**МІНІСТЕРСТВО ОСВІТИ І НАУКИ  УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"**

*Кафедра EOM*



**Звіт**

Лабораторна робота 2

з дисципліни «Моделювання комп’ютерних систем»

Варіант 4

Виконав: ст.гр. КІ-202

Коваль О.В.

                                                           Прийняв:

Козак Н.Б.

Львів – 2024

# Тема роботи:

# Структурний опис цифрового автомата Перевірка роботи автомата за допомогою стенда Elbert V2 – Spartan 3A FPGA

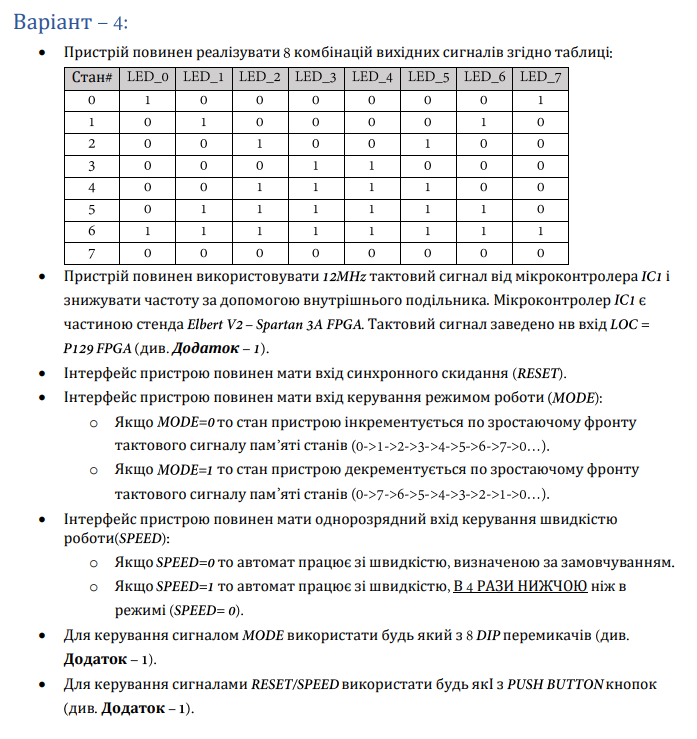
# Мета роботи:

На базі стенда Elbert V2 – Spartan 3A FPGA реалізувати цифровий автомат світлових ефектів згідно наступних вимог:

1. Інтерфейс пристрою та функціонал реалізувати згідно отриманого варіанту завдання
2. Логіку переходів реалізувати з використанням мови опису апаратних засобів VHDL. Заборонено використовувати оператори if, switch, for, when
3. Логіку формування вихідних сигналів реалізувати з використанням мови опису апаратних засобів VHDL. Заборонено використовувати оператори if, switch, for, when
4. Згенерувати Schematic символи для VHDL описів логіки переходів та логіки формування вихідних сигналів
5. Зінтегрувати всі компоненти (логіку переходів логіку формування вихідних сигналів та пам'ять станів) в єдину систему за допомогою ISE WebPACK Schematic Capture. Пам'ять станів реалізувати за допомогою графічних компонентів з бібліотеки
6. Промоделювати роботу окремих частин автомата та автомата в цілому за допомогою симулятора ISim
7. Інтегрувати створений автомат зі стендом Elbert V2 – Spartan 3A FPGA (додати подільник частоти для вхідного тактового сигналу призначити фізичні виводи на FPGA)
8. Згенерувати BIT файл та перевірити роботу за допомогою стенда Elbert V2 – Spartan 3A FPGA
9. Підготувати і захистити звіт

**Завдання:**

Мій номер по списку 10, тому номер варіанту 4.



# Виконання роботи:

1. У середовищі Xilinx ISE створив новий проєкт. Налаштував цільову FPGA, обрала інструменти для синтезу і симуляції.

2. Додав VHDL файл OutputLogic до проєкту та імплементував інтерфейс логіки формування вихідних сигналів, а також логічні вирази для формування кожного вихідного сигналу, залежно від поточного стану автомата.

# (VHDL OutputLogic)

# ----------------------------------------------------------------------------------

# -- Company:

# -- Engineer:

# --

# -- Create Date: 20:07:40 03/27/2024

# -- Design Name:

# -- Module Name: out\_logic\_intf - out\_logic\_arch

# -- Project Name:

# -- Target Devices:

# -- Tool versions:

# -- Description:

# --

# -- Dependencies:

# --

# -- Revision:

# -- Revision 0.01 - File Created

# -- Additional Comments:

# --

# ----------------------------------------------------------------------------------

# library IEEE;

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

# -- Uncomment the following library declaration if using

# -- arithmetic functions with Signed or Unsigned values

# --use IEEE.NUMERIC\_STD.ALL;

# -- Uncomment the following library declaration if instantiating

# -- any Xilinx primitives in this code.

# --library UNISIM;

# --use UNISIM.VComponents.all;

# entity out\_logic\_intf is

# port ( IN\_BUS : in std\_logic\_vector(2 downto 0) ;

# OUT\_BUS : out std\_logic\_vector(7 downto 0));

# end out\_logic\_intf;

# architecture out\_logic\_arch of out\_logic\_intf is

# begin

# OUT\_BUS(0) <= ((not(IN\_BUS(2)) and not(IN\_BUS(1)) and not(IN\_BUS(0))) or ((IN\_BUS(2)) and (IN\_BUS(1)) and not (IN\_BUS(0))));

# OUT\_BUS(1) <= ((not(IN\_BUS(2)) and not(IN\_BUS(1)) and(IN\_BUS(0))) or ((IN\_BUS(2)) and not (IN\_BUS(1)) and (IN\_BUS(0))) or ((IN\_BUS(2)) and (IN\_BUS(1)) and not (IN\_BUS(0))));

# OUT\_BUS(2) <= ((not(IN\_BUS(2)) and(IN\_BUS(1)) and not(IN\_BUS(0))) or ((IN\_BUS(2)) and not (IN\_BUS(1)) and not (IN\_BUS(0))) or ((IN\_BUS(2)) and not (IN\_BUS(1)) and (IN\_BUS(0))) or ((IN\_BUS(2)) and (IN\_BUS(1)) and not (IN\_BUS(0))));

# OUT\_BUS(3) <= ((not(IN\_BUS(2)) and(IN\_BUS(1)) and(IN\_BUS(0))) or ((IN\_BUS(2)) and not (IN\_BUS(1)) and not (IN\_BUS(0))) or ((IN\_BUS(2)) and not (IN\_BUS(1)) and (IN\_BUS(0))) or ((IN\_BUS(2)) and (IN\_BUS(1)) and not (IN\_BUS(0))));

# OUT\_BUS(4) <= ((not(IN\_BUS(2)) and(IN\_BUS(1)) and(IN\_BUS(0))) or ((IN\_BUS(2)) and not (IN\_BUS(1)) and not (IN\_BUS(0))) or ((IN\_BUS(2)) and not (IN\_BUS(1)) and (IN\_BUS(0))) or ((IN\_BUS(2)) and (IN\_BUS(1)) and not (IN\_BUS(0))));

# OUT\_BUS(5) <= ((not(IN\_BUS(2)) and(IN\_BUS(1)) and not(IN\_BUS(0))) or ((IN\_BUS(2)) and not (IN\_BUS(1)) and not (IN\_BUS(0))) or ((IN\_BUS(2)) and not (IN\_BUS(1)) and (IN\_BUS(0))) or ((IN\_BUS(2)) and (IN\_BUS(1)) and not (IN\_BUS(0))));

# OUT\_BUS(6) <= ((not(IN\_BUS(2)) and not(IN\_BUS(1)) and(IN\_BUS(0))) or ((IN\_BUS(2)) and not (IN\_BUS(1)) and (IN\_BUS(0))) or ((IN\_BUS(2)) and (IN\_BUS(1)) and not (IN\_BUS(0))));

# OUT\_BUS(7) <= ((not(IN\_BUS(2)) and not(IN\_BUS(1)) and not(IN\_BUS(0))) or ((IN\_BUS(2)) and (IN\_BUS(1)) and not (IN\_BUS(0))));

# end out\_logic\_arch;

3. За допомогою симулятора Isim провів моделювання роботи схеми при всіх можливих комбінаціях сигналів на входах.

# 

Рис.1: виконання роботи схеми

4. Додав до проєкту VHDL файл Transition\_Logic, в якому реалізував логіку формування переходів.

# (VHDL TransitionLogic)

----------------------------------------------------------------------------------

-- Company:

-- Engineer:

--

-- Create Date: 21:25:57 03/27/2024

-- Design Name:

-- Module Name: transition\_logic\_intf - transition\_logic\_arch

-- Project Name:

-- Target Devices:

-- Tool versions:

-- Description:

--

-- Dependencies:

--

-- Revision:

-- Revision 0.01 - File Created

-- Additional Comments:

--

----------------------------------------------------------------------------------

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--use IEEE.NUMERIC\_STD.ALL;

-- Uncomment the following library declaration if instantiating

-- any Xilinx primitives in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

entity transition\_logic\_intf is

Port ( CUR\_STATE : in std\_logic\_vector(2 downto 0);

MODE : in std\_logic;

NEXT\_STATE : out std\_logic\_vector(2 downto 0)

);

end transition\_logic\_intf;

architecture transition\_logic\_arch of transition\_logic\_intf is

begin

NEXT\_STATE(0) <= (not(MODE) and not(CUR\_STATE(2)) and not(CUR\_STATE(1)) and not(CUR\_STATE(0))) or

(not(MODE) and not(CUR\_STATE(2)) and CUR\_STATE(1) and not(CUR\_STATE(0))) or

(not(MODE) and CUR\_STATE(2) and not(CUR\_STATE(1)) and not(CUR\_STATE(0))) or

(not(MODE) and CUR\_STATE(2) and CUR\_STATE(1) and not(CUR\_STATE(0))) or

( MODE and not(CUR\_STATE(2)) and CUR\_STATE(1) and not(CUR\_STATE(0))) or

( MODE and CUR\_STATE(2) and not(CUR\_STATE(1)) and not(CUR\_STATE(0))) or

( MODE and CUR\_STATE(2) and CUR\_STATE(1) and not(CUR\_STATE(0))) or

( MODE and not(CUR\_STATE(2)) and not(CUR\_STATE(1)) and not(CUR\_STATE(0))) ;

NEXT\_STATE(1) <= (not(MODE) and not(CUR\_STATE(2)) and not(CUR\_STATE(1)) and CUR\_STATE(0)) or

(not(MODE) and not(CUR\_STATE(2)) and CUR\_STATE(1) and not(CUR\_STATE(0))) or

(not(MODE) and CUR\_STATE(2) and not(CUR\_STATE(1)) and CUR\_STATE(0)) or

(not(MODE) and CUR\_STATE(2) and CUR\_STATE(1) and not(CUR\_STATE(0))) or

( MODE and not(CUR\_STATE(2)) and CUR\_STATE(1) and CUR\_STATE(0)) or

( MODE and CUR\_STATE(2) and not(CUR\_STATE(1)) and not(CUR\_STATE(0))) or

( MODE and CUR\_STATE(2) and CUR\_STATE(1) and CUR\_STATE(0)) or

( MODE and not(CUR\_STATE(2)) and not(CUR\_STATE(1)) and not(CUR\_STATE(0))) ;

NEXT\_STATE(2) <= (not(MODE) and not(CUR\_STATE(2)) and CUR\_STATE(1) and CUR\_STATE(0)) or

(not(MODE) and CUR\_STATE(2) and not(CUR\_STATE(1)) and not(CUR\_STATE(0))) or

(not(MODE) and CUR\_STATE(2) and not(CUR\_STATE(1)) and CUR\_STATE(0)) or

(not(MODE) and CUR\_STATE(2) and CUR\_STATE(1) and not(CUR\_STATE(0))) or

( MODE and CUR\_STATE(2) and not(CUR\_STATE(1)) and CUR\_STATE(0)) or

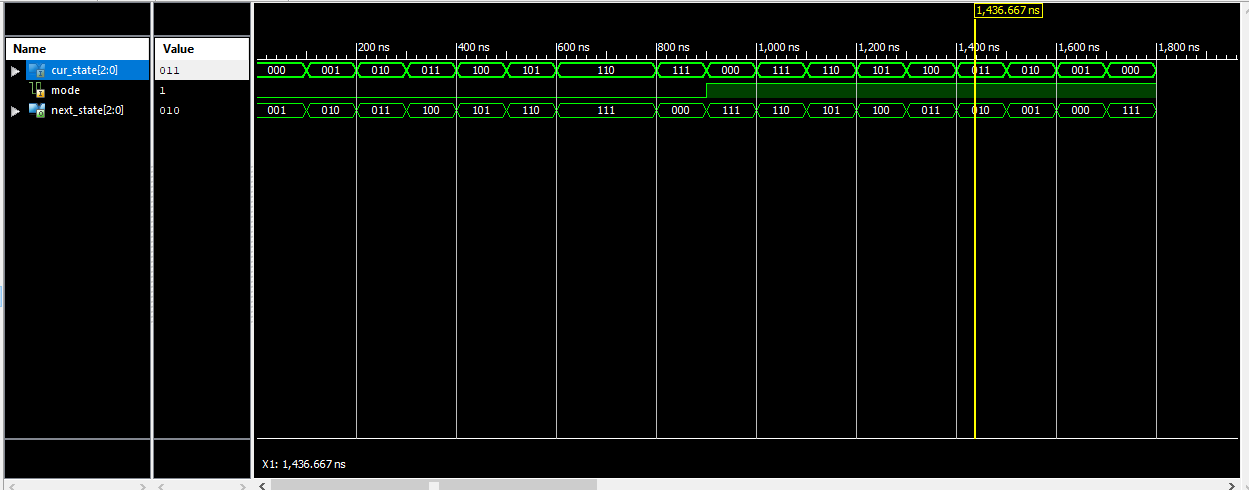
( MODE and CUR\_STATE(2) and CUR\_STATE(1) and not(CUR\_STATE(0))) or

( MODE and CUR\_STATE(2) and CUR\_STATE(1) and CUR\_STATE(0)) or

( MODE and not(CUR\_STATE(2)) and not(CUR\_STATE(1)) and not(CUR\_STATE(0))) ;

end transition\_logic\_arch;

5. За допомогою симулятора Isim провів моделювання роботи схеми при всіх можливих комбінаціях сигналів на входах.



# Рис.2: промодельована робота схеми формування вихідних сигналів з усіма можливими наборами сигналів

6. Додав до проєкту Schematic файл LightController, виконав для нього команду Set as Top Module. Згенерував Schematic символи для файлів OutputLogic і TransitionLogic. Використовуючи новостворені символи та елементи з бібліотеки, реалізував у файлі Light\_Controller.sch пам’ять стану автомата.

# 

Рис.3: згенеровані Schematic символи

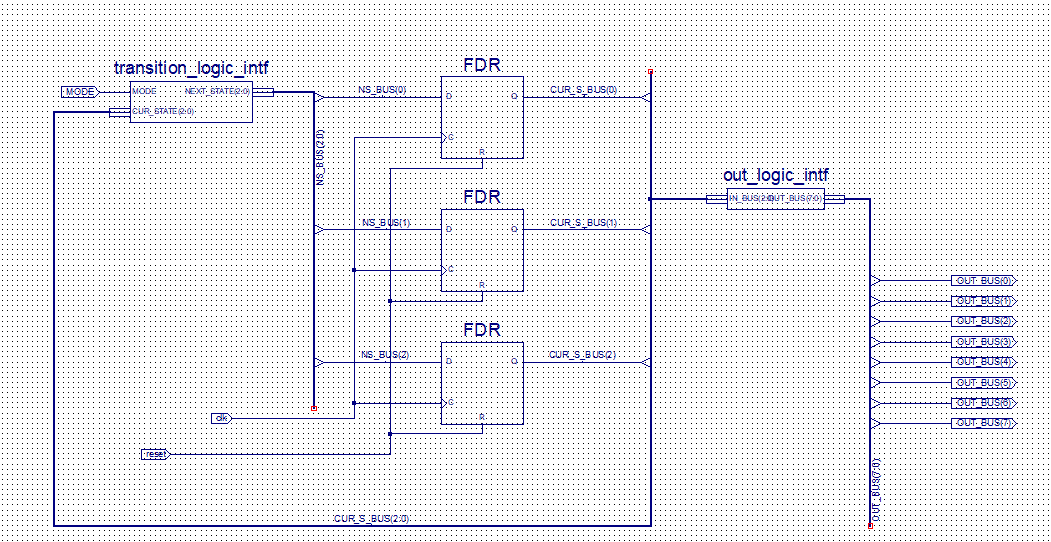


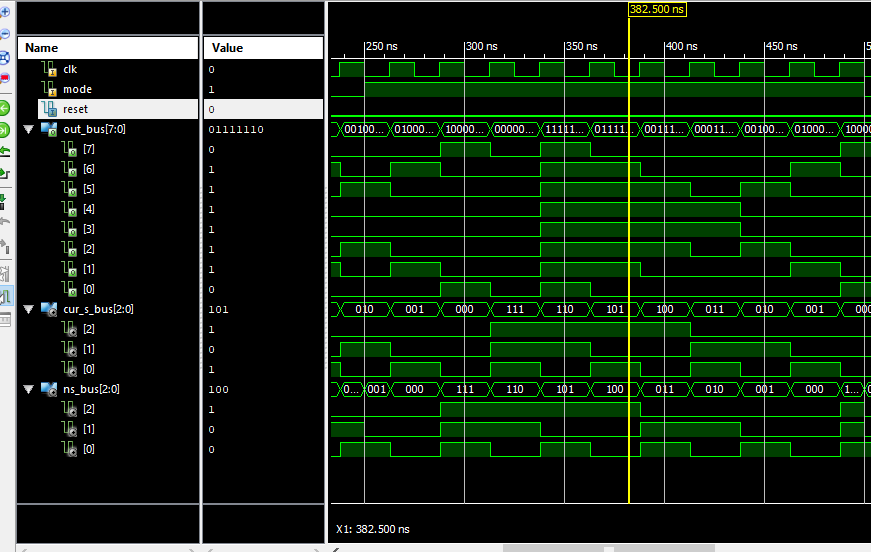
Рис.4: схема LightController

7. За допомогою симулятора Isim провів моделювання роботи схеми при всіх можливих комбінаціях сигналів на входах.

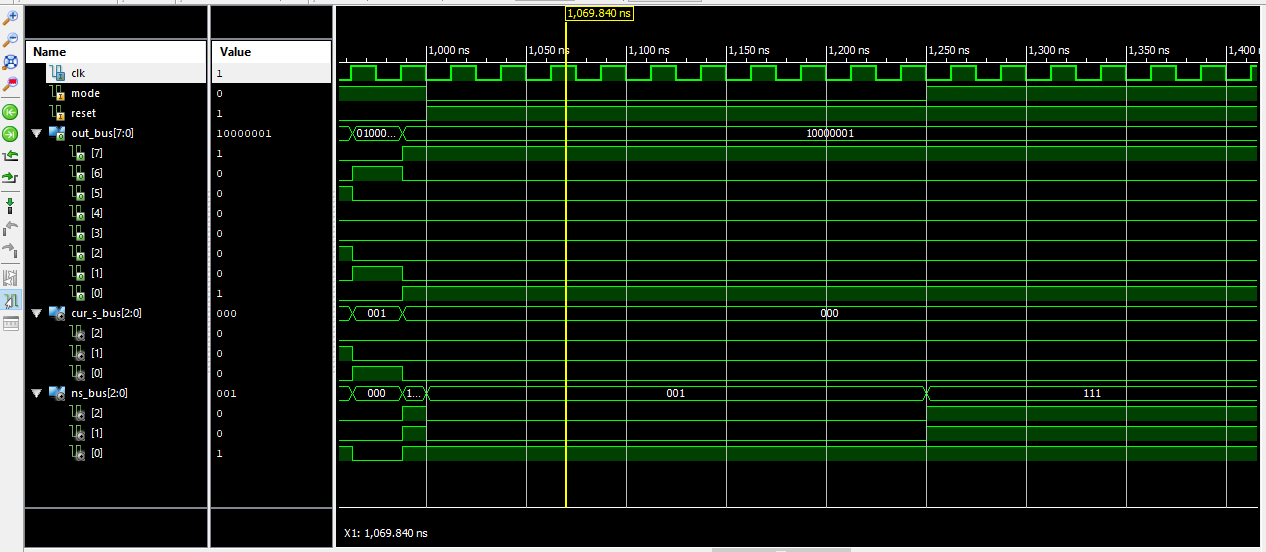
При MODE 0:



MODE 1:



RESET:



8. Додав до проєкту Schematic файл TopLevel, виконав для нього команду Set as Top Module. Згенерував Schematic символ для файлу LightController. Використовуючи новостворений символ та елементи з бібліотеки, реалізував у файлі подільник вхідної частоти та логіку сигналу SPEED.

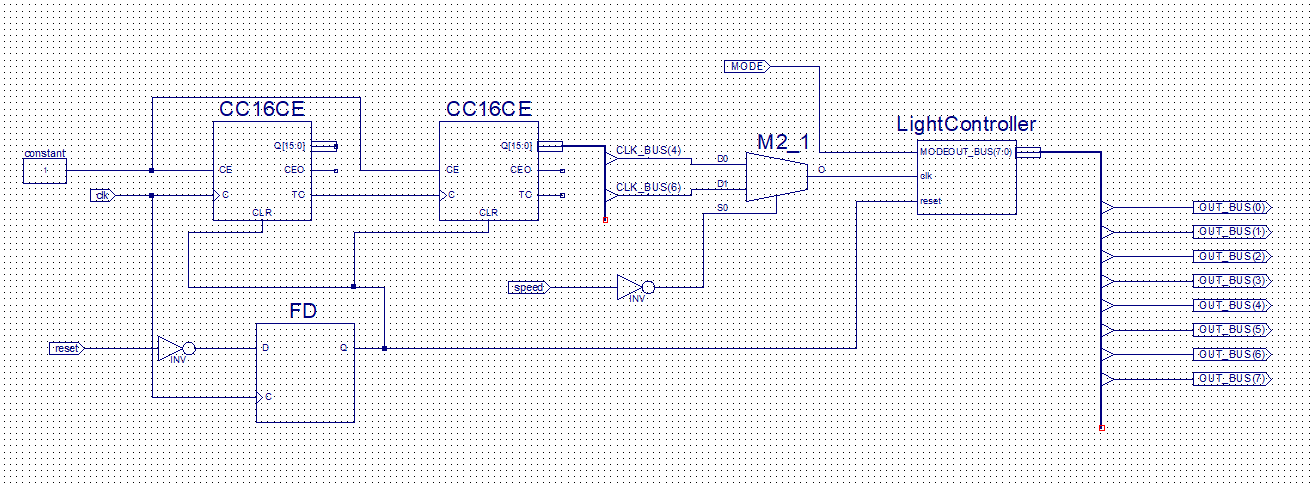
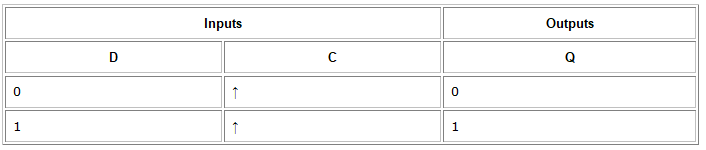
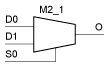


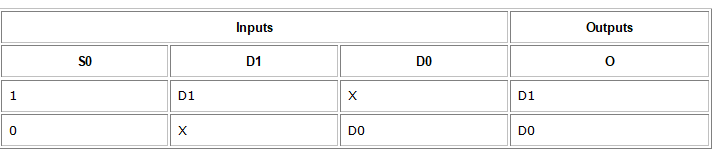
Рис.5: Schematic файл для кінцевої схеми.(TopLevel.sch)





Схематичне зображення D-тригера та його таблиця істинності





Схематичне зображення мультиплексора 2-1 та його таблиця істинності

9. За допомогою симулятора Isim провів моделювання роботи схеми з різними значеннями сигналів MODE, RESET та SPEED при подачі на вхід CLOCK тактового сигналу 12 MHz.

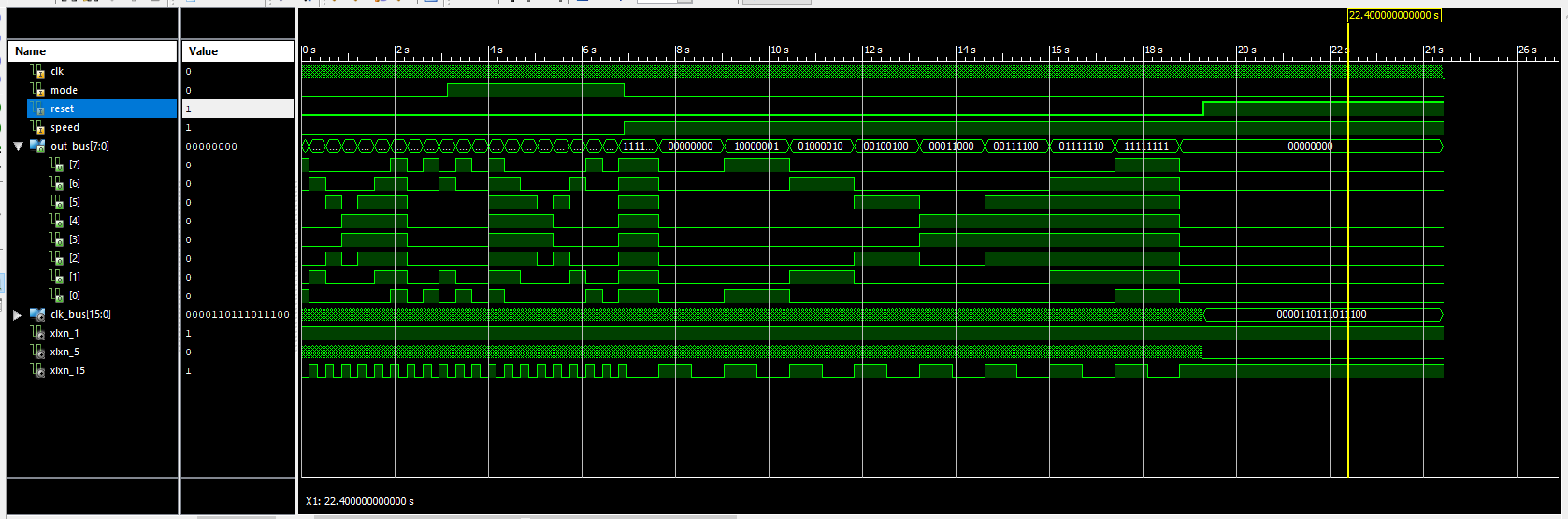


Рис.6: Діаграма проведеної симуляції для TopLevel

Автомат працює відповідно до завдання.

10. Додав до проєкту Constraint файл та призначив виводам схеми фізичні виводи цільової FPGA.

#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++#

# This file is a .ucf for ElbertV2 Development Board #

# To use it in your project : #

# \* Remove or comment the lines corresponding to unused pins in the project #

# \* Rename the used signals according to the your project #

#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++#

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

# UCF for ElbertV2 Development Board #

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

CONFIG VCCAUX = "3.3" ;

# Clock 12 MHz

NET "Clk" LOC = P129 | IOSTANDARD = LVCMOS33 | PERIOD = 12MHz;

####################################################################################################

# LED

####################################################################################################

NET "OUT\_BUS(0)" LOC = P46 | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

NET "OUT\_BUS(1)" LOC = P47 | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

NET "OUT\_BUS(2)" LOC = P48 | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

NET "OUT\_BUS(3)" LOC = P49 | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

NET "OUT\_BUS(4)" LOC = P50 | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

NET "OUT\_BUS(5)" LOC = P51 | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

NET "OUT\_BUS(6)" LOC = P54 | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

NET "OUT\_BUS(7)" LOC = P55 | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

####################################################################################################

# DP Switches

####################################################################################################

NET "MODE" LOC = P70 | PULLUP | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

####################################################################################################

# Switches

####################################################################################################

NET "reset" LOC = P80 | PULLUP | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

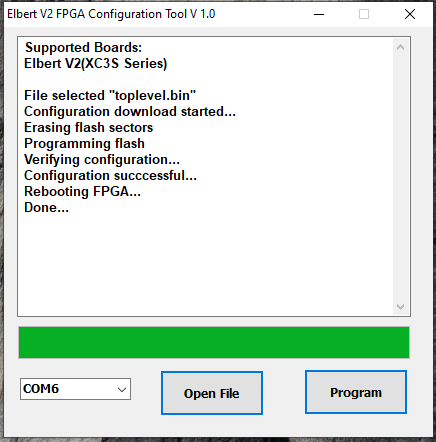
NET "speed" LOC = P79 | PULLUP | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

11. Генерую BIT файл, який названий *TopLevel.bin* для цільової FPGA. Для цього послідовно запускаю процеси Synthesize – XST, Implement Design та Generate Programming File.



Рис.7: згенерований бінарний файл

12. Перевіряю роботу на стенді. Програмування стенду згенерованим *TopLevel.bin* файлом.



**Висновок:** під час виконання цієї лабораторної роботи я реалізував цифровий автомат світлових ефектів у середовищі Xilinx ISE і стендом Elbert V2 - Spartan 3A FPGA. Я реалізував схему автомату та провів симуляцію його роботи.