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”

Варіант - 2

Виконав:
ст.гр. КІ-202
Побережник Ю.В.
Перевірив:
Козак Н.Б.

**Львів 2023**

**Мета роботи :** На базі стенда Elbert V2 – Spartan 3A FPGA, реалізувати цифровий автомат світлових ефектів.

## Завдання

### Варіант – 2:

- 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 | 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 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 5 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| 6 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 7 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

- 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 2 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:

***Файл OutputLogic:***

```
-- Company:
-- Engineer:
--
-- Create Date:    11:41:55 04/08/2023
-- 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(not(IN_BUS(2)) and not(IN_BUS(1)) and IN_BUS(0)) or (not(IN_BUS(2)) and
IN_BUS(1) and not(IN_BUS(0)))  or (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)))  or (IN_BUS(2) and IN_BUS(1) and IN_BUS(0)));

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

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

OUT_BUS(4) <= ((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)))  or (IN_BUS(2) and IN_BUS(1) and IN_BUS(0)));

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

OUT_BUS(6) <= ((IN_BUS(2) and IN_BUS(1) and not(IN_BUS(0)))  or (IN_BUS(2) and IN_BUS(1) and IN_BUS(0)));

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

end out_logic_arch;

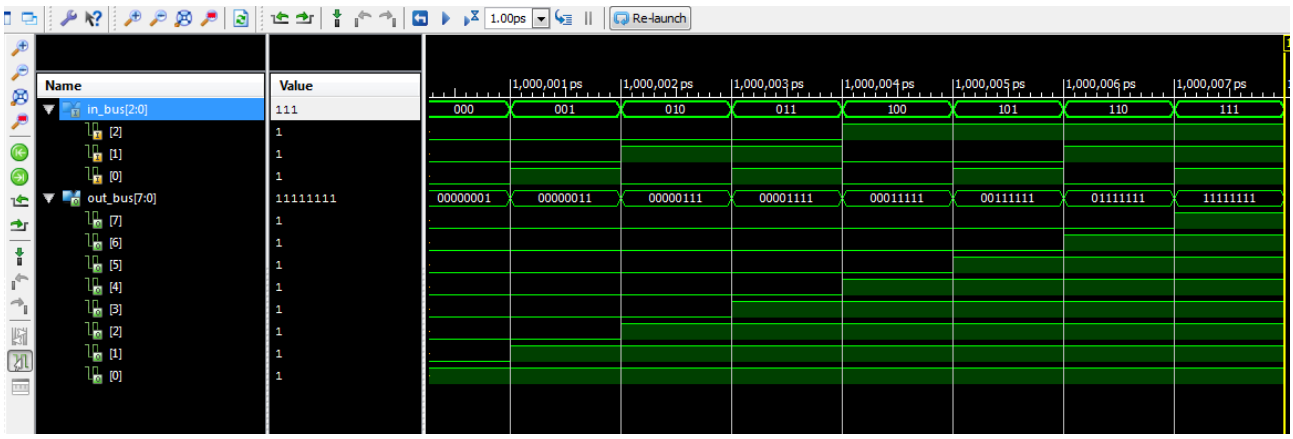


***Рис. 1 Діаграма проведеної симуляції для OutputLogic***

***Файл TransitionLogic:***

```
-- Company:
-- Engineer:
--
-- Create Date:    16:13:21 02/23/2020
-- 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);
         RESET : in  std_logic
         );
end transition_logic_intf;

architecture transition_logic_arch of transition_logic_intf is

begin

NEXT_STATE(0) <= (not(RESET) and not(MODE) and not(CUR_STATE(2)) and
not(CUR_STATE(1)) and not(CUR_STATE(0))) or -- 000 -> 001
                                            (not(RESET) and not(MODE) and
not(CUR_STATE(2)) and CUR_STATE(1) and not(CUR_STATE(0))) or -- 010 ->
011
```

(not(RESET) and not(MODE) and CUR_STATE(2) and not(CUR_STATE(1)) and not(CUR_STATE(0))) or -- 100 -> 101
(not(RESET) and not(MODE) and CUR_STATE(2) and CUR_STATE(1) and not(CUR_STATE(0))) or -- 110 -> 111
(not(RESET) and MODE and not(CUR_STATE(2)) and (CUR_STATE(1)) and not(CUR_STATE(0))) or -- 001 <- 010
(not(RESET) and MODE and (CUR_STATE(2)) and not(CUR_STATE(1)) and not(CUR_STATE(0))) or -- 011 <- 100
(not(RESET) and MODE and CUR_STATE(2) and (CUR_STATE(1)) and not(CUR_STATE(0))) or -- 101 <- 110
(not(RESET) and MODE and not(CUR_STATE(2)) and not(CUR_STATE(1)) and not(CUR_STATE(0))); -- 111 <- 000
NEXT_STATE(1) <= (not(RESET) and not(MODE) and not(CUR_STATE(2)) and not(CUR_STATE(1)) and CUR_STATE(0)) or -- 001 -> 010
(not(RESET) and not(MODE) and not(CUR_STATE(2)) and CUR_STATE(1) and not(CUR_STATE(0))) or -- 010 -> 011
(not(RESET) and not(MODE) and CUR_STATE(2) and not(CUR_STATE(1)) and CUR_STATE(0)) or -- 101 -> 110
(not(RESET) and not(MODE) and CUR_STATE(2) and CUR_STATE(1) and not(CUR_STATE(0))) or -- 110 -> 111
(not(RESET) and MODE and not(CUR_STATE(2)) and (CUR_STATE(1)) and (CUR_STATE(0))) or -- 010 <- 011
(not(RESET) and MODE and CUR_STATE(2) and not(CUR_STATE(1)) and not(CUR_STATE(0))) or -- 011 <- 100
(not(RESET) and MODE and CUR_STATE(2) and CUR_STATE(1) and CUR_STATE(0)) or -- 110 <- 111
(not(RESET) and MODE and not(CUR_STATE(2)) and not(CUR_STATE(1)) and not(CUR_STATE(0))); -- 111 <- 000
NEXT_STATE(2) <= (not(RESET) and not(MODE) and not(CUR_STATE(2)) and CUR_STATE(1) and CUR_STATE(0)) or -- 011 -> 100
(not(RESET) and not(MODE) and CUR_STATE(2) and not(CUR_STATE(1)) and not(CUR_STATE(0))) or -- 100 -> 101
(not(RESET) and not(MODE) and CUR_STATE(2) and not(CUR_STATE(1)) and CUR_STATE(0)) or -- 101 -> 110
(not(RESET) and not(MODE) and CUR_STATE(2) and CUR_STATE(1) and not(CUR_STATE(0))) or -- 110 -> 111
(not(RESET) and MODE and CUR_STATE(2) and not(CUR_STATE(1)) and CUR_STATE(0)) or -- 100 <- 101

(not(RESET) and MODE and CUR_STATE(2) and CUR_STATE(1) and not(CUR_STATE(0))) or -- 101 <- 110
(not(RESET) and MODE and CUR_STATE(2) and CUR_STATE(1) and CUR_STATE(0)) or -- 110 <- 111
(not(RESET) and MODE and not(CUR_STATE(2)) and not(CUR_STATE(1)) and not(CUR_STATE(0))); -- 111 <- 000
end transition_logic_arch;

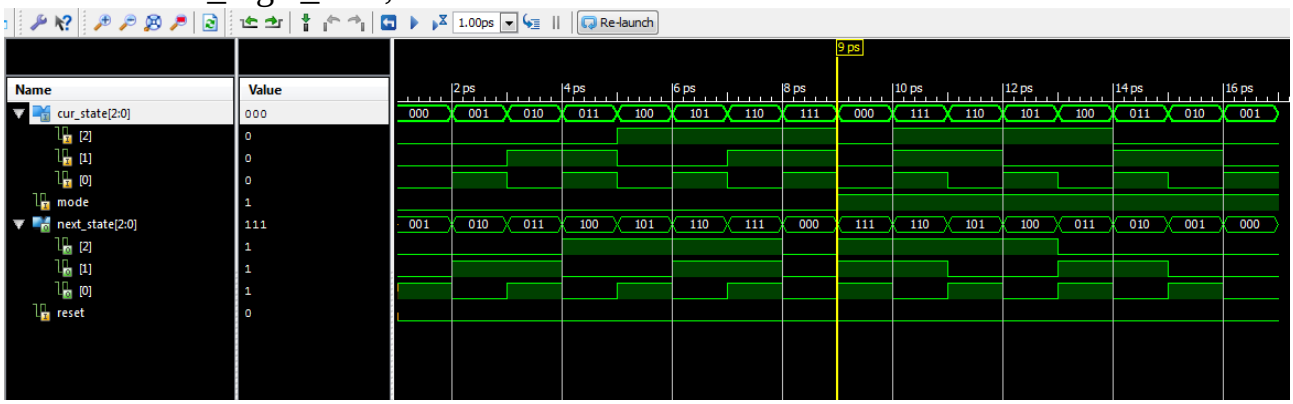


***Рис. 2 Діаграма проведеної симуляції для TransitionLogic***

## *Схема для Light controller*

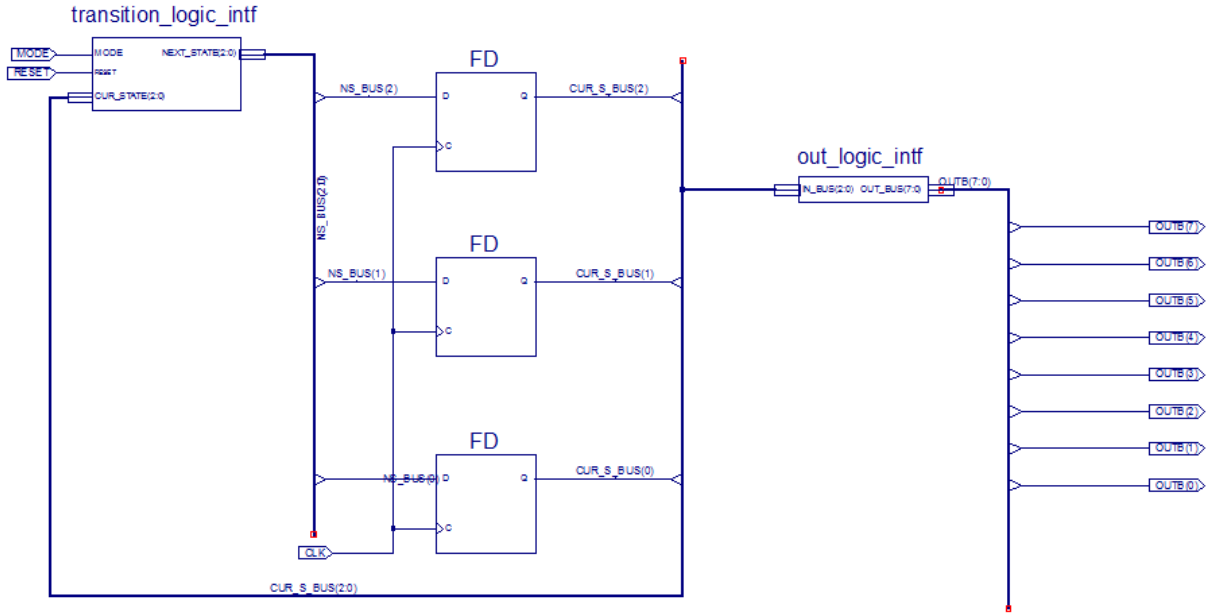


***Рис. 3 Схема LightController***

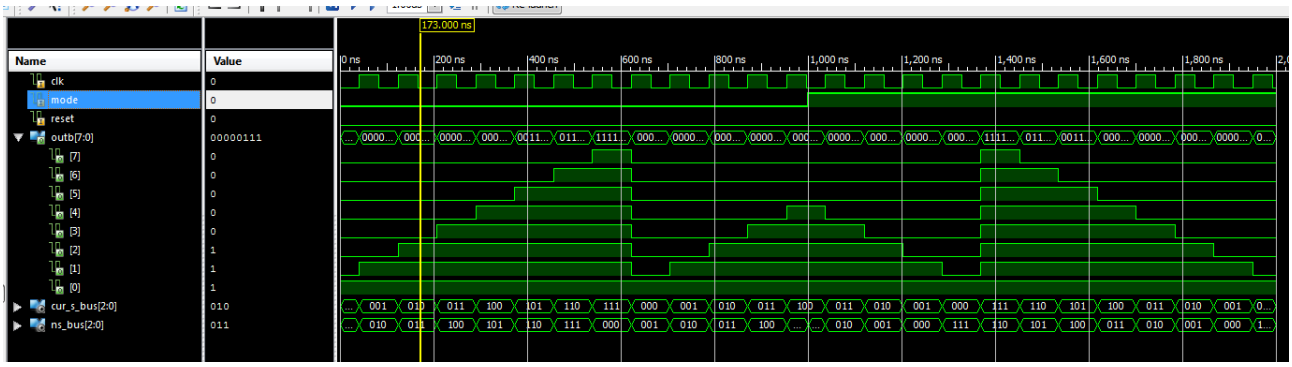


*Рис. 4 Діаграма проведеної симуляції для LightController*

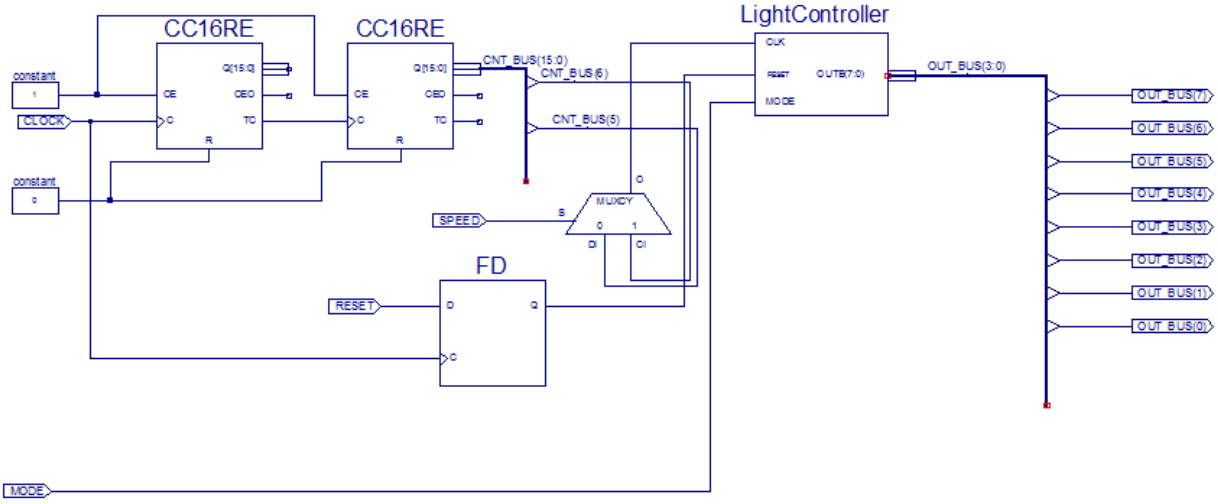


*Рис. 5 Схема TopLevel*

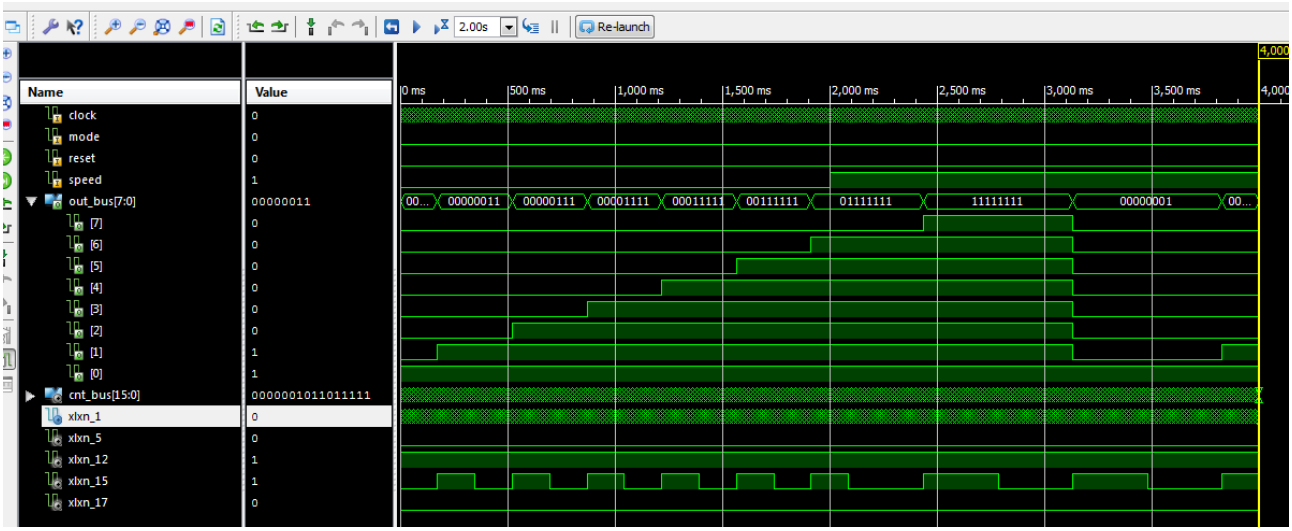


*Рис. 6 Діаграма проведеної симуляції для TopLevel*

## *Створення Constant файлу:*

```
# 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 "CLOCK"                 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 "SPEED"        LOC = P80   | PULLUP  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "RESET"        LOC = P79   | PULLUP  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
```

**Висновок:** на цій лабораторній роботі, я на базі стенда Elbert V2 – Spartan 3A FPGA, реалізувала цифровий автомат світлових ефектів.