Lab 5: Seven Segment Display

1) Purpose

Aim of this lab is to understand how to use 7 segment display on the Basys 3 FPGA board and design a digital circuit using 7 segment display. A design that can be show 4 different digits simultaneously should be made. In this lab a design that can take 16 values from 4 different switches for each 4 digits is used. The 4 digit display shows each digit simultaneously according to given 16 switch inputs (4 switch for each 4 digit).

2) Questions

1) What is the internal clock frequency of Basys 3?

a) A 100Mhz internal clock is included in Basys 3.

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

a) To create a slower clock frequency we can create another signal and assign its value to '1' and '0' according to a timer derived from the internal clock. For example, we can assign a number and increase it at the time when each rising edge of internal clock appears up to a constant value n. Then when the number reaches n the signal we define changes the value such as from '1' to '0' or vice versa. An example function is given below.

```
process(clk)
begin
 if(rising_edge(clk)) then
 divider <= divider + 1;
 end if;
 if (divider = n) then
 timer <= not(timer);
 Divider = 0;
end if;
end process;
```

Figure 2.1 Example lower frequency clock

3) Can you create a clock with any arbitrary frequency lower than that of the internal clock?

a) We can create lower frequencies using the method mentioned above. But its value will be 100MHz/n since in each cycle 1 rising edge appears in internal clock but we assign our signal to change value when n cycle appears. Therefore, we can get lower frequency signals with frequency 100MHz/n for n being positive integer.

3) Methodology

As the 4 digit's cathodes are in parallel it is impossible to display 4 different digits simultaneously. However, as human eye can detect 30-60 frames per second (30fps videos are 30 photos shown in 1 second and we interpret it as a continuous frame) and an optic effect called persistence of vision explained as as soon as photons stop encountering the eye, optic nerves do not respond simultaneously after a short period of time the perception of light disappears, we show each digit at a time with a frequency (100MHz/2^18 = 381.5Hz) higher than human eye detect. But we arrange the frequency so that the led of the 7 segment display is not damaged (frequency lower than 1kHz).

We assigned a 20 bit signal and incremented it by 1 with each rising edge appearing in internal clock of Basys 3 using IEEE.STD_LOGIC_UNSIGNED library. Then we used its most significant 2 digits as our digit selection signal with frequency (100MHz/2^18 = 381.5Hz). We used a decoder to choose which anode will be 0 and others 1 (o anode means digit is active and lighted on regarding to cathode signal). According to 2 bit signal we derived we choose digits and by using another multiplexer with the same selection signal we assign cathode signal derived from given 4 bit input for that digit by user revealing the

pattern of the digit. For each 4 digit and 16 characters determined by user given input assigned and sent to 7 segment display and shown with the frequency mentioned above. Schematic of the circuit can be seen in the figure below.

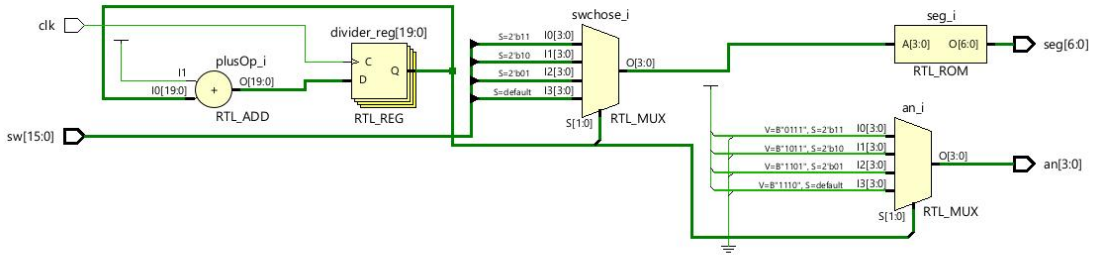

Figure 3.1 schematic of the design. sw user determined 4x4 input, clk internal clock of Basys 3, seg cathode output signal of 7 segment display, an anode output signal of 7 segment display

And the VHDL design code constraint file and test bench codes written in the appendix .

4) Results

Test bench simulation result and results gotten from implementation in Basys 3 can be seen the figures below. As in test bench mode signals assigned regarding to a given time period in order for the given 4 simulation value to be assigned its running time set to 1ms.

Figure 4.1 Test bench simulation results revealing true functionality of the design

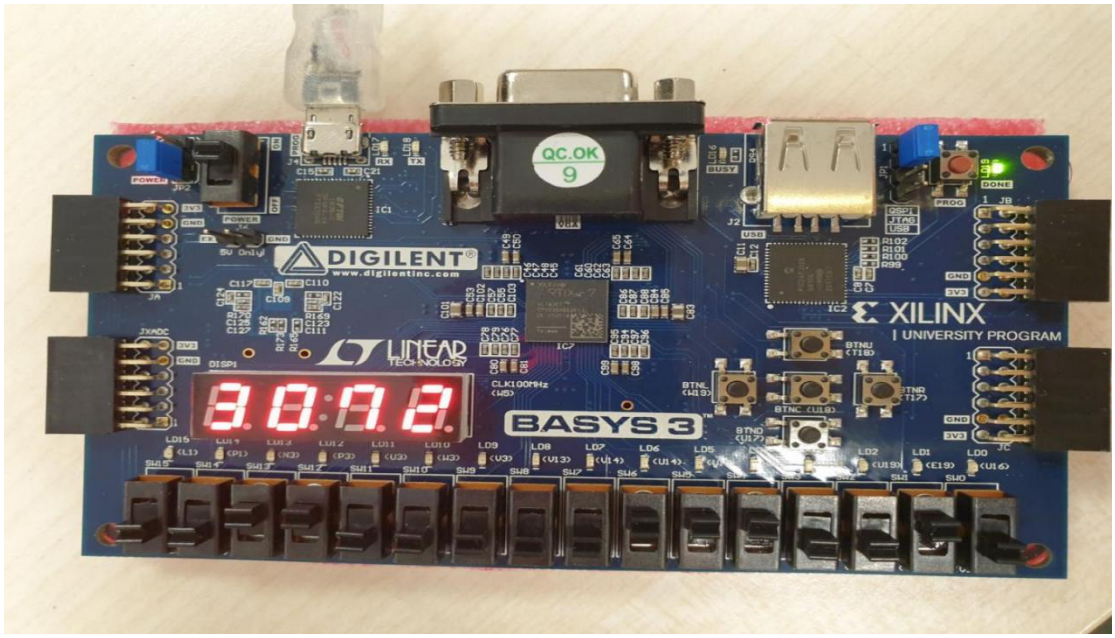

Figure 4.2 sw (0011 0000 0111 0010) showing 3072 as expected

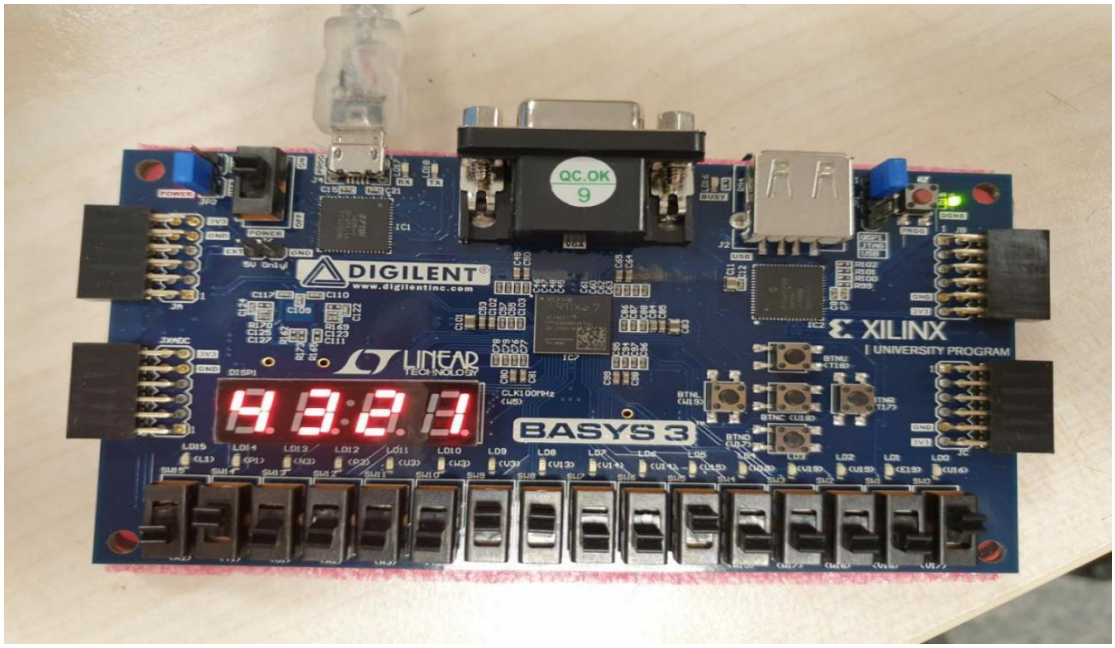

Figure 4.3 sw (0100 0011 0010 0001) showing 4321 as expected

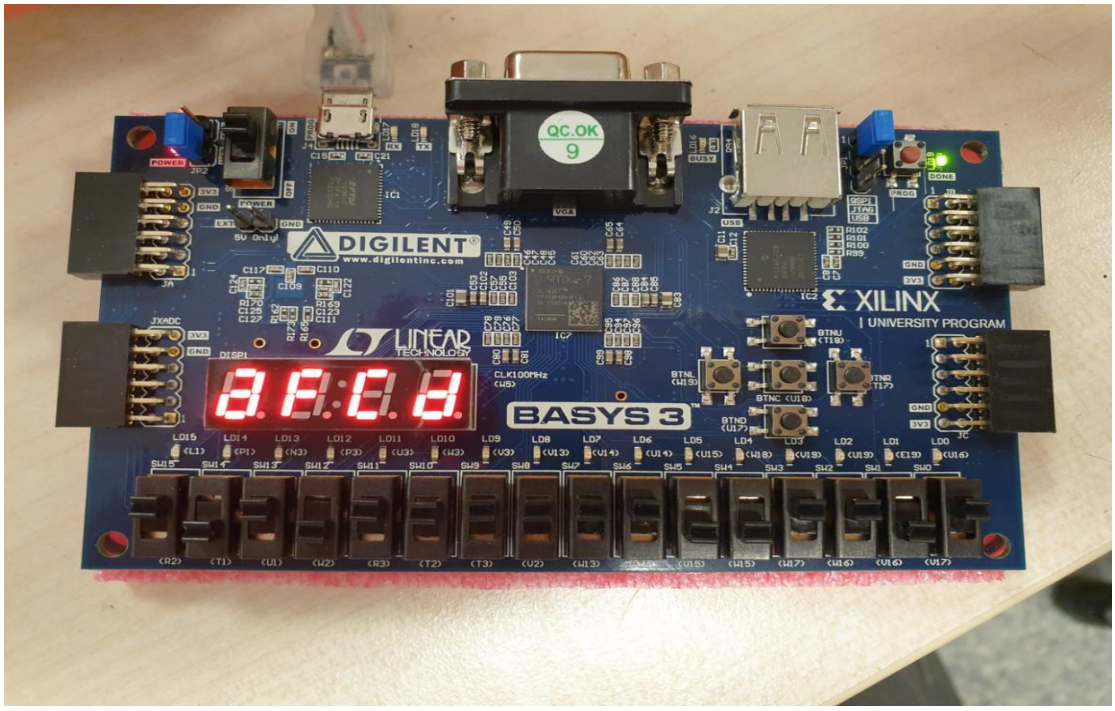

Figure 4.4 sw (1010 1111 1100 1101) showing afcd as expected (hexadecimal digits)

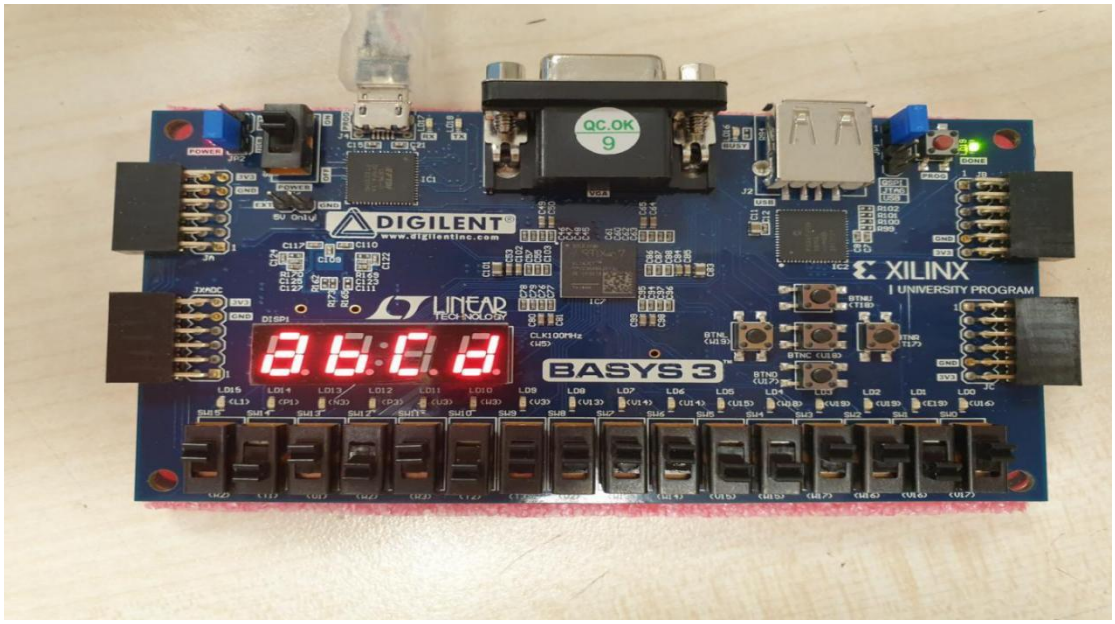

Figure 4.5 sw (1010 1011 1100 1101) showing abcd as expected (hexadecimal digits)

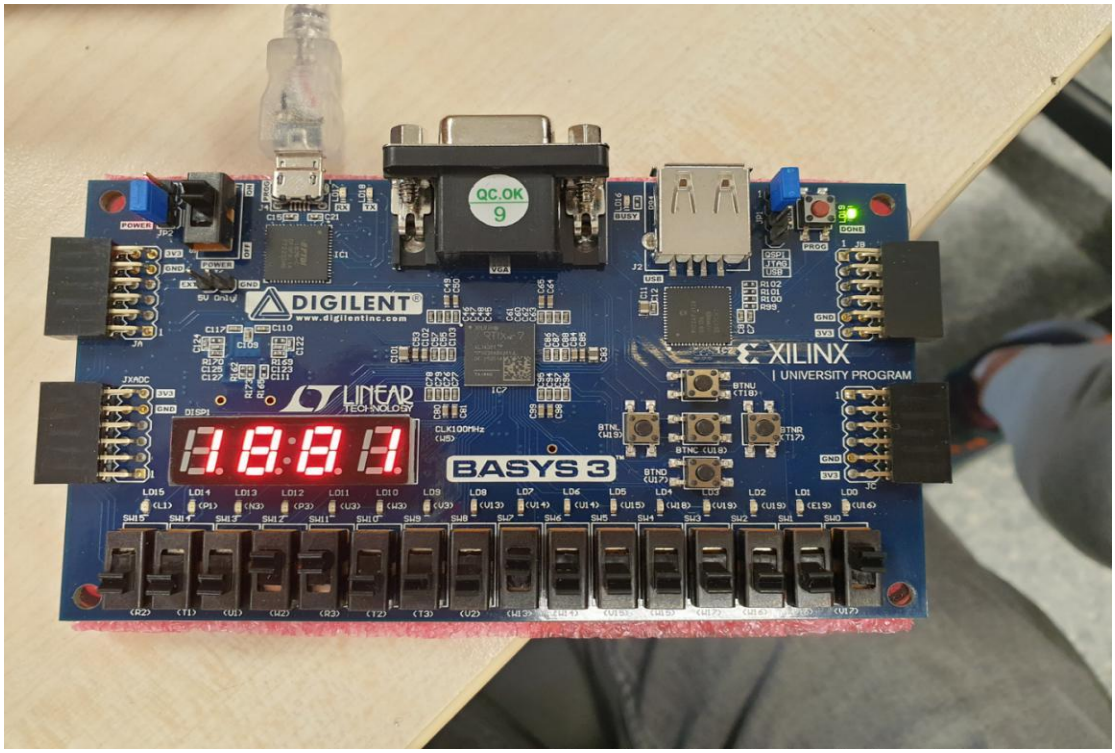

Figure 4.6 sw (0001 1000 1000 0001) showing 1881 as expected

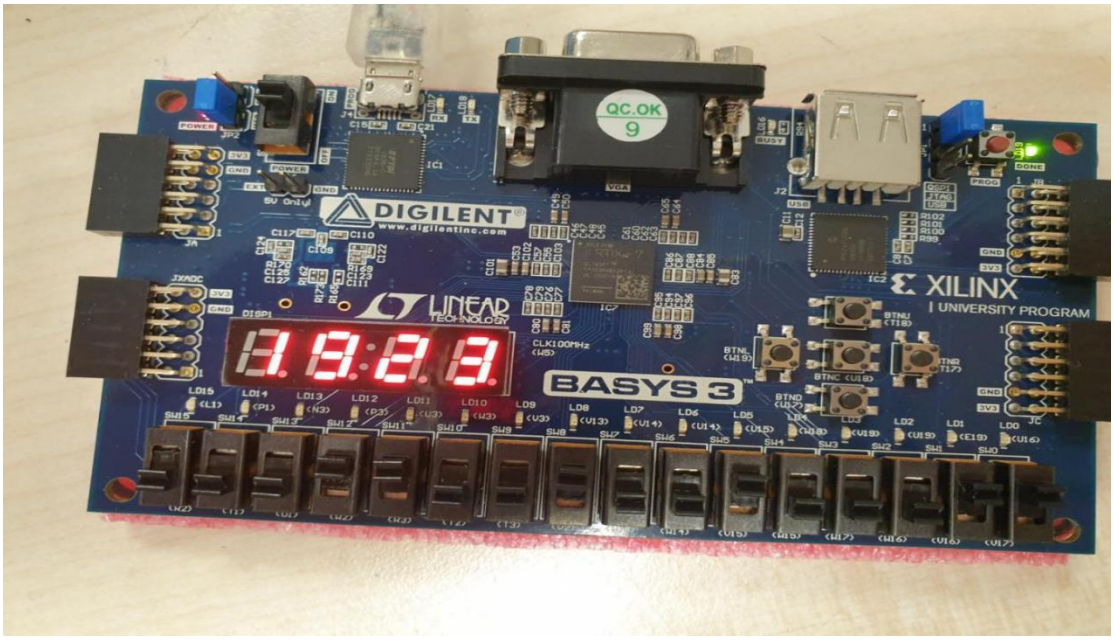

Figure 4.7 sw (0001 1001 0010 0011) showing 1923 as expected

5) Conclusion

In conclusion we learned internal structure of the 7 segment display (referance amnual of the Basys 3 board includes detailed explanations for the cathode parallelity anode configuration and the cathode signals that should be sent to display different configurations in each digit) and we learned how to divide internal clock signal of Basys 3 and used it for displaying 4 different digits simultaneously by using optic persistence phenomena of human eye.

6) Appendix

- Design code

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

entity deneme is
    Port ( sw : in std_logic_vector (15 downto 0);
           clk : in std_logic;
           seg : out std_logic_vector (6 downto 0);
           an : out std_logic_vector (3 downto 0));
end deneme;

architecture Behavioral of deneme is
signal divider: std_logic_vector (19 downto 0):="00000000000000000000";
signal swchose: std_logic_vector (3 downto 0):="0000";
begin

process(clk)
begin
 if(rising_edge(clk)) then
 divider <= divider + 1;
 end if;
end process;
```

```
with swchose select
 seg <= "0000001" when "0000", --0
        "1001111" when "0001", --1
        "0010010" when "0010", --2
        "0000110" when "0011", --3
        "1001100" when "0100", --4
        "0100100" when "0101", --5
        "0100000" when "0110", --6
        "0001101" when "0111", --7
        "0000000" when "1000", --8
        "0000100" when "1001", --9
        "0000010" when "1010", --A
        "1100000" when "1011", --B
        "0110001" when "1100", --C
        "1000010" when "1101", --D
        "0110000" when "1110", --E
        "0111000" when "1111", --F
        "1111111" when others; --default

with divider(19 downto 18) select
swchose <= sw(15 downto 12) when "11",
           sw(11 downto 8) when "10",
           sw(7 downto 4) when "01",
           sw(3 downto 0) when others;
```

```
process(divider(19 downto 18))
begin
case divider(19 downto 18) is
when "11" => an <= "0111";
when "10" => an <= "1011";
when "01" => an <= "1101";
when others => an <= "1110";
end case;
end process;

end Behavioral;
```

- Test bench code

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

entity test is
--  Port ( );
end test;

architecture Behavioral of test is
component deneme is
    Port ( sw : in std_logic_vector (15 downto 0);
           clk : in std_logic;
           seg : out std_logic_vector (6 downto 0);
```

```
        an : out std_logic_vector (3 downto 0));
end component deneme;

signal an : std_logic_vector(3 downto 0);
signal seg : std_logic_vector(6 downto 0);
signal sw : std_logic_vector(15 downto 0);
signal clk: std_logic := '0';
constant period : time := 0.1ns;

begin
UUT: deneme port map(
an => an,
seg => seg,
sw => sw,
clk => clk
);
clock : process
begin
wait for period /2;
clk <= '1';
wait for period/2;
clk <= '0';
end process;
test : Process
begin
```

```
sw <= "0000000000110000";
wait for 0.25 ms;
sw <= "0011000001110010";
wait for 0.25 ms;
sw <= "0010001000010000";
wait for 0.25 ms;
sw <= "1010101111001101";
wait for 0.25 ms ;
end process;
```

- Constraint file

```
set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
set_property PACKAGE_PIN V16 [get_ports {sw[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]
set_property PACKAGE_PIN W16 [get_ports {sw[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[2]}]
set_property PACKAGE_PIN W17 [get_ports {sw[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[3]}]
set_property PACKAGE_PIN W15 [get_ports {sw[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[4]}]
set_property PACKAGE_PIN V15 [get_ports {sw[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[5]}]
set_property PACKAGE_PIN W14 [get_ports {sw[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[6]}]
set_property PACKAGE_PIN W13 [get_ports {sw[7]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {sw[7]}]
set_property PACKAGE_PIN V2 [get_ports {sw[8]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[8]}]
set_property PACKAGE_PIN T3 [get_ports {sw[9]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[9]}]
set_property PACKAGE_PIN T2 [get_ports {sw[10]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[10]}]
set_property PACKAGE_PIN R3 [get_ports {sw[11]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[11]}]
set_property PACKAGE_PIN W2 [get_ports {sw[12]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[12]}]
set_property PACKAGE_PIN U1 [get_ports {sw[13]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[13]}]
set_property PACKAGE_PIN T1 [get_ports {sw[14]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[14]}]
set_property PACKAGE_PIN R2 [get_ports {sw[15]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[15]}]


set_property PACKAGE_PIN W7 [get_ports {seg[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[6]}]
set_property PACKAGE_PIN W6 [get_ports {seg[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[5]}]
set_property PACKAGE_PIN U8 [get_ports {seg[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[4]}]
```

```
set_property PACKAGE_PIN V8 [get_ports {seg[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[3]}]
set_property PACKAGE_PIN U5 [get_ports {seg[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[2]}]
set_property PACKAGE_PIN V5 [get_ports {seg[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[1]}]
set_property PACKAGE_PIN U7 [get_ports {seg[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[0]}]

set_property PACKAGE_PIN U2 [get_ports {an[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[0]}]
set_property PACKAGE_PIN U4 [get_ports {an[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[1]}]
set_property PACKAGE_PIN V4 [get_ports {an[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[2]}]
set_property PACKAGE_PIN W4 [get_ports {an[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {an[3]}]

set_property PACKAGE_PIN W5 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]
```