LAB 5 REPORT

EEE 102

Robin Umut Kızıl

22003260

Purpose of Lab:

In this lab, aim is the get use to use multiple seven-segment displays simultaneously while using basys3 with VHDL. while creating this simultaneous circuit, clock-reset structure will be used.

Design Specification:

In basys3's datasheet, it is shown that 7 segment display is connected by 'common anode' method, as shown in the figure 1. In this method, every led of the segments have same anodes and different cathodes. In order to light a led connected, the anode connected to it must be turned into HIGH and the cathode connected to it must be turned into LOW voltage. Additionally, anodes connected to inverter (mentioned in datasheet), so in the code, in order to light a led, both anode and cathode have to be LOW.

To create Persistence of vision, every segment of the 7 segment will be turn of in order with 16 Hz, so eye of the human cannot detect of the light change of leds. To create this illusion, CLK and RESET structure will be used.

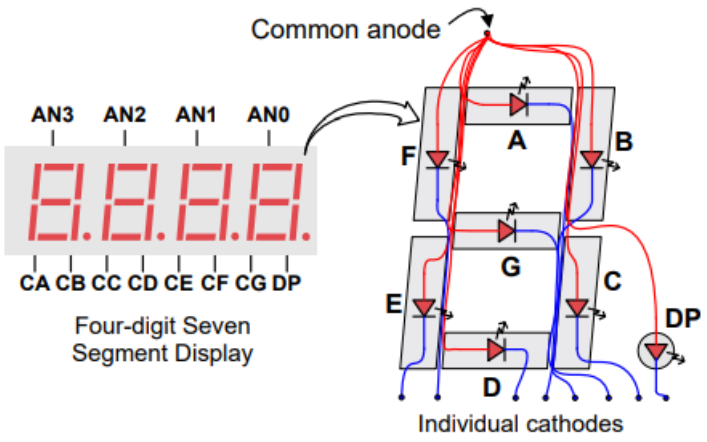

Figure 1 : Common anode connection in basys3

Answers for Questions:

1. Internal clock frequency of basys3 is 100MHz.
2. I create a loop that triggered when rising edge clock. Then I create, 16-bit clock counter signal that increase 1 every trigger. In this way, I get approximately 16 Hz clock which is lesser than the internal clock.
3. We cannot create every arbitrary frequency, We can only create clock frequencies that are the same as the input frequency or divide by powers of two as we do in this lab. We can also do proportional clock division operations by defining an integer that divides the clock frequency.

Methodology:

First SEVEN_SEGMENT Top module created with the port 16-bit switches (to control led display with switches), clk and reset, 4-bit anodes (to choice segment of display that show displayed), and 7-bit cathodes (to choice which led of segment will be light). Then another Component of decoder will be created to decode 4-bit switches to 7-bit leds. Then this component is added to top module and every 4 switches are decoded and represented with 4 different signals which are digit3_led, digit2_led, digit1_led, digit0_led.

Then clock cycle is coded. 16-bit clock counter is created and most significant 2 digits are connected as an anode mux. In this way, İnternal clock of basys3 with the values of 100MHz is decreased to approximately 16 Hz and this will be creating Persistence of vision.

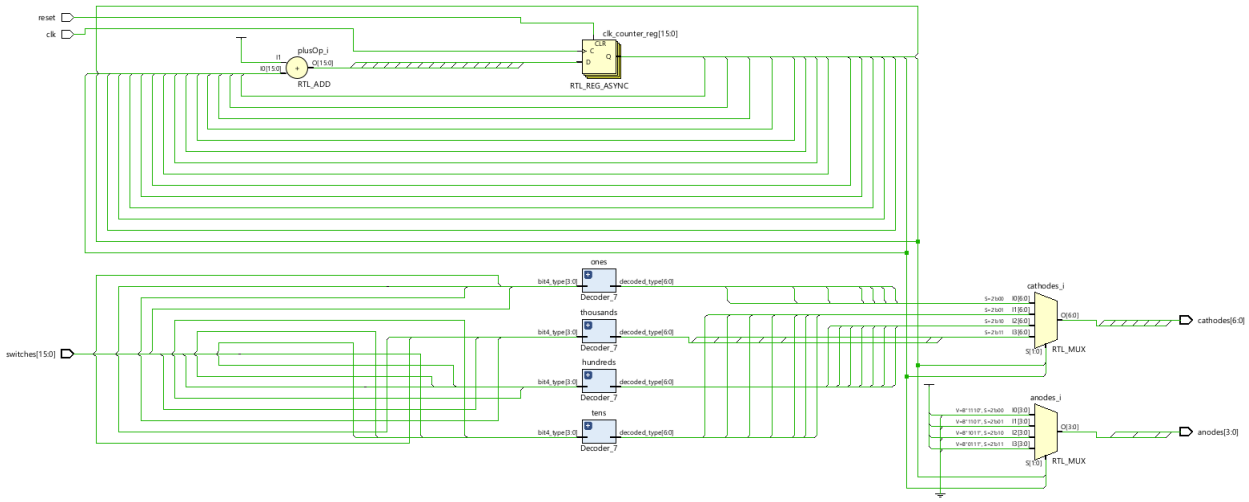

Figure 2 : RTL Schematic

Then segments are lighted while considering the anode mux. This creates the illusion of the eye, where every segment is fired at the same time, but showing different values.

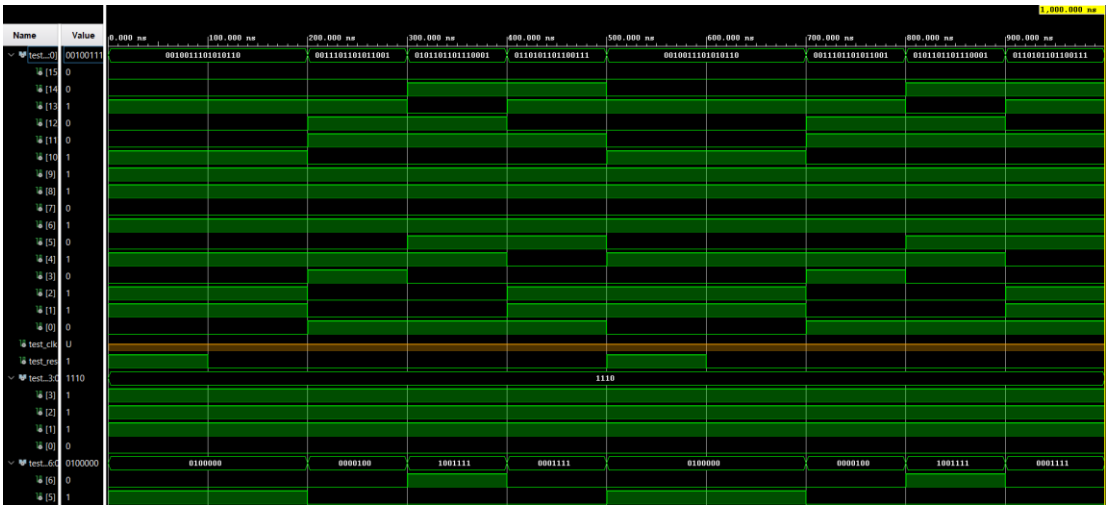

Figure 3 : First picture of Test bench of 7 segment display with the values of 2756 ,3959 ,5971 , 6567

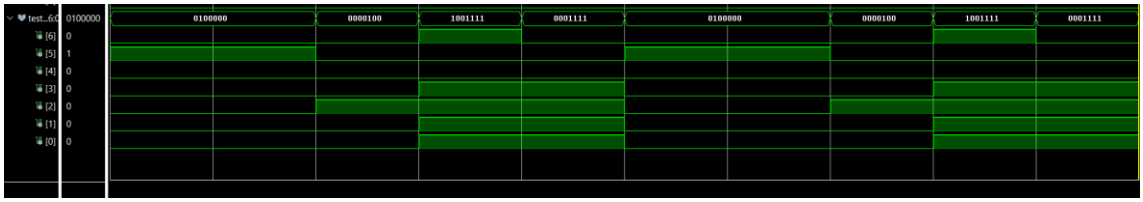

Figure 4 : Second picture of Test bench of 7 segment display with the values of 2756 ,3959 ,5971 , 6567

Conclusion:

7 segment display code has been successfully created, each segment shows different values independently from each other and these values are controlled by switches. The clock value was changed to a value close to 16Hz, which made Persistence of vision. That's why this lab has been a success.

Appendices A:

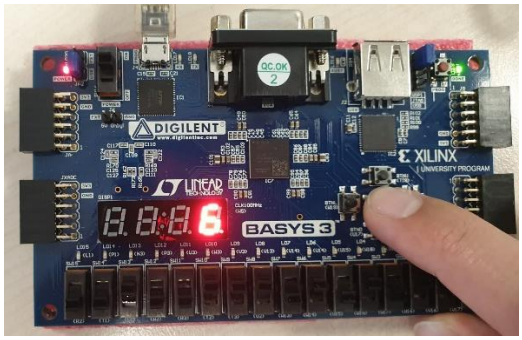

Figure 5 : When reset = '1'

While reset is ‘1’, 7 segment display shows the initial state which is anode “00” (which is shows just the less significant digit).

Figure 6: Situation of 3959

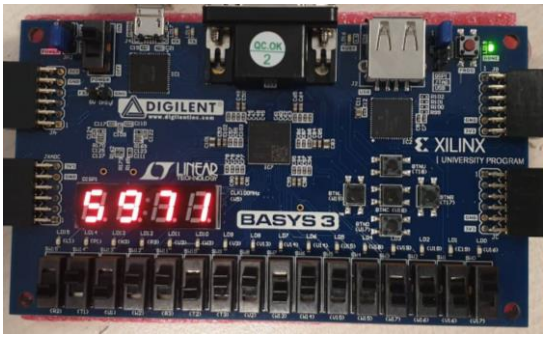

Figure 7 : Situation of 5971

Figure 8 : Situation of 6567

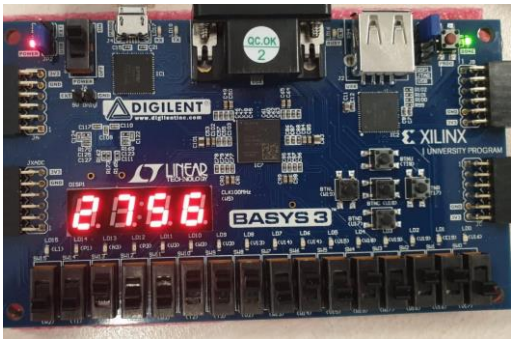

Figure 9 : Situation of 2756

Appendices B:

1. **SEVEN_SEGMENT CODE**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;


entity SEVEN_SEGMENT is

   Port ( switches : std_logic_vector(15 downto 0);
          clk : in STD_LOGIC;
          reset : in STD_LOGIC;
          anodes : out STD_LOGIC_VECTOR (3 downto 0);
          cathodes : out STD_LOGIC_VECTOR (6 downto 0)
         );

end SEVEN_SEGMENT;

architecture Behavioral of SEVEN_SEGMENT is

component Decoder_7
   Port ( bit4_type : in STD_LOGIC_VECTOR (3 downto 0);
          decoded_type : out STD_LOGIC_VECTOR (6 downto 0));
end component;

signal digit3, digit2, digit1, digit0 : std_logic_vector(3 downto 0);
signal digit3_led, digit2_led, digit1_led, digit0_led : std_logic_vector(6 downto 0);
signal anode_mux : std_logic_vector(1 downto 0);
signal clk_counter : std_logic_vector(15 downto 0);



begin

digit3 <= switches(15 downto 12);
digit2 <= switches(11 downto 8);
digit1 <= switches(7 downto 4);
digit0 <= switches(3 downto 0);
```

```
thousands : Decoder_7 port map(bit4_type =>digit3, decoded_type => digit3_led);
hundreds :  Decoder_7 port map(bit4_type =>digit2, decoded_type => digit2_led);
tens : Decoder_7 port map(bit4_type =>digit1, decoded_type => digit1_led);
ones : Decoder_7 port map(bit4_type =>digit0, decoded_type => digit0_led);


process(clk, reset)

 begin

 if reset = '1' then
   clk_counter <= (others => '0');

 elsif clk'event and clk = '1' then
   clk_counter <= clk_counter + 1;

 end if;

end process;

 anode_mux <= clk_counter(15 downto 14);


process(anode_mux,digit3_led, digit2_led, digit1_led, digit0_led)
begin

case anode_mux is

    when "00" => anodes <= "1110";
    cathodes <= digit0_led;
    when "01" => anodes <= "1101";
    cathodes <= digit1_led;
    when "10" => anodes <= "1011";
    cathodes <= digit2_led;
    when "11" => anodes <= "0111";
    cathodes <= digit3_led;
    when others => anodes <= "1111";

 end case;
```

```
end process;

end Behavioral;
```

2. **Decoder_7 CODE**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity Decoder_7 is
   Port ( bit4_type : in STD_LOGIC_VECTOR (3 downto 0);
          decoded_type : out STD_LOGIC_VECTOR (6 downto 0));
end Decoder_7;

architecture Behavioral of Decoder_7 is

begin

with bit4_type select

   decoded_type  <= "0000001" when "0000",
 "1001111" when "0001",
 "0010010" when "0010",
 "0000110" when "0011",
 "1001100" when "0100",
 "0100100" when "0101",
 "0100000" when "0110",
 "0001111" when "0111",
 "0000000" when "1000",
 "0000100" when "1001",
 "0000100" when others;



end Behavioral;
```

3. **Testbench_7_segment CODE**

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

```
entity Testbench_7_segment is
end Testbench_7_segment;

architecture Behavioral of Testbench_7_segment is

component SEVEN_SEGMENT
   Port ( switches : std_logic_vector(15 downto 0);
          clk : in STD_LOGIC;
          reset : in STD_LOGIC;
          anodes : out STD_LOGIC_VECTOR (3 downto 0);
          cathodes : out STD_LOGIC_VECTOR (6 downto 0)
         );
end component;

signal test_switches : std_logic_vector(15 downto 0);
signal test_clk, test_reset : STD_LOGIC;
signal test_anodes : STD_LOGIC_VECTOR (3 downto 0);
signal test_cathodes : STD_LOGIC_VECTOR (6 downto 0);

begin

test : SEVEN_SEGMENT port map(switches => test_switches,clk => test_clk, reset =>
test_reset, anodes => test_anodes, cathodes => test_cathodes );

sevensegment_TB : process
begin

test_reset <= '1';
test_switches <= "0010011101010110";
wait for 100 ns;

test_reset <= '0';
test_switches <= "0010011101010110";
wait for 100 ns;

test_reset <= '0';
test_switches <= "0011101101011001";
wait for 100 ns;

test_reset <= '0';
test_switches <= "0101101101110001";
```

```
wait for 100 ns;

test_reset <= '0';
test_switches <= "0110101101100111";
wait for 100 ns;


end process;

end Behavioral;
```