Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ЕОМ

Звіт

до лабораторної роботи № 2

з дисципліни «Моделювання комп’ютерних систем»
на тему:

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

Варіант №14

Виконав:
ст. гр. КІ-201
Когут Д.А.
Прийняв:
ст. викладач
каф. ЕОМ
Козак Н. Б.

Львів 2024

Мета роботи: На базі стенда реалізувати цифровий автомат світлових ефектів.

Згідно мого варіанту(2) потрібно реалізувати такі комбінації:

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

Та дотримуватися таких вимог:

- Пристрій повинен використовувати тактовий сигнал від мікроконтролера і знижувати частоту за допомогою внутрішнього подільника. Мікроконтролер є частиною стенда Elbert V2 – Spartan 3A 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.
    - Якщо MODE = 1 то стан пристрою декрементується по зростаючому фронту тактового сигналу пам’яті станів.
- 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 перемикачів
- Для керування сигналами RESET/SPEED використати будь які з кнопок

Виконання роботи:

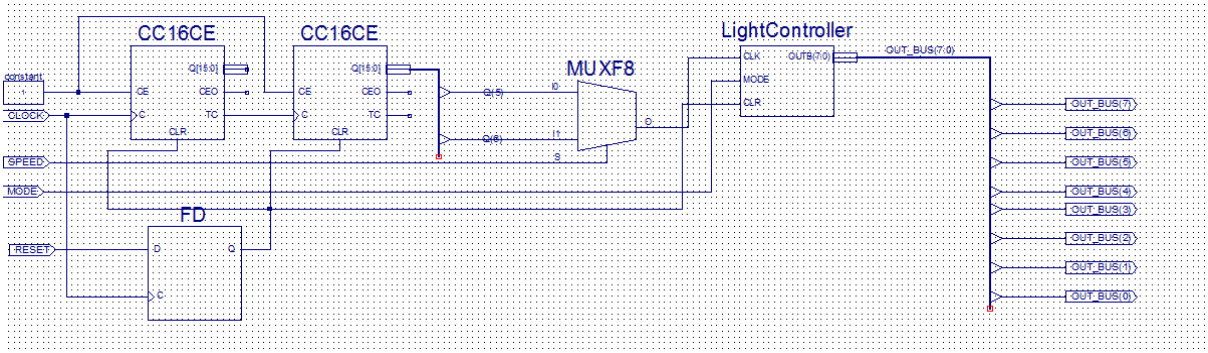

Рис. 1 – Top Level

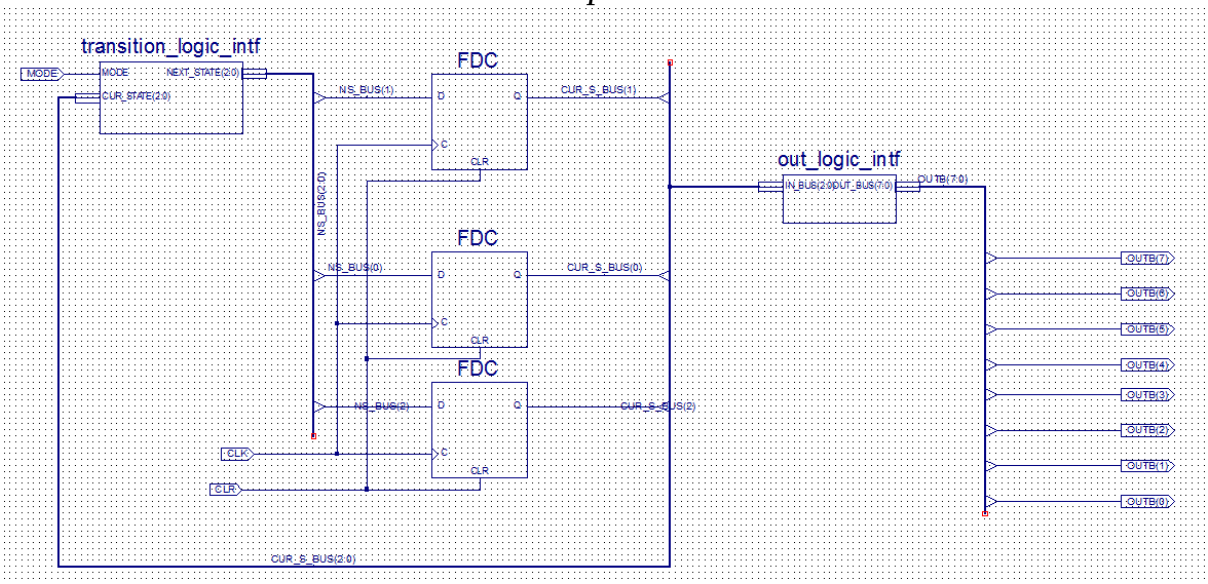

Рис. 2 – Light Controller

Файл TransitionLogic.vhd

```
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 CUR_STATE(2) and not CUR_STATE(1)
and not CUR_STATE(0)) or
                                                (not CUR_STATE(2) and
CUR_STATE(1) and not CUR_STATE(0)) or
                                                (CUR_STATE(2) and not
CUR_STATE(1) and not CUR_STATE(0)) or
                                                (CUR_STATE(2) and
CUR_STATE(1) and not CUR_STATE(0))) after 1ns;

        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 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 CUR_STATE(2)
and not CUR_STATE(1) and not CUR_STATE(0)) or
                                                (MODE and not CUR_STATE(2)
and CUR_STATE(1) and CUR_STATE(0))) after 1ns;

        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 not 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 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))) after 1ns;
end transition_logic_arch;
```

Файл OutputLogic.vhd

```
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) <= '1';
        OUT_BUS(1) <= not (not IN_BUS(2) and not IN_BUS(1) and not
IN_BUS(0)) after 1ns;
        OUT_BUS(2) <= not (not IN_BUS(2) and not IN_BUS(1)) after 1ns;
        OUT_BUS(3) <= not ((not IN_BUS(2) and not IN_BUS(1)) or (not
IN_BUS(2) and IN_BUS(1) and not IN_BUS(0))) after 1ns;
        OUT_BUS(4) <= IN_BUS(2) after 1ns;
        OUT_BUS(5) <= ((IN_BUS(2) and IN_BUS(1)) or (IN_BUS(2) and
not IN_BUS(1) and IN_BUS(0))) after 1ns;
        OUT_BUS(6) <= (IN_BUS(2) and IN_BUS(1)) after 1ns;
```

```
        OUT_BUS(7) <= (IN_BUS(2) and IN_BUS(1) and IN_BUS(0)) after
1ns;
end out_logic_arch;
```

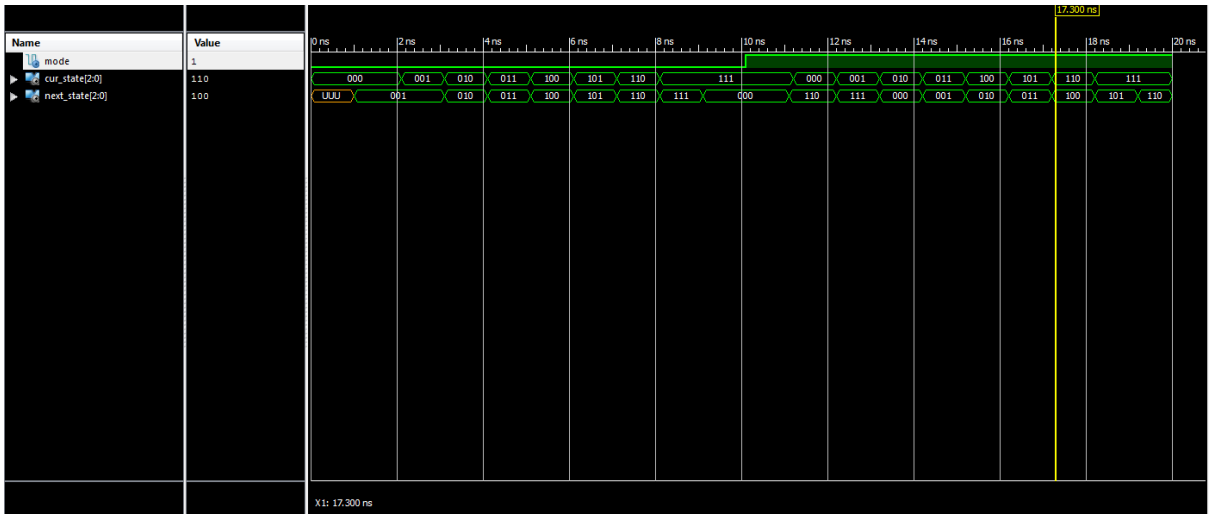

Рис. 3 – Часова діаграма TransitionLogic

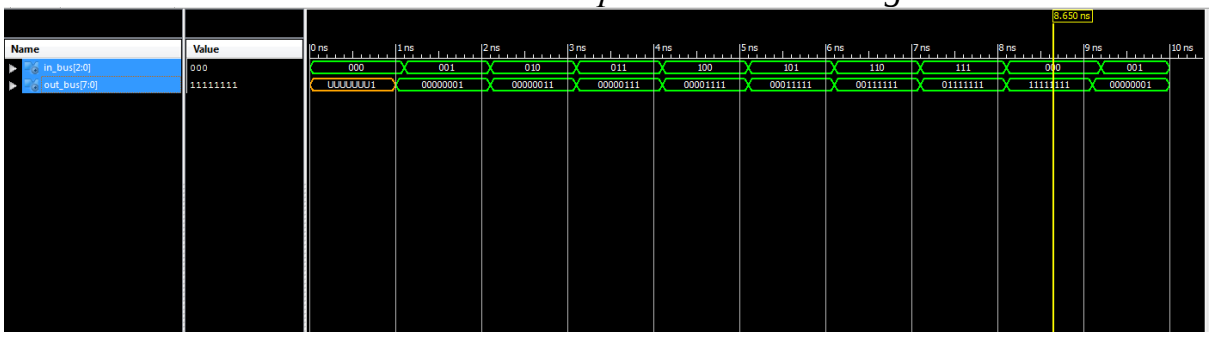

Рис. 4 – Часова діаграма OutputLogic

Рис. 5 – Часова діаграма LightController

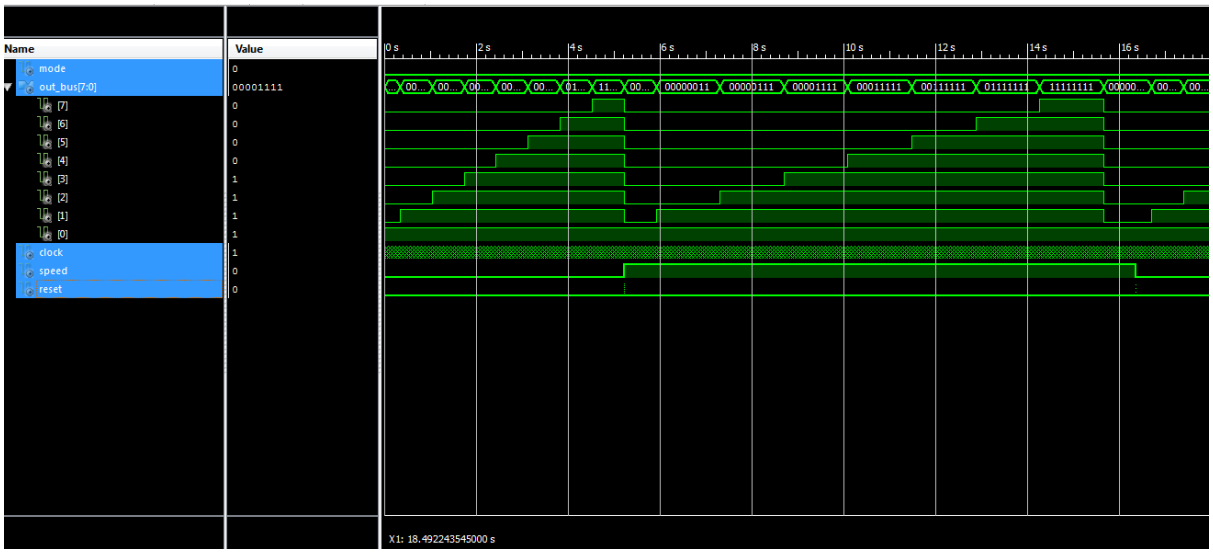

Рис 6. – Часова діграма TopLevel

```
Файл TestBechTopLevel.vsd
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
LIBRARY UNISIM;
USE UNISIM.Vcomponents.ALL;
ENTITY TopLevel_TopLevel_sch_tb IS
END TopLevel_TopLevel_sch_tb;
ARCHITECTURE behavioral OF TopLevel_TopLevel_sch_tb IS

  COMPONENT TopLevel
  PORT( MODE   :       IN     STD_LOGIC;
        OUT_BUS  :     OUT  STD_LOGIC_VECTOR (7 DOWNTO 0);
        CLOCK    :     IN     STD_LOGIC;
        SPEED    :     IN     STD_LOGIC;
        RESET    :     IN     STD_LOGIC);
  END COMPONENT;

  SIGNAL MODE :       STD_LOGIC;
  SIGNAL OUT_BUS    :       STD_LOGIC_VECTOR (7 DOWNTO 0);
  SIGNAL CLOCK      :       STD_LOGIC := '0';
  SIGNAL SPEED :      STD_LOGIC;
  SIGNAL RESET :      STD_LOGIC;

BEGIN
        CLOCK <= not CLOCK after 83ns;
  UUT: TopLevel PORT MAP(
                MODE => MODE,
                OUT_BUS => OUT_BUS,
```

```
                CLOCK => CLOCK,
                SPEED => SPEED,
                RESET => RESET
    );

-- *** Test Bench - User Defined Section ***
    tb : PROCESS
    BEGIN
                MODE <= '0';
                SPEED <= '0';
                RESET <= '1', '0' after 1ms;
                wait until RESET = '0';
                wait for 2ns;
                assert OUT_BUS = "00000001";
                wait for 348128us;
                assert OUT_BUS = "00000011";
                wait for 696255us;
                assert OUT_BUS = "00000111";
                wait for 696255us;
                assert OUT_BUS = "00001111";
                wait for 696255us;
                assert OUT_BUS = "00011111";
                wait for 696255us;
                assert OUT_BUS = "00111111";
                wait for 696255us;
                assert OUT_BUS = "01111111";
                wait for 696255us;
                assert OUT_BUS = "11111111";
                wait for 696255us;
                SPEED <= '1';
                RESET <= '1', '0' after 1ms;
                wait until RESET = '0';
                wait for 2ms;
                assert OUT_BUS = "00000001";
                wait for 1392509us;
                assert OUT_BUS = "00000011";
                wait for 1392509us;
                assert OUT_BUS = "00000111";
                wait for 1392509us;
                assert OUT_BUS = "00001111";
                wait for 1392509us;
                assert OUT_BUS = "00011111";
                wait for 1392509us;
```

```
            assert OUT_BUS = "00111111";
            wait for 1392509us;
            assert OUT_BUS = "01111111";
            wait for 1392509us;
            assert OUT_BUS = "11111111";
            wait for 1392509us;
            SPEED <= '0';
            RESET <= '1', '0' after 167ns;
            wait until RESET = '0';
   END PROCESS;
-- *** End Test Bench - User Defined Section ***

END;
```

Висновок: Виконуючи дану лабораторну роботу я навчився реалізовувати цифровий автомат світлових ефектів використовуючи засоби VHDL.