УНИВЕРЗИТЕТ
У НОВОМ САДУ

ФАКУЛТЕТ
ТЕХНИЧКИХ НАУКА

Трг Доситеја Обрадовића 6, 21000 Нови Сад, Југославија
Деканат: 021 350-413; 021 450-810; Централа: 021 350-122
Рачуноводство: 021 58-220; Студентска служба: 021 350-763
Телефакс: 021 58-133; e-mail: ftndean@uns.ns.ac.yu

PROJEKAT

iz Računarskog projektovanja digitalnih integrisanih kola

TEMA PROJEKTA:

Analiza rada Decimalnog brojača od 0 do 999 parametrizovanog u odnosu na opseg brojanja

TEKST ZADATKA:

1. U programskom jeziku VHDL napisati kod kojim se realizuje funkcionalnost decimalnog brojača koji podrazumevano broji od 000 do 999 i koji je sastavljen od tri brojaca koji broje od 0 do 9. Dodatno obezbediti mogućnost podešavanja opsega brojanja.
2. Koristeći simulator *NClaunch* programskog paketa *Cadence* prikazati rezultate simulacije kojom se potvrđuje funkcionalnost kola.
3. Koristeći alat *RTL Compiler* programskog paketa *Cadence* generisati električnu šemu u AMS 0.35μm (C35B4) CMOS tehnologiji.
4. Koristeći alat *First Encounter-a* programskog paketa *Cadence* generisati fizičku realizaciju (lejaut) projektovanog kola.

Mentor
Jelena Radić

Student/Studentkinja
Dejan Grubišić, EE20/2014

U Novom Sadu, 23.2.2018

1 Opis projekta

Brojači u digitalnim sistemima predstavljaju osnovne komponente i često su neophodni za rad digitalnog kola. Obično su implementirani kao binarni brojači, ali mogu biti implementirani i u drugim oblicima kao na primer Grejov, Džonsonov ili Decimalni brojač, što je slučaj i u ovom projektu.

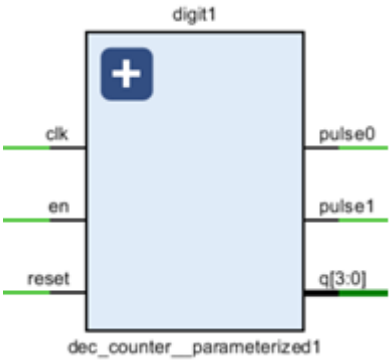
Decimalni brojač je u ovom projektu sastavljen od tri zasebna decimalna brojača koji broje od nula do devet. Pri prelazu sa devet na nula aktivira se signal **pulse0** koji je vezan za **enable** ulaz sledećeg brojača, što dalje dovodi do transakcije. U opštem slučaju broji od 0 do 999 nakon čega se vraća na 0. Drugi izlazni parametar je **pulse1**, koji postaje aktivan kada brojač dostigne gornju granicu do koje treba da broji.

Dizajn je parametrizovan sa dva parametra koji predstavljaju početak i kraj brojanja. Ovi parametri uvedeni su preko ***generic*** naredbe, što znači da prilikom implementacije ove vrednosti neće moći da budu promenjene. Zbog toga potrebno se unapred odlučiti koji će opseg brojanja biti. Ako je broj početka brojanja na primer 156, broj 1 će biti prosleđen prvom brojaču kao početna vrednost, 5 ka drugom, a 6 ka trećem. Nakon reseta ili dostizanja poslednjeg broja, tri brojača se postavljaju na ove vrednosti. Na isti način se dodeljuje i gornja granica.
Nakon što dostigne gornju granicu generiše se signal **pulse1** i kada svi brojači dostignu svoje granice aktivira se **reset** i oni se vraćaju na inicijalne vrednosti.

2 Projektovanje digitalnog integrisanog kola

VHDL model kola decimalnog brojača od 0 do 9

Izgled bloka je dat na slici 1.



Slika 1: Izgled pojedinačnog brojača

Model decimalnog brojača

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

entity dec_counter is
   Generic(start_num: natural := 0;
         end_num: natural := 9);
   Port ( en : in STD_LOGIC;
         reset : in STD_LOGIC;
         clk : in STD_LOGIC;
         pulse0 : out STD_LOGIC;  --pulse after 9
         pulse1 : out STD_LOGIC;  --pulse to indicate final number
         q : out STD_LOGIC_VECTOR (3 downto 0));
end dec_counter;

architecture Behavioral of dec_counter is
    signal counter: unsigned(3 downto 0) := (others => '0');
begin

    process (clk)
    begin
            if(rising_edge(clk))then
                    if(reset = '1')then
                            counter <= to_unsigned(start_num, 4);
                            pulse0 <= '0';
                            pulse1 <= '0';
                            if(start_num = 9) then
                               pulse0 <= '1';
```

```
                        end if;
                    else
                        if(en = '1')then
                            counter <= counter + 1;
                            pulse0 <= '0';
                            pulse1 <= '0';
                            if(counter = to_unsigned(9, 4))then
                                counter <= (others => '0');
                            end if;
                            if(counter = to_unsigned(8, 4))then
                pulse0 <= '1';
            end if;

            if(end_num = 0)then
                if(counter = to_unsigned(9, 4)) then
                    pulse1 <= '1';
                end if;
            elsif(counter = to_unsigned(abs(end_num-1) , 4))then
                pulse1 <= '1';
            end if;
                        else
                            counter <= counter;
                        end if;
                    end if;
            end if;
    end process;

    q <= std_logic_vector(counter);
end Behavioral;
```

Strukturni model čitavog sistema

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

entity counter999 is
    Generic(start_num: natural := 21;
            end_num  : natural := 256);
    Port ( en : in STD_LOGIC;
           reset : in STD_LOGIC;
           clk : in STD_LOGIC;
           q0 : out STD_LOGIC_VECTOR (3 downto 0);
           q1 : out STD_LOGIC_VECTOR (3 downto 0);
           q2 : out STD_LOGIC_VECTOR (3 downto 0)
```

```
        );
end counter999;

architecture Behavioral of counter999 is
    signal pulse0_s, pulse1_s : std_logic := '0';
    signal pulse0_and_pulse1_s_and_en, pulse0_s_and_en: std_logic := '0';

    signal pulse_reset0_s, pulse_reset1_s, pulse_reset2_s: std_logic := '0';
    signal reset_all: std_logic := '0';

    component dec_counter
    generic(start_num: natural;
            end_num: natural);
    port(   en : in STD_LOGIC;
            reset : in STD_LOGIC;
            clk : in STD_LOGIC;
            pulse0 : out STD_LOGIC;
            pulse1 : out STD_LOGIC;
            q : out STD_LOGIC_VECTOR (3 downto 0)
            );
    end component;
begin

    digit0: dec_counter
        generic map(start_num => start_num rem 10,
                    end_num => end_num rem 10)
        port map (  en => en,
                    reset => reset_all,
                    clk => clk,
                    pulse0 => pulse0_s,
                    pulse1 => pulse_reset0_s,
                    q => q0
                    );
    digit1: dec_counter
        generic map(start_num => ((start_num rem 100) - (start_num rem 10))/10,
                end_num => ((end_num rem 100) - (end_num rem 10))/10)
        port map (  en => pulse0_s_and_en,
                    reset => reset_all,
                    clk => clk,
                    pulse0 => pulse1_s,
                    pulse1 => pulse_reset1_s,
                    q => q1
                    );
    digit2: dec_counter
        generic map(start_num => (start_num - (start_num rem 100))/100,
```

```
            end_num => (end_num - (end_num rem 100))/100)
        port map (  en => pulse0_and_pulse1_s_and_en ,
                reset => reset_all,
                clk => clk,
                pulse0 => open,
                pulse1 => pulse_reset2_s,
                q => q2
                );
    pulse0_s_and_en <= pulse0_s and en;
    pulse0_and_pulse1_s_and_en <= pulse0_s and pulse1_s and en;
    reset_all <= (pulse_reset0_s and pulse_reset1_s and pulse_reset2_s) or
reset;

end Behavioral;
```

3 Simulaciona provera rezultata i diskusija dobijenih rezultata

Verifikacija ovog kola zasniva se na odabiru parametara koji iniciraju tranzicije na svim izlazima.

Verifikacioni kod

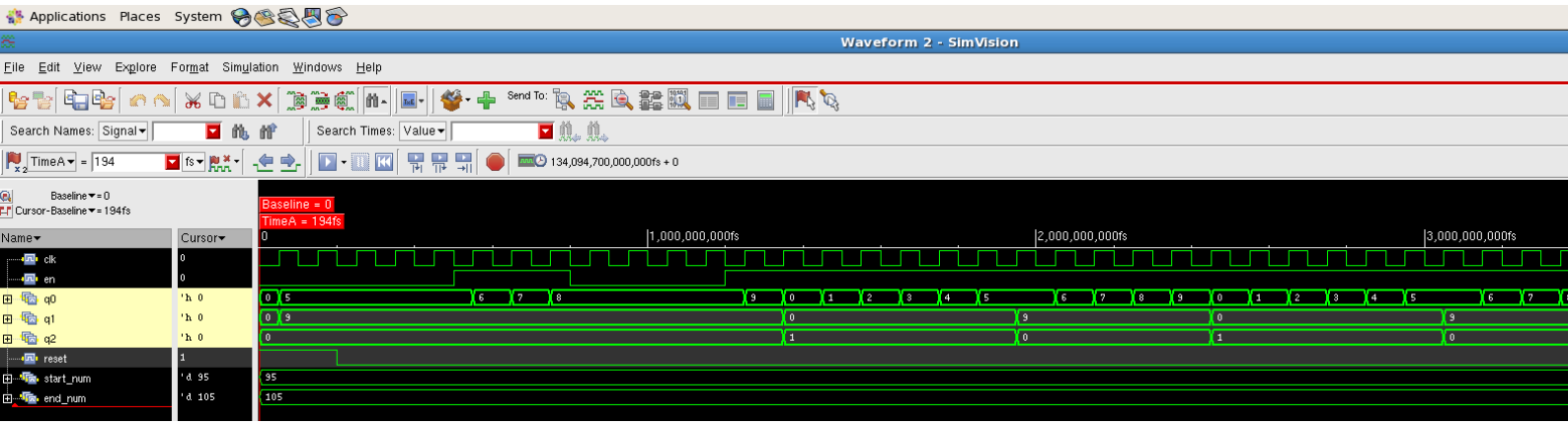
```
library IEEE;
use IEEE.Std_logic_1164.all;
use IEEE.Numeric_Std.all;
```

```
entity counter999_tb is
end;

architecture bench of counter999_tb is

  component counter999
     Generic(start_num: natural := 95;
             end_num  : natural := 105);
     Port ( en : in STD_LOGIC;
            reset : in STD_LOGIC;
            clk : in STD_LOGIC;
            q0 : out STD_LOGIC_VECTOR (3 downto 0);
            q1 : out STD_LOGIC_VECTOR (3 downto 0);
            q2 : out STD_LOGIC_VECTOR (3 downto 0)
            );
  end component;

  signal en: STD_LOGIC;
  signal reset: STD_LOGIC;
  signal clk: STD_LOGIC;
  signal q0: STD_LOGIC_VECTOR (3 downto 0);
  signal q1: STD_LOGIC_VECTOR (3 downto 0);
  signal q2: STD_LOGIC_VECTOR (3 downto 0) ;

begin

  -- Insert values for generic parameters !!
  uut: counter999 generic map ( start_num => 95,
                                end_num   => 105 )
                  port map ( en        => en,
                             reset     => reset,
                             clk       => clk,
                             q0        => q0,
                             q1        => q1,
                             q2        => q2 );

  clk_stimulus: process
  begin
    clk <= '0';
    wait for 50 ns;
    clk <= '1';
    wait for 50 ns;

  end process;

    stimulus: process
    begin
       reset <= '1' , '0' after 200 ns;

       en <= '0', '1' after 500 ns, '0' after 800 ns, '1' after 1200 ns;
```

```
        wait;
    end process;

end;
```

Rezultati simulacije

Nakon pokretanja simulacionog okruženja *NClaunch* dobijamo sledeće rezultate (Slika 2).



Slika 2: Rezultati verifikacije

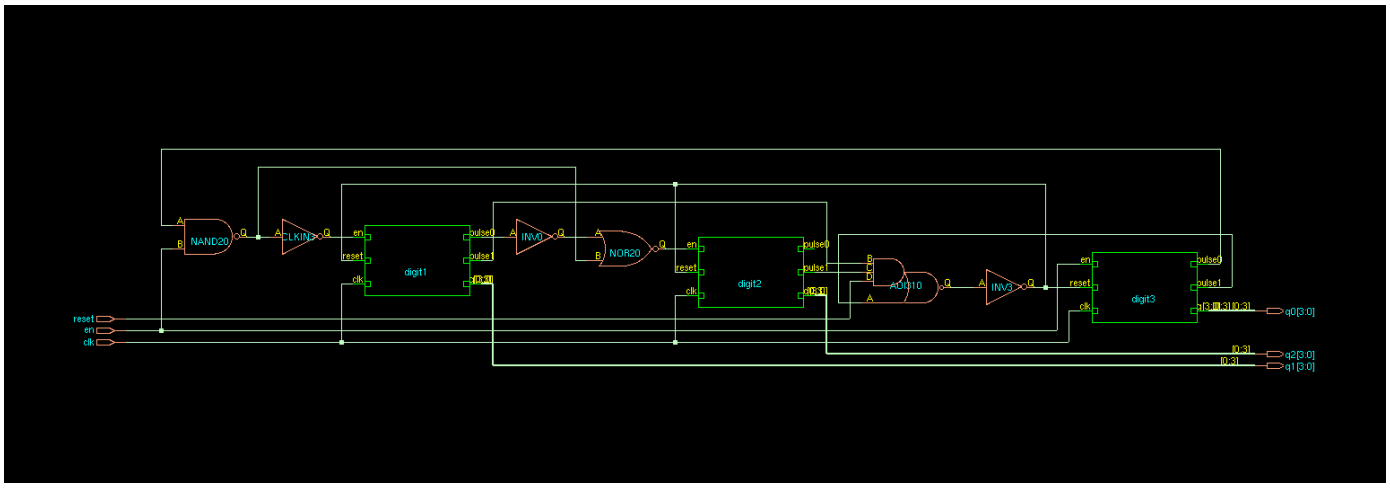
Diskusija dobijenih rezultata

Dobijeni rezultati ukazuju na vrednosti sitnala za odabrane parametre početka brojanja 95 i kraja brojanja 105. Ovi parametri su birani tako da se dese tranzicije na svim izlazima kako bi se jasno videla valjanost modela.

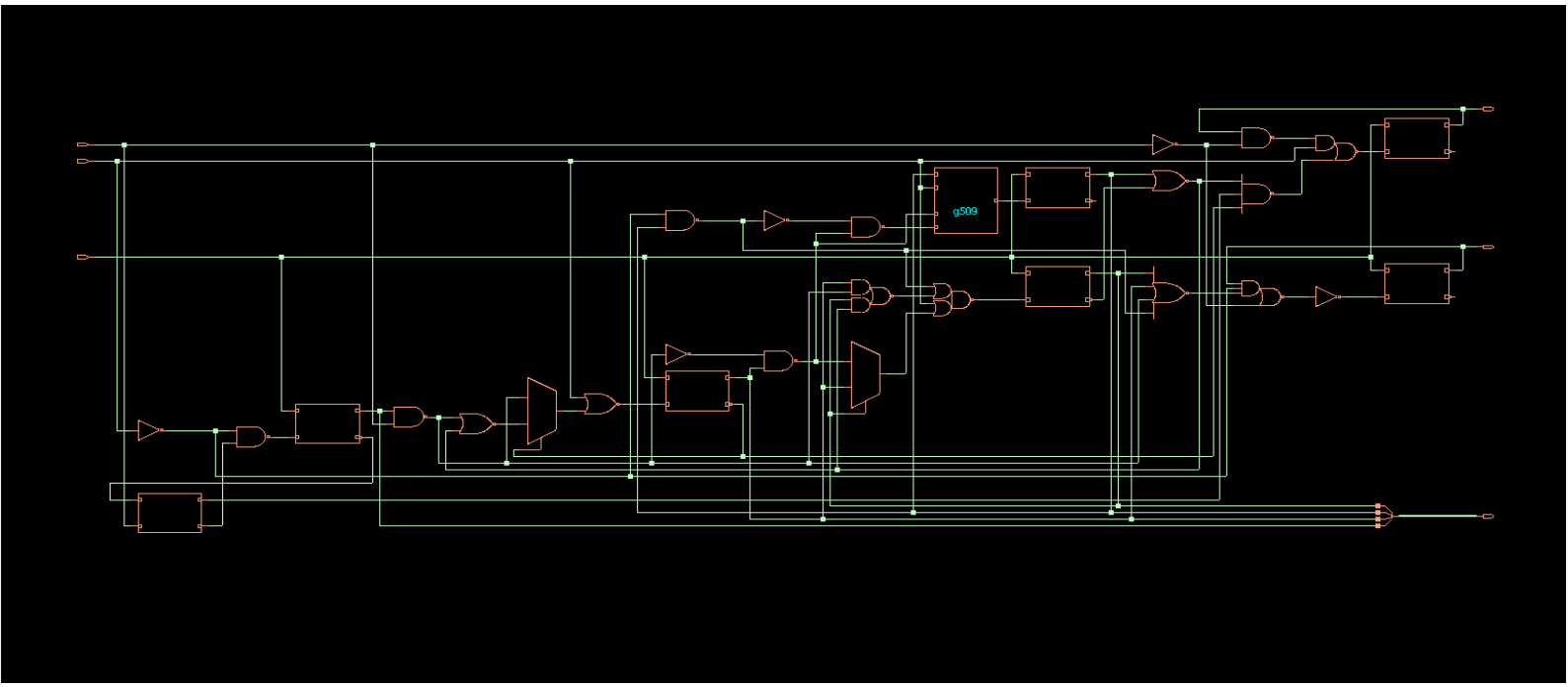
4 Projektovanje fizčke realizacije (lejauta) digitalnog integrisanog kola

Šematski prikaz kola

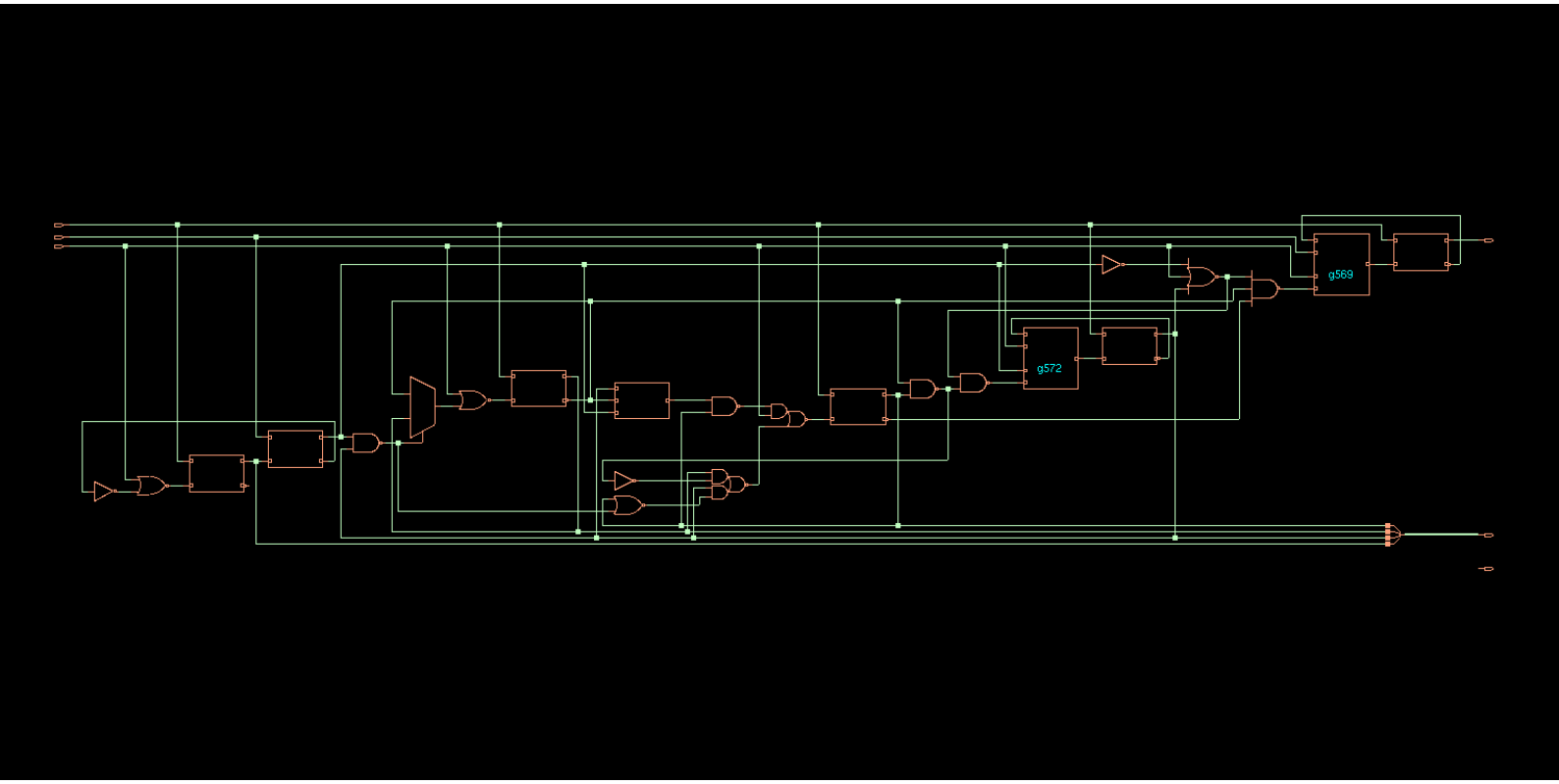
Na slici 3 je dat blok-dijagram sistema realizovan u AMS 0.35µm (C35B4) CMOS tehnologiji. Za ovu svrhu korišćen je RTL Compiler programskog paketa Cadence. Na slikama 4-6 su prikazane električne šeme pojedinačnih brojača.



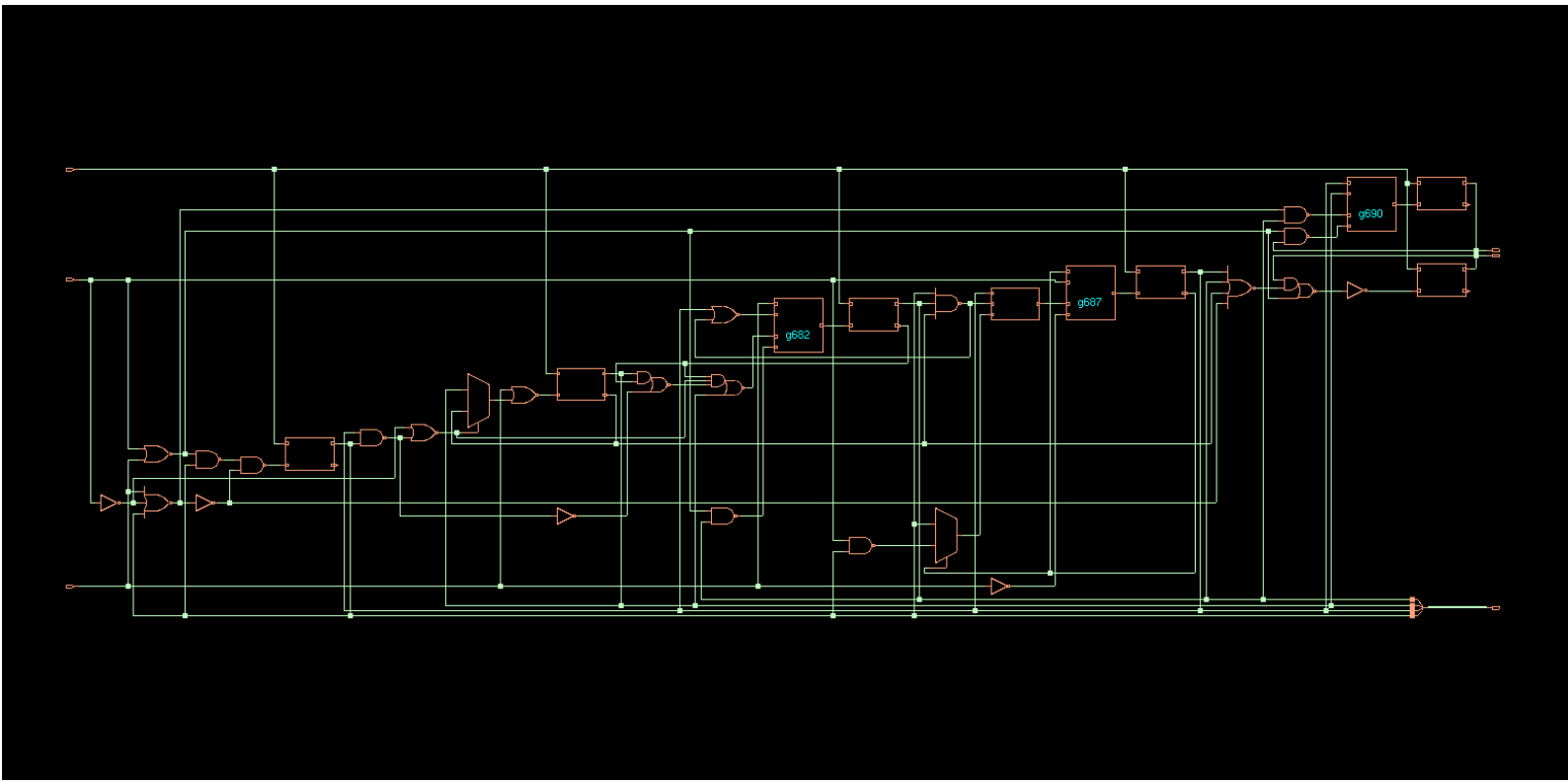
Slika 3: Strukturni model sistema



Slika 4: Električna šema brojača najviše cifre(odula digit3)



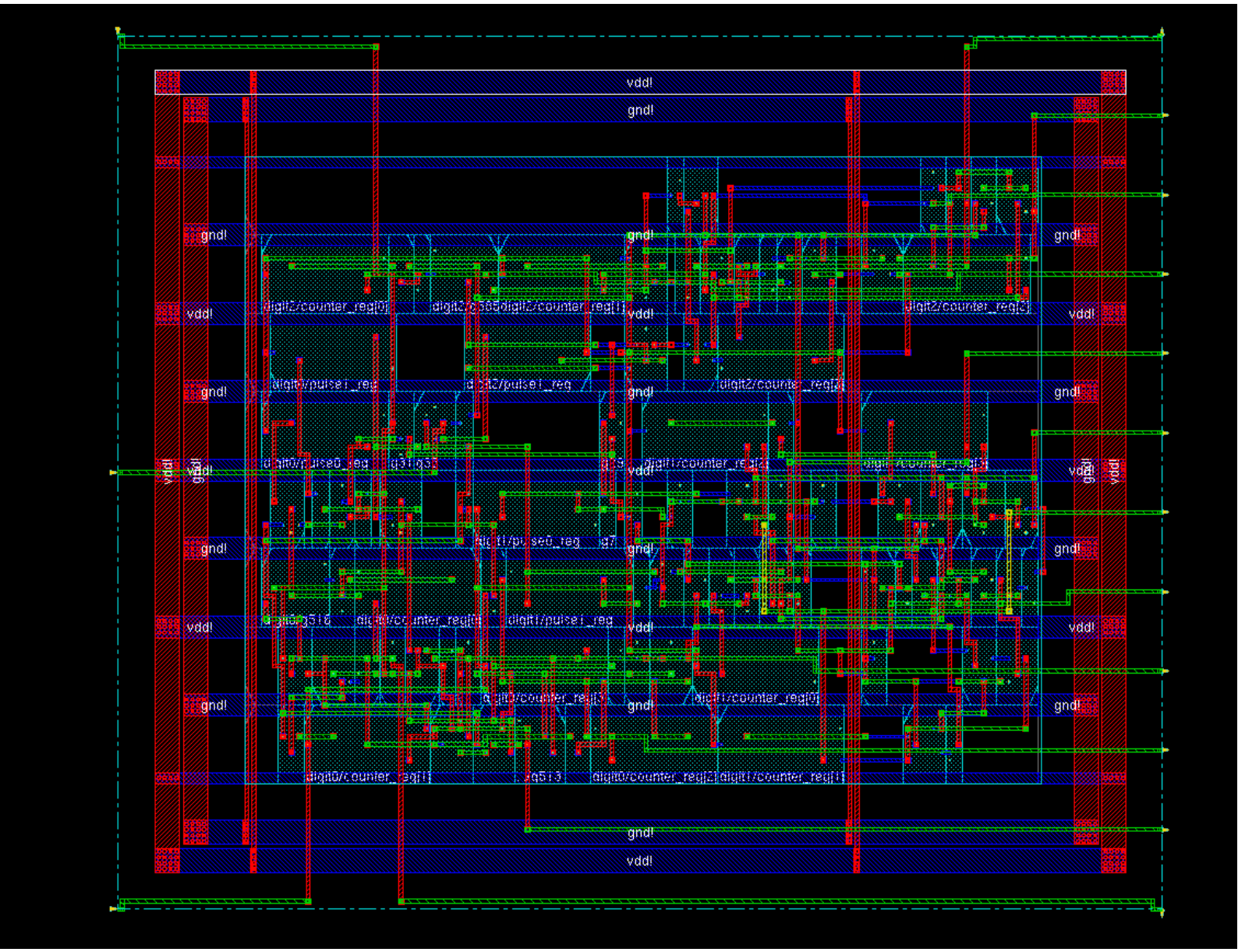
Slika 5: Električna šema brojača druge cifre(modula digit2)



Slika 6: Električna šema brojača prve cifre(modula digit1)

Lejaut (eng. Layout)

Korišćenjem alata *First Encounter* programskog paketa *Cadence* generisana je fizička realizacija projektovanog kola (Slika 7).



Slika 7: Lejaut dobijenog kola

5 Zaključak

Realizovan je trocifreni decimalni brojač prema očekivanjima. Kod prethodno urađene implementacije svaki pojedinačni brojač ima 4 bita od kojih ne koristi sve kombinacije. Zbog toga ukupan broj korišcenih bita je 3x4 = 12, dok bi korišćenjem binarnog brojača do 1000 taj broj bio 10. Ovim se može zaključiti da je običan binarni brojač optimalniji od trostruko decimalnog brojača, međutim implementirani brojač lako se kaskadno proširuje i lako se može povezati na sedmosegmentni displej tako da se rezultat brojanja može lako čitati.
Na osnovu verifikovanog modela zaključujemo još da brojač ispravno radi.