МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра ЕОМ

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 автоматсвітлових ефектів згідно наступних вимог:

1. Інтерфейс пристрою та функціонал реалізувати згідно отриманоговаріанту завдання
2. Логіку переходів реалізувати з використанням мови опису апаратнихзасобів VHDL. Заборонено використовувати оператори if, switch, for, when
3. Логіку формування вихідних сигналів реалізувати з використанням мовиопису апаратних засобів VHDL. Заборонено використовувати оператори if, switch, for, when
4. Згенерувати Schematic символи для VHDL описів логіки переходів талогіки формування вихідних сигналів
5. Зінтегрувати всі компоненти (логіку переходів логіку формування вихідних сигналів та пам'ять станів) в єдину систему за допомогою ISEWebPACK Schematic Capture. Пам'ять станів реалізувати за допомогоюграфічних компонентів з бібліотеки
6. Промоделювати роботу окремих частин автомата та автомата в цілому задопомогою симулятора ISim
7. Інтегрувати створений автомат зі стендом Elbert V2 – Spartan 3A FPGA (додати подільник частоти для вхідного тактового сигналу призначитифізичні виводи на FPGA)
8. Згенерувати BIT файл та перевірити роботу за допомогою стенда ElbertV2 – Spartan 3A FPGA
9. Підготувати і захистити звіт

Завдання:

Мій номер по списку 10, тому номер варіанту 4.

Варіант – 4:

- Пристрій повинен реалізувати 8 комбінацій вихідних сигналів згідно таблиці:

Стан#	LED_0	LED_1	LED_2	LED_3	LED_4	LED_5	LED_6	LED_7
0	1	0	0	0	0	0	0	1
1	0	1	0	0	0	0	1	0
2	0	0	1	0	0	1	0	0
3	0	0	0	1	1	0	0	0
4	0	0	1	1	1	1	0	0
5	0	1	1	1	1	1	1	0
6	1	1	1	1	1	1	1	1
7	0	0	0	0	0	0	0	0

- Пристрій повинен використовувати *12MHz* тактовий сигнал від мікроконтролера *IC1* і знижувати частоту за допомогою внутрішнього подільника. Мікроконтролер *IC1* є частиною стенда *Elbert V2 – Spartan 3A FPGA*. Тактовий сигнал заведено нв вхід *LOC = P129 FPGA* (див. ***Додаток – 1***).
- Інтерфейс пристрою повинен мати вхід синхронного скидання (*RESET*).
- Інтерфейс пристрою повинен мати вхід керування режимом роботи (*MODE*):
  - Якщо *MODE=0* то стан пристрою інкрементується по зростаючому фронту тактового сигналу пам'яті станів (0->1->2->3->4->5->6->7->0…).
  - Якщо *MODE=1* то стан пристрою декрементується по зростаючому фронту тактового сигналу пам'яті станів (0->7->6->5->4->3->2->1->0…).
- Інтерфейс пристрою повинен мати однорозрядний вхід керування швидкістю роботи(*SPEED*):
  - Якщо *SPEED=0* то автомат працює зі швидкістю, визначеною за замовчуванням.
  - Якщо *SPEED=1* то автомат працює зі швидкістю, В 4 РАЗИ НИЖЧОЮ ніж в режимі (*SPEED= 0*).
- Для керування сигналом *MODE* використати будь який з 8 *DIP* перемикачів (див. **Додаток – 1**).
- Для керування сигналами *RESET/SPEED* використати будь якІ з *PUSH BUTTON* кнопок (див. **Додаток – 1**).

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.

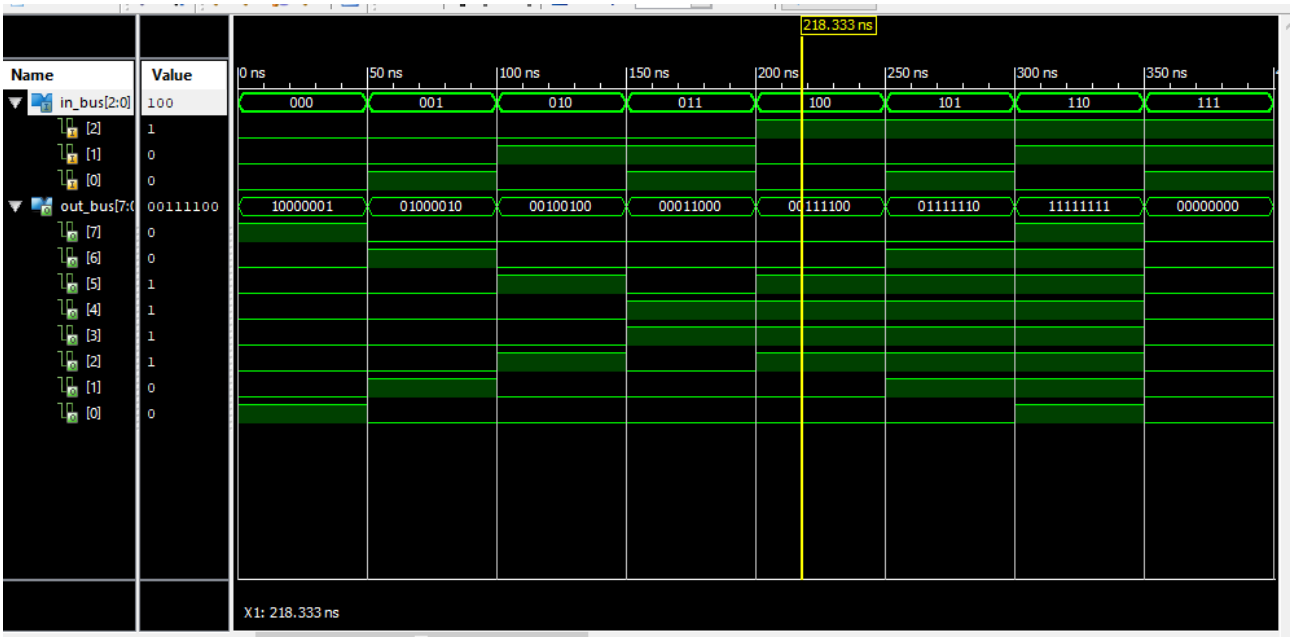


Рис.1: виконання роботи схеми

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.

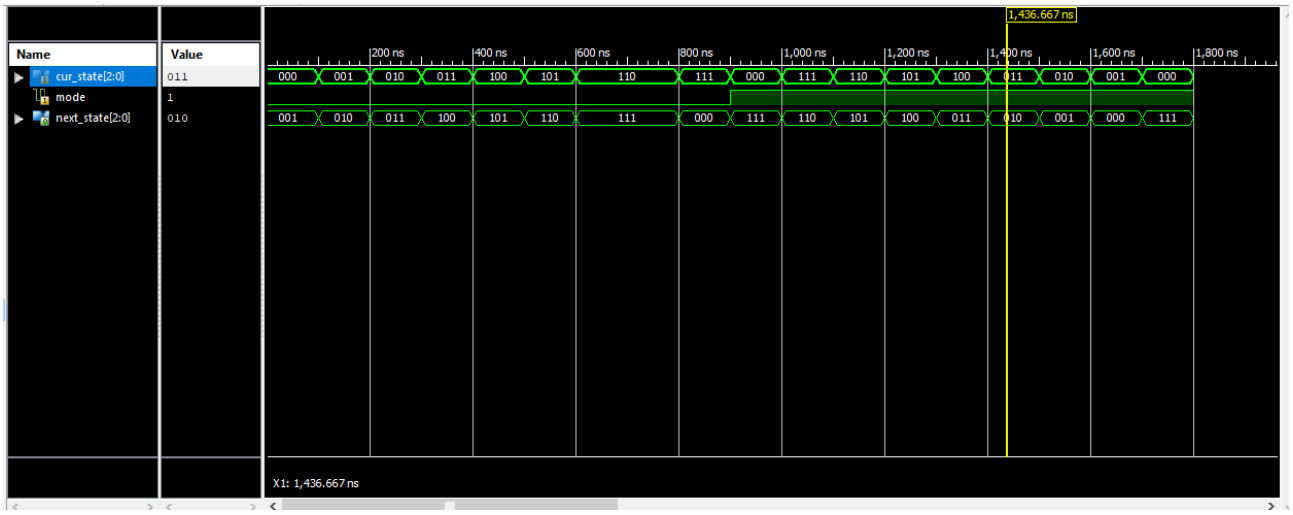


Рис.2: промодельована робота схеми формування вихідних сигналів з усіма можливими наборами сигналів

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.

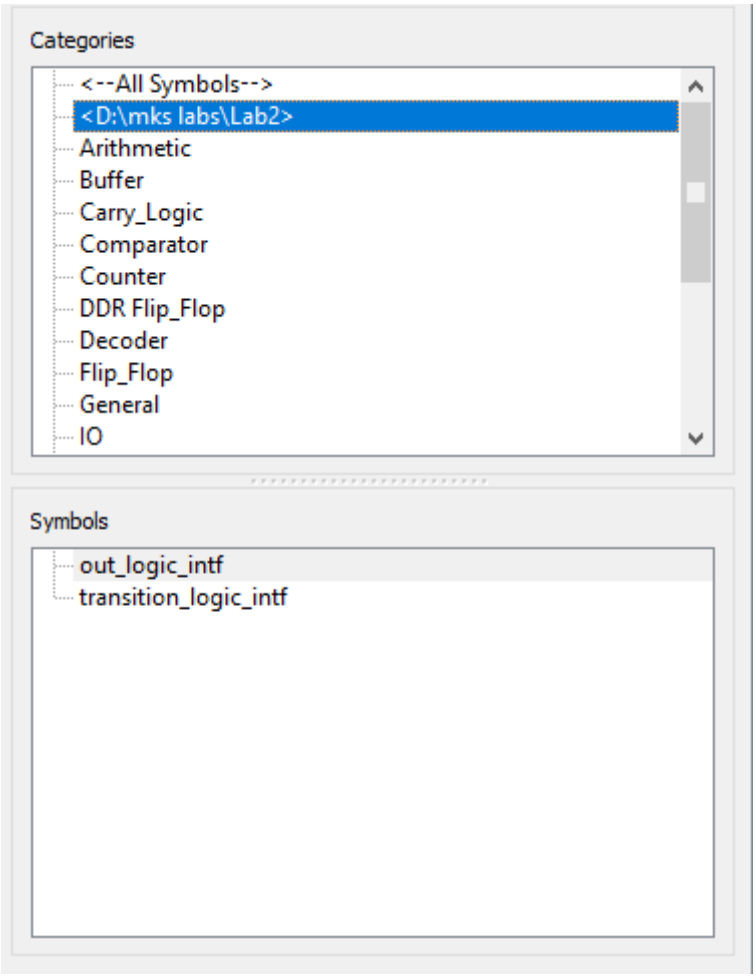


Рис.3: згенеровані Schematic символи

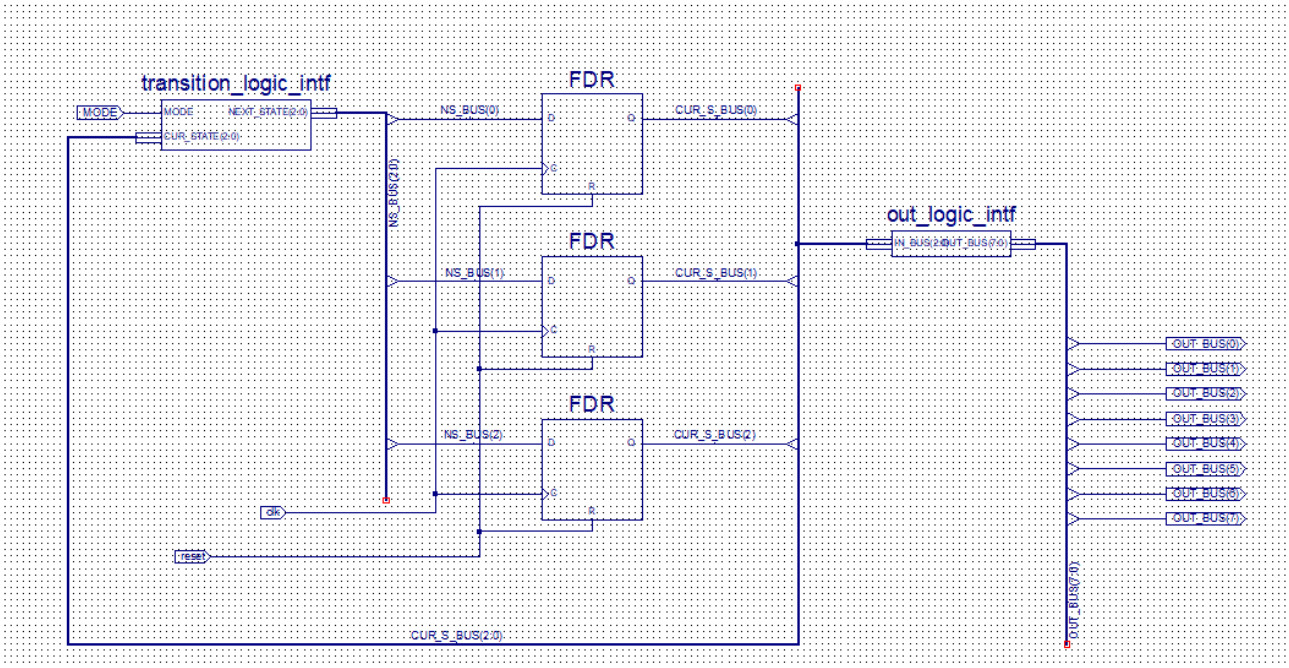
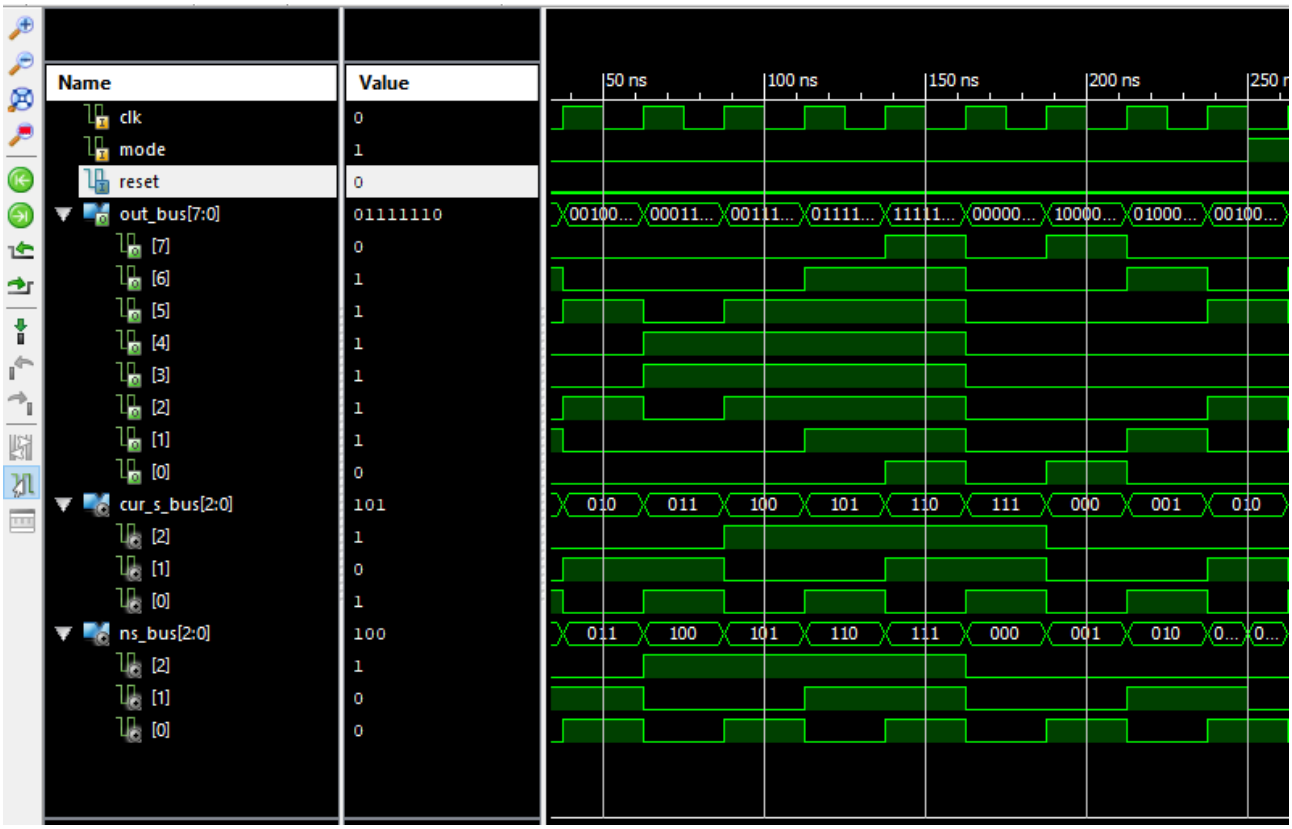


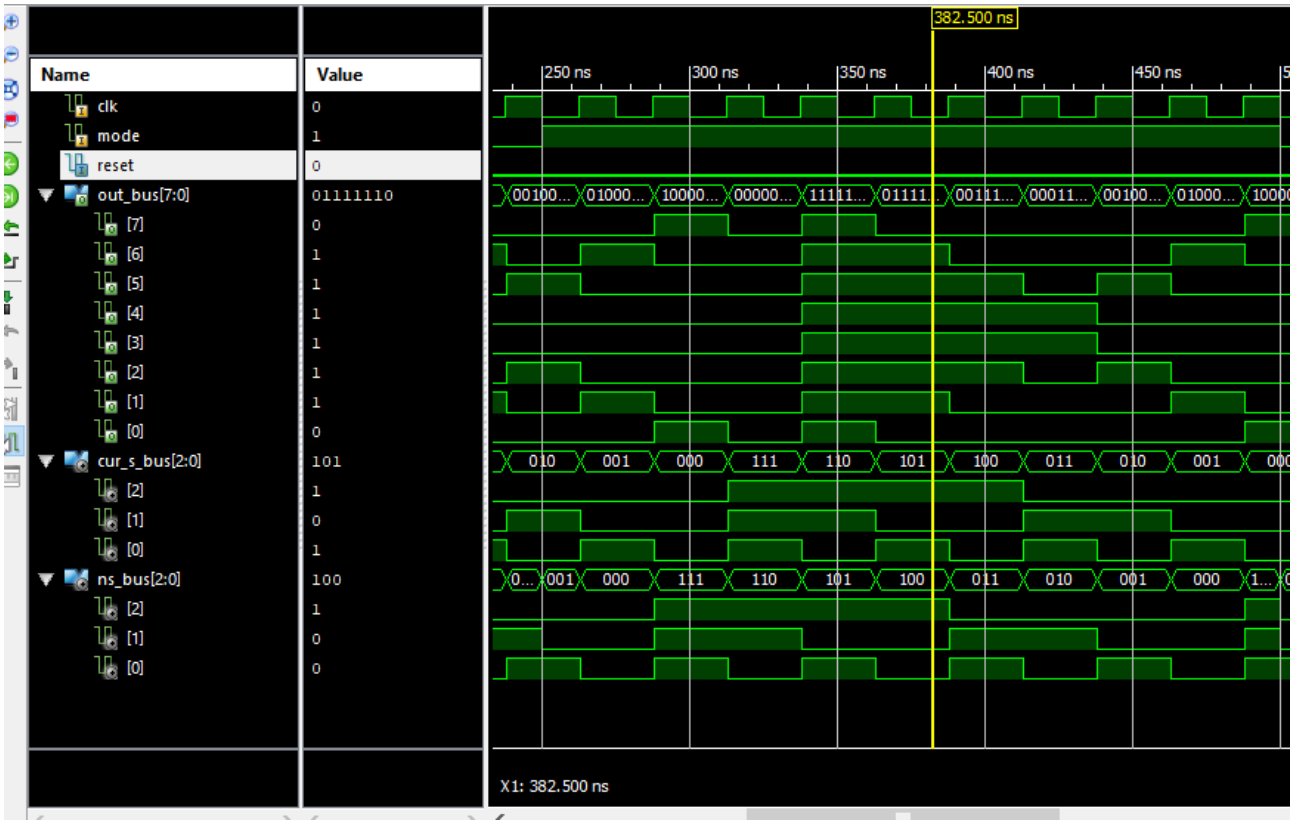
Рис.4: схема LightController

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

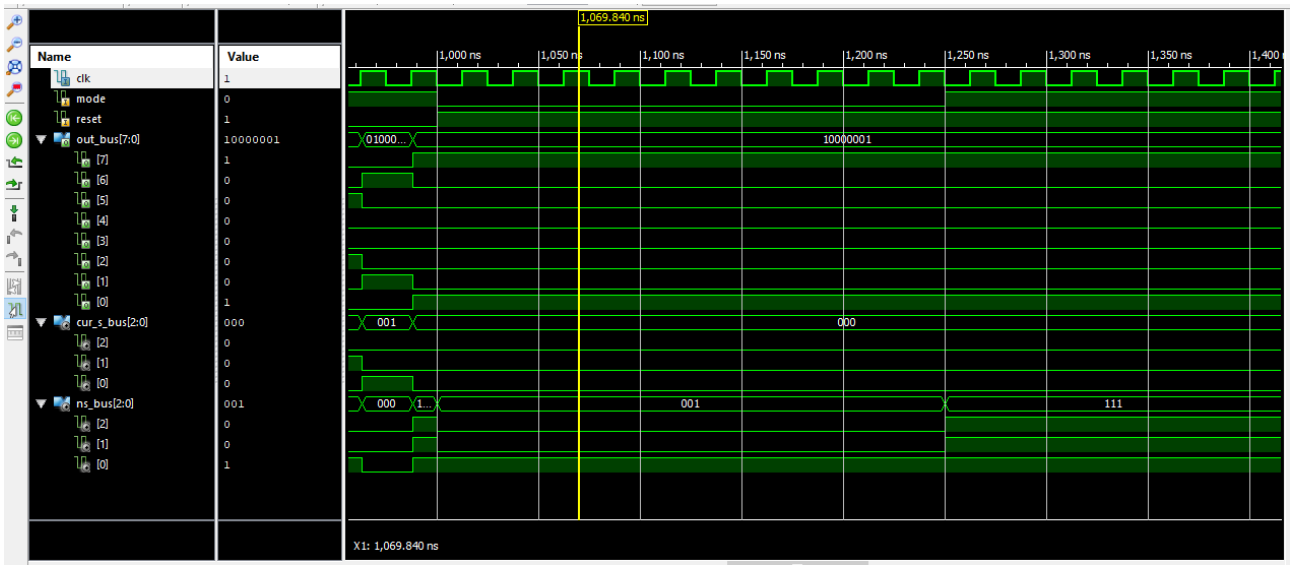
При MODE 0:



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.

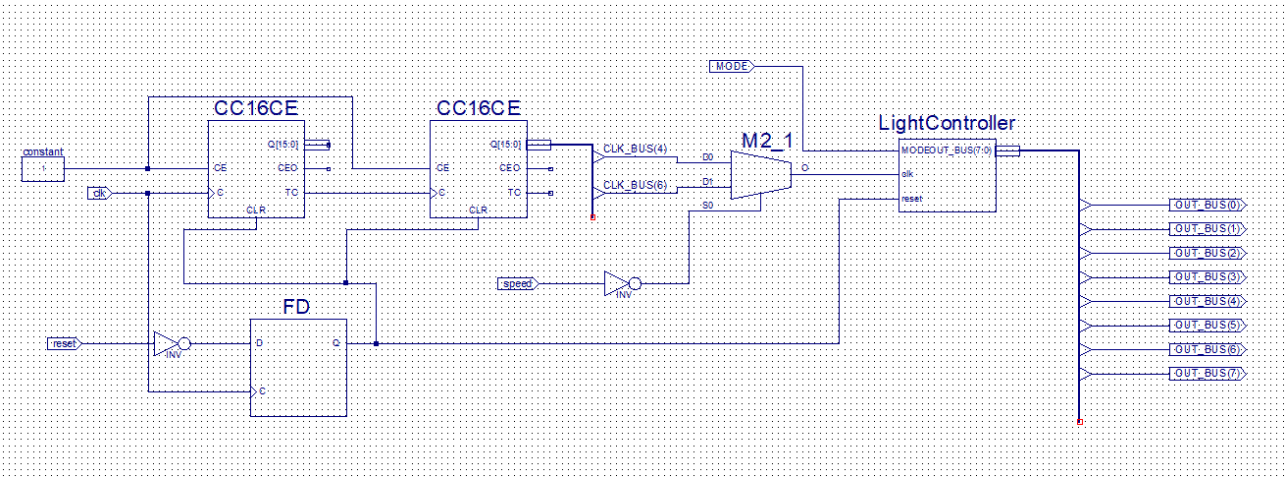
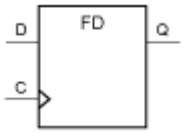
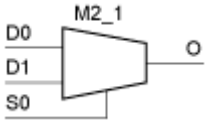


Рис.5: Schematic файл для кінцевої схеми.(TopLevel.sch)



Inputs		Outputs
D	C	Q
0	↑	0
1	↑	1

Схематичне зображення D-тригера та його таблиця істинності



Inputs			Outputs
S0	D1	D0	O
1	D1	X	D1
0	X	D0	D0

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.

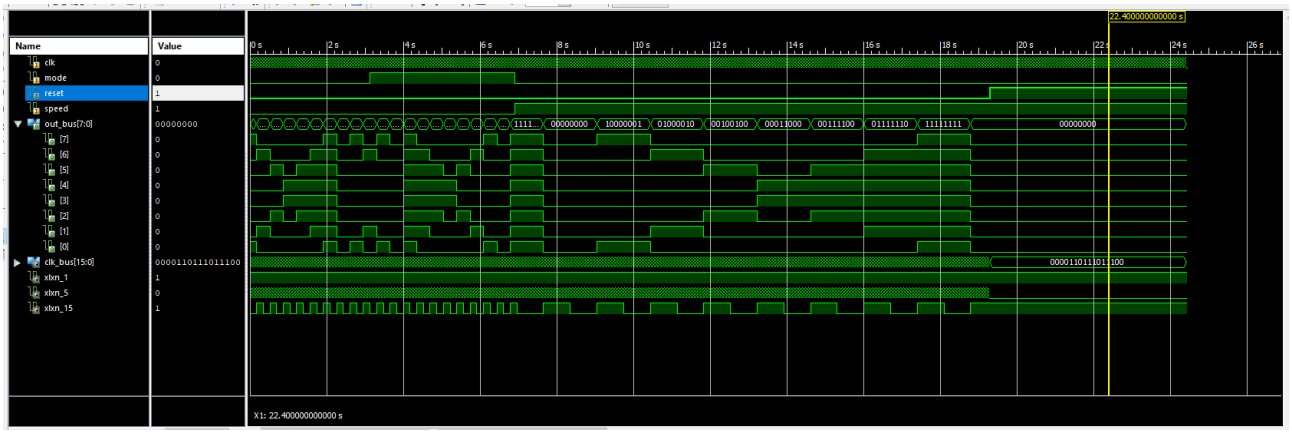


Рис.6: Діаграма проведеної симуляції для TopLevel

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.

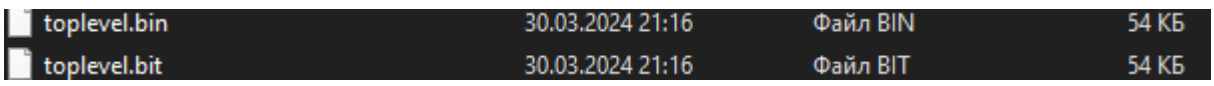
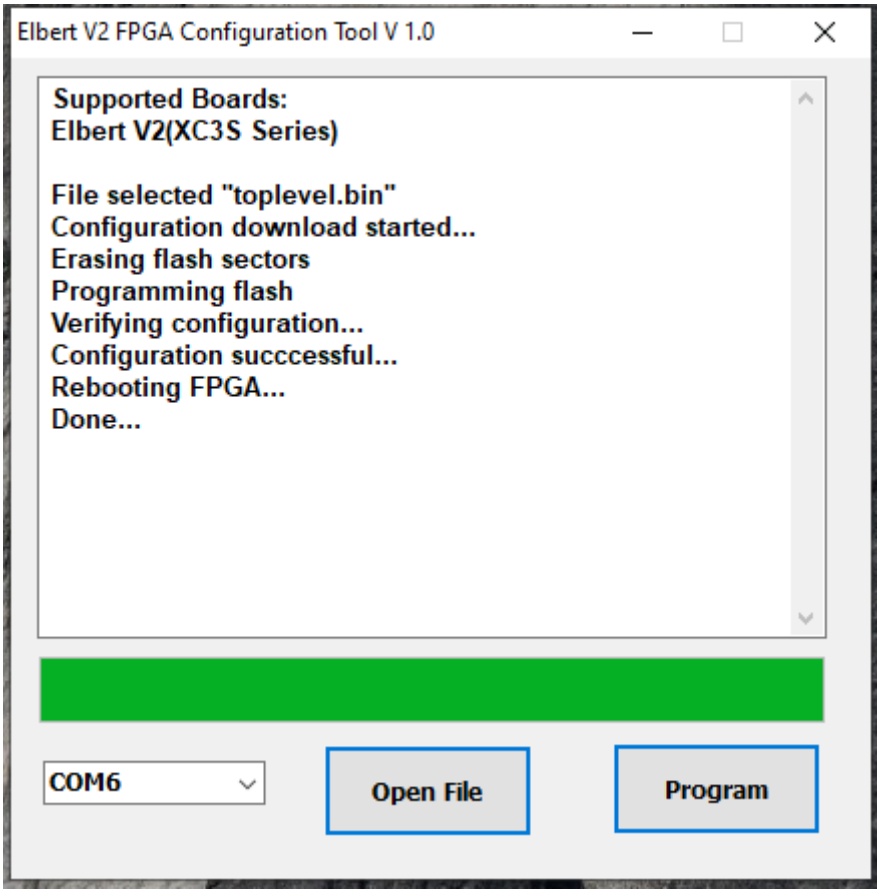


Рис.7: згенерований бінарний файл

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.