

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

**FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE**

**ELECTRONICĂ DIGITALĂ**

**Proiect**

**Tema: MODUL TIMER0 – v1**

Studenţi:

Cozma Andrei

Vieriu Gabriel

Georgita Adrian

Grupa : 1211A

Coordonator:

Asistent doctorand Ionica Pletea

**2022**

**Tema proiectului:**

**TIMER0 – v1**

**1. Specificaţiile proiectului:**

MODUL TIMER0 – v1

Să se implementeze în FPGA prin descriere în limbaj VHDL, utilizând programul VIVADO, modulul prezentat în figura 1 care este descris prin urmatoarele specificaţii:

a) registrul TMR0 are dimensiunea de 8 biţi

b) sursa de clock clk\_in este cea a plăcii de dezvoltare

c) sursa de clock clk\_ex va fi generată de un buton extern

d) iniţializarea registrului TMR0 de la butoanele externe se face pentru EN=’1’

e) valoarea registrului TMR0 se va afisa în binar prin LED-uri

f) la depăşirea valorii maxime bitul „flag” TMRIF ia valoarea ‚1’



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

**2. Modulul TIMER0\_v1**

Modulul TIMER0\_v1 este un timer/numărător pe 8 biți ce are în componența sa: un registru pe 8 biți, un clock intern și extern precum și un flag pentru semnalarea depășirii valorii maxime. Modulul va incrementa valoarea registrului plecând de la valoarea 0 sau de la o valoare introdusă, de fiecare dată când primește semnal de la clock-ul extern sau intern. Când modulul atinge valoarea maxima de “11111111”, se modifică valoarea flag-ului de overflow, se resetează valoarea și se întrerupe numărătoarea. De asemenea, valoarea precum și flag-ul de overflow pot fi resetate prin acționarea butonului de RESET. Pentru a fi vizibil procesul de numărare atunci când se folosește clock-ul intern, se folosește un divizor de frecvență.

**3. Metoda de implementare**

Implementarea modulului constă în:

* Crearea unui proiect nou în programul de sinteză Vivado
* Scrierea unui cod în limbajul VHDL pentru a descrie comportamentul modulului
* Editarea fișierului cu constrângeri
* Sinteza, Implementarea și generarea bitstream-ului
* Testarea modulului pe placa FPGA

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

BASYS 3 este un sistem de dezvoltare de placi de bază (development board) pentru proiectarea sistemelor programabile cu Xilinx Artix-7 FPGA. Acesta include o serie de componente precum un procesor Microblaze, memorie RAM, module de intrare/ieșire și interfață USB, care permit utilizatorilor să proiecteze și să testeze circuite digitale complexe. Sistemul este compatibil cu un mediu de dezvoltare integrat (IDE) precum Vivado și poate fi utilizat pentru aplicații precum procesarea imagine, comunicații digitale și controlul sistemelor.

**5. Editarea fişierului VHDL**

|  |
| --- |
| library IEEE;  use IEEE.STD\_LOGIC\_1164.ALL;  use IEEE.std\_logic\_unsigned.all;    entity TIMER0 is  port (  clk\_in : in STD\_LOGIC;  clk\_ex : in STD\_LOGIC;  set\_clk : in STD\_LOGIC;  EN : in STD\_LOGIC;  RESET : in STD\_LOGIC;  DATA\_init :in STD\_LOGIC\_VECTOR(7 downto 0);  TMRIF :out STD\_LOGIC;  TMR0 : out STD\_LOGIC\_VECTOR (7 downto 0)  );  end TIMER0;    architecture behavior of TIMER0 is    Signal out\_div : STD\_LOGIC;  Signal s\_bar : STD\_LOGIC;  Signal x\_int : STD\_LOGIC;  Signal y\_int : STD\_LOGIC;  Signal m : STD\_LOGIC;  signal a : std\_logic\_vector(7 downto 0);  Signal overflow : STD\_LOGIC;  begin    process(clk\_in)  variable n : integer range 0 to 1000000000;  begin  if clk\_in'event and clk\_in='1' then  if n < 100000000 then  n := n+1;  else  n := 0;  end if;    if n <= 50000000 then  out\_div <= '1';  else  out\_div <= '0';  end if;  end if;  end process;      s\_bar <= not set\_clk;  x\_int <= out\_div and s\_bar;  y\_int <= clk\_ex and set\_clk;  m <= x\_int or y\_int;    process(m, EN, RESET, DATA\_init)  begin    if RESET = '1' then  a <= (others => '0');  overflow <= '0';  TMRIF <= '0';  elsif EN = '1' then  a <= DATA\_init;  elsif (m'event and m = '1') then  if a = "11111111" then  overflow <= '1';  TMRIF <= '1';  a <= (others => '0');  else  if overflow = '0' then  a <= a + 1;  end if;  end if;  end if;  end process;      TMR0 <= a;  end behavior; |

**6. Editarea fişierului de constrângeri**

|  |
| --- |
| **## Clock signal**  **set\_property PACKAGE\_PIN W5 [get\_ports clk\_in]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports clk\_in]**  **## Switches**  **set\_property PACKAGE\_PIN V17 [get\_ports {DATA\_init[0]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {DATA\_init[0]}]**  **set\_property PACKAGE\_PIN V16 [get\_ports {DATA\_init[1]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {DATA\_init[1]}]**  **set\_property PACKAGE\_PIN W16 [get\_ports {DATA\_init[2]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {DATA\_init[2]}]**  **set\_property PACKAGE\_PIN W17 [get\_ports {DATA\_init[3]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {DATA\_init[3]}]**  **set\_property PACKAGE\_PIN W15 [get\_ports {DATA\_init[4]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {DATA\_init[4]}]**  **set\_property PACKAGE\_PIN V15 [get\_ports {DATA\_init[5]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {DATA\_init[5]}]**  **set\_property PACKAGE\_PIN W14 [get\_ports {DATA\_init[6]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {DATA\_init[6]}]**  **set\_property PACKAGE\_PIN W13 [get\_ports {DATA\_init[7]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {DATA\_init[7]}]**  **set\_property PACKAGE\_PIN T1 [get\_ports {EN}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {EN}]**  **set\_property PACKAGE\_PIN R2 [get\_ports {set\_clk}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {set\_clk}]**  **## LEDs**  **set\_property PACKAGE\_PIN U16 [get\_ports {TMR0[0]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {TMR0[0]}]**  **set\_property PACKAGE\_PIN E19 [get\_ports {TMR0[1]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {TMR0[1]}]**  **set\_property PACKAGE\_PIN U19 [get\_ports {TMR0[2]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {TMR0[2]}]**  **set\_property PACKAGE\_PIN V19 [get\_ports {TMR0[3]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {TMR0[3]}]**  **set\_property PACKAGE\_PIN W18 [get\_ports {TMR0[4]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {TMR0[4]}]**  **set\_property PACKAGE\_PIN U15 [get\_ports {TMR0[5]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {TMR0[5]}]**  **set\_property PACKAGE\_PIN U14 [get\_ports {TMR0[6]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {TMR0[6]}]**  **set\_property PACKAGE\_PIN V14 [get\_ports {TMR0[7]}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {TMR0[7]}]**  **set\_property PACKAGE\_PIN L1 [get\_ports {TMRIF}]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports {TMRIF}]**  **##Buttons**  **set\_property PACKAGE\_PIN U18 [get\_ports clk\_ex]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports clk\_ex]**  **set\_property clock\_dedicated\_route false[get\_nets counter\_clock\_ibuf]**  **set\_property PACKAGE\_PIN U17 [get\_ports RESET]**  **set\_property IOSTANDARD LVCMOS33 [get\_ports RESET]** |

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

1. Sinteza codului VHDL: constă în utilizarea programului de sinteză Vivado pentru a genera un fișier **“**netlist**”** care descrie circuitele logice necesare pentru a implementa modulul pe placa FPGA.
2. Implementarea: constă în plasarea și rutarea componentelor pe FPGA pentru a se potrivi cu configurația fizică a dispozitivului.
3. Generarea fișierului bitstream: constă în utilizarea programului Vivado pentru a genera un fișier care conține informații despre configurația fizică a circuitelor logice
4. Testarea: constă în încărcarea fișierului bitstream pe placa FPGA și testarea funcționalității modulului.

**8. Concluzii**

În concluzie, programul funcționează conform cerinței, permițându-i utilizatorului să folosească un numărător ce poate fi încărcat cu o valoare și care incrementat la fiecare semnal al clock-ului.

Bibliografie:

1. PIC16LF1937, datasheet, <http://ww1.microchip.com/downloads/en/DeviceDoc/41364E.pdf> (pag. 191)

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

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