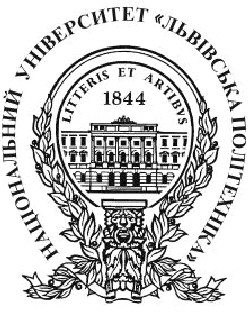
Міністерство освіти і науки України НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»



Кафедра ЕОМ

З лабораторної роботи № 1

З дисципліни « МОДЕЛЮВАННЯ КОМПЮТЕРНИХ СИСТЕМ »

На тему: « Структурний опис цифрового автомата. Перевірка роботи автомата за допомогою стенда Elbert V2-Spartan 3A FPGA»

Варіант 18

Виконав: студент гр. КІ-201  
Панасевич Ю. В.

Прийняв:   
Козак Н. Б.

Львів – 2023

# Мета роботи:

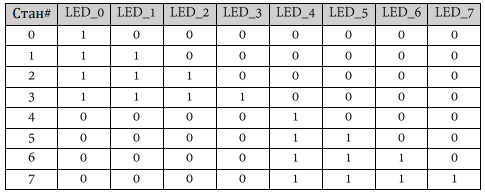
На базі стенда Elbert V2 – Spartan 3A FPGA реалізувати цифровий автомат світлових ефектів згідно наступних вимог:

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 18, тому номер варіант 6.

* Пристрій повинен реалізувати комбінацій вихідних сигналів згідно таблиці



* Пристрій повинен використовувати тактовий сигнал від мікроконтролера і знижувати частоту за допомогою внутрішнього подільника Мікроконтролер є частиною стенда 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 (012345670…).

Якщо то стан пристрою декрементується по зростаючому фронту тактового сигналу пам’яті станів (076543210…).

* Інтерфейс пристрою повинен мати однорозрядний вхід для подачі логічної на всі непарні виходи одночасно:

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:

# (VHDL OutputLogic)

# library IEEE;

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

# 

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

# 

# 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);

# 

# OUT\_BUS(4) <= IN\_BUS(2);

# 

# OUT\_BUS(5) <= IN\_BUS(2) and (IN\_BUS(1) or IN\_BUS(0));

# 

# OUT\_BUS(6) <= IN\_BUS(2) and IN\_BUS(1);

# 

# OUT\_BUS(7) <= IN\_BUS(2) and IN\_BUS(1) and IN\_BUS(0);

# end out\_logic\_arch;

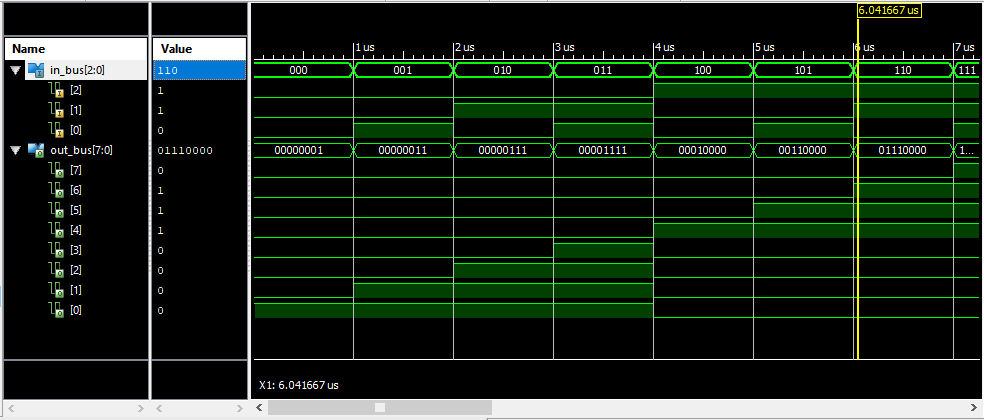


Рис.1: виконання роботи схеми

# (VHDL TransitionLogic)

library IEEE;

use IEEE.STD\_LOGIC\_1164.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(1)) and not (CUR\_STATE(0))) or

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

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

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

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

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

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

(MODE and CUR\_STATE(1) and CUR\_STATE(0));

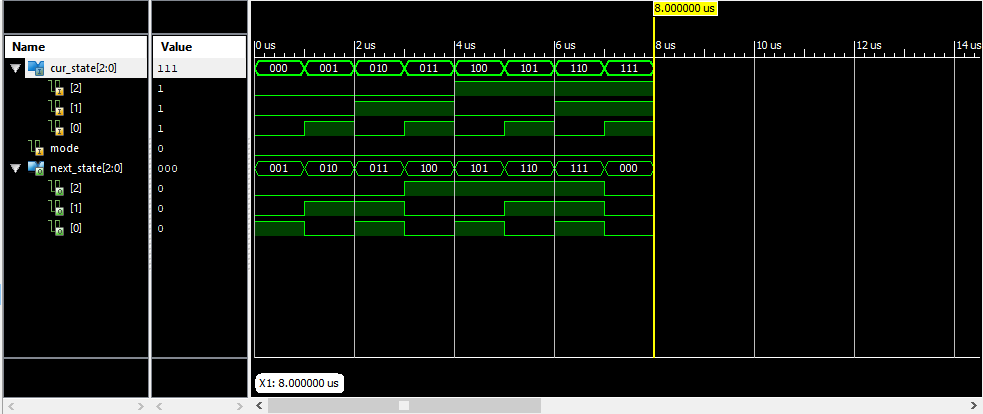
NEXT\_STATE(2) <= ((not(MODE) and CUR\_STATE(2)) and not (CUR\_STATE(1) and CUR\_STATE(0))) or

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

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

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

end transition\_logic\_arch;



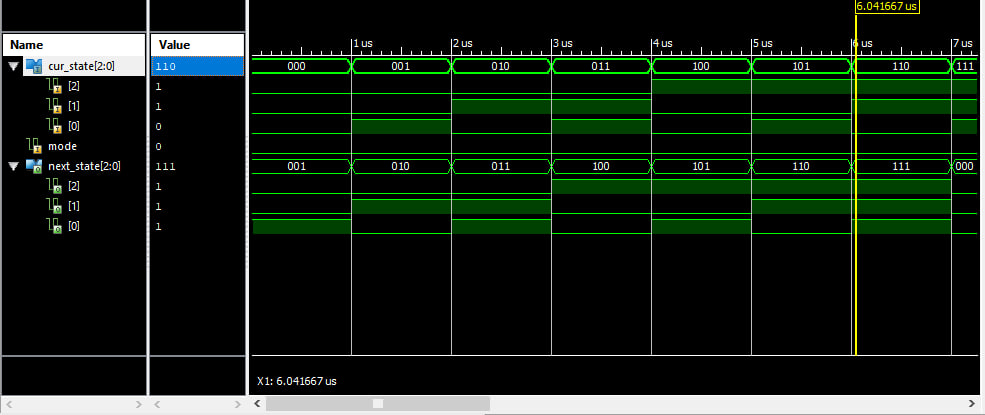


Рис.2: промодельована робота схеми формування вихідних сигналів з усіма можливими наборами сигналів

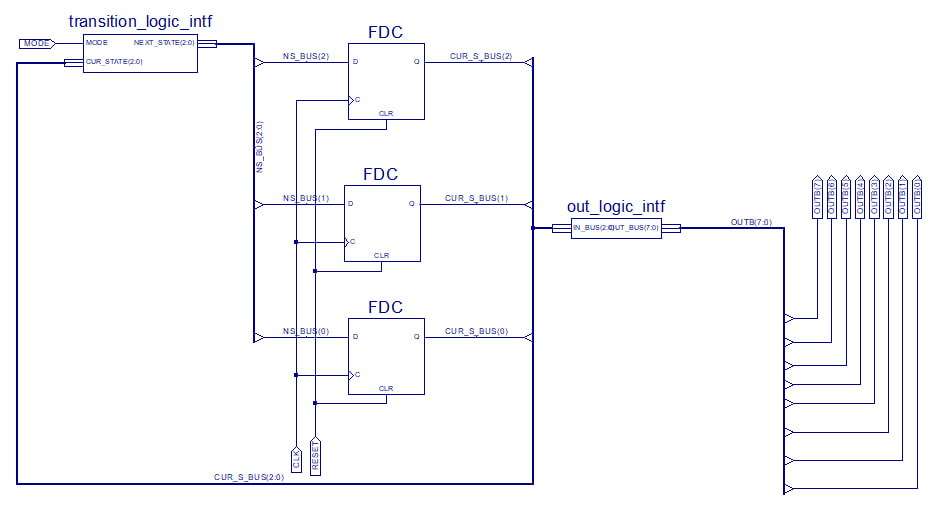


Рис.3: Schematic файл (LightController.sch)

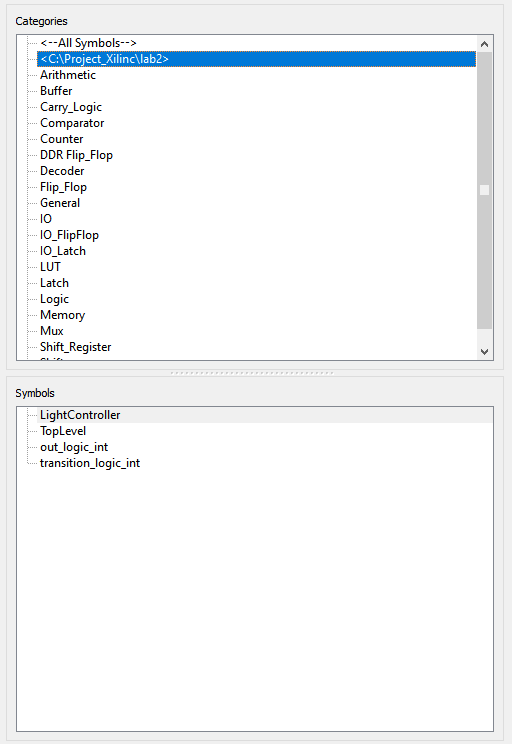


Рис.4: згенеровані Schematic символи

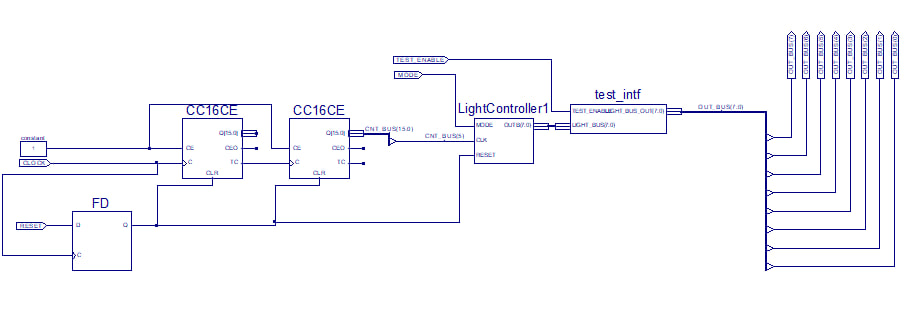


Рис.5: Schematic файл для кінцевої схеми.(TopLevel.sch)

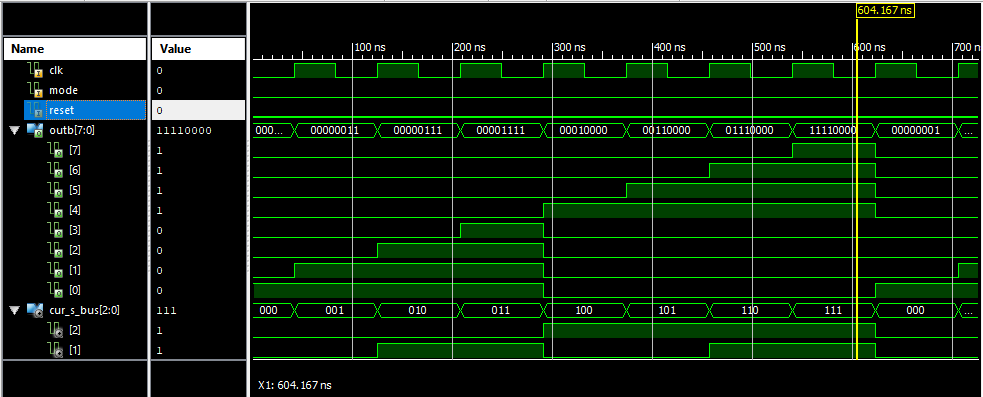


Рис.6: симуляція системи різними значеннями сигналів MODE/RESET/SPEED

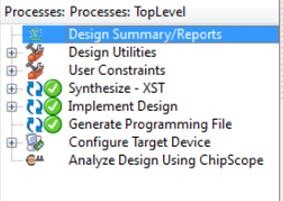


Рис.7: згенерований бінарний файл

**Висновок:** на лабораторній роботі на базі стенда Elbert V2- Spartan 3A FPGA реалізував цифровий автомат світлових ефектів. Навчився створювати нові елементи і описувати логіку їх роботи засобами VHDL.