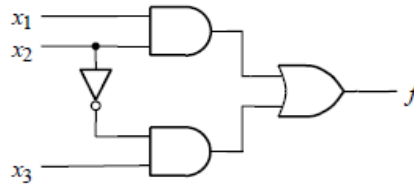


# VHDL, Lab1, Vježbe, 2018

## A) Kombinaciona kola

1. Za kolo prikazano na slici 1. napisati VHDL kod. Priložiti kod i simulacione dijagrame.



Slika 1.

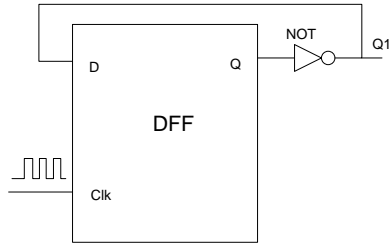
2. Za kolo polu-sabirača :
  - a) Napisati VHDL kod.
  - b) Kreirati simbol.
  - c) Koristeći simbol polu-sabirača i logičko kolo XOR u grafičkom editoru projektovati puni sabirač.
  - d) Simulacijom verifikovati ispravnost dizajna punog sabirača. Parametre *End Time* i *Grid Size* postaviti na vrijednosti  $2\mu s$  i  $200ns$ , respektivno.
3. a) Projektovati kolo u VHDLu 4-to-1 1-bit MUXa.

IO Pins	Description
a, b, c, d	Data Inputs
s[1:0]	MUX selector
o	Data Output

- b) Koristeći kreirani symbol Mux-a 4-1 i Mux 2-1, projektovati kolo multipleksora 8-1 i simulirati njegov rad.

## B) SEKVENCIJALNA KOLA

4. a) Projektovati u VHDL-u D Flip Flop sa ENABLE ulazom i klokom na pozitivnoj ivici  
b) Formirati simbol.  
c) Koristeci simbol DFF i inverter u grafičkom editoru napravite djelitelj frekvencije kloka sa 2.



5. Projektovati RS flip flop sa clear i set ulazima. Uočiti detla delay. Priložiti kod i simulacione dijagrame.
6. Donja tabela prikazuje definiciju pinova N bitnog brojača sa asinhronim clearom. Priložiti VHDL kod i simulacione dijagrame. Dizajnirati kod i sprovesti simulaciju.

IO Pins	Description
clock	Positive-Edge Clock
clear	Asynchronous Clear (active High)
enable	Enable input (active High)
Q[N:0]	Data Output

**Pomoc:**

```

library ieee ;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

-----

entity counter is

generic(N: natural :=2);
port(  clock: in std_logic;
       clear: in std_logic;
       enable: in std_logic;
       Q:      out std_logic_vector(N-1 downto 0)
);
end counter;

-----

architecture behv of counter is

```

```
    signal Pre_Q: std_logic_vector(N-1 downto 0);  
begin  
    -- princip rada opis  
    process(clock, count, clear)  
    begin  
        if clear = '1' then  
            Pre_Q <= Pre_Q - Pre_Q;  
        elsif (clock='1' and clock'event) then  
            if enable = '1' then  
                Pre_Q <= Pre_Q + 1; -- reset  
            end if;  
        end if;  
    end process;  
    -- concurrent assignment statement  
    Q <= Pre_Q;  
end behv;
```