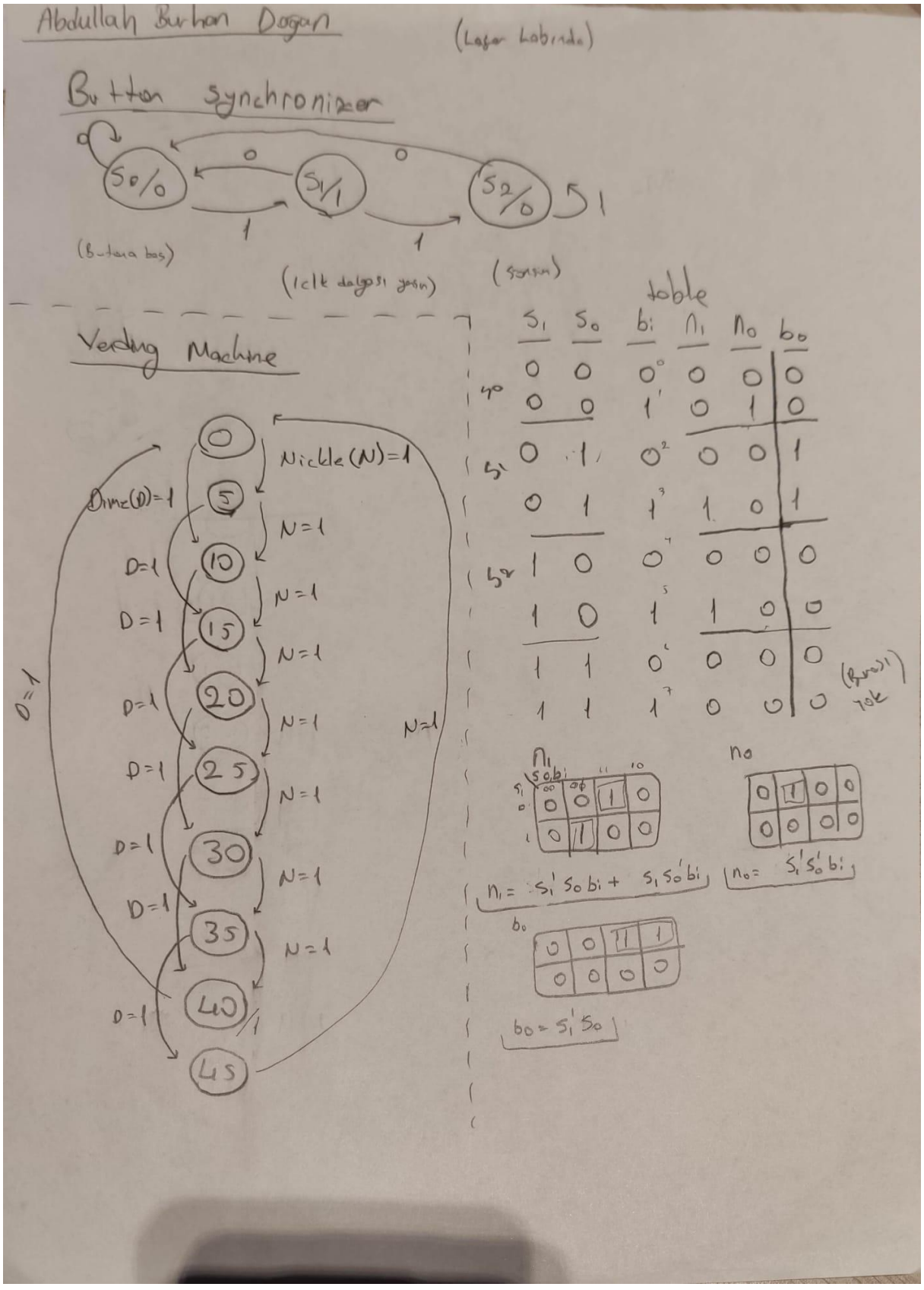
This endeavor encompasses the necessary VHDL files for the creation of a basic vending machine programmed to vend a 40-cent candy. The development was tailored for the BASYS3 board. To exhibit the inserted cents on the apparatus, a two-digit presentation approach was adopted, employing a pair of seven-segment displays affixed to the board.



Clk Divider

-VHD

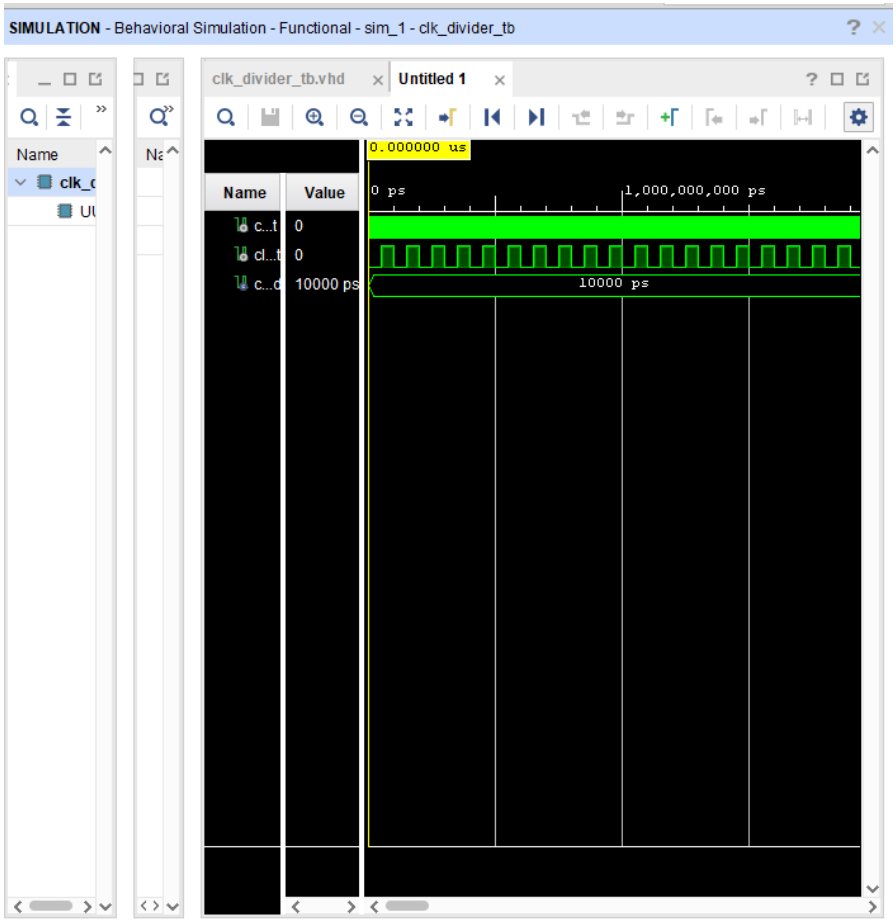
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity clk_divider is
port (clk_in : in std_logic;
      clk_out : out std_logic
    );
end clk_divider;

architecture Behavioral of clk_divider is

signal count : integer :=0;  --olayın gerçekleşme sayısını sayar
signal b : std_logic :='0';   -- değişken tanımlar ve başlangıç değeri olarak '0' atar

begin

-- 100MHz den 100 Hz üretir.
process(clk_in)
begin
if(rising_edge(clk_in)) then
count <=count+1;
if(count = 4999) then
b <= not b; -- clk_out
count <=0;
```

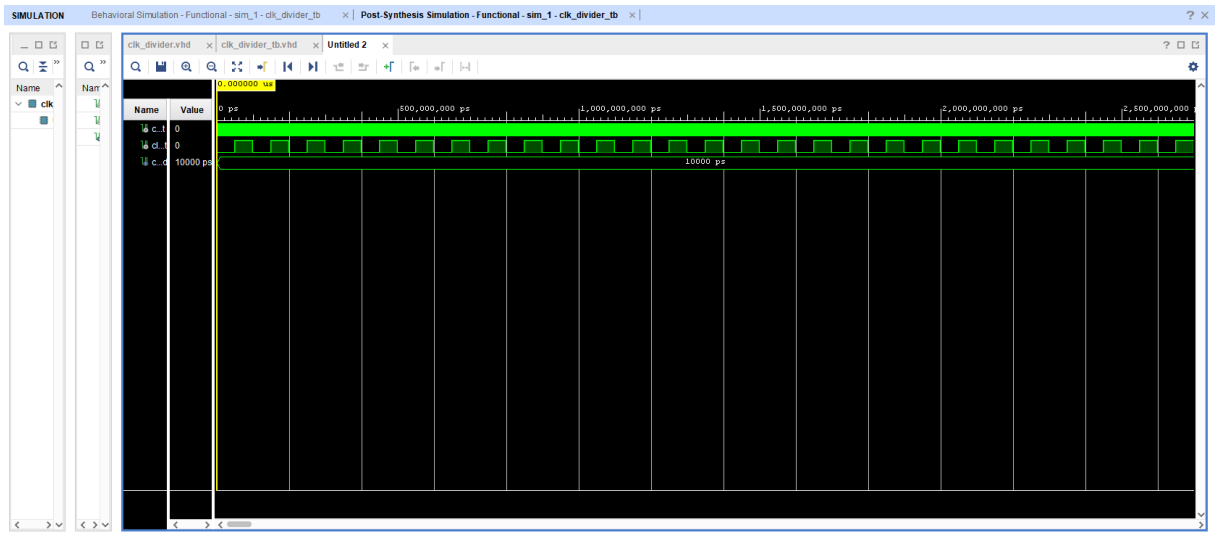
```
end if;
end if;
clk_out<=b;
end process;

end Behavioral;
```

-Testbench

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

entity clk_divider_tb is

end clk_divider_tb;

architecture testgate of clk_divider_tb is



component clk_divider is
port(clk_in : in std_logic;
     clk_out : out std_logic);

end component;

signal clk_in_t: std_logic;
signal clk_out_t : std_logic;
constant clk_period : time := 10 ns ;
```

```
begin
UUT: clk_divider port map(
clk_in => clk_in_t,
clk_out => clk_out_t
);




clk:
 process
begin
clk_in_t <= '0';
wait for clk_period/2;
clk_in_t <= '1';
wait for clk_period/2;

end process;




end testgate;
```





Button synchronizer

-VHD

```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;
```

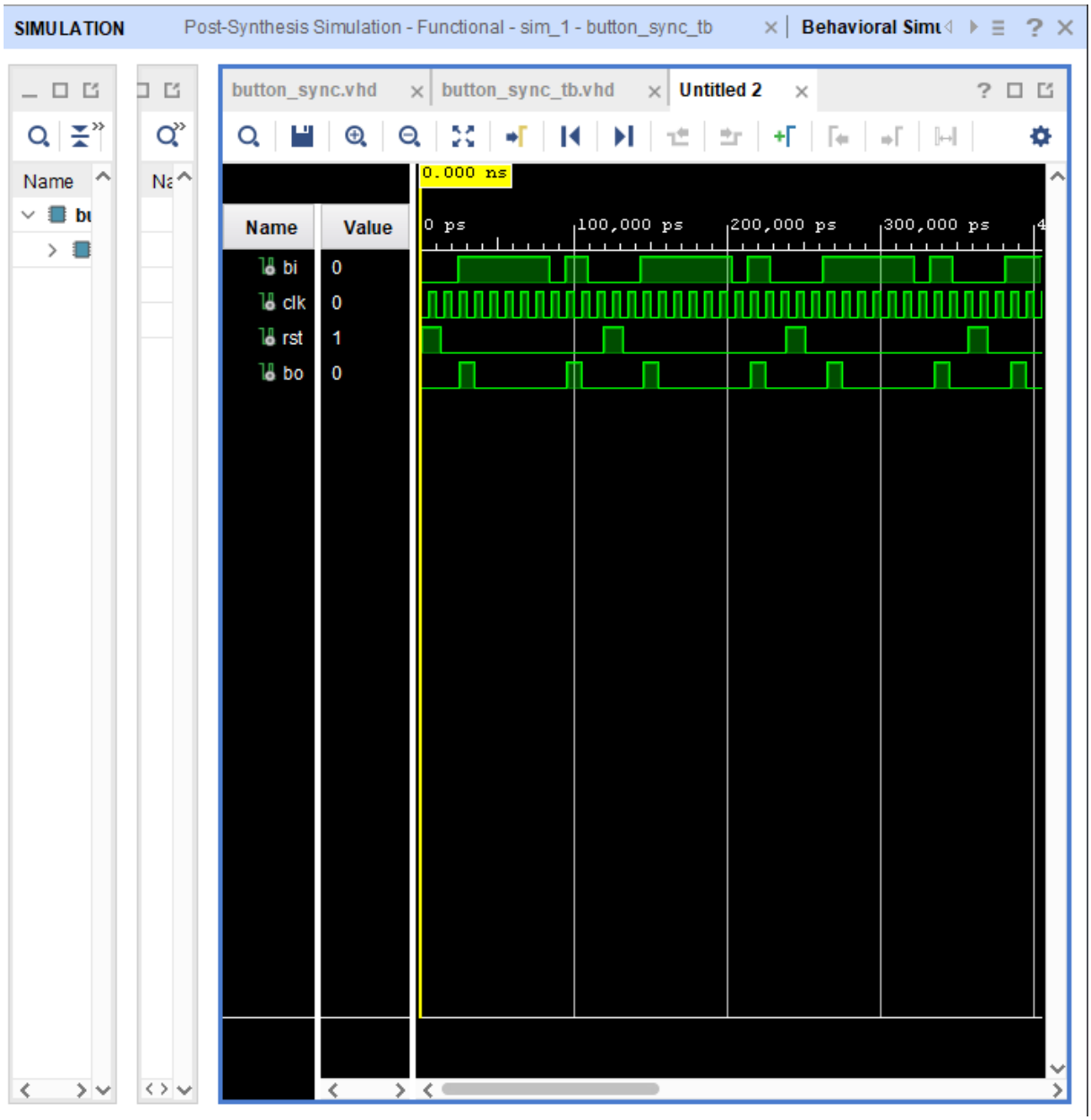
```
entity button_sync is
port( bi, clk, rst : in std_logic;
   bo: out std_logic);
end button_sync;

architecture Behavioral of button_sync is

component DFF is
port( Q : out std_logic;
     clk :in std_logic;
     rst: in std_logic;
     D :in  std_logic );
end component;

signal n0, n1, s0, s1: STD_LOGIC;
begin
bo <= s0 and (not s1);
n0 <= bi and (not s0) and (not s1);
n1 <= ((not s0) and s1 and bi) or ((not s1) and s0 and bi);

   D1: DFF port map (s0, clk, rst, n0);
   D2: DFF port map (s1, clk, rst, n1);

end Behavioral;
```

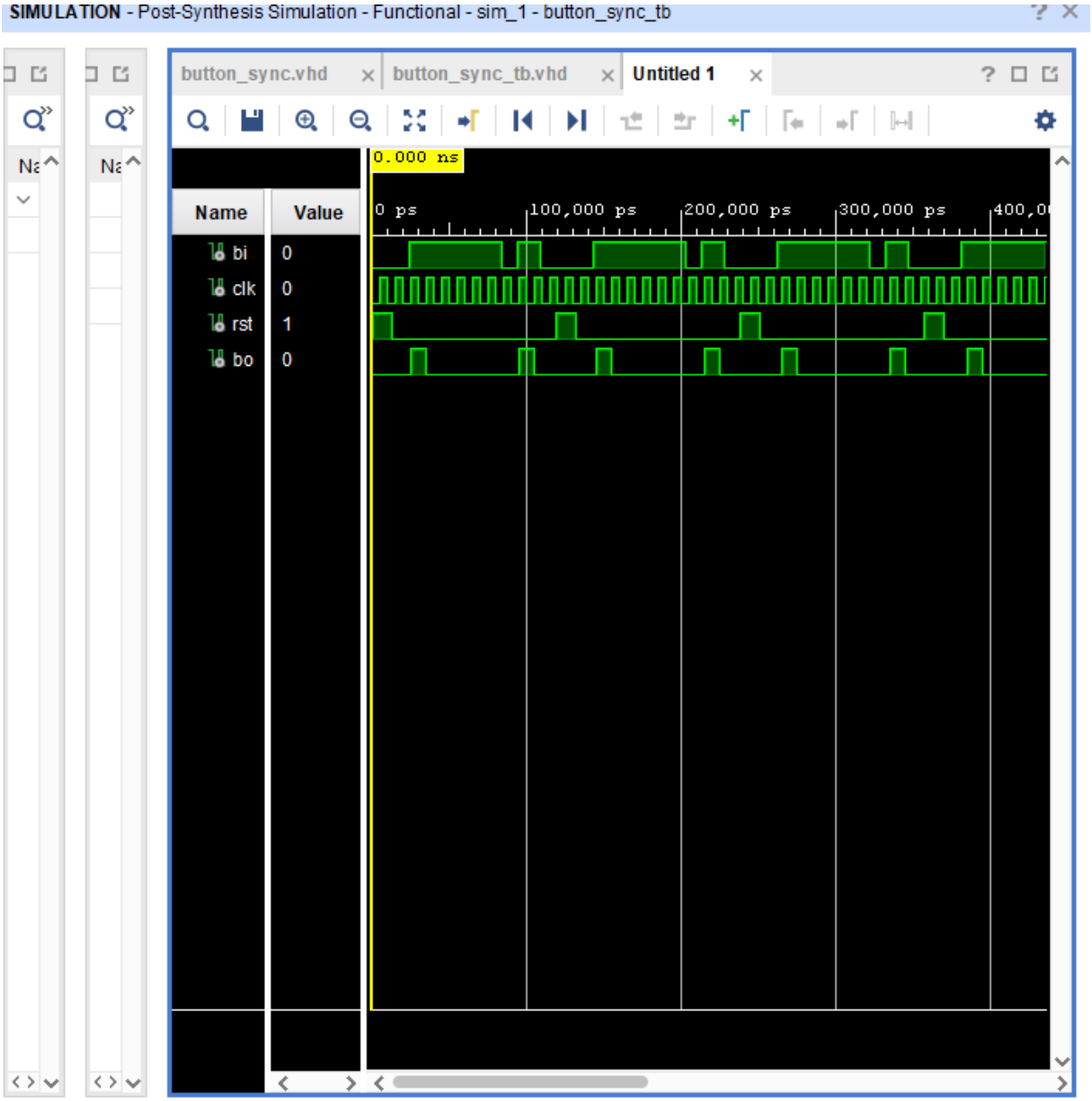
-Testbench

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

```
entity button_sync_tb is
end button_sync_tb;

architecture testgate of button_sync_tb is

component button_sync is
port( bi, clk, rst : in std_logic;
   bo: out std_logic);
end component;

signal bi, clk, rst : std_logic;
signal bo : std_logic;

begin

uut: button_sync port map(
bi => bi,
rst => rst,
clk => clk,
bo => bo);

clock: process
begin
clk <= '0';
wait for 5 ns;
clk <= '1';
wait for 5 ns;
end process;
```

```
Force: process
begin
bi <= '0';
rst <= '1';
wait for 13 ns;

bi <= '0';
rst <= '0';
wait for 11 ns;

bi <= '1';
rst <= '0';
wait for 60 ns;

bi <= '0';
rst <= '0';
wait for 10 ns;

bi <= '1';
rst <= '0';
wait for 15 ns;

bi <= '0';
rst <= '0';
wait for 10 ns;

end process;

end testgate;
```


SIMULATION
Post-Synthesis Simulation - Functional - sim_1 - button_sync_tb
Behavioral Simu
button_sync.vhd
button_sync_tb.vhd
Untitled 2
Name
Value
bi
clk
rst
bo
0
0
1
0
0.000 ns
0 ps
100,000 ps
200,000 ps
300,000 ps



Vending machine

-VHD

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


entity vending_machine is
 Port ( clk : in STD_LOGIC;
```

```
        rst : in STD_LOGIC;
        nickle_in : in STD_LOGIC;
        dime_in : in STD_LOGIC;
        display4seven :out std_logic_vector(7 downto 0);
        display : out std_logic_vector( 5 downto 0);
        coin_return : out STD_LOGIC;
        candy_out : out STD_LOGIC);
end vending_machine;

architecture Behavioral of vending_machine is

type state is (st0, st5, st10, st15, st20, st25, st30, st35, st40, st45); -- olabilecek durumlar
signal previous_state, next_state: state; -- basamaklara geçiş için kullanılacak

begin

SEQ: process(rst, clk)  -- bu noktada durumlar arasındaki geçişi gösterecek
begin
   if (rst = '1') then    -- reset 1'se previous_state sinyali st0 a atanır
      previous_state <= st0;
   elsif(clk' event and clk ='1') then   --buton rising adgeye denk geldiğinde next stage olur
      previous_state <= next_state;
   end if;
end process;

COM: process(previous_state, nickle_in, dime_in)
begin
next_state <= previous_state;
candy_out <= '0';
case previous_state is
   when st0 => candy_out <= '0'; -- atılan paralara göre verilen tepkiler
```

```
            coin_return <= '0';
            display <= "000000";
            display4seven <= "00000000"; -- two digit display kısmında çıkacak sayı
            if nickle_in = '1' then
            next_state <= st5;
            elsif dime_in = '1' then
            next_state <= st10;
            else
            next_state <= st0;
            end if;
   when st5 => candy_out <= '0';
            coin_return <= '0';
            display <= "000101"; --(binary)5 sayısını temsil eder devamında da aynı mantıkla devam
edecek
            display4seven <= "00000101"; -- 05 (0 ve 5 i ayrı ayrı 4 bite çevirip yazılır) display ekranında
okunabilecek hale getirildi. two digitte yapılan gibi
            if nickle_in = '1' then
            next_state <= st10;
            elsif dime_in = '1' then
            next_state <= st15;
            else
            next_state <= st5;
            end if;
   when st10 => candy_out <= '0';
            coin_return <= '0';
            display <= "001010";
            display4seven <= "00010000";
            if nickle_in = '1' then
            next_state <= st15;
            elsif dime_in = '1' then
            next_state <= st20;
            else
```

```
            next_state <= st10;
            end if;
    when st15 => candy_out <= '0';
            coin_return <= '0';
            display <= "001111";
            display4seven <= "00010101";--
            if nickle_in = '1' then
            next_state <= st20;
            elsif dime_in = '1' then
            next_state <= st25;
            else
            next_state <= st15;
            end if;
    when st20 => candy_out <= '0';
            coin_return <= '0';
            display <= "010100";
            display4seven <= "00100000";
            if nickle_in = '1' then
            next_state <= st25;
            elsif dime_in = '1' then
            next_state <= st30;
            else
            next_state <= st20;
            end if;
    when st25 => candy_out <= '0';
            coin_return <= '0';
            display <= "011001";-- binary de 25 buna denk geliyor
            display4seven <= "00100101"; -- seven segmentte oynatabilmek için 2 yi 4 bite çevirdik sonra yanına 5 i 4 bite çevirip yazdık
            if nickle_in = '1' then
            next_state <= st30;
```

```
            elsif dime_in = '1' then
            next_state <= st35;
            else
            next_state <= st25;
            end if;
when st30 => candy_out <= '0';
            coin_return <= '0';
            display <= "011110";
            display4seven <= "00110000";
             if nickle_in = '1' then
            next_state <= st35;
            elsif dime_in = '1' then
            next_state <= st40;
            else
            next_state <= st30;
            end if;
when st35 => candy_out <= '0';
            coin_return <= '0';
            display <= "100011";
            display4seven <= "00110101";
             if nickle_in = '1' then
            next_state <= st40;
            elsif dime_in = '1' then
            next_state <= st45;
            else
            next_state <= st35;
            end if;
when st40 => candy_out <= '1';
            coin_return <= '0';
            display <= "101000";
            display4seven <= "01000000";
```

```
            next_state <= st0;
 when st45 => candy_out <= '1';
            coin_return <= '1';
            display <= "101101";
            display4seven <= "01000101";
            next_state <= st0;

end case;
end process;

end Behavioral;
```

-Testbench

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

entity vending_machine_tb is
end vending_machine_tb;

architecture testgate of vending_machine_tb is

component top_deign
   Port ( clk : in STD_LOGIC;
          resetM : in STD_LOGIC;
          N : in STD_LOGIC;
          D : in STD_LOGIC;
          CR : out STD_LOGIC;
          candy : out STD_LOGIC;
          display : out std_logic_vector( 5 downto 0);
```

```
        selectedAnode: out STD_LOGIC_VECTOR (3 downto 0);
        displaySevenSegment : out STD_LOGIC_VECTOR (6 downto 0));
end component;

signal clk: STD_LOGIC;
signal resetM: STD_LOGIC := '0';
signal N: STD_LOGIC := '0';
signal D: STD_LOGIC := '0';
signal CR: STD_LOGIC := '0';
signal candy: STD_LOGIC;
signal display: std_logic_vector( 5 downto 0);
signal selectedAnode: STD_LOGIC_VECTOR (3 downto 0);
signal displaySevenSegment : STD_LOGIC_VECTOR (6 downto 0);


begin -- genel devre parti
uut: top_deign

port map(
clk => clk,
resetM => resetM,
N => N,
D => D,
display => display,
CR => CR,
candy => candy,
displaySevenSegment => displaySevenSegment,
selectedAnode => selectedAnode );

clock: process
begin
```

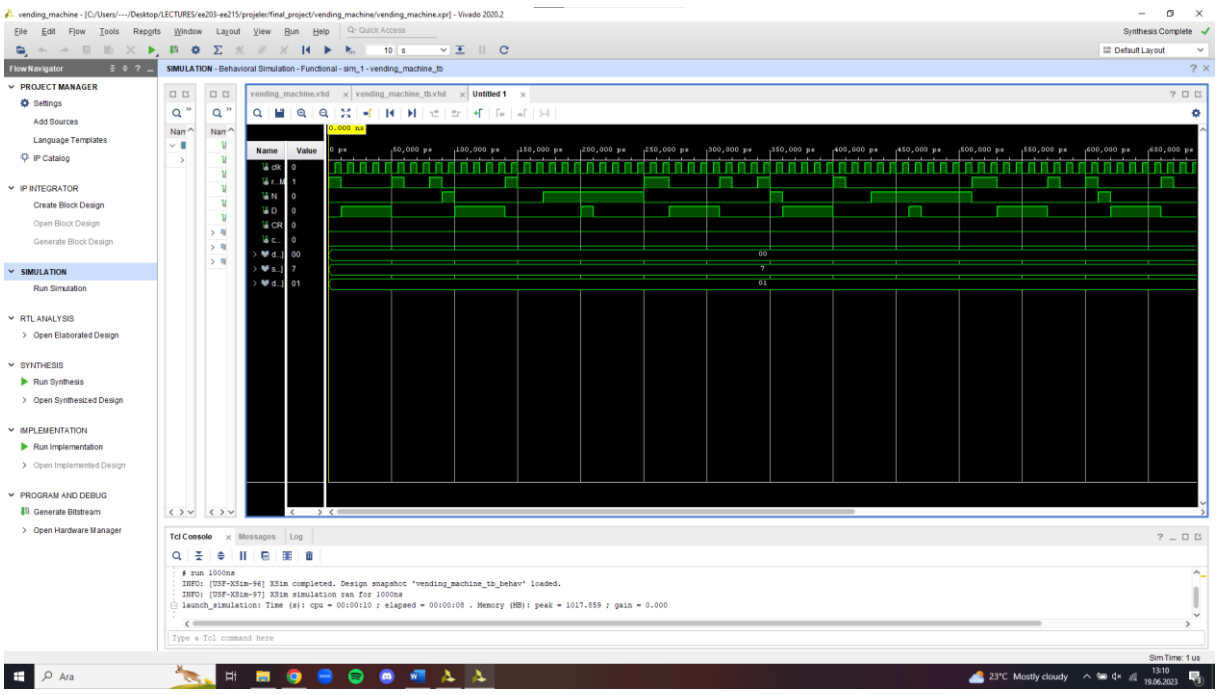
```
clk <= '0';
wait for 5 ns;
clk <= '1';
wait for 5 ns;
end process;

Force: process
begin

resetM <= '1';
N <= '0';
D <= '0';
wait for 10 ns;

resetM <= '0';
N <= '0';
D <= '1';
wait for 10 ns;

resetM <= '0';
N <= '0';
D <= '1';
wait for 10 ns;

resetM <= '0';
N <= '0';
D <= '1';
wait for 10 ns;

resetM <= '0';
```

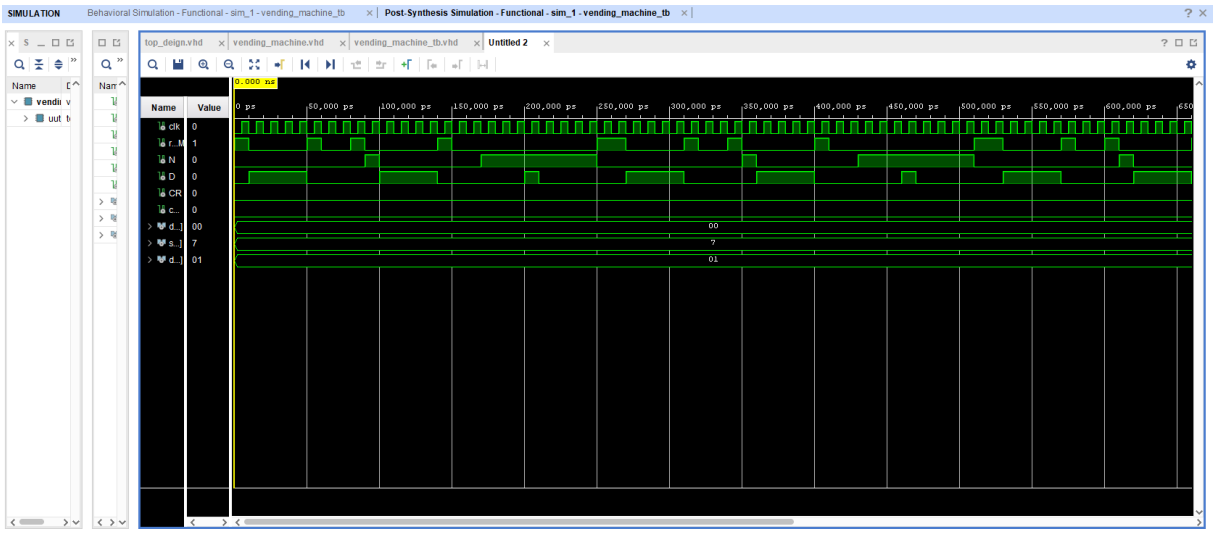
```
N <= '0';
D <= '1';
wait for 10 ns;

resetM <= '1';
N <= '0';
D <= '0';
wait for 10 ns;

resetM <= '0';
N <= '0';
D <= '0';
wait for 20 ns;-- boş bekleme daha net gözüksün diye

 --2 part
resetM <= '1';
N <= '0';
D <= '0';
wait for 10 ns;



resetM <= '0';
N <= '1';
D <= '0';
wait for 10 ns;

resetM <= '0';
N <= '0';
D <= '1';
wait for 10 ns;
```

```
resetM <= '0';
N <= '0';
D <= '1';
wait for 10 ns;

resetM <= '0';
N <= '0';
D <= '1';
wait for 10 ns;

resetM <= '0';
N <= '0';
D <= '1';
wait for 10 ns;

resetM <= '1';
N <= '0';
D <= '0';
wait for 10 ns;

resetM <= '0';
N <= '0';
D <= '0';
wait for 20 ns; -- boş bekleme

--3. part

resetM <= '0';
N <= '1';
D <= '0';
wait for 10 ns;
```

```
resetM <= '0';
N <= '1';
D <= '0';
wait for 10 ns;

resetM <= '0';
N <= '1';
D <= '0';
wait for 10 ns;

resetM <= '0';
N <= '1';
D <= '1';
wait for 10 ns;

resetM <= '0';
N <= '1';
D <= '0';
wait for 10 ns;

resetM <= '0';
N <= '1';
D <= '0';
wait for 10 ns;

resetM <= '0';
N <= '1';
D <= '0';
wait for 10 ns;
```

```
resetM <= '0';

N <= '1';

D <= '0';

wait for 10 ns;


resetM <= '1';

N <= '0';

D <= '0';

wait for 10 ns;


end process;


end testgate;
```





TwoDigit

- VHD

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


entity twodigit is  -- daha önceki lab görevinden
 Port ( clk : in STD_LOGIC;
        reset : in STD_LOGIC;
        display_number: in STD_LOGIC_VECTOR (7 downto 0);
        selectedAnode: out STD_LOGIC_VECTOR (3 downto 0);
        displaySevenSegment : out STD_LOGIC_VECTOR (6 downto 0));
end twodigit;

architecture Behavioral of twodigit is

signal refresh_rate: STD_LOGIC_VECTOR (1 downto 0);
signal binary_coded_decimal: STD_LOGIC_VECTOR (3 downto 0);
```

```
signal clk_divider: STD_LOGIC_VECTOR(22 downto 0);

begin

Clock_Divider:process(reset,clk)
begin
   if(reset='1') then
      clk_divider   <= (others=>'0');
   elsif(clk'event and clk='1') then
      clk_divider   <= clk_divider + 1;
 end if;
end process;
refresh_rate<= clk_divider(11 downto 10); -- yenileme süresi

Anode_Selection:process(refresh_rate) -- süreki yanıp sönerek yan yana gözükeceği içn onun ratesi
begin
   case refresh_rate is
      when "00" => selectedAnode <= "0111";
                   binary_coded_decimal<="0000";
      when "01" => selectedAnode <= "1011";
                   binary_coded_decimal<="0000";
      when "10" => selectedAnode <= "1101";
                   binary_coded_decimal<=display_number(7 downto 4);
      when "11" => selectedAnode <= "1110";
                   binary_coded_decimal<=display_number(3 downto 0);
      when others => selectedAnode <= "0111";
                  binary_coded_decimal<="0000";
   end case;
end process;

Segment_Display_Decoder:process(binary_coded_decimal)
```

```
begin
   case binary_coded_decimal is -- önceki two-digit (LAB4) görevlerinde oluşturulan tablo kullanıldı
      when "0000" => displaySevenSegment <= "0000001";
      when "0001" => displaySevenSegment <= "1001111";
      when "0010" => displaySevenSegment <= "0010010";
      when "0011" => displaySevenSegment <= "0000110";
      when "0100" => displaySevenSegment <= "1001100";
      when "0101" => displaySevenSegment <= "0100100";
      when "0110" => displaySevenSegment <= "0100000";
      when "0111" => displaySevenSegment <= "0001111";
      when "1000" => displaySevenSegment <= "0000000";
      when "1001" => displaySevenSegment <= "0000100";
      when others => displaySevenSegment <= "1111111"; -- 9 a kadar gerisi öndemsiz
   end case;
end process;

end Behavioral;
```

TopDesign

-VHD

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

Top design

- VHD

```
entity top_deign is
 Port ( clk : in STD_LOGIC;
        resetM : in STD_LOGIC;
        N : in STD_LOGIC;
```

```
        D : in STD_LOGIC;
        CR : out STD_LOGIC;
        candy : out STD_LOGIC;
        display : out std_logic_vector( 5 downto 0);
        selectedAnode: out STD_LOGIC_VECTOR (3 downto 0); -- en_out kısmı burası
        displaySevenSegment : out STD_LOGIC_VECTOR (6 downto 0));
end top_deign;

architecture Behavioral of top_deign is

component clk_divider is
 port (clk_in : in std_logic;
      clk_out : out std_logic );
end component;

component button_sync is
port( bi, clk, rst : in std_logic;
   bo: out std_logic);
end component;

component vending_machine is
Port ( clk : in STD_LOGIC;
        rst : in STD_LOGIC;
        nickle_in : in STD_LOGIC;
        dime_in : in STD_LOGIC;
        display4seven :out std_logic_vector(7 downto 0);
        display : out std_logic_vector( 5 downto 0);
        coin_return : out STD_LOGIC;
        candy_out : out STD_LOGIC);

end component;
```

```
component twodigit is
Port ( clk : in STD_LOGIC;
       reset : in STD_LOGIC;
       display_number: in STD_LOGIC_VECTOR (7 downto 0);
       selectedAnode: out STD_LOGIC_VECTOR (3 downto 0);
       displaySevenSegment : out STD_LOGIC_VECTOR (6 downto 0));
end component;



signal slw_clock: std_logic; ---
signal Ns,Ds: std_logic;
signal twodigitt : std_logic_vector(7 downto 0);
begin
COM1: clk_divider port map(clk,slw_clock); -- reset 0 olduğundan reset yazılmadı
COM2: button_sync port map(N,slw_clock,resetM,Ns); --Ns çıkışı(b0), slw_clc = clk_out
COM3: button_sync port map(D,slw_clock,resetM,Ds);
COM4: vending_machine port map(slw_clock,resetM , Ns, Ds,twodigitt, display, CR, candy);
COM5: twodigit port map(clk,resetM, twodigitt,selectedAnode, displaySevenSegment);

end Behavioral;
```

- Testbech

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

entity top_design_tb is
end top_design_tb;
```

```
architecture testgate of top_design_tb is

component vending_machine
    Port ( clk : in STD_LOGIC;
           rst : in STD_LOGIC;
           nickle_in : in STD_LOGIC;
           dime_in : in STD_LOGIC;
           display : out std_logic_vector( 5 downto 0);
           coin_return : out STD_LOGIC;
           candy_out : out STD_LOGIC);
end component;

signal clk: STD_LOGIC;
signal rst: STD_LOGIC;
signal nickle_in: STD_LOGIC;
signal dime_in: STD_LOGIC;
signal display: std_logic_vector( 5 downto 0);
signal coin_return: STD_LOGIC;
signal candy_out: STD_LOGIC;

begin

uut: vending_machine
port map(
clk => clk,
 rst        => rst,
 nickle_in   => nickle_in,
 dime_in     => dime_in,
 display     => display,
 coin_return => coin_return,
```

```
 candy_out   => candy_out );



 clock: process
begin
clk <= '0';
wait for 5 ns;
clk <= '1';
wait for 5 ns;
end process;



Force: process
begin
  -- 1. istenen durum
rst <= '1';
nickle_in <= '0';
dime_in <= '0';
wait for 10 ns;

rst <= '0'; --dime attik sonra bekleyip 3 tane daha dime attık
nickle_in <= '0';
dime_in <= '1';
wait for 10 ns;

rst <= '0'; -- para attığktan sonra kullanıcı biraz bekledi
nickle_in <= '0';
dime_in <= '0';
wait for 20 ns;

rst <= '0';
```

```
nickle_in <= '0';
dime_in <= '1';
wait for 10 ns;

rst <= '0';
nickle_in <= '0';
dime_in <= '1';
wait for 10 ns;

rst <= '0';
nickle_in <= '0';
dime_in <= '1';
wait for 10 ns;

rst <= '0';
nickle_in <= '0';
dime_in <= '0';
wait for 20 ns;
-----2. istenene durum
rst <= '1';
nickle_in <= '0';
dime_in <= '0';
wait for 10 ns;

rst <= '0';
nickle_in <= '1';
dime_in <= '0';
wait for 10 ns;

rst <= '0';
nickle_in <= '0';
```

```
dime_in <= '1';
wait for 10 ns;

rst <= '0';
nickle_in <= '0';
dime_in <= '1';
wait for 10 ns;

rst <= '0';
nickle_in <= '0';
dime_in <= '1';
wait for 10 ns;

rst <= '0';
nickle_in <= '0';
dime_in <= '1';
wait for 10 ns;


rst <= '0';
nickle_in <= '0';
dime_in <= '0';
wait for 20 ns;



-- 3. istenen N ve D nin =1 olduğu durumlar var, iki coin aynı anda atılamayacağı iiçn dime yanlış
rst <= '1';
nickle_in <= '0';
dime_in <= '0';
wait for 10 ns;
```

```
rst <= '0';
nickle_in <= '1';
dime_in <= '0';
wait for 10 ns;

rst <= '0';
nickle_in <= '1';
dime_in <= '0';
wait for 10 ns;

rst <= '0';
nickle_in <= '1';
dime_in <= '0';
wait for 10 ns;

rst <= '0';
nickle_in <= '1';
dime_in <= '1';
wait for 10 ns;

rst <= '0';
nickle_in <= '1';
dime_in <= '0';
wait for 10 ns;

rst <= '0';
nickle_in <= '1';
dime_in <= '0';
wait for 10 ns;
```

```
rst <= '0';
nickle_in <= '1';
dime_in <= '0';
wait for 10 ns;

 rst <= '0';
nickle_in <= '1';
dime_in <= '0';
wait for 10 ns;

 rst <= '0';
nickle_in <= '0';
dime_in <= '0';
wait for 20 ns;




end process;

end testgate;
```

- XDC

```
##Buttons                %% Canvasta paylaşılan Basys-3-Master.xdc dosyanın içindeki kodu kendi koduma göre yorumladım
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]

set_property PACKAGE_PIN T18 [get_ports resetM]
```

```
    set_property IOSTANDARD LVCMOS33 [get_ports resetM]

set_property PACKAGE_PIN W19 [get_ports N]
    set_property IOSTANDARD LVCMOS33 [get_ports N]

set_property PACKAGE_PIN T17 [get_ports D]
    set_property IOSTANDARD LVCMOS33 [get_ports D]

    ## LEDs
set_property PACKAGE_PIN U16 [get_ports {candy}]
    set_property IOSTANDARD LVCMOS33 [get_ports {candy}]

set_property PACKAGE_PIN E19 [get_ports {CR}]
    set_property IOSTANDARD LVCMOS33 [get_ports {CR}]

    ## Anodes  %%two digit display anotları
set_property PACKAGE_PIN U2 [get_ports {selectedAnode[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {selectedAnode[0]}]
set_property PACKAGE_PIN U4 [get_ports {selectedAnode[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {selectedAnode[1]}]
set_property PACKAGE_PIN V4 [get_ports {selectedAnode[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {selectedAnode[2]}]
set_property PACKAGE_PIN W4 [get_ports {selectedAnode[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {selectedAnode[3]}]

    ##7 segment display
set_property PACKAGE_PIN W7 [get_ports {displaySevenSegment[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {displaySevenSegment[0]}]
set_property PACKAGE_PIN W6 [get_ports {displaySevenSegment[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {displaySevenSegment[1]}]
set_property PACKAGE_PIN U8 [get_ports {displaySevenSegment[2]}]
```

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

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

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

set_property PACKAGE_PIN U5 [get_ports {displaySevenSegment[4]}]

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

set_property PACKAGE_PIN V5 [get_ports {displaySevenSegment[5]}]

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

set_property PACKAGE_PIN U7 [get_ports {displaySevenSegment[6]}]

set_property IOSTANDARD LVCMOS33 [get_ports {displaySevenSegment[6]}]

set_property PACKAGE_PIN U19 [get_ports {display[0]}]

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

set_property PACKAGE_PIN V19 [get_ports {display[1]}]

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

set_property PACKAGE_PIN W18 [get_ports {display[2]}]

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

set_property PACKAGE_PIN U15 [get_ports {display[3]}]

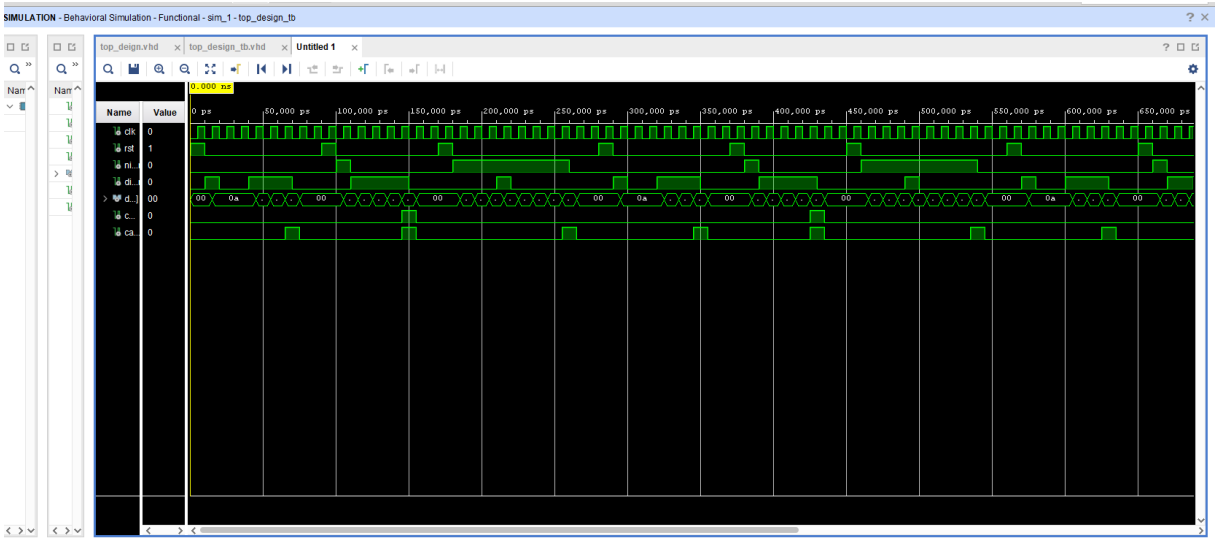
set_property IOSTANDARD LVCMOS33 [get_ports {display[3]}]

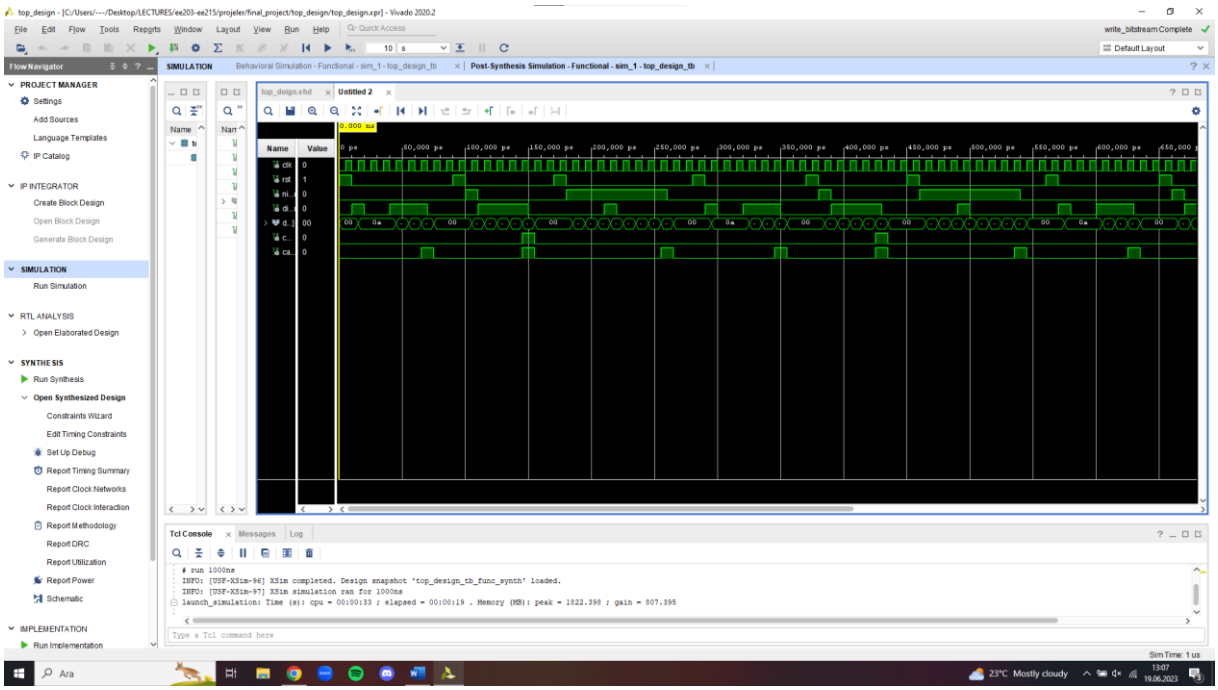
set_property PACKAGE_PIN U14 [get_ports {display[4]}]

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

set_property PACKAGE_PIN V14 [get_ports {display[5]}]

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




top_design - [C:/Users/---/Desktop/LECTURES/ee203-ee215/projeler/final_project/top_design/top_design.xpr] - Vivado 2020.2
File Edit Flow Tools Reports Window Layout View Run Help
write_bitstream Complete
Default Layout
Flow Navigator
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
Constraints Wizard
Edit Timing Constraints
Set Up Debug
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Utilization
Report Power
Schematic
IMPLEMENTATION
Run Implementation
SIMULATION
Behavioral Simulation - Functional - sim_1 - top_design_tb
Post-Synthesis Simulation - Functional - sim_1 - top_design_tb
top_design.vhd
Untitled 2
Name
Value
clk
rst
Tcl Console
Messages
Log
run 1000ns
INFO: [USF-XSim-96] XSim completed. Design snapshot 'top_design_tb_func_synth' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:33 ; elapsed = 00:00:19 . Memory (MB): peak = 1822.398 ; gain = 807.395
Type a Tcl command here
Sim Time: 1 us
Ara
23°C Mostly cloudy
13:07
19.06.2023