Universitatea Tehnică "Gheorghe Asachi" din Iaşi

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE

ELECTRONICĂ DIGITALĂ

Proiect

Tema: COUNTER BCD – v3

Studenţi:
Aluchienesei Georgiana
Barbaliu Larisa-Bianca
Rosca Ana-Maria-Madalina
Grupa : 1211A

Coordonator:
Asistent doctorand Ionica Pletea

2021

Tema proiectului:

COUNTER BCD – v3

1. Specificaţiile proiectului:

COUNTER BCD – v3

Să se implementeze în FPGA prin descriere în limbaj VHDL, un sistem secvenţial de 4 biţi : cu reset prioritar activ pe 0; cu două intrări de selecţie din care să se stabilească funcţionare de registru paralel sau numărător BCD direct. Frecventa clockului va fi de 1Hz. Afisarea se va face pe Displayul 7 segmente.

Fişierul bitstream rezultat în urma procesului de implementare va fi verificat utilizând placa de dezvoltare BASYS3.

2. Modulul COUNTER BCD – v3

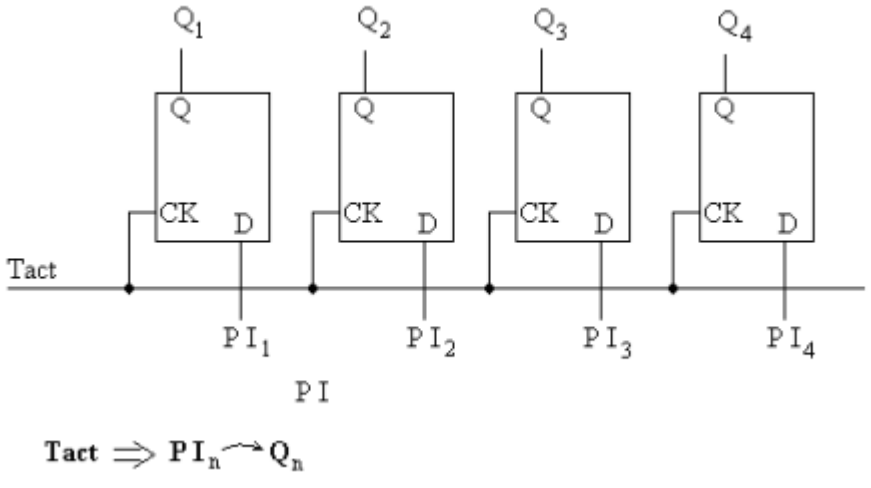
- se va descrie în câteva fraze funcţionalităţile modulului

Registru paralel

Regiştrii sunt circute digitale secvenţiale de stocare a informaţiei sub formă binară ce permit scrierea sau citirea simultană a tuturor biţilor.

Din punct de vedere structural, regiştri, sunt constituiţi din mai multe circute de tip flip-flop ce sunt grupate impreuna. Descrierea unui proces secvenţial in limbaj VHDL se face prin urmărirea evenimentelor unui semnal de tact (a unui semnal de ceas pe care îl vom denumi generic clk).

Un registru paralel constă de fapt într-un număr de bistabile de tip D care au tactul în comun.



$\text{Tact} \Rightarrow PI_n \rightsquigarrow Q_n$

La aplicarea tactului fiecare bistabil va încărca data de la propria intrare (fiecare circuit va acţiona conform funcţionării unui bistabil tip D). Se remarcă sincronismul acţiunii de încărcare de unde şi numele ansamblului care este registru.

Intrările D ale bistabilelor constituie intrarea registrului iar ieşirile bistabilelor sunt ieşirea registrului. Intrările bistabilelor poartă numele de intrări paralele, „parallel input", de unde şi notarea PI. Evident, la aplicarea tactului, data de la intrarea PI cu indicele n va fi transferată la ieşirea Q cu acelaşi indice, de unde denumirea de „încărcare paralel"(parallel load).

Counter BCD

Numărătoarele sunt circuite secveţiale de „integrare" a semnalului de clock ce se aplică la intrare. În funcţie de modul de efectuare a resetului se evidenţiaza numărătoare cu reset asincron (atunci când semnalul de reset este prioritat faţă de semnalul de clock) şi numărătoare cu reset sincron (resetarea are loc în situaţia îndeplinirii a două condiţii: activarea semnalului de reset şi apariţia unei eveniment de clock.

Tabelul de succesiune a stărilor pentru numărătorul binar de 4 biţi

Stare	Q3 Q2 Q1 Q0
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	XXXX
11	XXXX
12	XXXX
13	XXXX
14	XXXX
15	XXXX

Un numărător care evoluează ciclic prin exact 10 stări se numește zecimal sau decadic. Dacă cele 10 stări sunt 0, 1, 2, ..., 9 atunci el se mai numește numărător BCD (Binary Coded Decimal).

Bistabilele utilizate în construcția numărătoarelor sunt de tip T realizate de obicei din bistabile JK sau D-MS, cu T = 1 permanent sau uneori cu validarea accesibila în exterior.

3. Metoda de implementare

Utilizarea resurselor: circuit FPGA, limbajul VHDL, programul de sinteză Vivado

4. Descrierea (scurtă) a sistemului de dezvoltare BASYS 3

Placa Basys 3 este o platformă de dezvoltare a circuitelor digitale completă, gata de utilizat, bazată pe cel mai recent Artix®-7 Field Programmable Gate Array (FPGA) de la Xilinx®. Cu FPGA de mare capacitate (număr de piesă Xilinx XC7A35T1CPG236C), costul general scăzut și colecția de porturi USB, VGA și alte porturi, Basys 3 poate găzdui designuri variind de la circuite combinaționale introductive până la circuite secvențiale complexe, cum ar fi procesoare și controlere încorporate.

Artix-7 FPGA este optimizat pentru logica de înaltă performanță și oferă mai multă capacitate, performanță mai mare și mai multe resurse decât modelele anterioare. Caracteristicile Artix-7 35T includ:

- 33.280 de celule logice în 5200 de secțiuni (fiecare secțiune conține patru LUT-uri cu 6 intrări și 8 flip-flop);
- 1.800 Kbiți de RAM bloc rapid;
- Cinci plăci de gestionare a ceasului, fiecare cu o buclă blocată în fază (PLL);
- 90 de felii DSP;
- Viteze interne de ceas care depășesc 450 MHz;
- Convertor analog-digital pe cip (XADC).

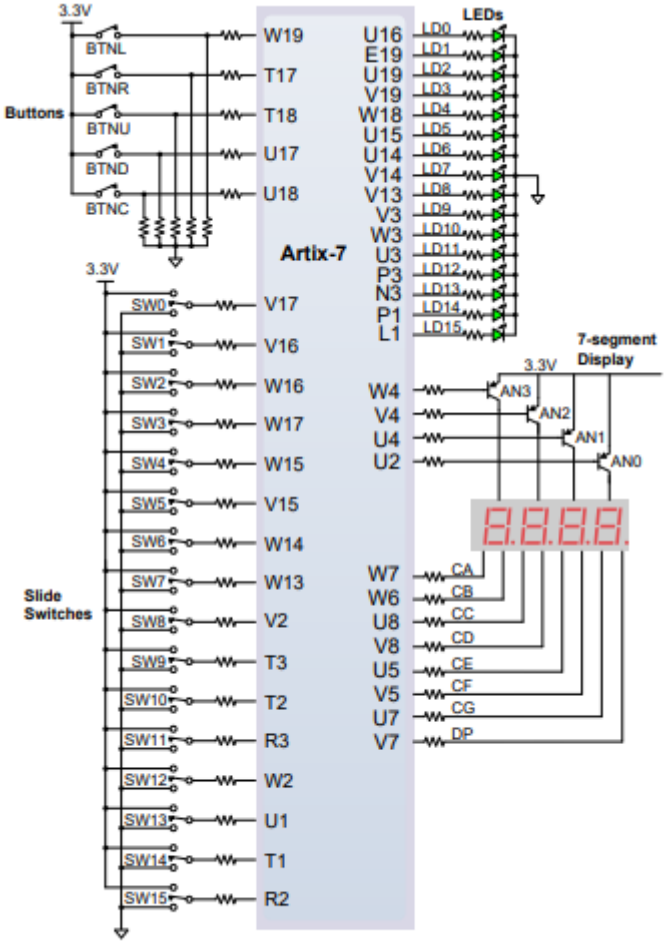


Figure 16. General purpose I/O devices on the Basys 3.

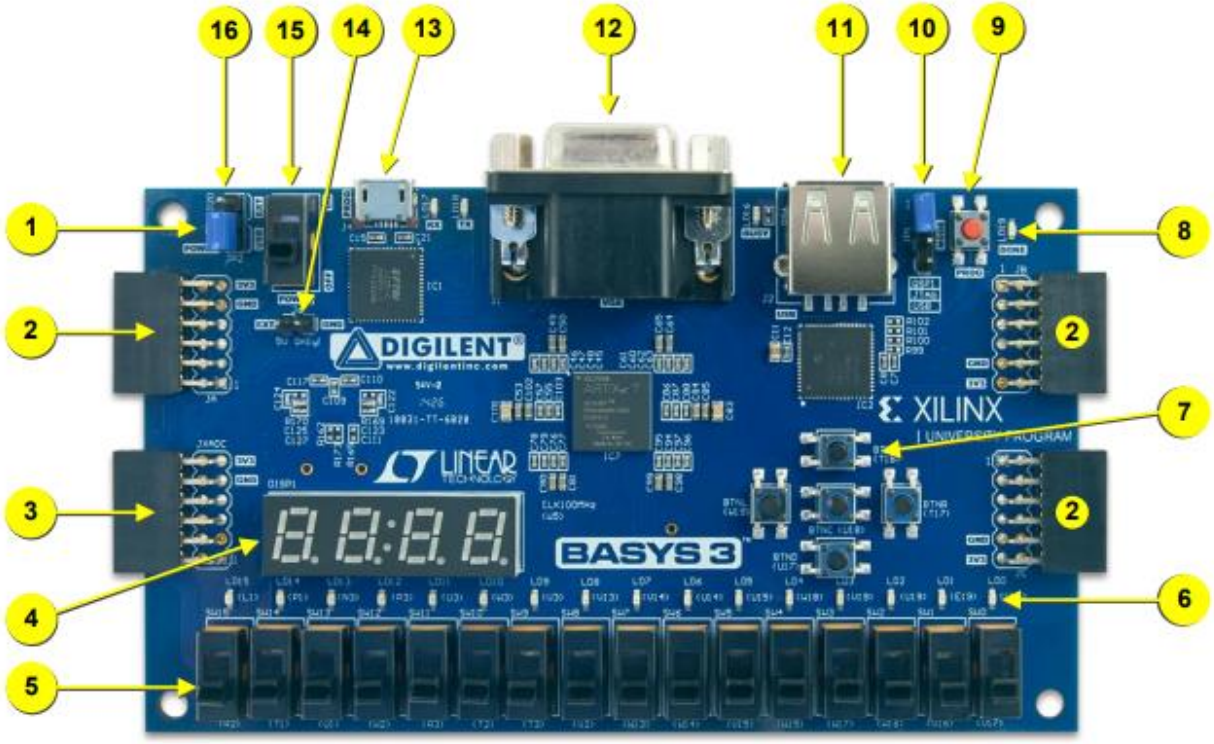


Figure 1. Basys 3 FPGA board with callouts.

Callout	Component Description	Callout	Component Description
1	Power good LED	9	FPGA configuration reset button
2	Pmod port(s)	10	Programming mode jumper
3	Analog signal Pmod port (XADC)	11	USB host connector
4	Four digit 7-segment display	12	VGA connector
5	Slide switches (16)	13	Shared UART/ JTAG USB port
6	LEDs (16)	14	External power connector
7	Pushbuttons (5)	15	Power Switch
8	FPGA programming done LED	16	Power Select Jumper

Table 1. Basys 3 Callouts and component descriptions.

5. Editarea fişierului VHDL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_unsigned.all;

entity bcdcounter is
Port (
counter_clk : in STD_LOGIC; --clock de la placuta

sel_in1 : in STD_LOGIC; --Registru Paralel
sel_in2 : in STD_LOGIC; --Counter BCD
reset_in : in STD_LOGIC;
```

```
D : in STD_LOGIC_VECTOR ( 3 downto 0 );
load : in STD_LOGIC;

seg : out STD_LOGIC_VECTOR(6 downto 0);
an : out STD_LOGIC_VECTOR ( 3 downto 0)

);
end bcdcounter;

architecture Behavioral of bcdcounter is
signal a : std_logic_vector ( 3 downto 0 );
signal count : std_logic_vector(25 downto 0); --folosit pentru divizor de frecventa (100Mhz corespunde la 26 bits)
signal outClock : std_logic := '0'; --clock 1Hz

begin
an(0) <= '0';
an(1) <= '1';
an(2) <= '1';
an(3) <= '1';

process(counter_clk) --DIVIZOR DE FRECVENTA
begin
if(rising_edge(counter_clk)) then
count <= count + 1;
if(count = 50000000 - 1) then
outClock <= not outClock; -- semnal de 1Hz
count <= (OTHERS => '0');
end if;
end if;
end process;

process(outClock,sel_in1, sel_in2, reset_in)
begin
if (reset_in = '0') then
a <= (others => '0');
end if;
if (sel_in1 = '0' and sel_in2 = '1') then -- BCD
if (rising_edge(outClock) and reset_in = '1') then
a <= a + 1;
end if;
if ( a = "1010" or reset_in = '0') then
a <= (others => '0');
end if;

elsif (sel_in1 = '1' and sel_in2 = '0') then --R.P.
if (rising_edge(outClock)) then
if (load = '1' and reset_in = '1') then
a <= D;
elsif(load = '0') then
a <= (others => '0');
```

```
end if;
if (D > 9) then
a <= (others => '0');
end if;
end if;
end if;
end process;

seg(0) <= (a(2) and not(a(1)) and not(a(0))) or (not(a(3)) and not(a(2)) and not(a(1)) and a(0));

seg(1) <= (a(2) and not(a(1)) and a(0)) or (a(2) and a(1) and not(a(0)));

seg(2) <= (not(a(2)) and a(1) and not(a(0)));

seg(3) <= (a(2) and not(a(1)) and not(a(0))) or (a(2) and a(1) and a(0)) or (not(a(3)) and not(a(2)) and not(a(1)) and
a(0));

seg(4) <= a(0) or (a(2) and not(a(1)));

seg(5) <= (not(a(3)) and not(a(2)) and a(0)) or (a(1) and a(0)) or (not(a(3)) and not(a(2)) and a(1));

seg(6) <= (not(a(3)) and not(a(2)) and not(a(1))) or (a(2) and a(1) and a(0));

end Behavioral;
```

6. Editarea fişierului de constrângeri

```
##Clock signal
set_property PACKAGE_PIN W5 [get_ports counter_clk]
set_property IOSTANDARD LVCMOS33 [get_ports counter_clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports counter_clk]

## Switches
set_property PACKAGE_PIN V17 [get_ports {sel_in1}]
set_property IOSTANDARD LVCMOS33 [get_ports {sel_in1}]
set_property PACKAGE_PIN V16 [get_ports {sel_in2}]
set_property IOSTANDARD LVCMOS33 [get_ports {sel_in2}]
set_property PACKAGE_PIN W16 [get_ports {load}]
set_property IOSTANDARD LVCMOS33 [get_ports {load}]
set_property PACKAGE_PIN W17 [get_ports {reset_in}]
set_property IOSTANDARD LVCMOS33 [get_ports {reset_in}]
set_property PACKAGE_PIN W15 [get_ports {D[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[3]}]
set_property PACKAGE_PIN V15 [get_ports {D[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[2]}]
```

```
set_property PACKAGE_PIN W14 [get_ports {D[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[1]}]
set_property PACKAGE_PIN W13 [get_ports {D[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[0]}]

##7 segment display
set_property PACKAGE_PIN W7 [get_ports {seg[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[0]}]
set_property PACKAGE_PIN W6 [get_ports {seg[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[1]}]
set_property PACKAGE_PIN U8 [get_ports {seg[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[2]}]
set_property PACKAGE_PIN V8 [get_ports {seg[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[3]}]
set_property PACKAGE_PIN U5 [get_ports {seg[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[4]}]
set_property PACKAGE_PIN V5 [get_ports {seg[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[5]}]
set_property PACKAGE_PIN U7 [get_ports {seg[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[6]}]

#set_property PACKAGE_PIN V7 [get_ports dp]
#set_property IOSTANDARD LVCMOS33 [get_ports dp]

set_property PACKAGE_PIN U2 [get_ports {an[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[0]}]
set_property PACKAGE_PIN U4 [get_ports {an[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[1]}]
set_property PACKAGE_PIN V4 [get_ports {an[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[2]}]
set_property PACKAGE_PIN W4 [get_ports {an[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[3]}]
```

7. Descrierea paşilor de sinteză şi testarea circuitului rezultat

Pentru rezolvarea problemei propuse, am creat o entitate numită „bcdcounter”, în care am declarat următoarele variabile:

- counter_clk: semnalul de clock de la plăcuță (100MHz);
- sel_in1: prima variabilă de selecție, pentru Registrul Paralel;
- sel_in2: a doua variabilă de selecție, pentru Counter BCD;
- reset_in: variabilă de reset;
- D: vector de 4 biți, folosit pentru citirea celor 4 celule a Registrului Paralel;
- load: variabilă de selecție pentru încărcarea vectorului D pe display;
- seg: vector de 7 biți, folosit pentru afișarea pe 7 Segment Display;
- an: vector de 4 biți, folosit pentru selecția unei singure cifre dintre cele 4 ale display-ului.

Din cauza incapacității ochiului uman de a percepe un semnal vizual care depășește 60 de Hz, semnalul de clock oferit de plăcuța Basys 3 are o frecvență mult prea mare pentru a fi receptată (100MHz). Așadar, se conturează necesitatea implementării unui divizor de frecvență care să primească ca variabilă de intrare clock-ul de la plăcuță și să genereze ca variabilă de ieșire un semnal de clock cu valoare de 1Hz (stocată în variabila outClock).

În conceperea instrucțiunilor în limbaj VHDL, s-a ținut cont de prioritatea resetului (reset_in) care este activ pe valoarea 0 logic. Prin urmare, dacă switch-ul corespondent variabilei de reset este pornit (generăm un semnal de 1 logic), permitem rularea programului, în caz contrar, afișăm valoarea 0 pe display.

În funcție de valoarea selecțiilor, programul rulează astfel:

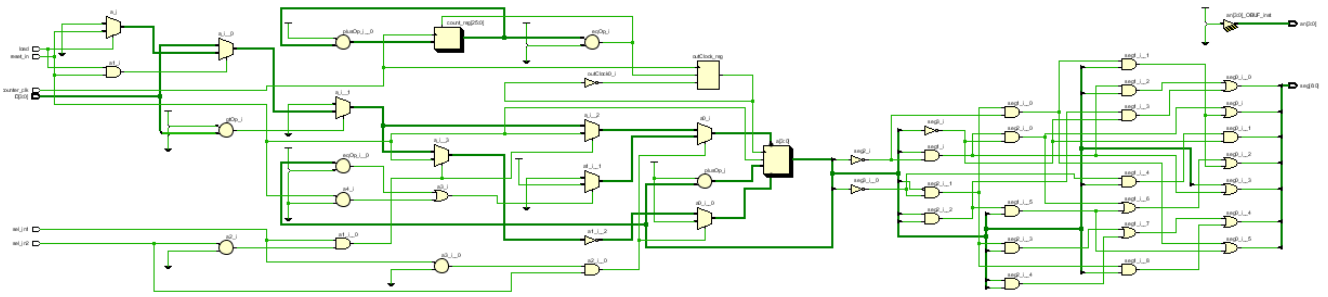
- pentru sel_in1 = ’0’ si sel_in2 = ’1’: testează funcționalitatea Numaratorului in Binar;
- pentru sel_in1 = ’1’ si sel_in2 = ’0’: testeaza funcționalitatea Registrului Paralel, condiționată de valoarea variabilei load :
  - load = ’1’: afișăm valoarea citită de vectorul D pe 7 Segment Display;
  - load = ’0’: inhibăm afișarea valorii citite de vectorul D pe 7 Segment Display, afișând valoarea 0 în schimb.

Fiecare bit din vectorul D corespunde unui switch de pe placa de dezvoltare Basys 3, iar în funcție de valoarea acestor biți, vom afișa cifre de la 0 la 9.

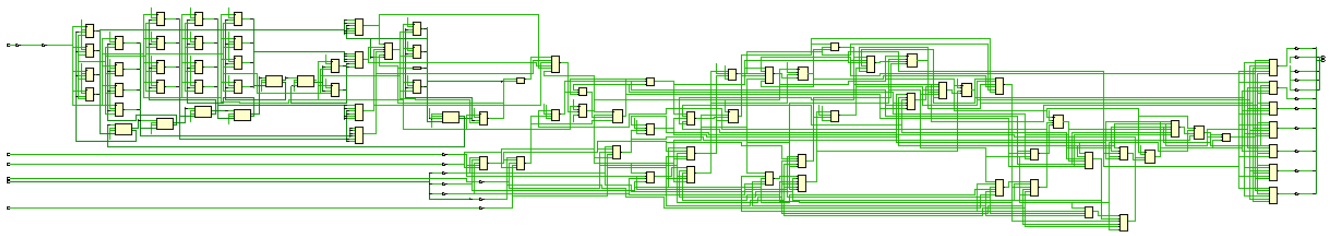
Pe durata utilizării Counter-ului BCD, vom numara de la 0 până la valoarea 9, cu ajutorul semnalului de clock divizat (de 1Hz), iar în momentul în care se încearcă afișarea valorii 10, Counter-ul se resetează la 0.

Pentru realizarea proiectului s-a folosit mediul de lucru Vivado și s-a utilizat circuitul FPGA „xc7a35tcpg236-1”. Mai apoi, design-ul elaborat a fost încărcat pe plăcuța Basys 3.

- Schema sintetizată în RTL Analysis

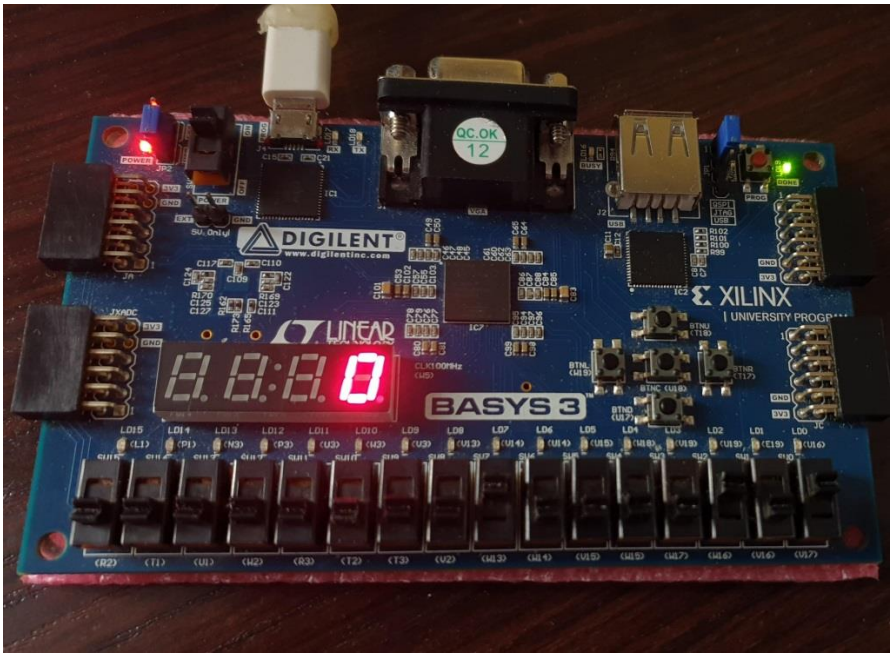


- Schematic obținut în Synthesized Design

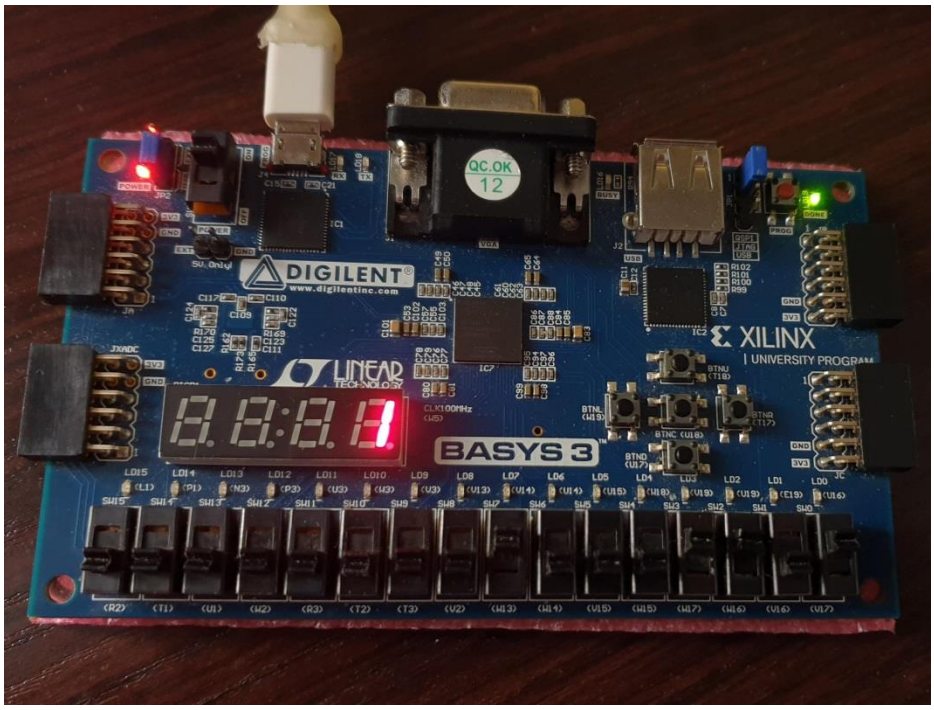


Am validat funcționarea corectă a proiectului, prin următoarele exemple, pentru Registrul Paralel:

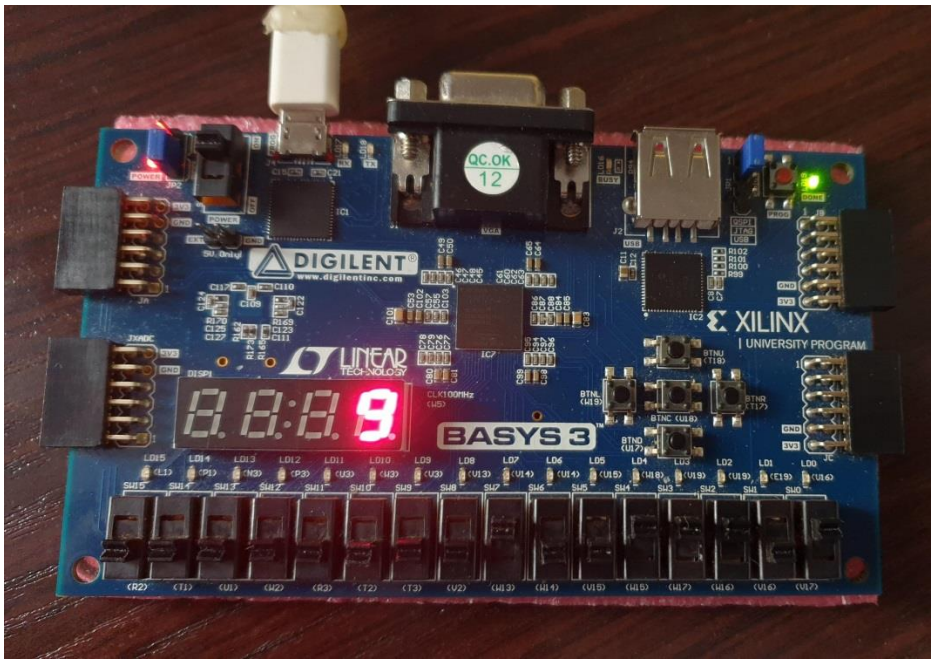
- sel_in1=’1’, sel_in2=’0’, load =’1’ si D=“0001”, dar reset_in = ‘0’



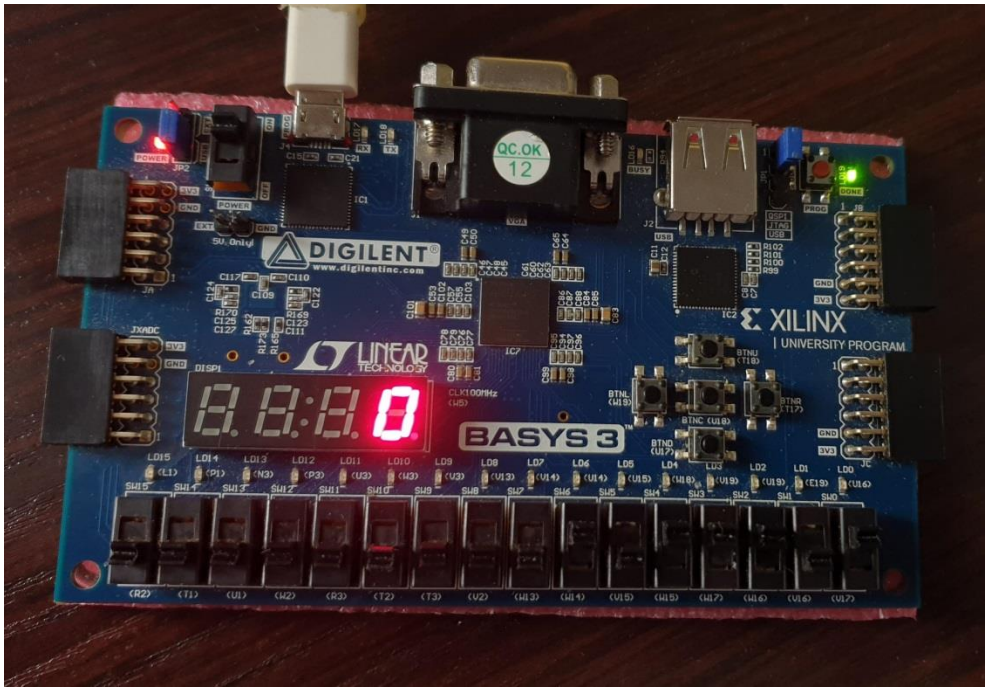
- sel_in1=’1’, sel_in2=’0’, load =’1’ si D=”0001”, dar reset_in=’1’



- sel_in1=’1’, sel_in2=’0’, load =’1’ si D=”1001”, dar reset_in=’1’



- sel_in1=’1’, sel_in2=’0’, load =’1’ si D=”1010”, dar reset_in=’1’



Pentru testarea Counter-ului BCD se va prezenta un scurt videoclip.

8. Concluzii

În concluzie, prin acest proiect am reușit să înțelegem funcționalitatea numărătorului binar, al registrului paralel și ne-am dezvoltat cunoștințele in limbajul VHDL. Lucrând interactiv, am reușit să înțelegem mai bine codul scris în Vivado pentru a vedea în mod practic ceea ce a fost implementat.

Bibliografie:

1. VHDL Reference Manual, http://www.ics.uci.edu/~jmoorkan/vhdlref/Synario%20VHDL%20Manual.pdf

2. BASYS 3 Reference Manual, https://reference.digilentinc.com/reference/programmable-logic/basys-3/reference-manual