

**Universitatea Tehnică “Gheorghe Asachi” din Iaşi**

**FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE**

**ELECTRONICĂ DIGITALĂ**

**Proiect**

**Tema: Counter – Up-Down – v5**

Studenţi:

Avramescu Andrei

Popa Alexandru-Șerban

Musteață Raluca-Elena

Grupa : 1212B

Coordonator:

Asistent doctorand Ionica Pletea

**2023**

**Tema proiectului:**

**Counter – Up-Down – v5**

**1. Specificaţiile proiectului:**

COUNTER – Up-Down – v5

Să se implementeze în FPGA prin descriere în limbaj VHDL, un sistem secvenţial pe 4 biţi: cu reset asincron activ pe 1; cu o intrare de selecţie din care să se stabilească funcţionarea de registru serie sau numărător in cod gray. Clockul va avea perioada de o secunda.

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

**2. Modulul Counter – Up-Down – v5**

Modulul este implementat astfel încât placa FPGA, va aprinde ledurile corespunzătoare valorii binare, după fiecare tact aplicat, având astfel funcția de a selecta tipul de funcționare a acestuia, dintre registru serie sau numărător în cod gray. De asemenea, pentru numărătorul în cod gray, putem alege și ordinea numărării. Acesta mai conține funcția de a reseta valoarea afișată, adică de a stinge ledurile aprinse.

**3. Metoda de implementare**

În cadrul acestui proiect vor fi utilizate următoarele resurse:

1) circuitul FPGA xc7a35tcpg236-1, din familia ARTIX 7 fiind produs de XILINX, acest circuit fiind utilizat prin intermediul plăcii de dezvoltare Basys3

2) tool-ul de sinteză VIVADO

3) limbajul de descriere VHDL (Very High Speed Integrated Circuit Hardware Description Language)

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

Placa Basys3 este un circuit integrat digital configurabil, bazat pe tehnologia Artix-7, de la Xilinx. Datorită costului redus și multiplelor porturi de intrare-ieșire, circuitul este perfect pentru începători, deoarece acesta facilitează dezvoltarea de la circuite combinaționale simple până la procesoare și controlere incorporate.

**5. Editarea fişierului VHDL**

*library IEEE;*

*use IEEE.STD\_LOGIC\_1164.ALL;*

*use IEEE.std\_logic\_unsigned.all;*

*entity my\_counter is*

*Port ( counter\_clk : in STD\_LOGIC;*

*ud\_in : in STD\_LOGIC;*

*reset\_in : in STD\_LOGIC;*

*count\_mode : in STD\_LOGIC;*

*Data\_out : out STD\_LOGIC\_VECTOR (3 downto 0)*

*);*

*end my\_counter;*

*architecture Behavioral of my\_counter is*

*signal a : std\_logic\_vector(3 downto 0);*

*signal a\_gray : std\_logic\_vector(3 downto 0);*

*signal a\_selectat : std\_logic\_vector(3 downto 0);*

*begin*

*process(counter\_clk, ud\_in, reset\_in)*

*variable n : integer range 0 to 1000000000;*

*begin*

*if reset\_in = '1' then --1*

*a <= (others => '0');*

*a\_gray <= (others => '0');*

*elsif counter\_clk'event and counter\_clk='1' then*

*if n < 100000000 then*

*n := n+1;*

*else*

*n := 0;*

*end if;*

*if n <= 50000000 then*

*if ud\_in = '1' then --2*

*a <= a + 1;*

*elsif ud\_in = '0' then*

*a <= a - 1;*

*end if;*

*else*

*a<=a+0;*

*end if;*

*end if;*

*case a is*

*when "0000" => a\_gray <= "0000";*

*when "0001" => a\_gray <= "0001";*

*when "0010" => a\_gray <= "0011";*

*when "0011" => a\_gray <= "0010";*

*when "0100" => a\_gray <= "0110";*

*when "0101" => a\_gray <= "0111";*

*when "0110" => a\_gray <= "0101";*

*when "0111" => a\_gray <= "0100";*

*when "1000" => a\_gray <= "1100";*

*when "1001" => a\_gray <= "1101";*

*when "1010" => a\_gray <= "1111";*

*when "1011" => a\_gray <= "1110";*

*when "1100" => a\_gray <= "1010";*

*when "1101" => a\_gray <= "1011";*

*when "1110" => a\_gray <= "1001";*

*when "1111" => a\_gray <= "1000";*

*end case;*

*end process;*

*a\_selectat <= a when (count\_mode = '0') else a\_gray;*

*Data\_out <= a\_selectat;*

*end Behavioral;*

**6. Editarea fişierului de constrângeri**

Pentru început, am decomentat primele 2 linii de cod din categoria “Clock signal” pentru a avea acces la clock și l-am numit astfel încat să fie în concordanță cu prima noastră intrare “*counter\_clk”.*

*set\_property PACKAGE\_PIN W5 [get\_ports counter\_clk]*

*set\_property IOSTANDARD LVCMOS33 [get\_ports counter\_clk]*

Apoi, am decomentat liniile pentru pinii “V17”, “V16” și “W16” primele 3 switch-uri de la stânga la dreapta, primul reprezentând “*ud\_in”* care semnifică modul în care se dorește a se efectua numărătoarea, (crescător sau descrescător), al doilea fiind “*reset\_in”* pentru a asigura o modalitate de a reseta procesul, iar ultimul “*count\_mode”* prin intermediul căruia stabilim ce tip de numărătoare vrem să avem, în codul binar sau în codul Gray.

*set\_property PACKAGE\_PIN V17 [get\_ports {ud\_in}]*

*set\_property IOSTANDARD LVCMOS33 [get\_ports {ud\_in}]*

*set\_property PACKAGE\_PIN V16 [get\_ports {reset\_in}]*

*set\_property IOSTANDARD LVCMOS33 [get\_ports {reset\_in}]*

*set\_property PACKAGE\_PIN W16 [get\_ports {count\_mode}]*

*set\_property IOSTANDARD LVCMOS33 [get\_ports {count\_mode}]*

În final, am decomentat primele linii de cod din categoria “LEDs” astfel încat să ne asigurăm că avem suficienți biți, 4 - la număr, pentru a reprezenta toate numerele de la 0 la 15.

*set\_property PACKAGE\_PIN U16 [get\_ports {Data\_out[3]}]*

*set\_property IOSTANDARD LVCMOS33 [get\_ports {Data\_out[3]}]*

*set\_property PACKAGE\_PIN E19 [get\_ports {Data\_out[2]}]*

*set\_property IOSTANDARD LVCMOS33 [get\_ports {Data\_out[2]}]*

*set\_property PACKAGE\_PIN U19 [get\_ports {Data\_out[1]}]*

*set\_property IOSTANDARD LVCMOS33 [get\_ports {Data\_out[1]}]*

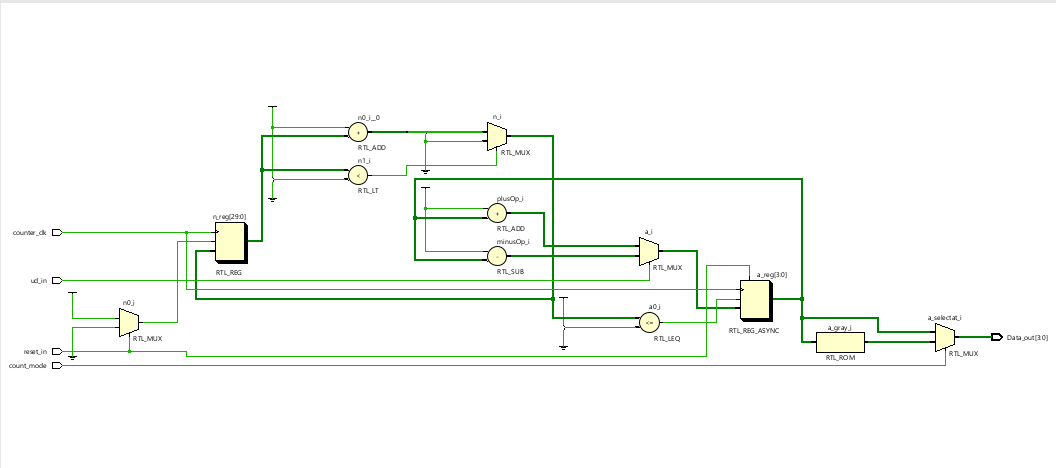
*set\_property PACKAGE\_PIN V19 [get\_ports {Data\_out[0]}]*

*set\_property IOSTANDARD LVCMOS33 [get\_ports {Data\_out[0]}]*

**7. Descrierea paşilor de sinteză şi testarea circuitului rezultat**

Prin intermediul programului Vivado, am creat un nou proiect ce să corespundă plăcii FPGA, am creat un nou fișier sursă, iar pentru fișierul de constrângeri, am preluat din documentația plăcii FPGA fișierului respectiv, și l-am modificat pentru a asigura funcționarea corectă.

Am pornit procesul de sinteză și de implementare și am verificat pentru a nu avea niciun fel de erori după ce cele două procese s-au terminat. În urma acestora, am obținut schema respectivă pentru circuitul nostru:

****

În continuare, am generat fișierul BITSTREAM, necesar pentru programarea plăcii, am inițiat procesul de programare, iar în final am verificat funcționalitatea plăcii.

**8. Concluzii**

În concluzie, putem spune că placa Basys3 este o placă puternică și capabilă pentru diferite proiecte de diverse complexități, motiv pentru care este foarte importantă cunoașterea unui limbaj HDL, mai ales VHDL, fiind unul dintre cele mai folosite.

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