Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ

Звіт

З лабораторної роботи №2

З дисципліни: «Моделювання комп’ютерних систем»

На тему: «Структурний опис цифрового автомата Перевірка роботи автомата за допомогою стенда Elbert V2 – Spartan3A FPGA»

Варіант 22

Виконав: ст. гр. КІ-201

Вітик С.А

Прийняв:

Козак Н.Б

Львів 2024

Мета роботи:

На базі стенда реалізувати цифровий автомат світлових ефектів згідно заданих вимог.

Етапи роботи:

1. Інтерфейс пристрою та функціонал реалізувати згідно отриманого варіанту завдання.
2. Логіку переходів реалізувати з використанням мови опису апаратних засобів.
3. Логіку формування вихідних сигналів реалізувати з використанням мови опису апаратних засобів.
4. Згенерувати символи для 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. Пам’ять станів реалізувати за допомогою графічних компонентів з бібліотеки.
6. Промоделювати роботу окремих частин автомата та автомата вцілому за допомогою симулятора ISim.
7. Інтегрувати створений автомат зі стендом додати подільник частоти для вхідного тактовового сигналу призначити фізичні виводи на FPGA.
8. Згенерувати файал та перевірити роботу за допомогою стенда Elbert V2 – Spartan3A FPGA.
9. Підготувати і захистити звіт.

Варіант виконання роботи:

Пристрій повинен реалізувати комбінацій вихідних сигналів згідно таблиці:

Табл.1.1 Вихідні сигнали для кожного стану..

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	0
1	1	1	0	0	0	0	0	0
2	1	1	1	0	0	0	0	0
3	1	1	1	1	0	0	0	0
4	0	0	0	0	1	0	0	0
5	0	0	0	0	1	1	0	0
6	0	0	0	0	1	1	1	0
7	0	0	0	0	1	1	1	1

- Пристрій повинен використовувати тактовий сигнал 12MHz від мікроконтролера і знижувати частоту за допомогою внутрішнього подільника Мікроконтролер є частиною стенда Elbert V2 – Spartan3A FPGA. Тактовий сигнал заведено на вхід LOC = P129 FPGA.

- 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 (TEST) для подачі логічної <1> на всі непарні виходи:
  - Якщо *TEST=0* то автомат перемикає сигнали на виходах згідно заданого алгоритму.
  - Якщо *TEST=1* то на непарних виходах (7, 5, 3, 1) повинна бути логічна «1» (непарні *LED* увімкнені).

- Для керування сигналом MODE використати будь який з 8 DIP перемикачів.
- Для керування сигналами RESET/TEST використати будь які з PUSH BUTTON кнопок.

Виконання роботи:

1) Логіку переходів реалізувати з використанням мови опису апаратних засобів.

Табл.2.2.1-2.2.3. Логіка переходів для всіх станів автомата.

1)

MODE	CUR_STATE(2)	CUR_STATE(1)	CUR_STATE(0)	NEXT_STATE(0)
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

2)

MODE	CUR_STATE(2)	CUR_STATE(1)	CUR_STATE(0)	NEXT_STATE(1)
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

3)

MODE	CUR_STATE(2)	CUR_STATE(1)	CUR_STATE(0)	NEXT_STATE(2)
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Мінімізовані функції наступних станів автомата:

NEXT_STATE(0) = not(CURR_STATE(0));

NEXT_STATE(1) = ((not(MODE) and not(CURR_STATE(1)) and CURR_STATE(0)) or (not(MODE) and CURR_STATE(1) and not(CURR_STATE(0))) or (MODE and not(CURR_STATE(1)) and not(CURR_STATE(0))) or (MODE and CURR_STATE(1) and CURR_STATE(0)));

NEXT_STATE(2) <= ((not(MODE) and CURR_STATE(2) and not(CURR_STATE(1))) or (CURR_STATE(2) and CURR_STATE(1) and not(CURR_STATE(0))) or (MODE and CURR_STATE(2) and CURR_STATE(0)) or (not(MODE) and not(CURR_STATE(2)) and CURR_STATE(1) and CURR_STATE(0)) or (MODE and not(CURR_STATE(2)) and not(CURR_STATE(1)) and not(CURR_STATE(0))));

```
19  ----------------------------------------------------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  entity TRANSITION_LOGIC is
24     Port (CURR_STATE : in  std_logic_vector(2 downto 0);
25           MODE : in std_logic;
26           NEXT_STATE : out std_logic_vector(2 downto 0)
27           );
28  end TRANSITION_LOGIC;
29
30  architecture TRANSITION_LOGIC_ARCH of TRANSITION_LOGIC is
31
32  begin
33     NEXT_STATE(0) <= not(CURR_STATE(0));
34     NEXT_STATE(1) <= ((not(MODE) and not(CURR_STATE(1)) and CURR_STATE(0)) or
35                       (not(MODE) and CURR_STATE(1) and not(CURR_STATE(0))) or
36                       (MODE and not(CURR_STATE(1)) and not(CURR_STATE(0))) or
37                       (MODE and CURR_STATE(1) and CURR_STATE(0)));
38     NEXT_STATE(2) <= ((not(MODE) and CURR_STATE(2) and not(CURR_STATE(1))) or
39                       (CURR_STATE(2) and CURR_STATE(1) and not(CURR_STATE(0))) or
40                       (MODE and CURR_STATE(2) and CURR_STATE(0)) or
41                       (not(MODE) and not(CURR_STATE(2)) and CURR_STATE(1) and CURR_STATE(0)) or
42                       (MODE and not(CURR_STATE(2)) and not(CURR_STATE(1)) and not(CURR_STATE(0))));
43
44
45  end TRANSITION_LOGIC_ARCH;
```

Рис.2.1. VHDL опис логіки переходів.

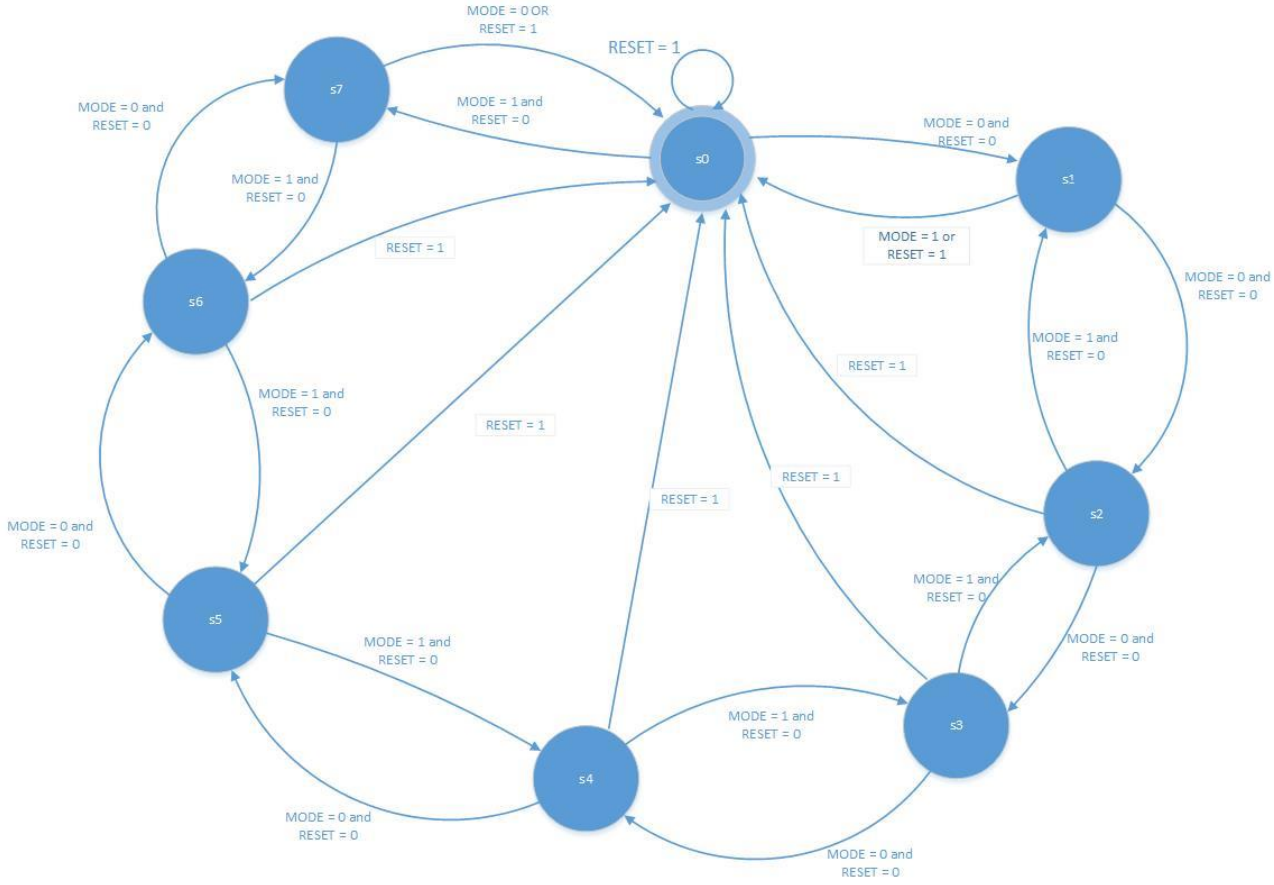

Рис.2.2. Граф переходів автомата між станами.

2) Логіку формування вихідних сигналів реалізувати з використанням мови опису апаратних засобів VHDL.

Логічні вирази для вихідних сигналів:

OUT_BUS(0) <= not(IN_BUS(2));

OUT_BUS(1) <= ((not(IN_BUS(2)) and IN_BUS(0)) or (not(IN_BUS(2)) and IN_BUS(1))) or TEST;

OUT_BUS(2) <= (not(IN_BUS(2)) and IN_BUS(1));

OUT_BUS(3) <= (not(IN_BUS(2)) and IN_BUS(1) and IN_BUS(0)) or TEST;

OUT_BUS(4) <= IN_BUS(2);

OUT_BUS(5) <= ((IN_BUS(2) and IN_BUS(0)) or (IN_BUS(2) and IN_BUS(1))) or TEST;

OUT_BUS(6) <= (IN_BUS(2) and IN_BUS(1));

OUT_BUS(7) <= (IN_BUS(2) and IN_BUS(1) and IN_BUS(0)) or TEST;

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3
4  entity OUTPUT_LOGIC is
5      Port ( IN_BUS : in  std_logic_vector(2 downto 0);
6             TEST : in  std_logic;
7             OUT_BUS : out  std_logic_vector(7 downto 0)
8             );
9  end OUTPUT_LOGIC;
10
11 architecture OUTPUT_LOGIC_ARCH of OUTPUT_LOGIC is
12
13 begin
14
15     OUT_BUS(0) <= not(IN_BUS(2));
16     OUT_BUS(1) <= ((not(IN_BUS(2)) and IN_BUS(0)) or (not(IN_BUS(2)) and IN_BUS(1))) or TEST;
17     OUT_BUS(2) <= (not(IN_BUS(2)) and IN_BUS(1));
18     OUT_BUS(3) <= (not(IN_BUS(2)) and IN_BUS(1) and IN_BUS(0)) or TEST;
19     OUT_BUS(4) <= IN_BUS(2);
20     OUT_BUS(5) <= ((IN_BUS(2) and IN_BUS(0)) or (IN_BUS(2) and IN_BUS(1))) or TEST;
21     OUT_BUS(6) <= (IN_BUS(2) and IN_BUS(1));
22     OUT_BUS(7) <= (IN_BUS(2) and IN_BUS(1) and IN_BUS(0)) or TEST;
23
24 end OUTPUT_LOGIC_ARCH;
25
26
27
```

Рис.2.4. VHDL опис вихідних сигналів.

3) Згенерувати символи для описів логіки переходів та логіки формування вихідних сигналів.

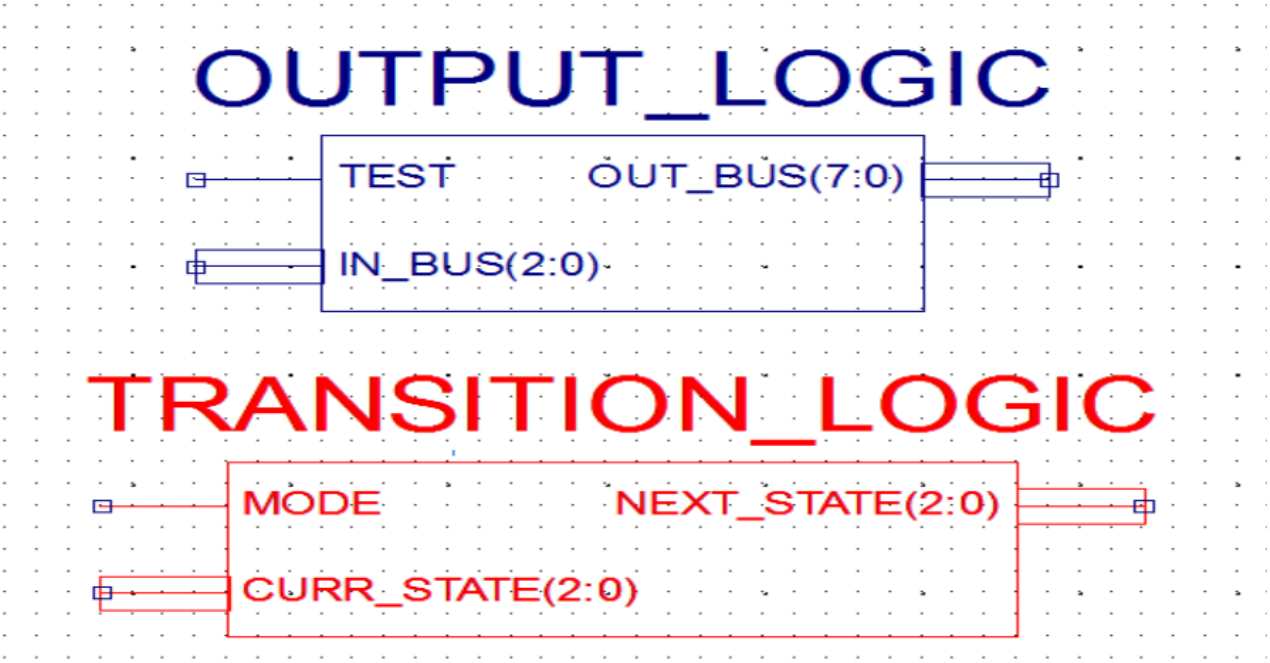

Рис.2.5. Згенеровані схематичні символи.

4) 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.

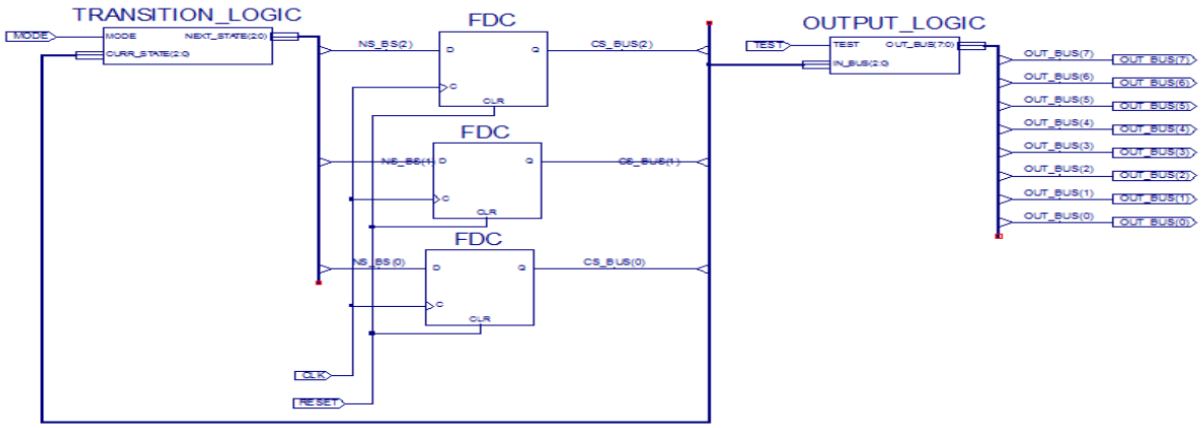

Рис.2.6. Інтеграція всіх створених компонентів разом з пам'ятю стану автомата.

5) Промоделювати роботу окремих частин автомата та автомата вцілому за допомогою симулятора ISim.

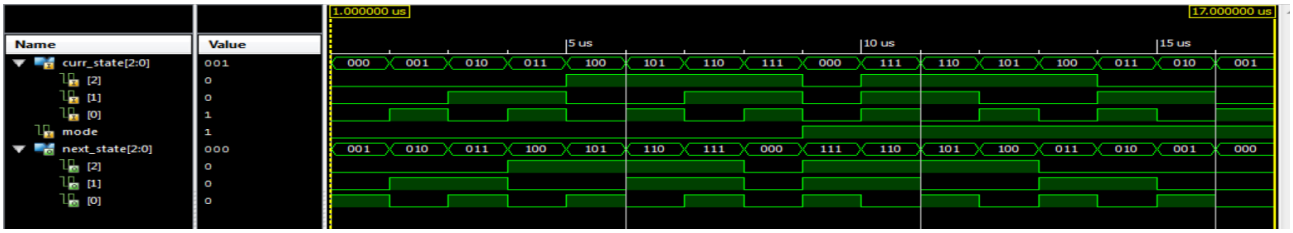

Рис.2.7. Результати симуляції логіки переходів в ISim.

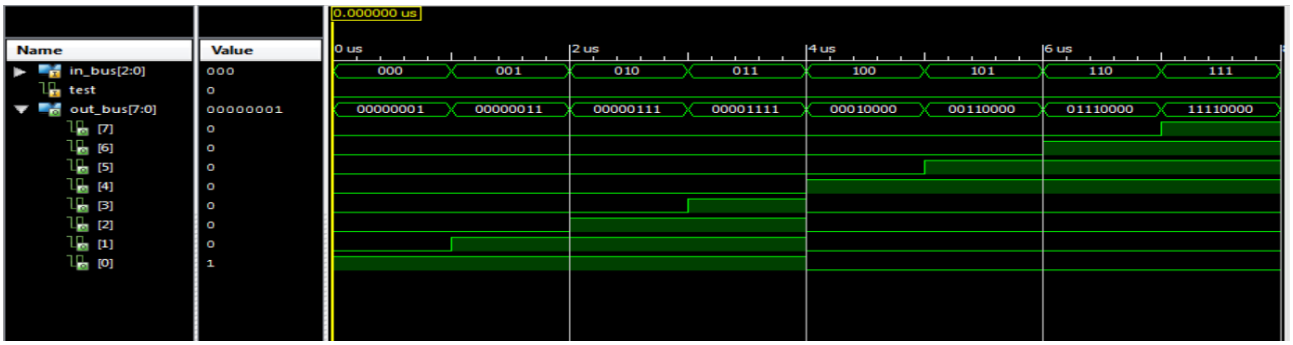

Рис.2.8.1. Результати симуляції логіки вихідних сигналів в ISim(TEST = 0).

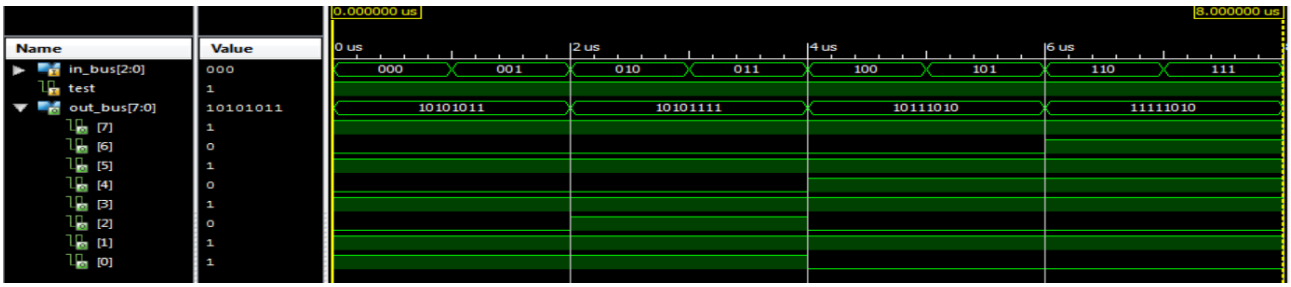

Рис.2.8.2. Результати симуляції логіки вихідних сигналів в ISim(TEST = 1).

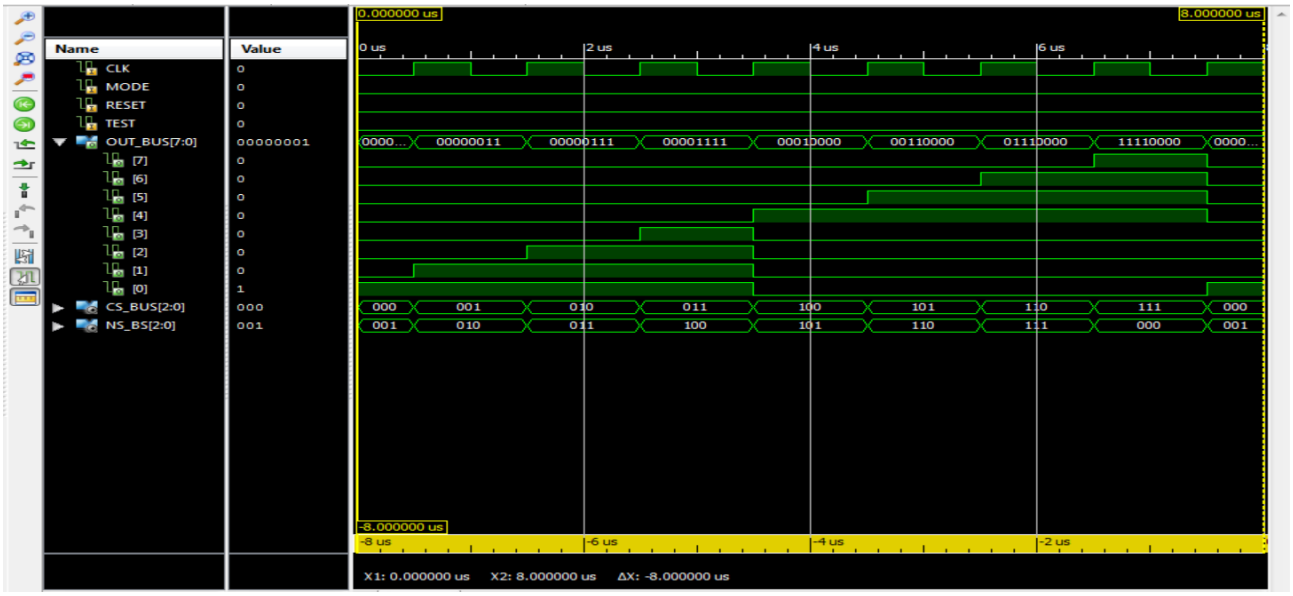

Рис.2.9. Результати симуляції автомата (MODE = 0, TEST = 0, RESET = 0).

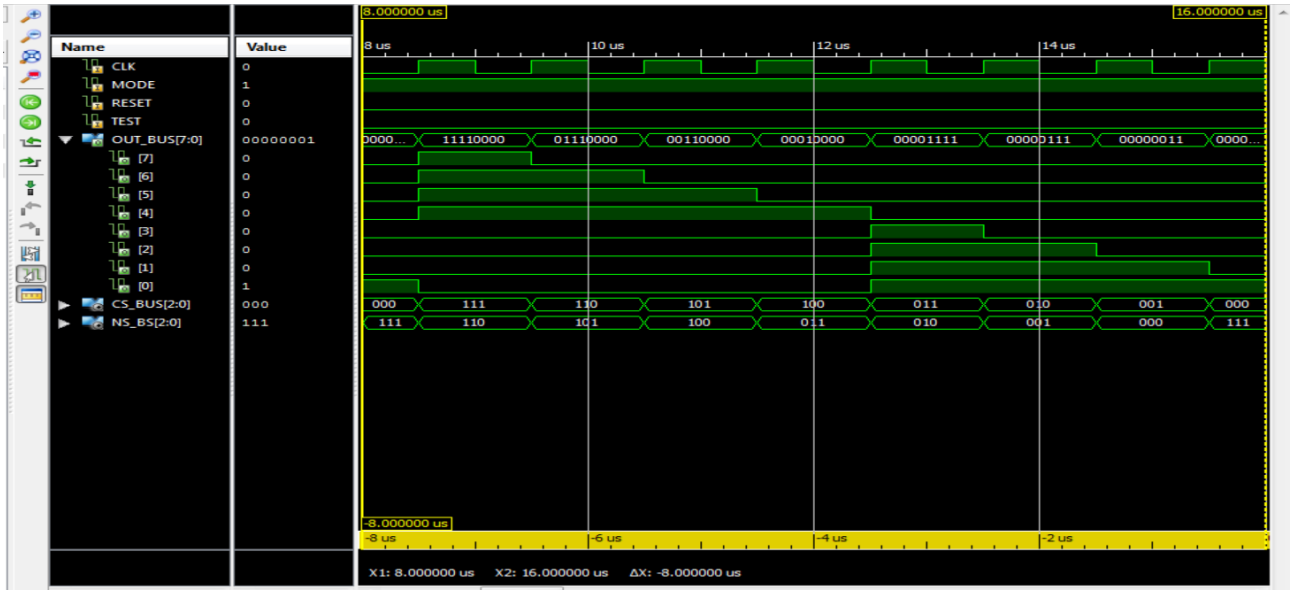

Рис.2.10. Результати симуляції автомата (MODE = 1, TEST = 0, RESET = 0).

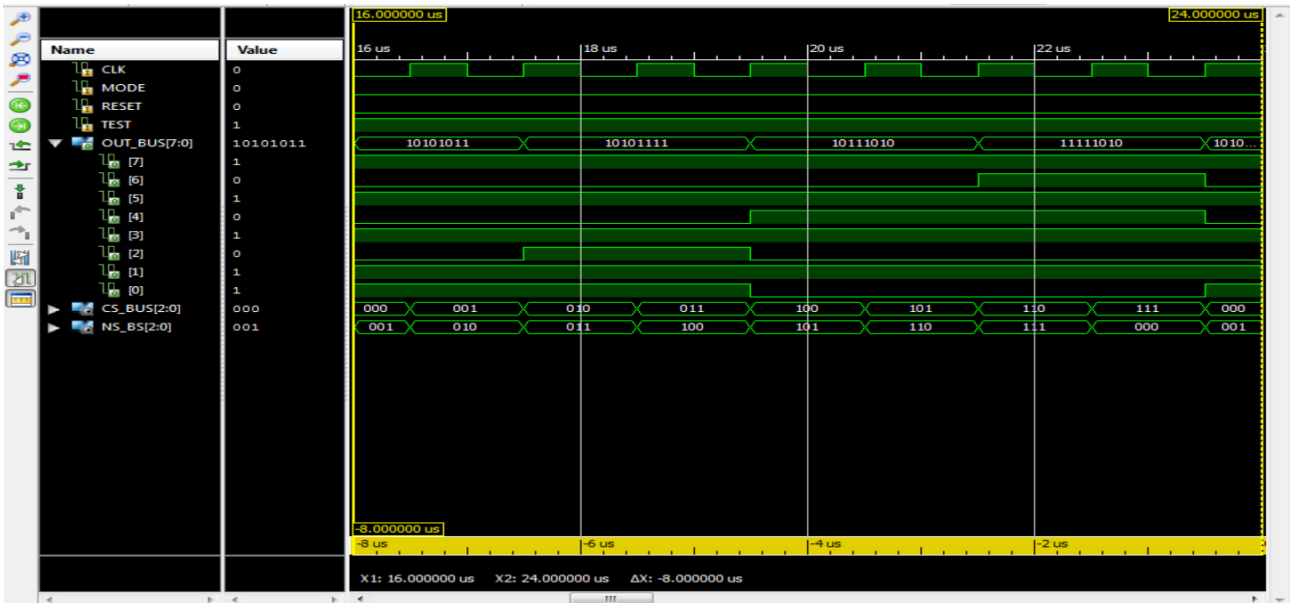

Рис.2.11. Результати симуляції автомата (MODE = 0, TEST = 1, RESET = 0).

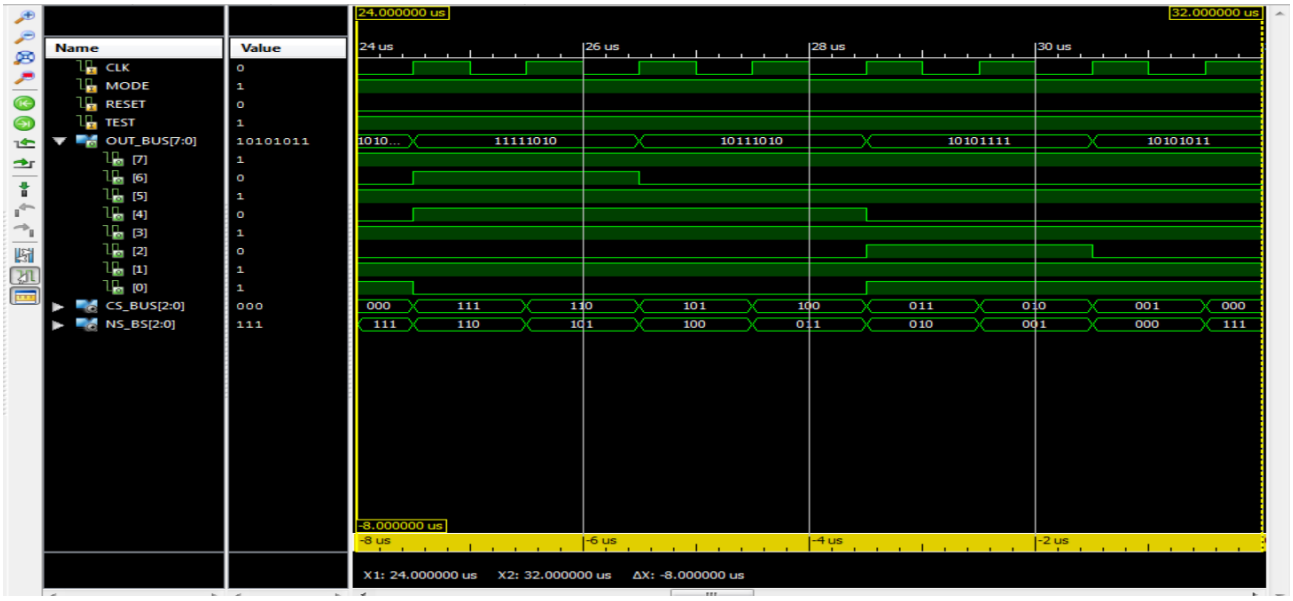

Рис.2.12. Результати симуляції автомата (MODE = 1, TEST = 1, RESET = 0).

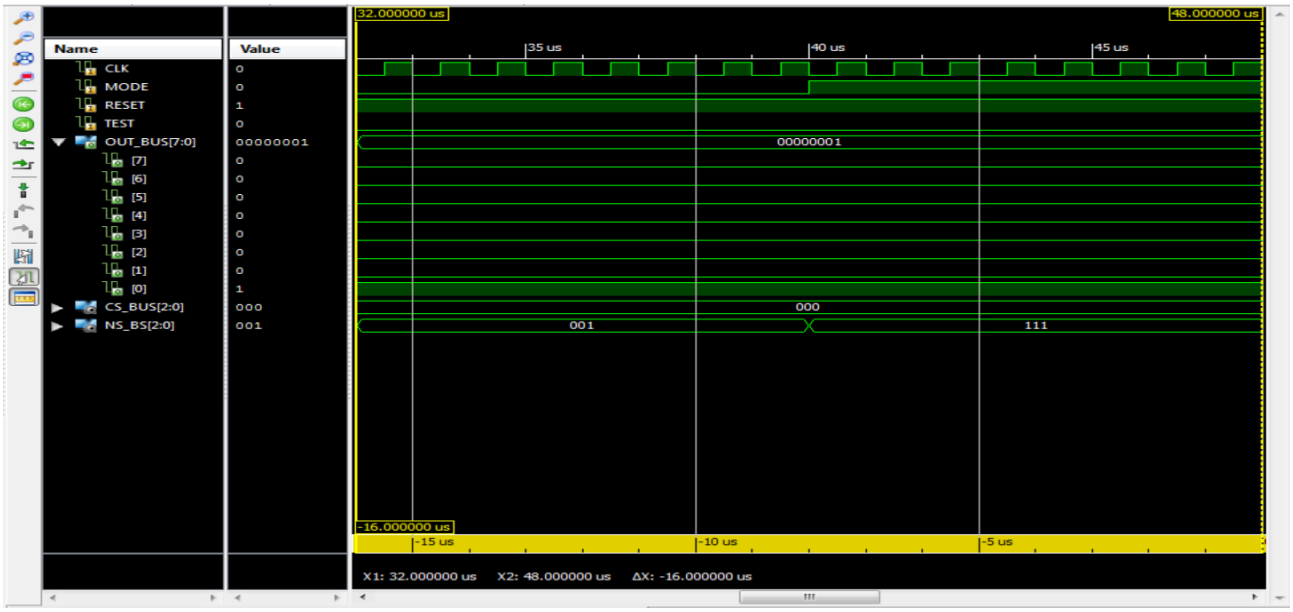

Рис.2.13. Результати симуляції автомата (MODE = 0, TEST = 0, RESET = 1).

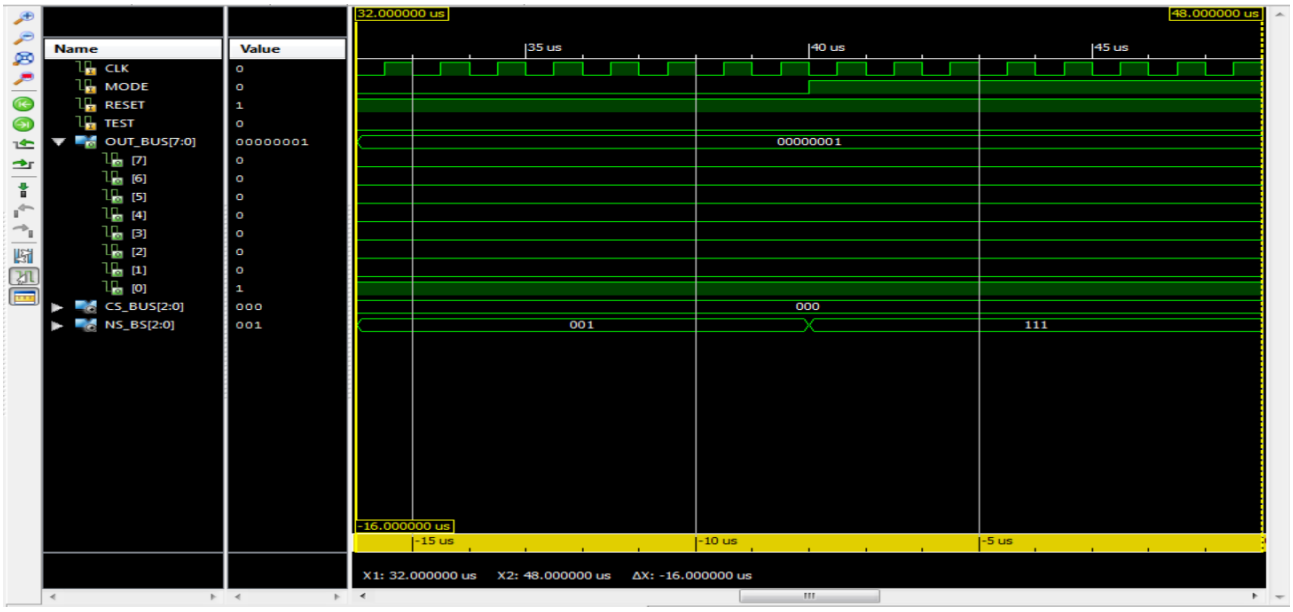

Рис.2.14. Результати симуляції автомата (MODE = 1, TEST = 0, RESET = 1).

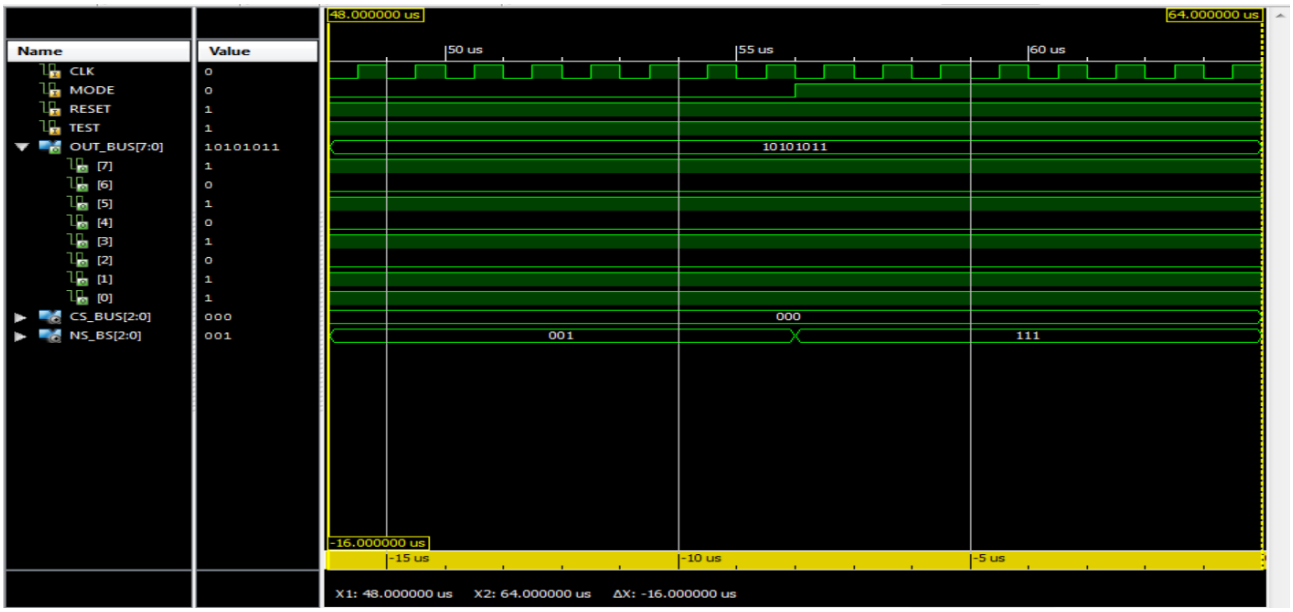

Рис.2.15. Результати симуляції автомата (MODE = 0, TEST = 1, RESET = 1).

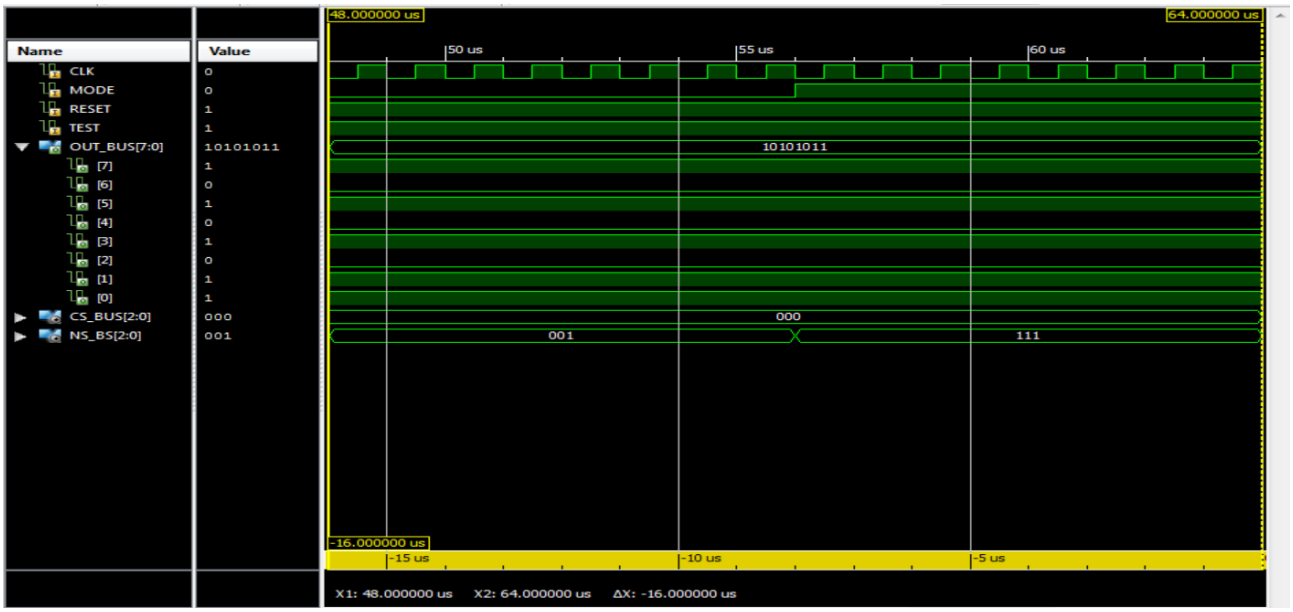

Рис.2.16. Результати симуляції автомата (MODE = 1, TEST = 1, RESET = 1).

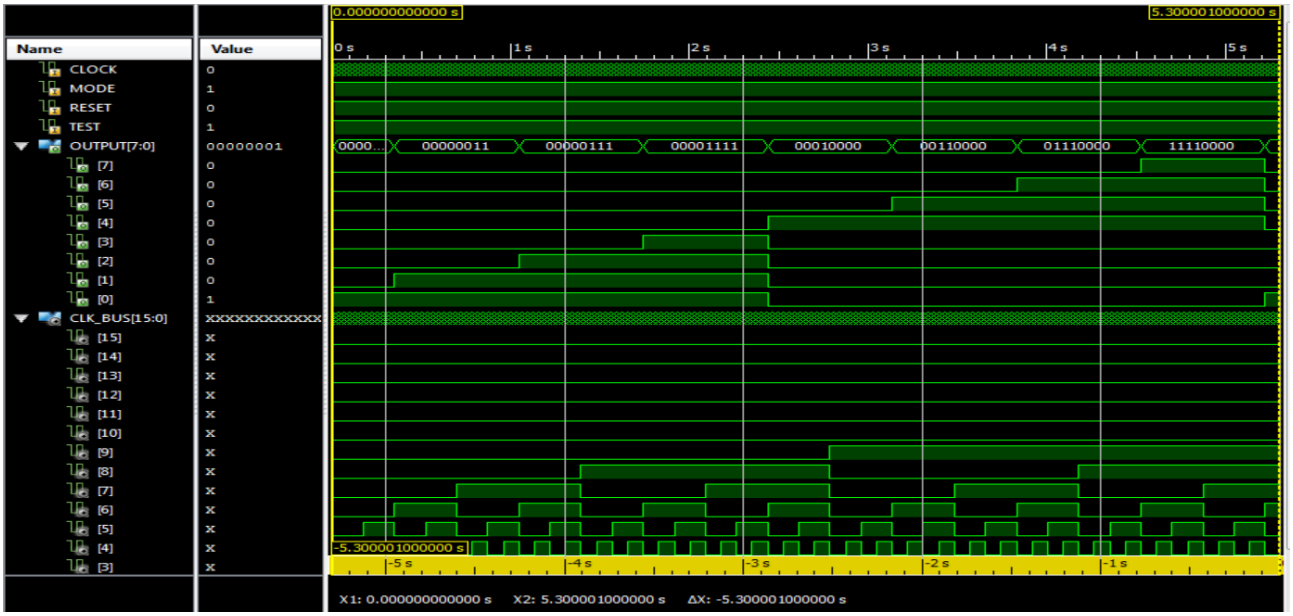

Рис.2.17. Результати симуляції фінальної схеми (MODE = 0, TEST = 0, RESET = 0).

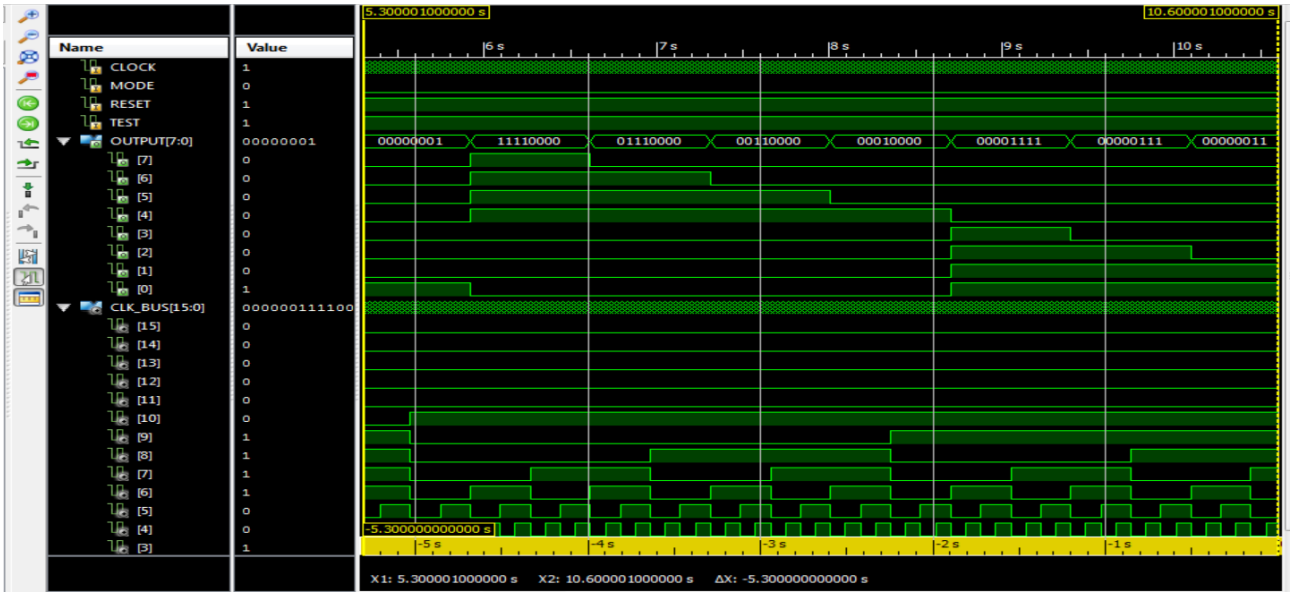

Рис.2.18. Результати симуляції фінальної схеми (MODE = 1, TEST = 0, RESET = 0).

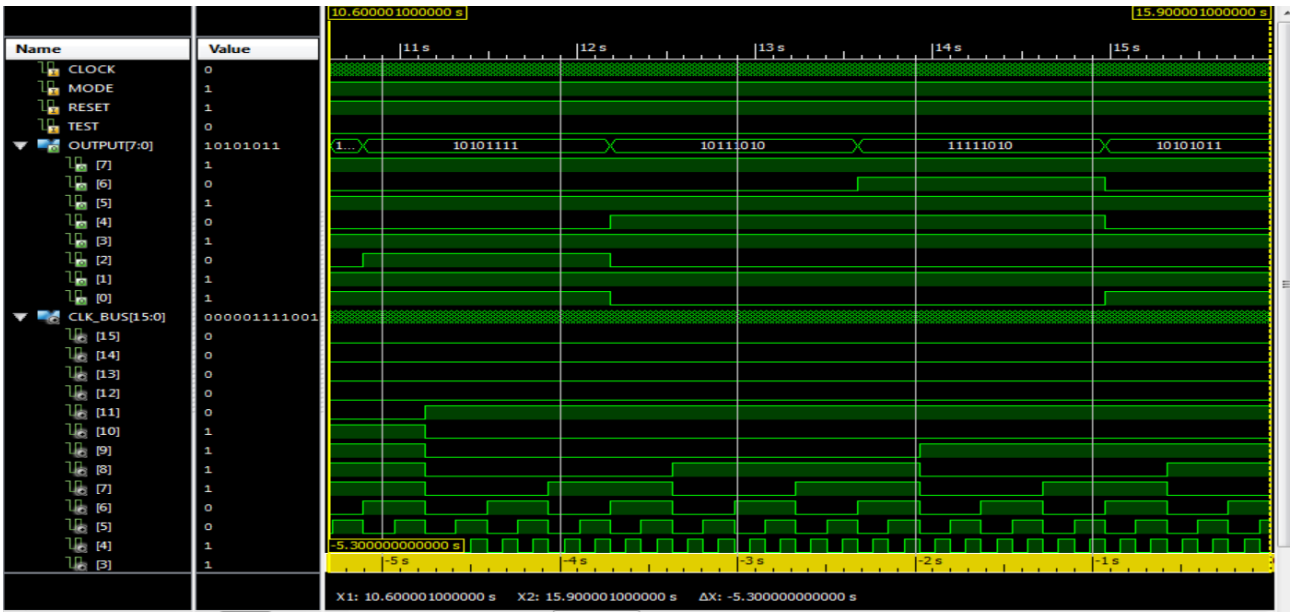

Рис.2.19. Результати симуляції фінальної схеми (MODE = 0, TEST = 1, RESET = 0).

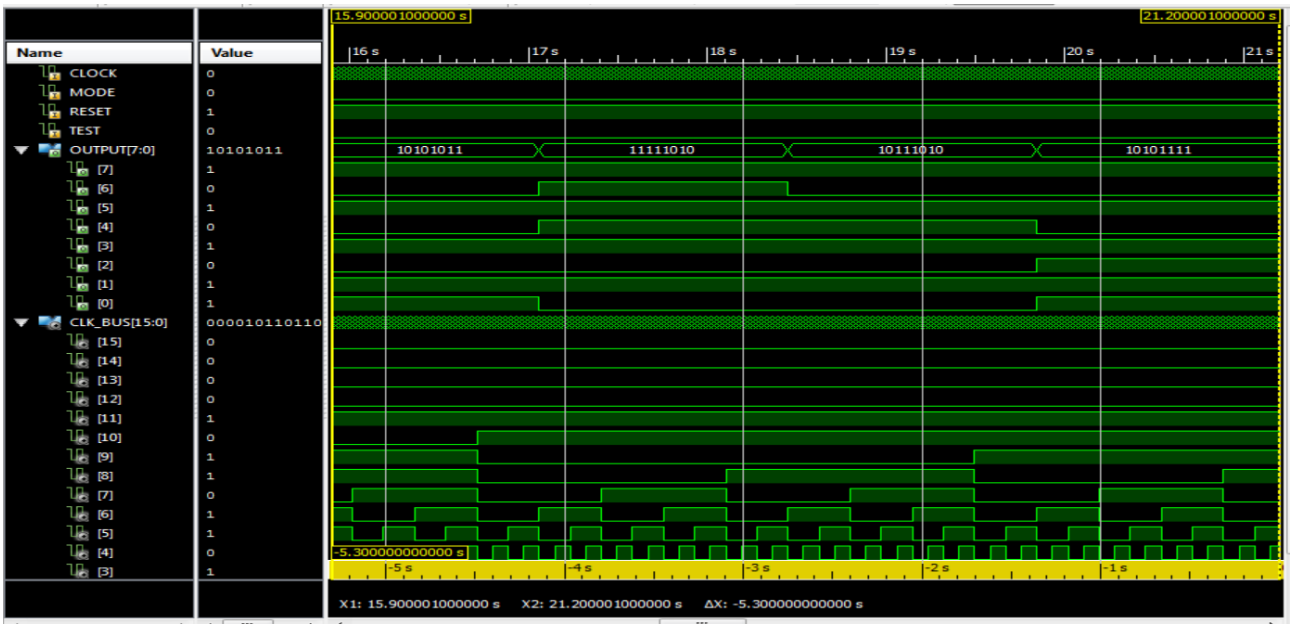

Рис.2.20. Результати симуляції фінальної схеми (MODE = 1, TEST = 1, RESET = 0).

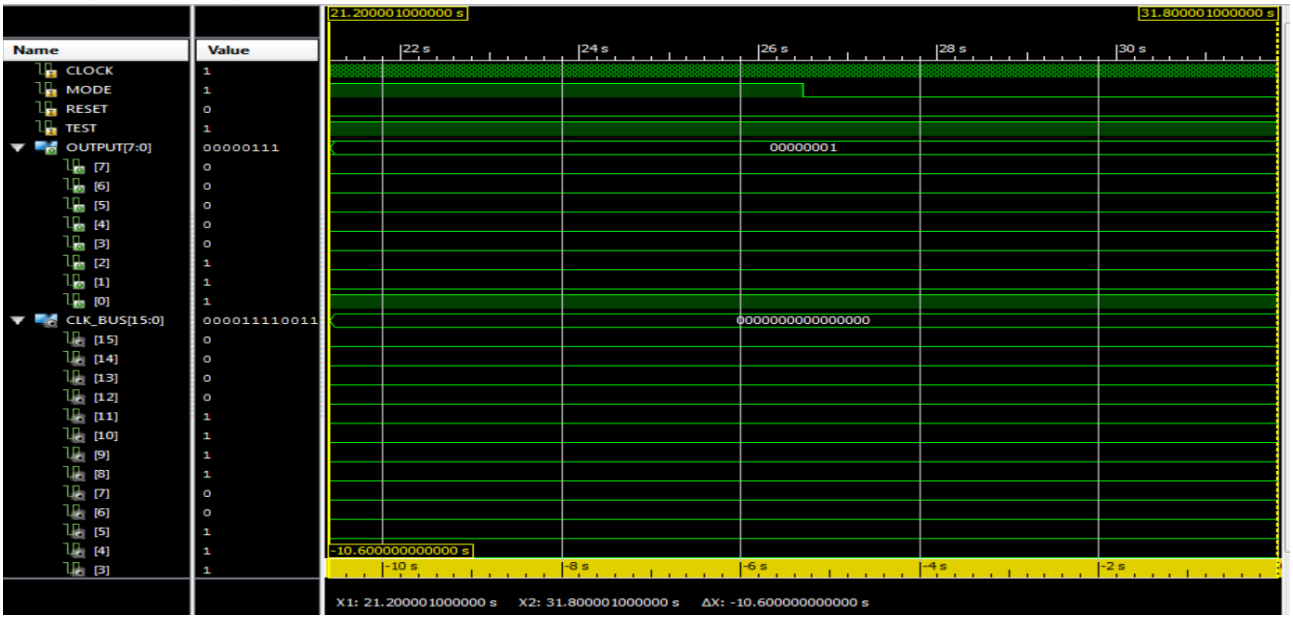

Рис.2.21. Результати симуляції фінальної схеми (MODE = 0, TEST = 0, RESET = 1).

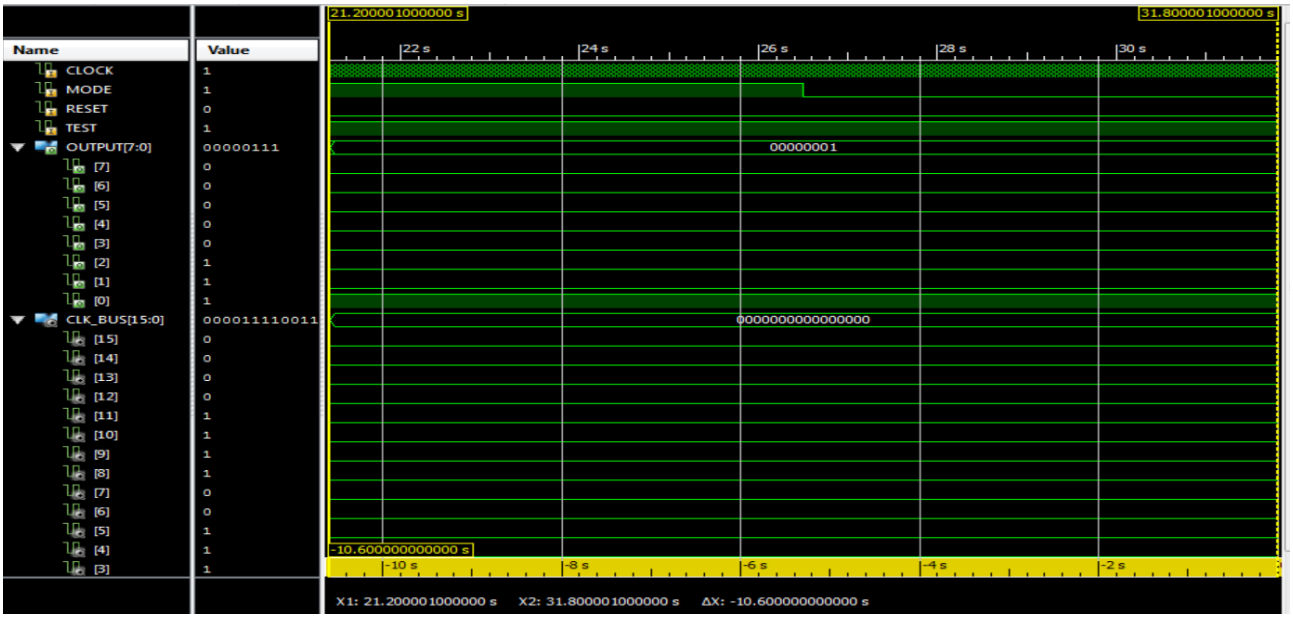

Рис.2.22. Результати симуляції фінальної схеми (MODE = 1, TEST = 0, RESET = 1).

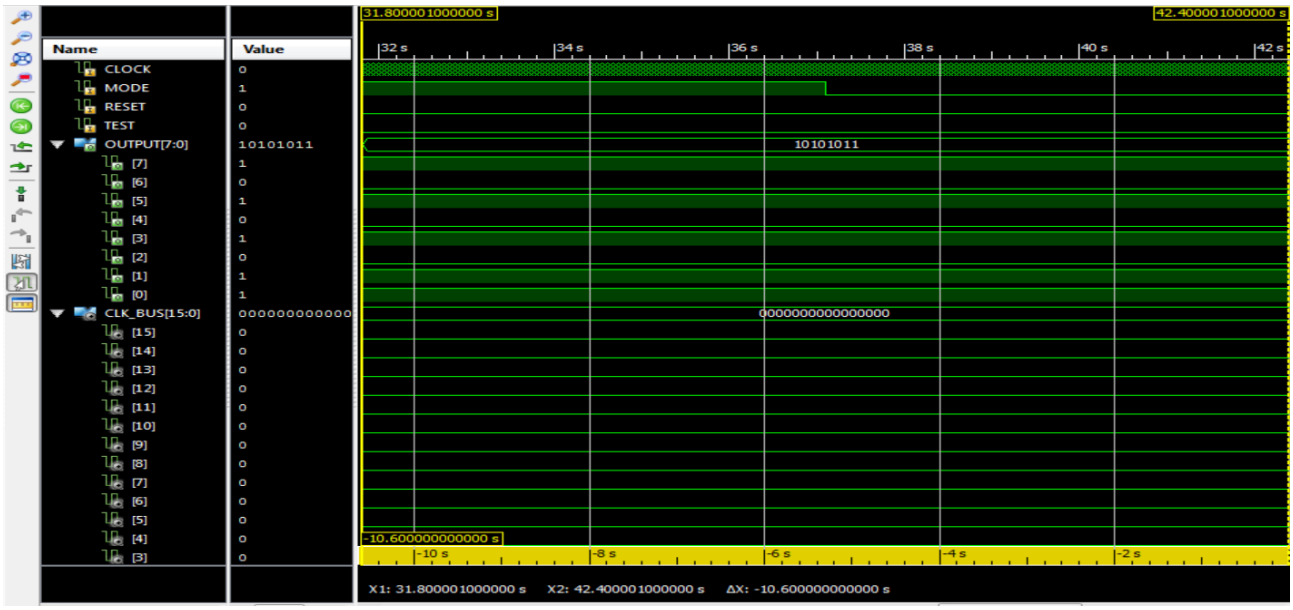

Рис.2.23. Результати симуляції фінальної схеми (MODE = 0, TEST = 1, RESET = 1).

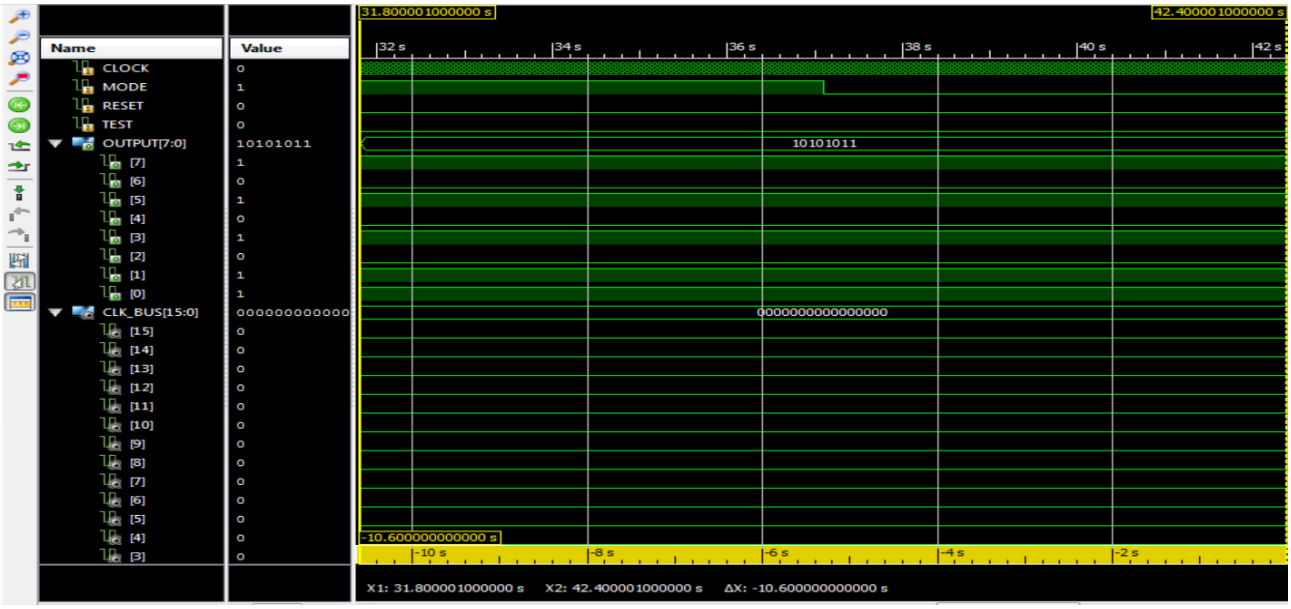

Рис.2.24. Результати симуляції фінальної схеми (MODE = 1, TEST = 1, RESET = 1).

6) Інтегрувати створений автомат зі стендом Elbert V2 – Spartan3A FPGA. Додати подільник частоти для вхідного тактовового сигналу призначити фізичні виводи на FPGA.

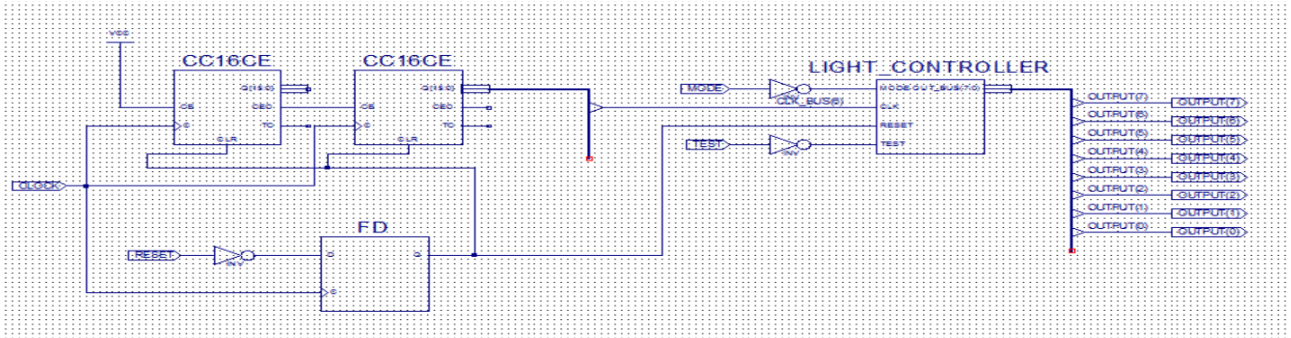

Рис.2.25. Автомат світлових сигналів та подільник тактового сигналу.

```
#**************************************************************************************************************
#                                          UCF for ElbertV2 Development Board
#**************************************************************************************************************
CONFIG VCCAUX = "3.3" ;

 # Clock 12 MHz
  NET "CLOCK"                   LOC = P129  | IOSTANDARD = LVCMOS33 | PERIOD = 12MHz;

##############################################################################################################
#                                            LED
##############################################################################################################

    NET "OUTPUT(0)"              LOC = P46   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "OUTPUT(1)"              LOC = P47   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "OUTPUT(2)"              LOC = P48   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "OUTPUT(3)"              LOC = P49   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "OUTPUT(4)"              LOC = P50   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "OUTPUT(5)"              LOC = P51   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "OUTPUT(6)"              LOC = P54   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "OUTPUT(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 "TEST"           LOC = P79   | PULLUP  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
```

Рис.2.26. Призначення фізичних входів та виходів.

Висновок:

В ході виконання цієї лабораторної роботи я реалізував на базі стенда Elbert V2 – Spartan3A FPGA цифровий автомат світлових ефектів згідно заданих вимог.