Eray Gündoğdu 22101962 22.11.2022

EEE102 LAB05:SEVEN SEGMENT DISPLAY

Purpose:

Aim of this experiment is to get a better understanding of seven segment display and design a four digit seven segment display counter.

Research Questions:

-What is the internal clock frequency of Basys 3?

Internal clock frequency of BASYS3 is 100MHz.

- How can you create a slower clock signal from this one?

By initializing a n bit logic vector from the value “00000…0”.By incrementing this value by one in each rising edge of the clock,a clock with $100MHz/2^n$ is obtained.

-- Can you create a clock with any arbitrary frequency lower than that of the internal clock? If not, which frequencies can you create?

Only clock frequencies with 100MHz/n can be obtained where n is an integer.

Methadology:

The working mechanism of seven segment display is very clear.In each segment there are 7 LEDs.They are controlled by signalization of anodes and cathodes of the LEDs.Anodes control the segments,cathodes control the LEDs.Initially the frequency of the clock in this design is determined as 100MHz.In order to turn anodes on and off a phase clock is selected .For the last clock,another signal is written in order to count seconds.Upcoming steps are writing the codes for segment driver and seven segment LED configuration(test1a and test1b).Later on testbench code is written for obtaining a simulation.Finally constraints file is written and design is implemented on BASYS3.

Variables in the Design:

Inputs and Outputs:

“clock”(input)(test1)

“rst”(input)(test1)

“anode_actv”(output)(test1)

“ledout”(output)(test1)

“clock_count”(input).(test1a)

“number_x”(input)(test1a)

“resret”(input)(test1a)

“anode”(output)(test1a)

“led”(output)(test1a)

“a”(input),”b”(output)(test1b),names are meaningless these are only for obtaining 4 bit and 7 bit input and output.

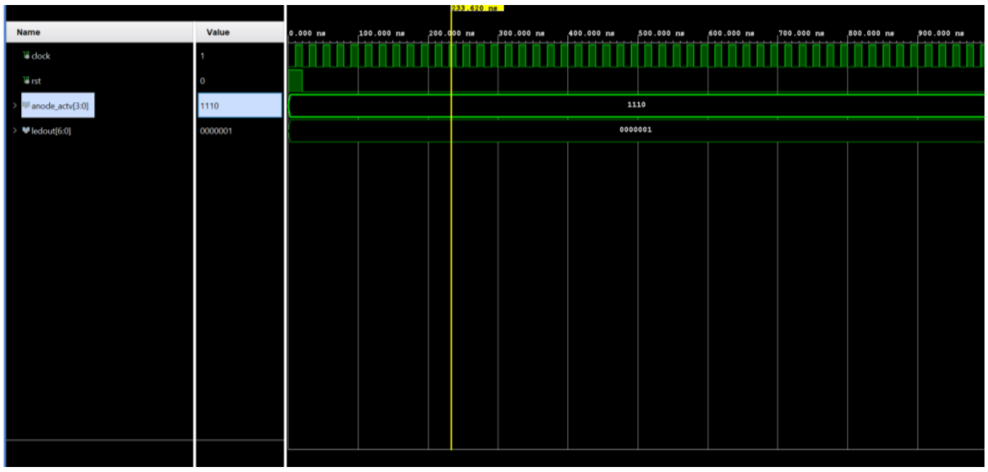
Signals

“enable”,” second_num”,” number_count”,” Ledactivating”(test1)

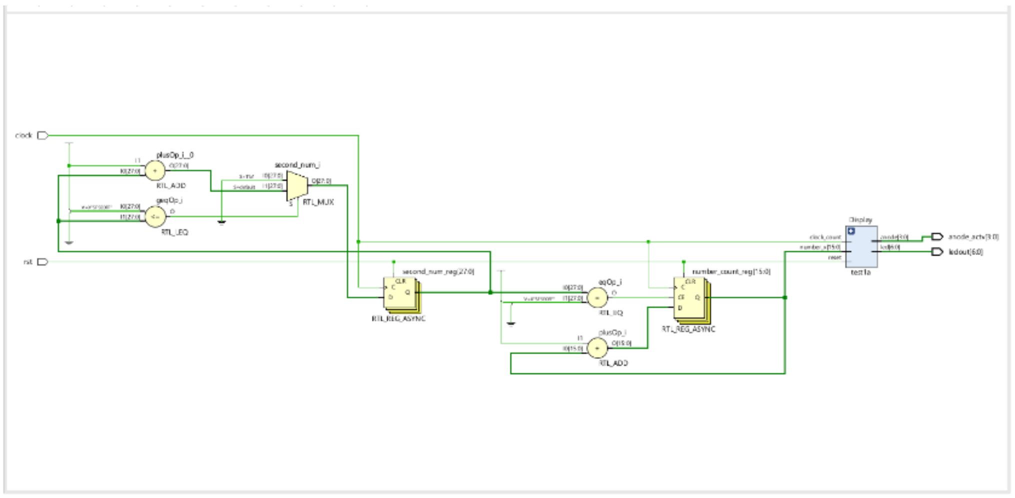
“Led_Bcd”(test1a),” refresh”(test1a),” sel”(test1a)

Results:

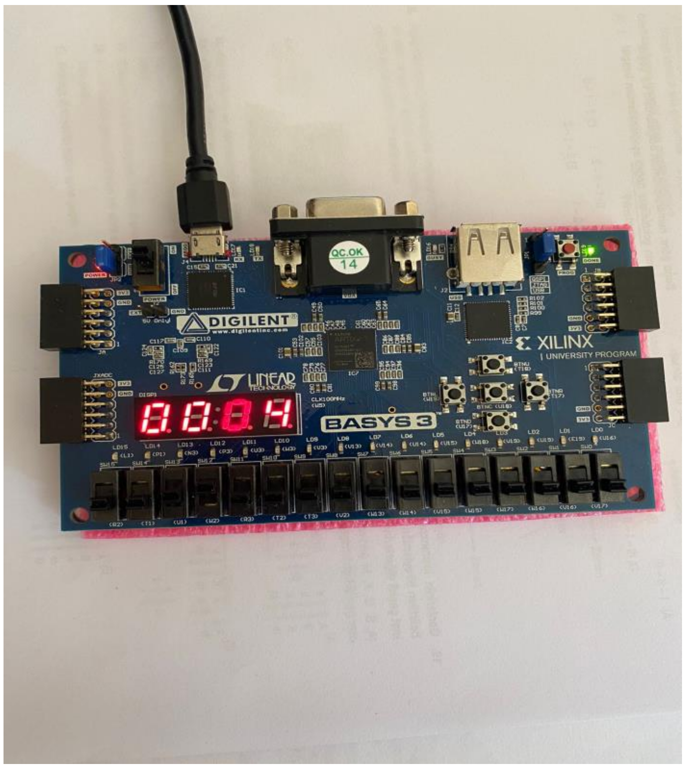
1)Simulation: This simulation demonstrates the relation between anode controller,cathode controller and clock.



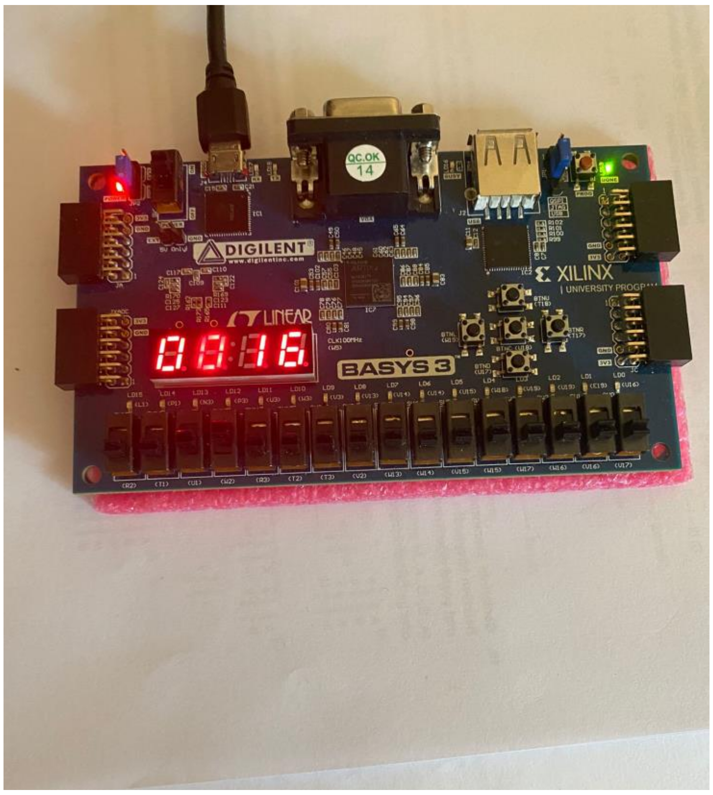
2)RTL Schematic



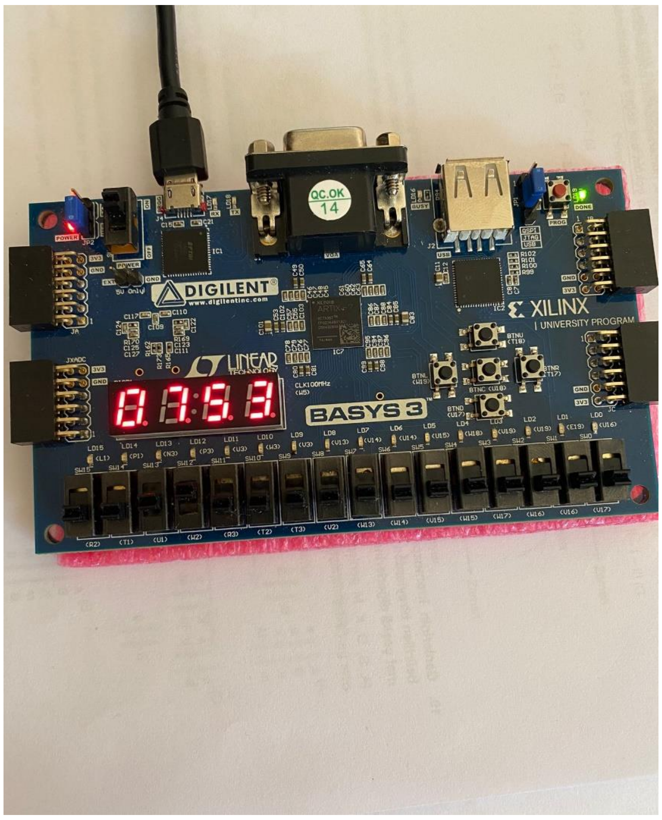
3)Demonstration of 4 digit seven segment display on BASYS3:

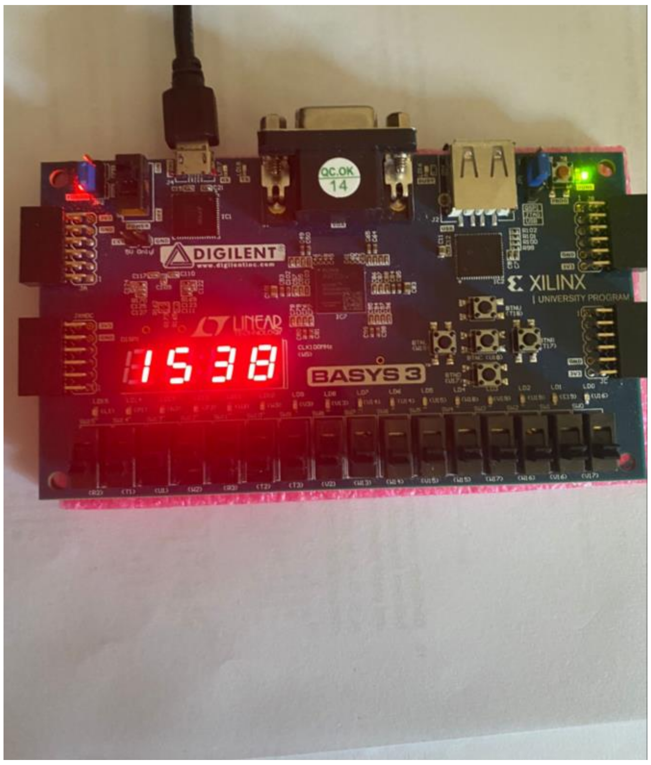


1 digit demonstration



2 digit demonstration


3 digit demonstration


4 digit demonstration

Conclusion:

In this experiment,ımportant concepts in digital design is familiarized such as decoder,multiplexer,clock implementation.Personally ı gained detailed information about seven segment display.

Appendices

Codes of test1(four bit counter):

```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;

use IEEE.std_logic_unsigned.all;




entity test1 is

   Port ( clock : in STD_LOGIC;

          rst : in STD_LOGIC;

          anode_actv : out STD_LOGIC_VECTOR (3 downto 0);

          ledout : out STD_LOGIC_VECTOR (6 downto 0));

end test1;


architecture Behavioral of test1 is


signal second_num : std_logic_vector(27 downto 0);




signal enable : std_logic;



signal number_count: std_logic_vector(15 downto 0);
signal Ledactivating: std_logic_vector(1 downto 0);

Component test1a is
   Port (
         number_x : in std_logic_vector(15 downto 0);
         reset: in std_logic;
         anode : out STD_LOGIC_VECTOR (3 downto 0);
         clock_count: in std_logic;
         led : out STD_LOGIC_VECTOR (6 downto 0)
         );
 end Component;

begin
Display: test1a
   Port Map(
      number_x => number_count,
      reset => rst,
      anode => anode_actv,
      clock_count => clock,
      led => ledout);



process(clock, rst)
begin
   if(rst='1') Then
      second_num <= (others => '0');
   elsif(rising_edge(clock)) Then
      if(second_num>=x"5F5E0FF") then
            second_num <= (others => '0');
            else
                second_num<= second_num + "0000001";
            end if;
         end if;
end Process;
enable <= '1' when second_num=x"5F5E0FF" else '0';

process(clock, rst)
begin
    if(rst='1') Then
       number_count <= (others => '0');
    elsif(rising_edge(clock)) Then
        if(enable = '1') then
                number_count <= number_count + x"0001";
         end if;
    end if;
end process;
end Behavioral;
```

Codes of test1a(segment driver):

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



entity test1a is
    Port (
           number_x : in std_logic_vector(15 downto 0);
           reset: in std_logic;
           anode : out STD_LOGIC_VECTOR (3 downto 0);
           clock_count: in std_logic;
           led : out STD_LOGIC_VECTOR (6 downto 0)
```

```
        );
end test1a;

architecture Behavioral of test1a is

signal Led_Bcd : STD_LOGIC_VECTOR(3 downto 0);

Component test1b is
   Port(
   a : in STD_LOGIC_VECTOR (3 downto 0);
   b : out STD_LOGIC_VECTOR (6 downto 0)
   );
   end Component;


signal refresh: STD_LOGIC_VECTOR (19 downto 0);



Signal sel : STD_LOGIC_VECTOR (1 downto 0);

begin



process(clock_count, reset)
begin
   if(reset = '1') Then
      refresh <= (others => '0' );
    elsif(rising_edge(clock_count)) then
      refresh <= refresh + 1;
     end if;
end process;
```

```
sel <= refresh(19 downto 18);


process(sel)
begin
   case sel is
      when "00" => anode <= "1110";


      Led_Bcd <= number_x(15 downto 12);
      when "01" => anode <= "1101";


      Led_Bcd <= number_x(11 downto 8);
      when "10" => anode <= "1011";


      Led_Bcd <= number_x(7 downto 4);
      when "11" => anode <= "0111";


      Led_Bcd <= number_x(3 downto 0);
      when others => anode <= "1111";
   end case;
 end process;


onebitseven_segment: test1b Port Map(Led_Bcd,led);


end Behavioral;
```

Codes of test1b(seven segment configuration):

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



entity test1b is
   Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
          b : out STD_LOGIC_VECTOR (6 downto 0));
end test1b;

architecture Behavioral of test1b is


begin
process(a)
begin
   case a is
       when "0000" => b <= "0000001";
       when "0001" => b <= "1001111";
       when "0010" => b <= "0010010";
       when "0011" => b <= "0000110";
       when "0100" => b <= "1001100";
       when "0101" => b <= "0100100";
       when "0110" => b <= "0100000";
       when "0111" => b <= "0001111";
       when "1000" => b <= "0000000";
       when "1001" => b <= "0000100";
       when "1010" => b <= "0000010";
       when "1011" => b <= "1100000";
       when "1100" => b <= "0110001";
       when "1101" => b <= "1000010";
       when "1110" => b <= "0110000";
       when others => b <= "0111000";
    end case;
 end process;


end Behavioral;
```

Codes of testbench:

```
library IEEE;
```

```
use IEEE.STD_LOGIC_1164.ALL;



entity testbench1 is

end testbench1;

architecture Behavioral of testbench1 is

Component test1 is
   Port ( clock : in STD_LOGIC;
          rst : in STD_LOGIC;
          anode_actv : out STD_LOGIC_VECTOR (3 downto 0);
          ledout : out STD_LOGIC_VECTOR (6 downto 0));
end Component;



Signal clock: std_logic := '0' ;
Signal rst: std_logic := '0';




Signal anode_actv : std_logic_vector(3 downto 0);
Signal led: std_logic_vector(6 downto 0);

begin

uut: test1 Port Map(
   clock => clock,
   rst => rst,
   anode_actv => anode_actv,
   ledout => led
);

clk_process: process
begin
   clock <= '0';
      wait for 10ns;
   clock <= '1';
      wait for 10ns;
end Process;



stim_proc: process
begin
   rst <= '1';
      wait for 20ns;
   rst <= '0';
      wait;
end Process;
end Behavioral;
```

Codes of constraints:

```
set_property PACKAGE_PIN W5 [get_ports clock]
set_property IOSTANDARD LVCMOS33 [get_ports clock]
set_property PACKAGE_PIN V17 [get_ports rst]
set_property IOSTANDARD LVCMOS33 [get_ports rst]




set_property PACKAGE_PIN W7 [get_ports {ledout[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {ledout[6]}]
set_property PACKAGE_PIN W6 [get_ports {ledout[5]}]

set_property IOSTANDARD LVCMOS33 [get_ports {ledout[5]}]

set_property PACKAGE_PIN U8 [get_ports {ledout[4]}]

set_property IOSTANDARD LVCMOS33 [get_ports {ledout[4]}]

set_property PACKAGE_PIN V8 [get_ports {ledout[3]}]

set_property IOSTANDARD LVCMOS33 [get_ports {ledout[3]}]

set_property PACKAGE_PIN U5 [get_ports {ledout[2]}]

set_property IOSTANDARD LVCMOS33 [get_ports {ledout[2]}]

set_property PACKAGE_PIN V5 [get_ports {ledout[1]}]

set_property IOSTANDARD LVCMOS33 [get_ports {ledout[1]}]

set_property PACKAGE_PIN U7 [get_ports {ledout[0]}]

set_property IOSTANDARD LVCMOS33 [get_ports {ledout[0]}]


set_property PACKAGE_PIN U2 [get_ports {anode_actv[3]}]

set_property IOSTANDARD LVCMOS33 [get_ports {anode_actv[3]}]

set_property PACKAGE_PIN U4 [get_ports {anode_actv[2]}]

set_property IOSTANDARD LVCMOS33 [get_ports {anode_actv[2]}]

set_property PACKAGE_PIN V4 [get_ports {anode_actv[1]}]

set_property IOSTANDARD LVCMOS33 [get_ports {anode_actv[1]}]

set_property PACKAGE_PIN W4 [get_ports {anode_actv[0]}]

set_property IOSTANDARD LVCMOS33 [get_ports {anode_actv[0]}]
```