Informatikos fakultetas

SKAITMENINĖS LOGIKOS PRADMENYS

Individualios užduoties Nr. 109

Atliko: Vytenis Kriščiūnas gr. Stud. IFF-1/1

Primėmė: dėst. Stasys Maciulevičius

Kaunas, 2022

TURINYS

1. ILIUSTRACIJŲ SĄRAŠAS

2. ĮVADAS

Šio individualaus darbo tikslas buvo išmokti įvairių skaitiklių ir daliklių veikimo principus, suprasti jų realizavimą VHDL kalboje, įsigilinti į projektavimo ir taikymo galimybes. Atliekant užduotis patikrinti skaitiklių ir daliklių veikimą simuliacijoje ir programuojamos logikos schemose. Remiantis pateiktais pavyzdžiais teorinėje medžiagoje pilnai atlikti priskirtą individualią užduotį.

3. NAGRINĖJAMOS TEORINĖS DARBO PRIELAIDOS

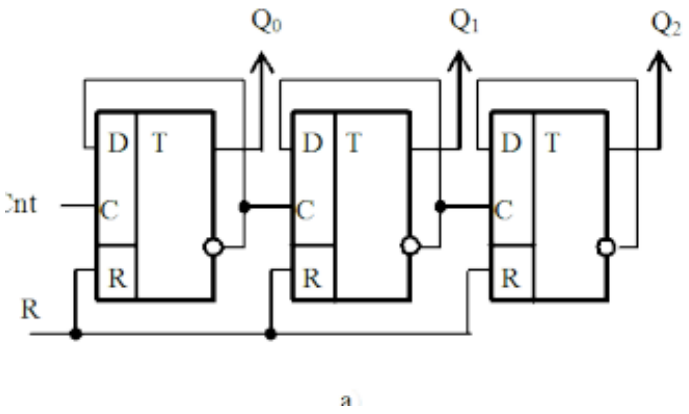
Siekdamas atlikti individualią užduotį turėjau išnagrinėti teorinę šio darbo dalį. Teorinėje medžiagoje išsisaiškinau, kad skaitikliais yra vadinami operaciniai elementai, atliekantys skaičiavimo mikrooperacijas, pavyzdžiui: pridedantys skaitikliai, atimantys skaitikliai arba abi šias operacijas. Skaitikliai, kurios teko nagrinėti buvo sudaromi iš trigerių. O pats trigerių skaičius buvo nagrinėjamas pagal formulę:

$$m \geq [\log M], \text{ kur } M - \text{skaitiklio skaičiavimo modulis (skaitiklio būsenų skaičius).}$$

1 pav. Trigerių skaičiaus radimo formulė

Skaitikliui pasiekus skaičiavimo ribą, jis suformuoja pernašą, kurią gali naudoti kitas skaitiklis. Trigerių skaitikliai yra skirstomi į asinchroninius ir sinchroninius. Sinchroninių skaitiklių visi trigeriai būna perjungiami tuo pačiu metu, kai yra išsiunčiamas tas pats sinchronizavimo signalas.

Asinchroniniai skaitikliai susidaro iš D trigerių, kaip parodyta apačioje esančiame paveiklėlyje.



2 pav. Asichroninis skaitiklis

Individualioje užduotyje pateiktus skaitiklius reikėjo realizuoti VHDL kalboje. Pagal pateiktas formules 3 ir 4 paveiklėliuose buvo galima apskaičiuoti JM1 ir JM2 skaitiklių nustatymo į nulinę būseną sąlygas.

N2 = (**JM** – 1) div **M1**, (div – sveikoji dalis)

N1 = (**JM** – 1) mod **M1**;

3 pav. JM1 skaitiklio vidinio perkrovimo sąlygos skaičiavimo formulė

N3 = (**JM** – 1) div (**M1** × **M2**),

L1 = (**JM** – 1) mod (**M1** × **M2**),

N2 = **L1** div **M1**,

N1 = **L1** mod **M1**;

4 pav. JM2 skaitiklio vidinio perkrovimo sąlygos skaičiavimo formulė

4. INDIVIDUALIOS UŽDUOTIES PROJEKTAVIMO ETAPAI

4.1 Skaitiliai.

Nagrinėjau man priskirtos individualios užduoties skaitiklius, remiantis pateikta teorija.

10	26	54	243	1080

5 pav. skaitikliai

4.2 M1 skaitiklio VHDL kodas.

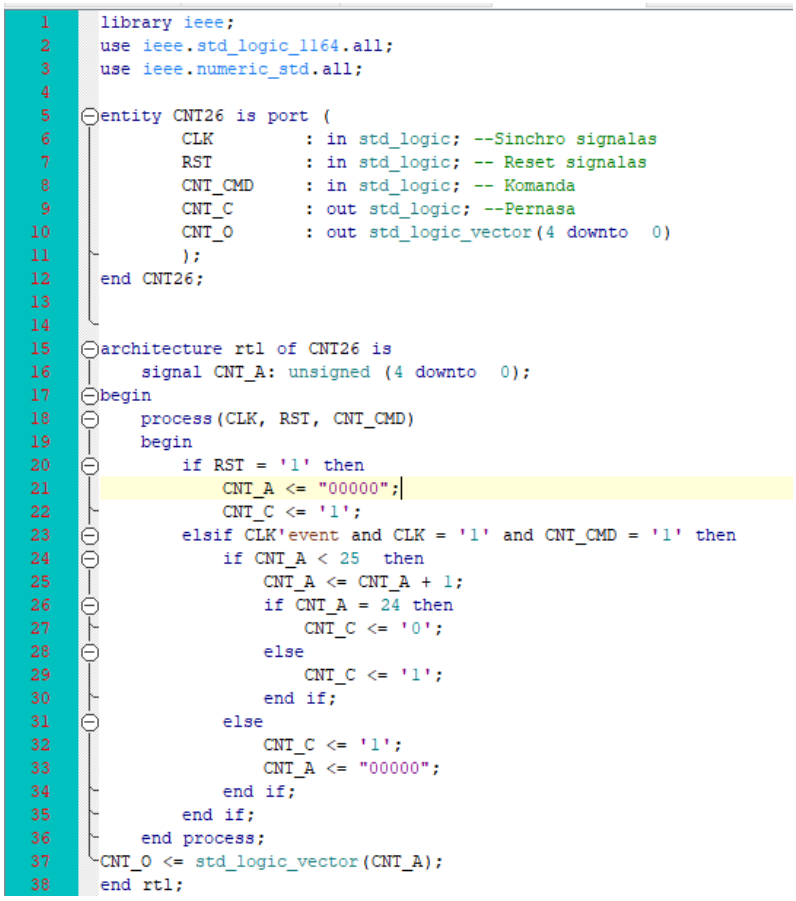
Pakeičiau teorinėje mežiagoje pateiktą M1 VHDL kodą, kad atitiktų man priskirtoje salygoje M1 skaitiklį.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity CNT10 is port (
        CLK         : in std_logic; --Sinchro signalas
        RST         : in std_logic; -- Reset signalas
        CNT_CMD     : in std_logic; -- Komanda
        CNT_C       : out std_logic; --Pernasa
        CNT_O       : out std_logic_vector(3 downto  0)
        );
end CNT10;

architecture rtl of CNT10 is
    signal CNT_A: unsigned (3 downto  0);
begin
    process(CLK, RST, CNT_CMD)
    begin
        if RST = '1' then
            CNT_A <= "0000";
            CNT_C <= '1';
        elsif CLK'event and CLK = '1' and CNT_CMD = '1' then
            if CNT_A < 9  then
                CNT_A <= CNT_A + 1;
                if CNT_A = 8 then
                    CNT_C <= '0';
                else
                    CNT_C <= '1';
                end if;
            else
                CNT_C <= '1';
                CNT_A <= "0000";
            end if;
        end if;
    end process;
CNT_O <= std_logic_vector(CNT_A);
end rtl;
```

6 pav. M1 skaitiklio VHDL kodas

4.3 M2 skaitiklio VHDL kodas.

Pakeičiau teorinėje medžiagoje pateiktą M2 VHDL kodą, kad atitiktų man priskirtoje sąlygoje M2 skaitiklį.

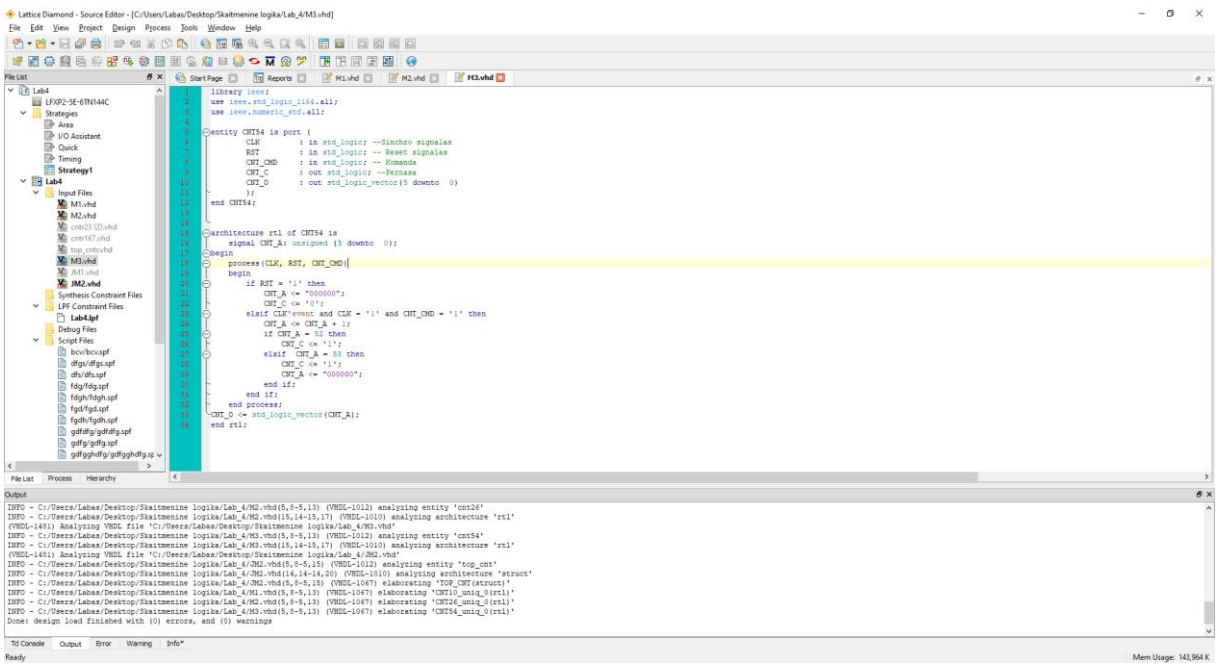


```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity CNT26 is port (
        CLK         : in std_logic; --Sinchro signalas
        RST         : in std_logic; -- Reset signalas
        CNT_CMD     : in std_logic; -- Komanda
        CNT_C       : out std_logic; --Pernasa
        CNT_O       : out std_logic_vector(4 downto  0)
        );
end CNT26;


architecture rtl of CNT26 is
    signal CNT_A: unsigned (4 downto  0);
begin
    process(CLK, RST, CNT_CMD)
    begin
        if RST = '1' then
            CNT_A <= "00000";
            CNT_C <= '1';
        elsif CLK'event and CLK = '1' and CNT_CMD = '1' then
            if CNT_A < 25  then
                CNT_A <= CNT_A + 1;
                if CNT_A = 24 then
                    CNT_C <= '0';
                else
                    CNT_C <= '1';
                end if;
            else
                CNT_C <= '1';
                CNT_A <= "00000";
            end if;
        end if;
    end process;
CNT_O <= std_logic_vector(CNT_A);
end rtl;
```

7 pav. M2 skaitiklio VHDL kodas

4.4 M3 skaitiklio VHDl kodas.

Pakeičiau teorinėje medžiagoje pateiktą M3 VHDL kodą, kad atitiktų man priskirtoje sąlygoje M3 skaitiklį.



8 pav. M3 skaitiklio VHDL kodas

4.5 JM1 skaitiklio nustatymo į nulinę būseną sąlygos skaičiavimas.

$$N2 = (243 - 1) \text{ div } 10 = 24$$

$$N1 = (243 - 1) \text{ mod } 10 = 2$$

9 pav. JM1 nustatymo į nulinę būseną sąlygos skaičiavimas

4.6 JM1 skaitiklio VHDL kodas.

Pakeičiau teorinėje medžiagoje pateiktą JM1 VHDL kodą, kad atitiktų man priskirtoje sąlygoje JM1 skaitiklį.

```vhdl
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity TOP_CNT is port (

        CLK_I   : in std_logic; --Sinchro signalas
        RST_I   : in std_logic; -- Reset signalas
        ENBL_I  : in std_logic; -- Aktyvavimo signalas
        CNT_CO  : out std_logic --Pernasa
        );
end TOP_CNT;

architecture struct of TOP_CNT is

signal C,RST_internal,C1,C2 : std_logic;
signal CNT_1_O :  std_logic_vector(3 downto 0);
signal CNT_2_O :  std_logic_vector(4 downto 0);

component    CNT10
    port    (
        CLK : in std_logic; --Sinchro signalas
        RST : in std_logic; -- Reset signalas
        CNT_CMD  : in std_logic; -- Komanda
        CNT_C    : out std_logic; --Pernasa
        CNT_O    : out std_logic_vector(3 downto  0));
end component;

component    CNT26
    port  (
        CLK : in std_logic; --Sinchro signalas
        RST : in std_logic; -- Reset signalas
        CNT_CMD  : in std_logic; -- Komanda
        CNT_C    : out std_logic; --Kai pasiekia 0
        CNT_O    : out std_logic_vector(4 downto  0) );
end component;

```

```vhdl
begin
    CNT_1:  CNT10   port map (CLK=>CLK_I,
        RST=>RST_internal, CNT_CMD=>ENBL_I,
        CNT_C=>C1, CNT_O=>CNT_1_O);
    CNT_2:  CNT26  port map (CLK=> C1,
        RST=>RST_internal, CNT_CMD=>ENBL_I,
        CNT_C=>C2, CNT_O=>CNT_2_O);
    process(CLK_I,RST_I)
    begin
        if  (RST_I = '1') then
            RST_internal <= '1';  CNT_CO <= '0';
        elsif CLK_I'event and CLK_I = '1' then
            if ((CNT_2_O(4) = '1')
            and (CNT_2_O(3) = '1')
            and (CNT_1_O(1) = '1'))  then
                RST_internal <= '1';
                CNT_CO <= '1';
            else
                RST_internal <= '0';
                CNT_CO <= '0';
            end if;
        end if;
    end process;
end struct;
```

10 pav. JM1 skaitiklio VHDL kodas

4.7 JM2 nustatymo į nulinę būseną sąlygos skaičiavimas.

N3 = (1080 – 1) div (10 x 26) = 4

L1 = (1080 – 1) mod (10 x 26) = 39

N2 = 39 div 10 = 3

N1 = 39 mod 10 = 9

11 pav. JM2 nustatymo į nulinę būseną sąlygos skaičiavimas

4.8 JM2 skaitiklio VHDL kodas.

Pakeičiau teorinėje medžiagoje pateiktą JM2 VHDL kodą, kad atitiktų man priskirtoje sąlygoje JM2 skaitiklį.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity TOP_CNT is port (

        CLK_I   : in std_logic; --Sinchro signalas
        RST_I   : in std_logic; -- Reset signalas
        ENBL_I  : in std_logic; -- Aktyvavimo signalas
        CNT_CO  : out std_logic --Pernasa
        );
end TOP_CNT;

architecture struct of TOP_CNT is

signal C,RST_internal,C1,C2,C3 : std_logic;
signal CNT_1_O :  std_logic_vector(3 downto 0);
signal CNT_2_O :  std_logic_vector(4 downto 0);
signal CNT_3_O :  std_logic_vector(5 downto  0);

component    CNT10
    port    (
        CLK : in std_logic; --Sinchro signalas
        RST : in std_logic; -- Reset signalas
        CNT_CMD  : in std_logic; -- Komanda
        CNT_C    : out std_logic; --Pernasa
        CNT_O    : out std_logic_vector(3 downto  0));
end component;

component    CNT26
    port  (
        CLK : in std_logic; --Sinchro signalas
        RST : in std_logic; -- Reset signalas
        CNT_CMD  : in std_logic; -- Komanda
        CNT_C    : out std_logic; --Kai pasiekia 0
        CNT_O    : out std_logic_vector(4 downto  0));
end component;
```

```
component    CNT54
    port  (
        CLK : in std_logic; --Sinchro signalas
        RST : in std_logic; -- Reset signalas
        CNT_CMD  : in std_logic; -- Komanda
        CNT_C    : out std_logic; --Kai pasiekia 0
        CNT_O    : out std_logic_vector(5 downto  0));
end component;

begin
    CNT_1:  CNT10   port map (CLK=>CLK_I,
        RST=>RST_internal, CNT_CMD=>ENBL_I,
        CNT_C=>C1, CNT_O=>CNT_1_O);
    CNT_2:  CNT26  port map (CLK=> C1,
        RST=>RST_internal, CNT_CMD=>ENBL_I,
        CNT_C=>C2, CNT_O=>CNT_2_O);
    CNT_3:  CNT54  port map (CLK=> C2,
        RST=>RST_internal, CNT_CMD=>ENBL_I,
        CNT_C=>C3, CNT_O=>CNT_3_O);
    process(CLK_I,RST_I)
    begin
        if  (RST_I = '1') then
            RST_internal <= '1';  CNT_CO <= '0';
        elsif CLK_I'event and CLK_I = '1' then
            if ((CNT_3_O(2) = '1')
            and (CNT_2_O(1) = '1')
            and (CNT_2_O(0) = '1')
            and (CNT_1_O(3) = '1')
            and (CNT_1_O(0) = '1'))  then
                RST_internal <= '1';
                CNT_CO <= '1';
            else
                RST_internal <= '0';
                CNT_CO <= '0';
            end if;
        end if;
    end process;
end struct;
```

12 pav. JM2 skaitiklio VHDL kodas

4.9 ModelSim testai.

Susidariau testavimo direktyvas.

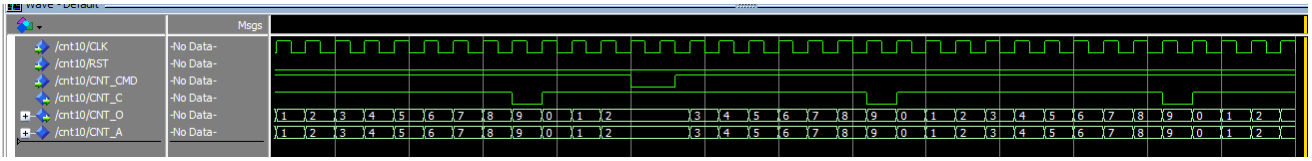
*New Text Document.txt - Notepad

File Edit Format View Help

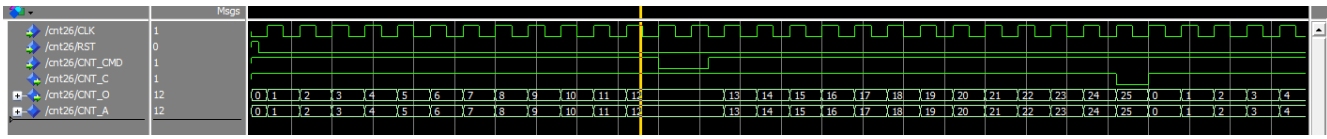
```
restart -f
force -freeze sim:/cnt10/Rst 1 0, 0 {10 ps}
force -freeze sim:/cnt10/CLK 0 0, 1 {20 ps} -r 40
force -freeze sim:/cnt10/CNT_CMD 1 0, 0 {500 ps}
force -freeze sim:/cnt10/CNT_CMD 1 560
run 1400

restart -f
force -freeze sim:/cnt26/Rst 1 0, 0 {10 ps}
force -freeze sim:/cnt26/CLK 0 0, 1 {20 ps} -r 40
force -freeze sim:/cnt26/CNT_CMD 1 0, 0 {500 ps}
force -freeze sim:/cnt26/CNT_CMD 1 560
run 1400

restart -f
force -freeze sim:/cnt54/Rst 1 0, 0 {10 ps}
force -freeze sim:/cnt54/CLK 0 0, 1 {20 ps} -r 40
force -freeze sim:/cnt54/CNT_CMD 1 0, 0 {500 ps}
force -freeze sim:/cnt54/CNT_CMD 1 560
run 3000

force -freeze sim:/top_cnt/CLK_I 1 0, 0 {10 ps} -r 20
force -freeze sim:/top_cnt/ENBL_I 1 0, 0 {500 ps}
force -freeze sim:/top_cnt/ENBL_I 1 560
force -freeze sim:/top_cnt/RST_I 1 0, 0 {5 ps}
run 10000

force -freeze sim:/top_cnt/CLK_I 1 0, 0 {10 ps} -r 20
force -freeze sim:/top_cnt/ENBL_I 1 0, 0 {500 ps}
force -freeze sim:/top_cnt/ENBL_I 1 560
force -freeze sim:/top_cnt/RST_I 1 0, 0 {5 ps}
run 100000
```

13 pav. Testavimo direktyvos

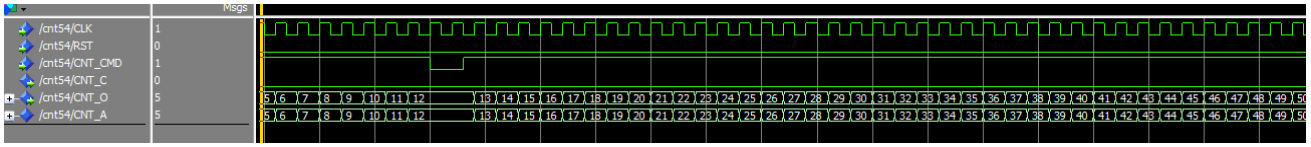
4.10 ModelSim programinės įrangos gauti rezultatai.



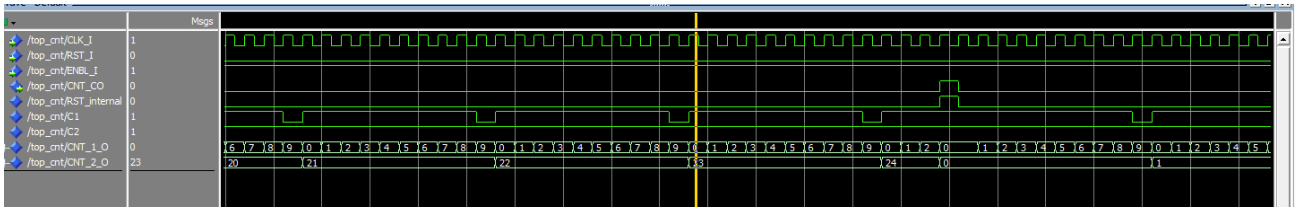
14 pav. M1 skaitiklio ModelSim rezultatai



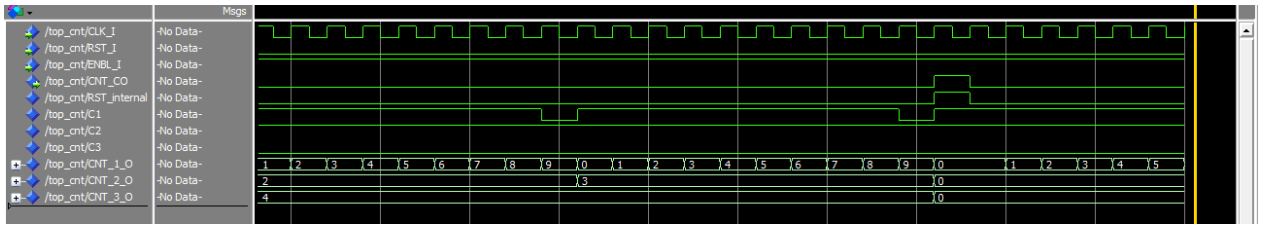
15 pav. M2 skaitiklio ModelSim rezultatai



16 pav. M3 skaitiklio ModelSim rezultatai

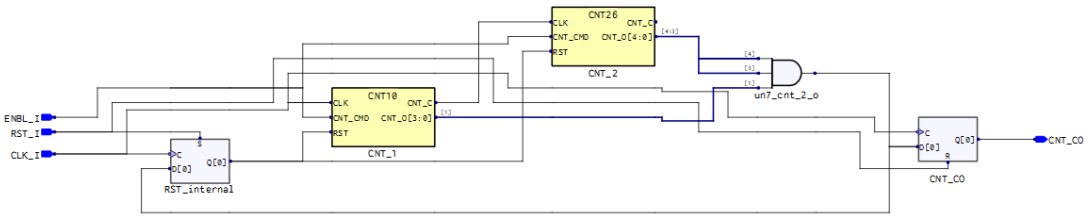


17 pav. JM1 skaitiklio ModelSim rezultatai

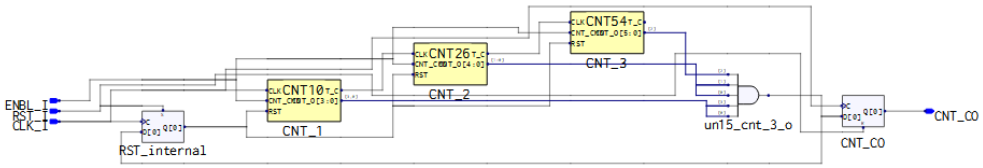


18 pav. JM2 skaitiklio ModelSim rezultatai

4.11 RTL hierarchinė realizalicja.



19 pav. Symplify Pro JM1 skaitiklio schema



20 pav. Simplify Pro JM2 skaitiklio schema

5. IŠVADOS

Ši individuali užduotis sukėlė man įvairiausių iššūkių, kuriuos pavyko įveikti pasiskaičius pateiktą teorinę paskaitų medžiagą. Sunkiausia darbo dalis buvo atlikti nustatymo į nulinę būseną sąlygos skaičiavimus, pasitelkus duotas formules. Patikrinti skaitiklius ModelSim programoje, naudojant sudarytas testines direktyvas, sunku nebuvo. Daugiausiai laiko skyriau teorinės darbo dalies supratimui. Pateikta informacija buvo labai naudinga sudarinėjant VHDL kodus. Įgytos žinios privedė prie individualios užduoties realizacijos.