```
--5.
type Tip1 is range 1 to 10;
type Tip2 is range 17 downto 10;
type Tip3 is range 1 to Tip1'high/2;
type Tjedan is (PON, UTO, SRI, CET, PET, SUB, NED);
subtype Vikend is Tjedan range SUB to NED;
type Tip4 is range 1 to 12;
type Tip5 is range Tip4'high downto Tip4'low;
type nadmorska_visina is range -1000 to 1000
    units
        mm;
        cm = 10 mm;
        dm = 10 cm;
        m = 10 dm;
        km = 1000 m;
    end units nadmorska_visina;

subtype visina is nadmorska_visina range 0 to 200 cm;
subtype prirodni_brojevi_bez_nule is integer range 1 to integer'high;
type Tip6 is range 100 downto 0;
type Tip7 is range Tip6'low to Tip6'high;



--12.
--Proces ovisi samo o promjeni signala a, te se izlazima dodjeljuju vrijednosti samo kad se
promjeni a.
--Ako se gleda sklop, izlazima se dodjeljuje vrijednost i kad se mijenja a i kad se mijenja
b.
--I valjda se ne crtaju delta kašnjenja kada se crta dijagram za shemu.

--23.
--Najmanje 1 um, najviše 100 km (bar mislim da je tako)..

--14, 16, 17, 18 – po meni se ne mogu nacrtati.

--19.

--Tu je caka da se  wait for ne računa nakon dodjeljivanja vrijednosti a1 neg se računa od
početka
--(1. Wait), a drugi se računa od 50 ns (odnosno od kraja 1. Waita)...vidi se na slici...

--32.

entity Sklop is
    Port ( din1 : in  STD_ULOGIC_VECTOR (11 downto 0);
           din2 : in  STD_ULOGIC_VECTOR (9 downto 0);
           gsr : in  STD_ULOGIC;
           reset : in  STD_ULOGIC;
           clk : in  STD_ULOGIC;
           dout : out  STD_ULOGIC_VECTOR (23 downto 0));
end Sklop;

--33.

entity Sklop is
    Port ( din : in  STD_LOGIC_VECTOR (11 downto 0);
```

```
           control : in  STD_LOGIC_VECTOR (2 downto 0);
           gsr : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           clk : in  STD_LOGIC;
           ack : out  STD_LOGIC;
           dout : out  STD_LOGIC_VECTOR (7 downto 0);
           comm : inout  STD_LOGIC_VECTOR (3 downto 0));
end Sklop;

--37.

entity Sklop is
    Port ( a : in  STD_LOGIC;
           b : in  STD_LOGIC;
           c : in  STD_LOGIC;
           x : out  STD_LOGIC;
           y : out  STD_LOGIC);
end Sklop;

architecture Behavioral of Sklop is

begin
    process(a, b, c) is
    variable t1, t2: std_logic;

    begin
        t1 := not c;
        t2 := a xor b;
        x <= t2;
        y <= t1 and t2;
    end process;


end Behavioral;

--38.

entity Sklop is
    Port ( a : in  STD_LOGIC;
           b : in  STD_LOGIC;
           c : in  STD_LOGIC;
           x : out  STD_LOGIC;
           y : out  STD_LOGIC);
end Sklop;

architecture Behavioral of Sklop is
    signal t1, t2: std_logic;

begin
    t1 <= not c;
    t2 <= a xor b;
    x <= t2;
    y <= t1 and t2;

end Behavioral;

--39.
```

```
entity Sklop is
    Port ( a : in  STD_LOGIC;
           b : in  STD_LOGIC;
           c : in  STD_LOGIC;
           x : out  STD_LOGIC;
           y : out  STD_LOGIC);
end Sklop;

architecture Behavioral of Sklop is
    component xor_vrata
        port(in1, in2: in bit;
                    o: out);
    end component;

    component and_vrata
        port(in1, in2: in bit;
                    o: out);
    end component;

    component not_vrata
        port(i: in bit;
                    o: out);
    end component;

    signal t1, t2: std_logic;


begin
    I0: component not_vrata port map (i => c, o => t1);
    X0: component xor_vrata port map (in1 => a, in2 => b, o => t2);
    A0: component and_vrata port map (in1 => t2, in2 => t1, o => y);


end Behavioral;

--41. ovo baš i nisam siguran da je dobro

entity Sklop is
    Port ( data : in  integer;
           sel : in  integer;
           izl1 : out  integer;
           izl2 : out  integer;
           izl3 : out  integer;
           izl4 : out  integer);
end Sklop;

architecture Behavioral of Sklop is

begin
    process (data, sel) is
    begin
        case sel is
            when 0 =>
                izl1 <= data;
            when 1 =>
                izl2 <= data;
```

```
            when 2 =>
                izl3 <= data;
            when 3 =>
                izl4 <= data;
            when others =>
                izl1 <= 0;
                izl2 <= 0;
                izl3 <= 255;
                izl4 <= 255;
        end case;
    end process;


end Behavioral;

--55.

entity brojilo3b is
    Port ( CLK : in  STD_LOGIC;
           CE : in  STD_LOGIC;
           R : in  STD_LOGIC;
           B : out  STD_LOGIC_VECTOR (2 downto 0));
end entity brojilo3b;

architecture brojilo3b_arch of brojilo3b is

    component D_bistabil
        port(D, CLK, CE, R: in std_logic;
             Q, Q_inv: out std_logic);
    end component;

    signal q: std_logic_vector(2 downto 0);
    signal t: std_logic;

begin

    D0: component D_bistabil port map(D => t, CLK => CLK, CE => CE, R => R, Q => q(0), Q_inv
=> open);
    D1: component D_bistabil port map(D => q(0), CLK => CLK, CE => CE, R => R, Q => q(1),
Q_inv => open);
    D2: component D_bistabil port map(D => q(1), CLK => CLK, CE => CE, R => R, Q => q(2),
Q_inv => t);

    B <= Q;

end brojilo3b_arch;


--59. redom – 0, 0, X, X, 1
```