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

!!! (PN mod 7) + 1 !!!

For example, if your PN ends in 730, then (730 mod 7) + 1 = 3, meaning you must answer question 3.

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.

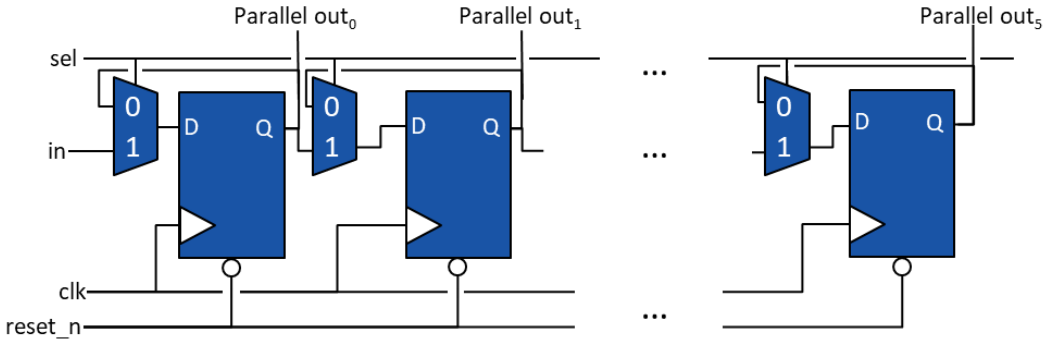
QUESTIONS

1.1 QUESTION 1

1. Model using HDL, a **6-bit shift register** that can operate in a **serial-in-parallel-out** (SIPO) fashion. The design should accept **a single-bit input**, a **clock**, a **reset**, a **select**, and a **6-bit output**. The select signal should control the loading of the SIPO, i.e. as long as the select signal is asserted ('1'), the design should load new bits serially in the register. When the select signal is de-asserted ('0'), the design should output in parallel all the bits that are stored inside the shift register. The design should use **D-flops as components**; their description is given here. The schematic of the design and the module declaration can also be found below.

```
module d_flop (
  input logic clk, rst_n, d,
  output logic q
);
  always_ff @(posedge clk) begin
    if (!rst_n) begin
      q <= 1'b0;
    end else begin
      q <= d;
    end
  end
endmodule
```

```
module SIPO (
  input logic clk, rst_n,
  input logic serial_in, select,
  output logic [5:0] parallel_out
);
  …
endmodule
```

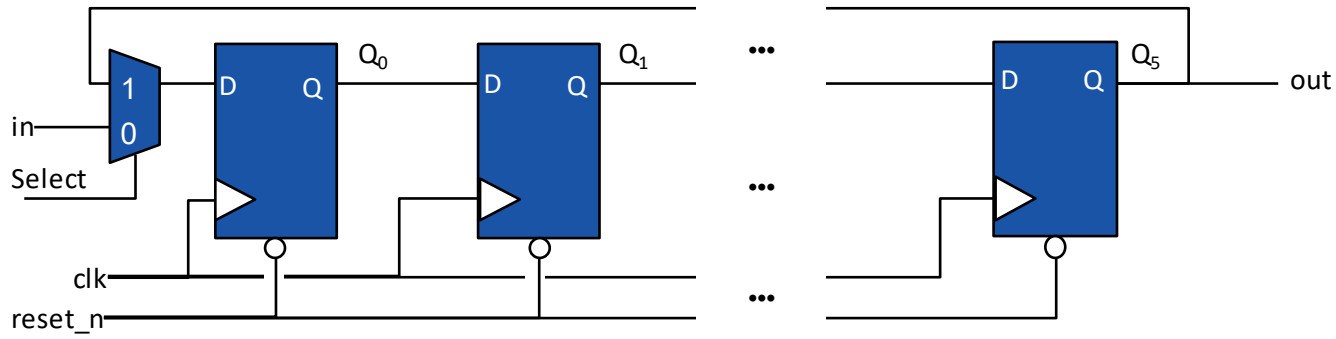


1.2 Question 2

Model using HDL, a **6-bit shift register** that can also be used as a rotator. The design should accept **a single-bit input**, a **clock**, a **reset**, a **select**, and a **single-bit output**. The select signal should control if the design should work as a shift register or a rotator, i.e. as long as the select signal is asserted ('1'), the design should not load new bits serially in the register but instead rotate. When the select signal is de-asserted ('0'), the design should input new bits and operate as a shift register. The design should use **D-flops as components**; their description is given here. The schematic of the design and the module declaration can also be found below.

```
module d_flop (
  input logic clk, rst_n, d,
  output logic q
);
  always_ff @(posedge clk) begin
    if (!rst_n) begin
      q <= 1'b0;
    end else begin
      q <= d;
    end
  end
endmodule
```

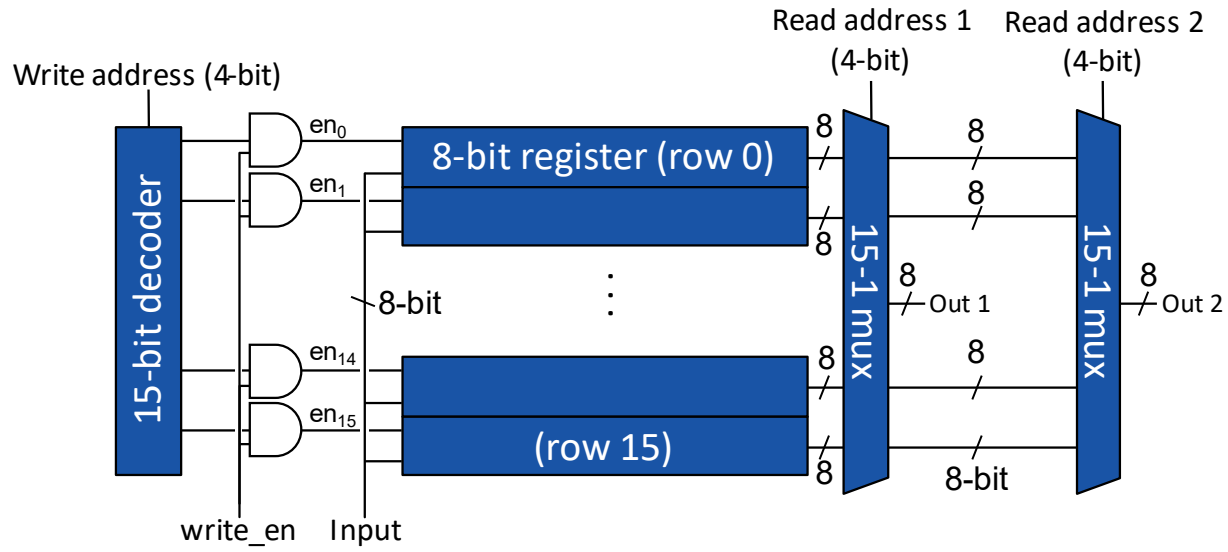
```
module Rotator (
  input logic clk, rst_n,
  input logic in, select,
  output logic out
);
  …
endmodule
```



1.3 Question 3

Model using HDL is a design that implements a register file with 8-bit words, and that can store 16 words (rows). The design should have one write port but two write ports. Each read-and-write port should have its own address. The write port should also have a write enable signal that will disable the enable signal to each register row so no new data would be registered in the row, even if a valid write address is given. The schematic and the module definition are given below.

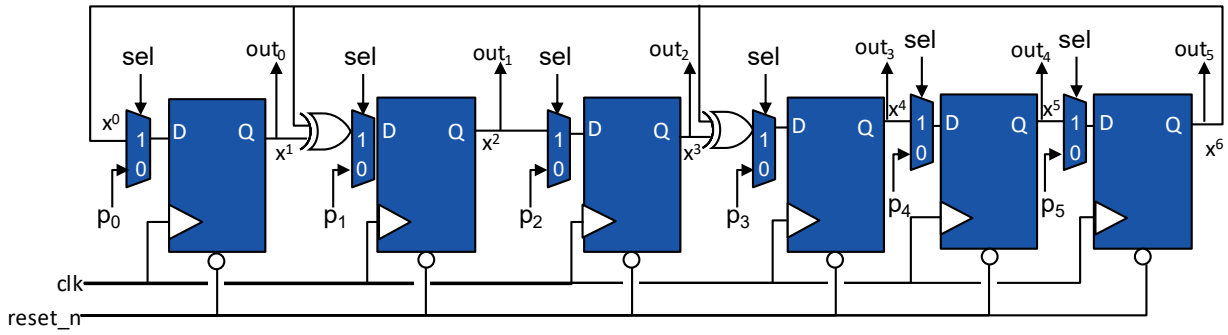
```
module register_file (
  input  logic       clk, rst_n,
  input  logic [7:0] data      ,
  input  logic       write_en  ,
  input  logic [4:0] r_address1, r_address2 , w_address,
  output logic [7:0] out1, out2
);
  // …
  // Add your description here
  // …
endmodule
```



1.4 QUESTION 4

Model a design using HDL that implements a **6-bit LFSR**. The LFSR should implement the polynomial $x^6+x^3+x^1+1$. The design should be able to load initial values in parallel using a **select** signal. When the select is '0', new bits should be loaded to the registers in parallel using a 6-bit input. When the select signal is '1', the design should not load new bits from the input but work as an LFSR. The LFSR should have a **6-bit output** that is read directly from the output of each flop. The schematic and module definition are given below.

```
module LFSR_6 (
  input  logic clk, rst_n,
  input  logic sel,
  input  logic [5:0] p,
  output logic [5:0] out
);
  // …
  // Add your description here
  // …
endmodule
```



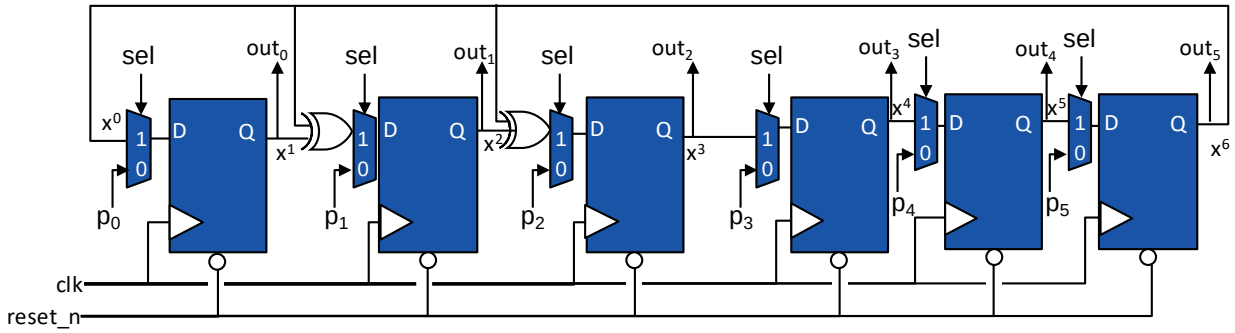
6 bit LFSR $x^6+x^3+x^1+1$

1.5 QUESTION 5

Model a design using HDL that implements a **6-bit LFSR**. The LFSR should implement the polynomial $x^6+x^2+x^1+1$. The design should be able to load initial values in parallel using a **select** signal. When the select is '0', new bits should be loaded to the registers in parallel using a 6-bit input. When the select signal is '1', the design should not load new bits from the input but work as an LFSR. The LFSR should have a **6-bit output** that is read directly from the output of each flop. The schematic and module definition are given below.

```
module LFSR_6 (
  input  logic clk, rst_n,
  input  logic sel,
  input  logic [5:0] p,
  output logic [5:0] out
);
  // …
  // Add your description here
```

```
  // …
endmodule
```

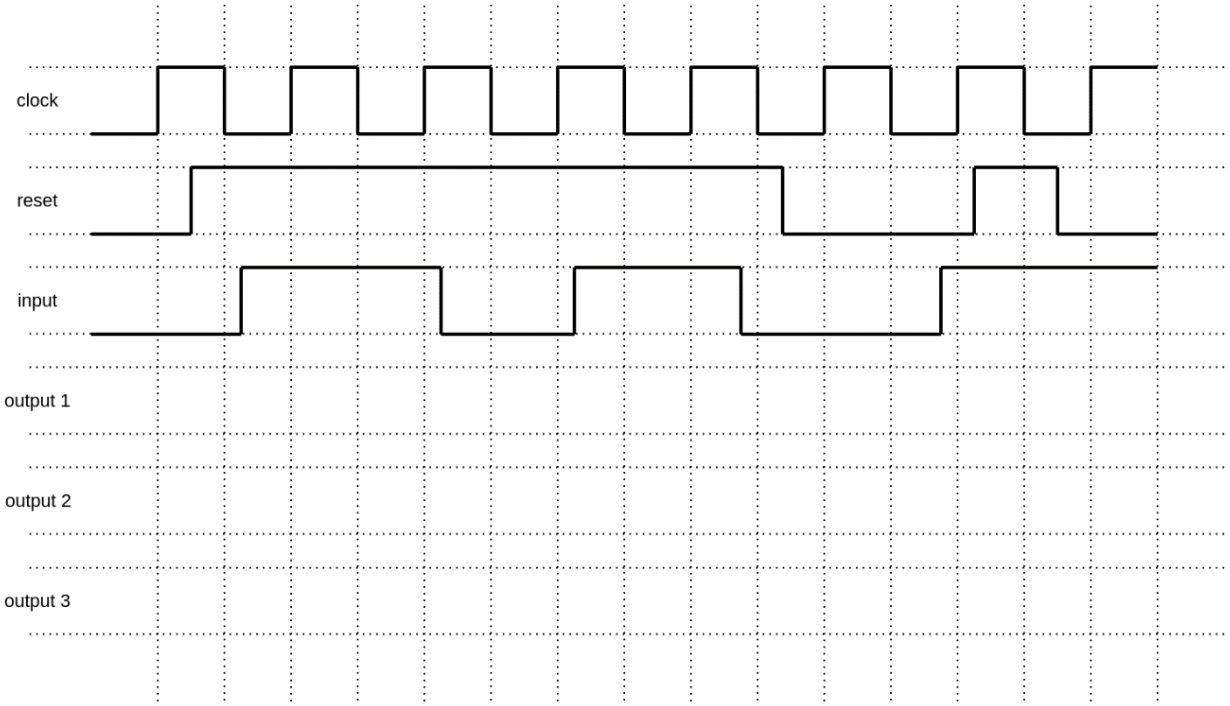


6 bit LFSR $x^6+x^2+x^1+1$

1.6 Question 5

Complete the timing diagram for:

1. Positive triggered D-latch
2. Positive edge triggered D flip-flop with active low asynchronous reset
3. Negative edge triggered D flip-flop with active low synchronous reset



1.7 QUESTION 7

Verify the setup and hold constraints for the following circuit. Calculate the and hold slack.

Clock = 10 ns
Setup time = 1 ns
Hold time = 0.2ns
C2Q = 0.5ns