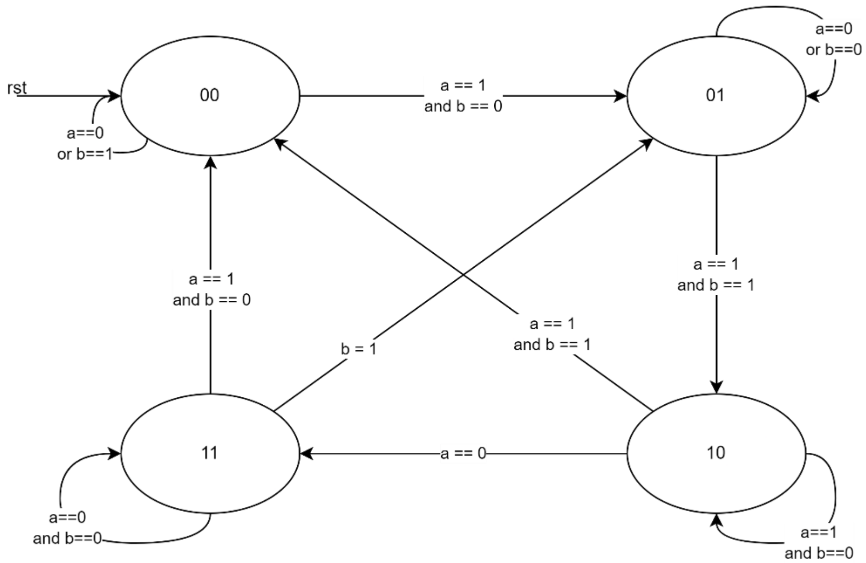
**If you have questions or need clarifications, you can ask in the discussion forum in Canvas.**

# Questions

## 1.1 Question 1

Model an FSM in HDL that implements the following state transition diagram. The output of the state machine should be as follows.

- State 00: c=1, d=1
- State 01: c=1, d=0
- State 10: c=0, d=0
- State 11: c=0, d=1



| Name | Direction | Width | Description |
|---|---|---|---|
| clk | in | 1 | Clock signal |
| rstn | in | 1 | Active low reset |
| a,b | in | 1 | Inputs |
| c,d | out | 1 | Outputs |

## 1.2 Question 2

Given the following SystemVerilog code, derive the state transition diagram of the FSM.

```
module fsm (
  input logic A, B,
  input logic clk, // Clock input
  input logic reset, // Reset input
  output logic [1:0] state // Output representing the current state
);
  typedef enum logic [1:0] {
     S00 = 2'b00,
     S01 = 2'b01,
     S10 = 2'b10,
     S11 = 2'b11
   } fsm_states;
   fsm_states current_state, next_state;
  always_ff @(posedge clk or posedge reset) begin
    if (reset) begin
        current_state <= S00;
    end else begin
        current_state <= next_state;
    end
  end
  always_comb begin
     next_state = current_state;
    case(current_state)
        S00: begin
         if (A) begin
           if (B) begin
                next_state = S11;
           end else begin
                next_state = S01;
           end
         end else begin
           if (B) begin
                next_state = S10;
           end
         end
      end
        S01: begin
         if (!A) begin
           if (B) begin
                next_state = S10;
           end else begin
                next_state = S00;
           end
         end
      end
        S10: begin
         if (A) begin
           if (!B) begin
                next_state = S01;
           end else begin
                next_state = S11;
           end
         end
      end
        S11: begin
         if (!A) begin
           if (!B) begin
                next_state = S00;
           end else begin
                next_state = S10;
           end
         end
      end
    endcase
  end
  assign state = current_state;
endmodule
```

## 1.3 QUESTION 3

Design an FSM that can detect an input sequence of 5 consecutive zeros. Assume that a new input comes at every clock cycle. Model the FSM in HDL.

| Name | Direction | Width | Description |
|---|---|---|---|
| clk | in | 1 | Clock signal |
| rstn | in | 1 | Active low reset |
| input | in | 1 | Input |
| detected | out | 1 | 1 if 5 consecutive zeros are detected |

## 1.4 QUESTION 4

Design an FSM that can detect an input sequence of 5 consecutive ones. Assume that a new input comes at every clock cycle. Model the FSM in HDL.

| Name | Direction | Width | Description |
|---|---|---|---|
| clk | in | 1 | Clock signal |
| rstn | in | 1 | Active low reset |
| input | in | 1 | Input |
| detected | out | 1 | 1 if 5 consecutive ones are detected |