An-Najah National University
Department of Computer Engineering
Digital design 2 LAB -10636391-
Second Semester 2020/2021

جامعة النجاح الوطنية
قسم هندسة الحاسوب
مختبر تصميم دوائر الكترونية 2 - 10636391-
الفصل الأول 2021/2020

Instructor: Anas Toma

NAME	ID NUM	SEC
1- أحمد عبد الله جبر القرم	11819195	3/8:00-11:00
-	-	-

Experiment 3:

Part 1 : Asynchronous Ripple Counter.

Tools used in Lap :

1- Computer lap.
2- Vivado software .
3- Zboard from Xilinx.
4- VHDL

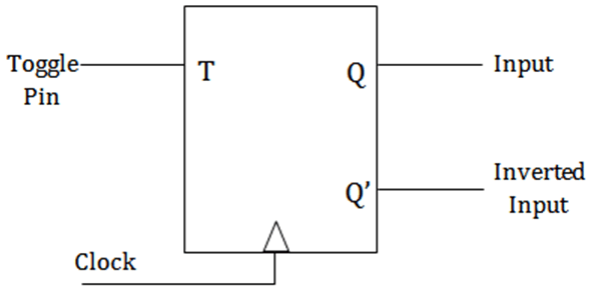
Introduction :

In this part, I will try to build a 4-bit Asynchronous ripple counter using T flip-flops.

1- T Flip-Flop

Implementation for T flip-flop with following pins:

a. **T** : Synchronous Input (0: No change , 1: Toggle)
b. **CLK**: positive edge trigger clock.
c. **Clear**: Asynchronous active low clear.
d. **Q** : output.
e. **QBAR** : output .



T	Q	Q'
0	0	0
1	0	1
0	1	0
1	1	0

T Flip-Flop VHDL CODE :

Project Summary | TFF.vhd

E:/projects/EX3_4BRC_Part1/EX3_4BRC.srcs/sources_1/new/TFF.vhd

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  entity TFF is
4      Port ( T : in STD_LOGIC;
5             CLK : in STD_LOGIC;
6             Clear : in STD_LOGIC;
7             Q : out STD_LOGIC;
8             Qbar : out STD_LOGIC);
9  end TFF;
10 architecture Behavioral of TFF is
11 signal temp : std_logic := '0';
12 begin
13 process (CLK,Clear)
14 begin
15     if Clear='0'then
16         temp<='0';
17     elsif (CLK'event and CLK='1' and T='1') then
18         temp<=not temp;
19     end if;
20 end process;
21 Q <= temp;
22 Qbar <= not temp;
23 end Behavioral;
24
```

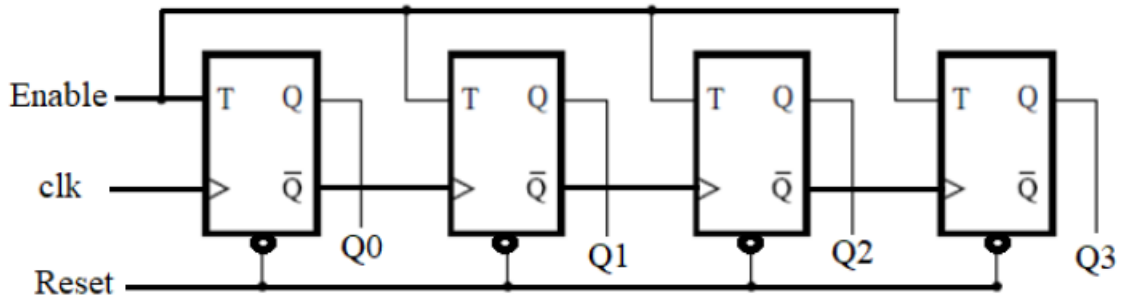
Simulation for T Flip-Flop:

2- 4-bit Asynchronous Counter :

I had Designed a 4-bit asynchronous ripple counter using the previous T flip-flop as a component.

My counter ports:

a. **CLK.**
b. **Enable**: Active High Enable.
c. **Reset**: Active Low Reset.
d. **Q**: (4-bit output).



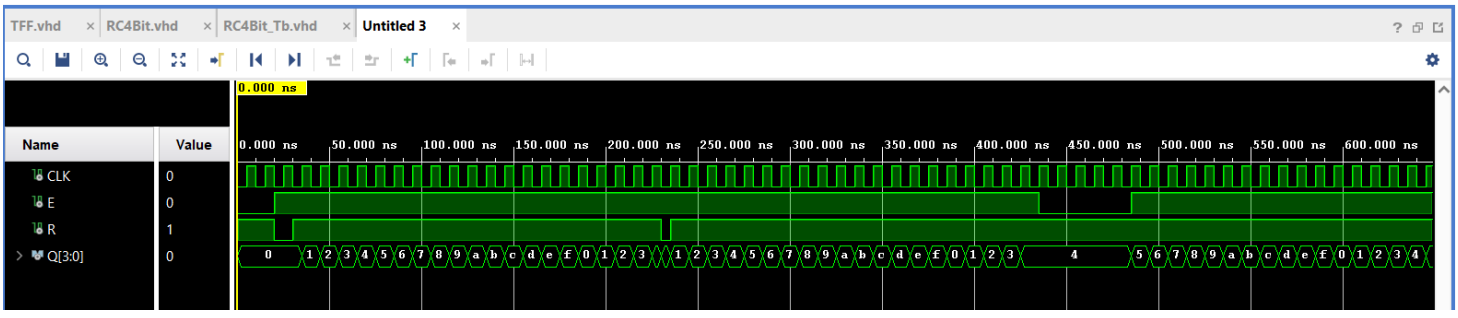
4-bit Asynchronous Counter VHDL CODE :

TFF.vhd | RC4Bit.vhd | Untitled 1

E:/projects/EX3_4BRC_Part1/EX3_4BRC.srcs/sources_1/new/RC4Bit.vhd

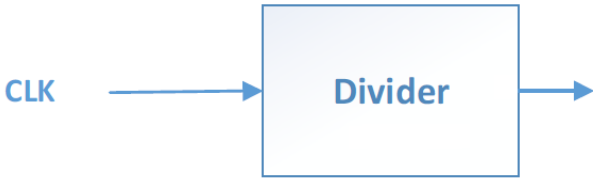
```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  entity RC4Bit is
4      Port ( CLK : in STD_LOGIC;
5             E : in STD_LOGIC;
6             R : in STD_LOGIC;
7             Q : out STD_LOGIC_VECTOR (3 downto 0));
8  end RC4Bit;
9  architecture Behavioral of RC4Bit is
10 component TFF port (
11     T : in STD_LOGIC;
12     CLK : in STD_LOGIC;
13     Clear : in STD_LOGIC;
14     Q : out STD_LOGIC;
15     Qbar : out STD_LOGIC
16 );
17 end component;
18 signal Qb:std_logic_vector(3 downto 0);
19 begin
20 mod1 : TFF port map(E,CLK,R,Q(0),Qb(0));
21 mod2 : TFF port map(E,Qb(0),R,Q(1),Qb(1));
22 mod3 : TFF port map(E,Qb(1),R,Q(2),Qb(2));
23 mod4 : TFF port map(E,Qb(2),R,Q(3),Qb(3));
24 end Behavioral;
```

4-bit Asynchronous Counter simulation :



3- Clock Divider :

Because the system is operating on 100MHz frequency, I completed clock divider to generate an output clock with 1Hz frequency.



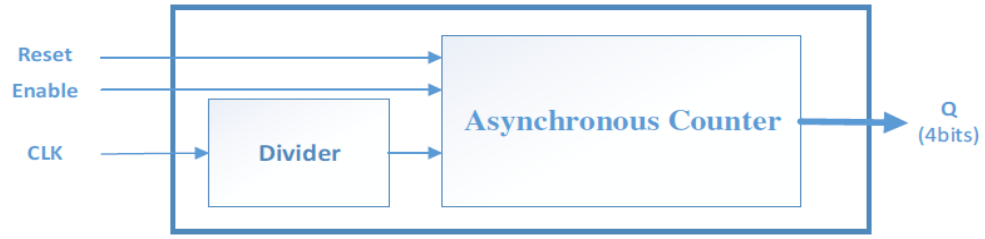
Clock Divider VHDL CODE :

E:/projects/EX3_4BRC_Part1/EX3_4BRC.srcs/sources_1/new/Divider.vhd

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  use IEEE.std_logic_unsigned.all;
4  use IEEE.numeric_std.ALL;
5  entity Divider is
6      Port ( CLK_IN : in STD_LOGIC;
7             CLK_OUT : out STD_LOGIC);
8  end Divider;
9
10 architecture Behavioral of Divider is
11 signal count:integer range 0 to 50000000:=0;
12 signal CLK : std_logic := '0';
13 begin
14     process (CLK_IN)
15     begin
16         if (CLK_IN'event and CLK_IN='1') then
17         if (count=50000000)then
18             CLK<=not CLK;
19             count<=0;
20             else
21              count<= count +1;
22           end if;
23             end if;
24     end process;
25     CLK_OUT <= CLK;
26 end Behavioral;
27
```

4- Top-Level Entity:

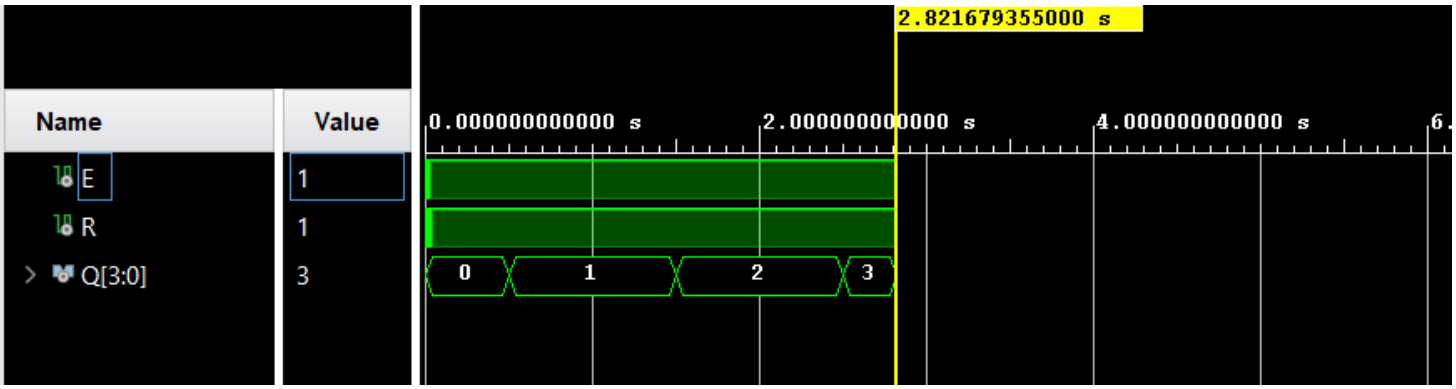
The top level entity include two component : clock divider & asynchronous counter as shown in the figure.



Top-Level Entity VHDL CODE:

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  entity TLEofDand4BRC is
4      Port ( CLK : in STD_LOGIC;
5             E : in STD_LOGIC;
6             R : in STD_LOGIC;
7             Q : out STD_LOGIC_VECTOR (3 downto 0));
8  end TLEofDand4BRC;
9
10 architecture Behavioral of TLEofDand4BRC is
11 component RC4Bit port (
12     CLK : in STD_LOGIC;
13     E : in STD_LOGIC;
14     R : in STD_LOGIC;
15     Q : out STD_LOGIC_VECTOR (3 downto 0)
16 );
17 end component;
18 component Divider port (
19     CLK_IN : in STD_LOGIC;
20     CLK_OUT : out STD_LOGIC
21 );
22 end component;
23 signal CLKto:std_logic;
24 begin
25 Divmod:Divider port map (
26 CLK_IN=>CLK,
27 CLK_OUT=>CLKto
28 );
29 RC4Bitmod:RC4Bit port map (CLKto,E,R,Q);
30 end Behavioral;
31
```

Top-Level Entity asynchronous counter Simulation:



This is not the full simulation because it's hard to simulate real time in personal PC

5- Synthesis, Implementation, and Bitstream Generation:

I connected the pins as shown and tested the circuit .

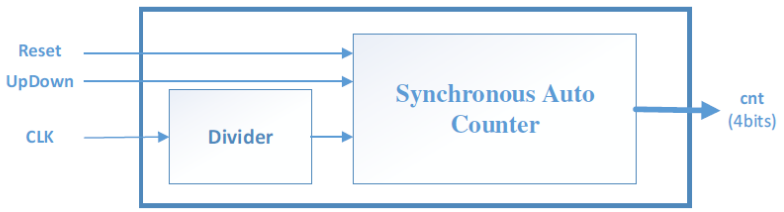
a. Reset: SW0
b. Enable: SW1
c. CLK: Y9
d. Q (4bits): LD3, LD2, LD1, LD0.

Inputs	Package Pin	I/O Std
SW0	F22	LVCMOS18
SW1	G22	LVCMOS18
CLK	Y9	LVCMOS33
Outputs	**Package Pin**	**I/O Std**
LD0	T22	LVCMOS33
LD1	T21	LVCMOS33
LD2	U22	LVCMOS33
LD3	U21	LVCMOS33

Part 2 : Auto Up Down Synchronous Counter.

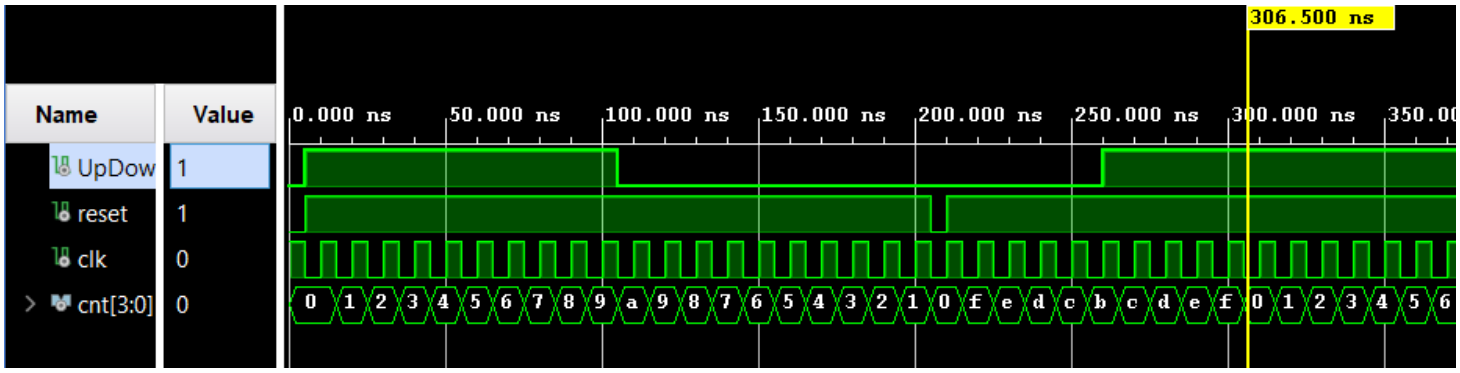
The top level entity for Auto Up Down Counter contain two component : divider & Auto Up Down Counter(when UpDown pin is 1 will count rise way and otherwise will count down)
I implemented auto up down counter using VHDL with beaver discretion as shown:

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_unsigned.ALL;
entity AutoUpDownCounter is
    Port (
        UpDown : in STD_LOGIC;
        reset : in STD_LOGIC;
        clk : in STD_LOGIC;
        cnt : out STD_LOGIC_VECTOR(3 downto 0)
        );
end AutoUpDownCounter;
architecture Behavioral of AutoUpDownCounter is
signal temp:std_logic_vector(3 downto 0):="0000";
begin
process (clk,reset)
begin
    if (reset ='0') then
        temp <= "0000";
    elsif (clk'event and clk='1')then
        if (UpDown = '1')then
            temp <= temp + '1';
        else
            temp <= temp - '1';
        end if;
    end if;
cnt <= temp;
end process;
end Behavioral;
```



And used the same divider in the part 1 .

Auto Up Down Synchronous Counter Simulation:



Synthesis, Implementation, and Bitstream Generation:

I connected the pins as shown and tested the circuit .

a. **Reset**: SW0
b. **UpDown**: SW1
c. **CLK**: Y9
d. **Q (4bits)**: LD3, LD2, LD1, LD0.

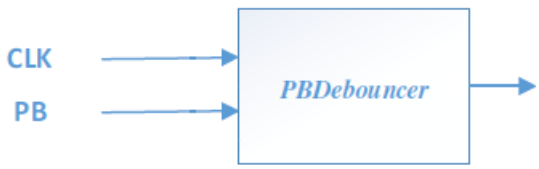
Inputs	Package Pin	I/O Std
SW0	F22	LVCMOS18
SW1	G22	LVCMOS18
CLK	Y9	LVCMOS33
Outputs	**Package Pin**	**I/O Std**
LD0	T22	LVCMOS33
LD1	T21	LVCMOS33
LD2	U22	LVCMOS33
LD3	U21	LVCMOS33

Part 3: Push Button UpDown Counter.

In this part I used the same counter in the previous part (part 2).and instead of using direct clock I used a Push Button .as a result new problem Appeared (noise in the Push Button)so the counter will count several times when I push the button. we can fix it easily with Push Button Debouncer algorithm.

A- Build Debouncer Block :

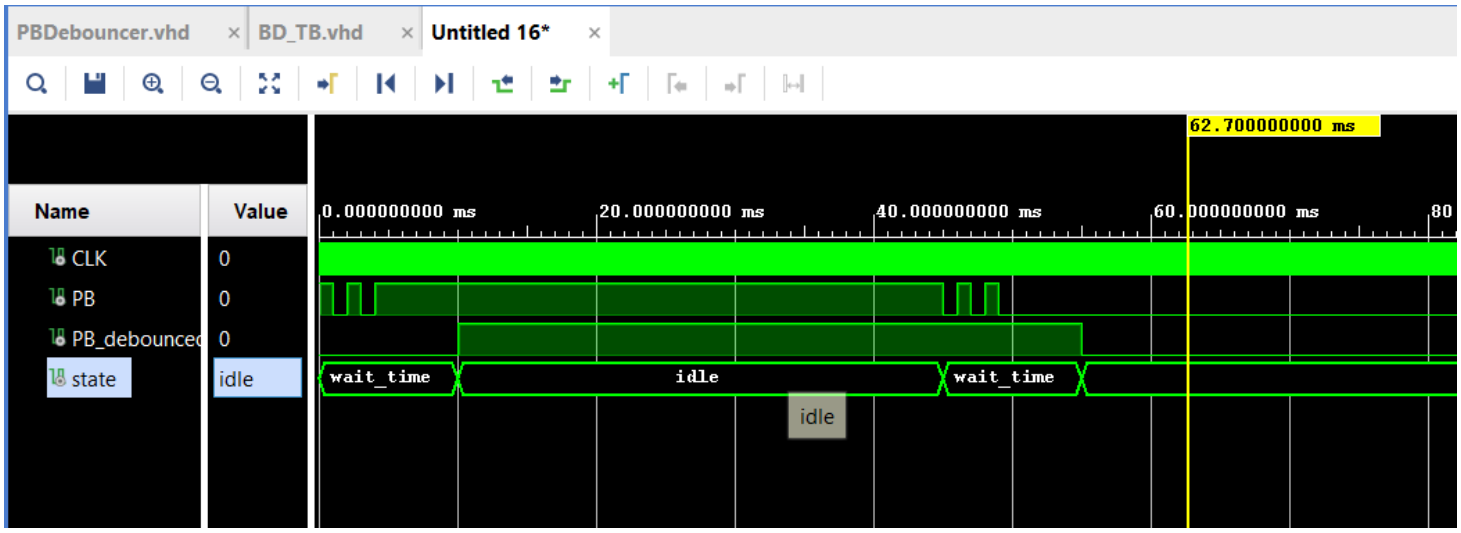
When the Debouncer detect noise
It move to wait state for 10 ms
And then output the result.



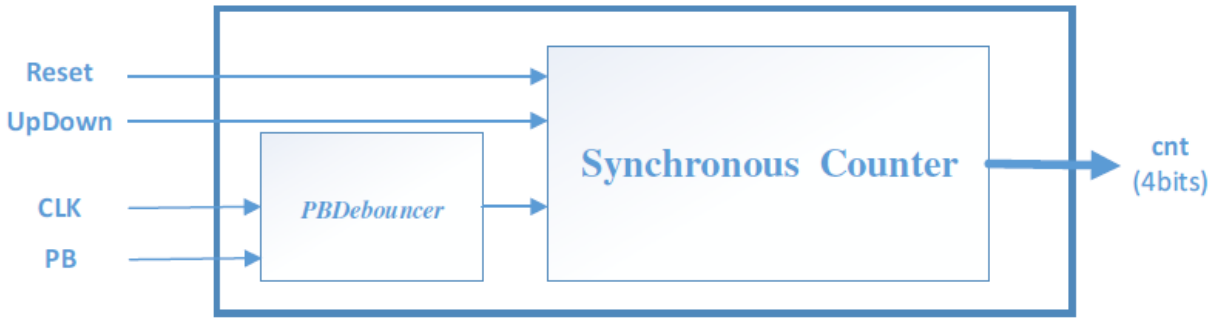
Debouncer VHDL CODE:

```
11  architecture Behavioral of PBDebouncer is
12  constant COUNT_MAX:integer := 1000000;
13  SIGNAL BTN_ACTIVE : std_logic := '0';
14  signal count : integer := 0;
15  type state_type is (idle,wait_time);
16  signal state : state_type := idle;
17  begin
18  process(CLK)
19  begin
20     if(CLK'event and CLK='1') then
21          case (state) is
22              when idle =>
23                  if(PB /= BTN_ACTIVE) then
24                      state <= wait_time;
25                  else
26                      state <= idle;
27                  end if;
28              when wait_time =>
29                  if(count = COUNT_MAX) then
30                      count <= 0;
31                      BTN_ACTIVE <= NOT BTN_ACTIVE;
32                      state <= idle;
33                  else
34                      count <= count + 1;
35                  end if;
36          end case;
37      end if;
38  end process;
39  PB_debounced <= BTN_ACTIVE;
40  end Behavioral;
```

Debouncer Simulation :



B- Top level Push Button UpDown Counter:



VHDL CODE :

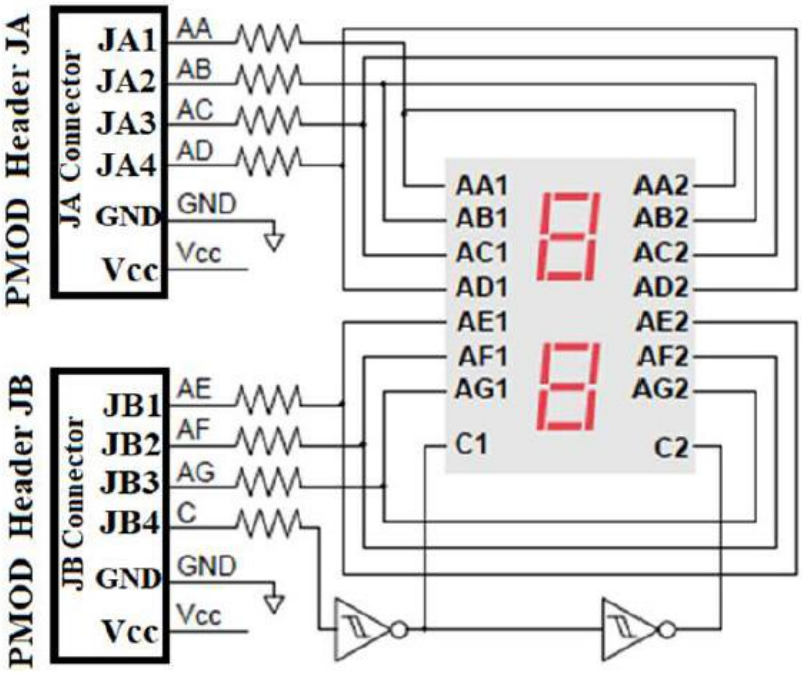
```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  entity PBUpDownCounter is
4      Port (
5          CLK : in STD_LOGIC;
6          Reset : in STD_LOGIC;
7          UpDown : in STD_LOGIC;
8          PB : in STD_LOGIC;
9          cnt : out STD_LOGIC_VECTOR (3 downto 0)
10         );
11 end PBUpDownCounter;
12 architecture Behavioral of PBUpDownCounter is
13 component AutoUpDownCounter Port (
14         UpDown : in STD_LOGIC;
15         reset : in STD_LOGIC;
16         clk : in STD_LOGIC;
17         cnt : out STD_LOGIC_VECTOR(3 downto 0)
18         );
19 end component;
20 component PBDebouncer Port (
21         CLK : in STD_LOGIC;
22         PB : in STD_LOGIC;
23         PB_debounced: out STD_LOGIC
24         );
25 end component;
26 signal PB_debounced2:std_logic;
27 begin
28 mod1 : PBDebouncer port map(CLK,PB,PB_debounced2);
29 mod2 : AutoUpDownCounter port map (UpDown,reset,PB_debounced2,cnt);
30 end Behavioral;
```

Synthesis, Implementation, and Bitstream Generation:
I connected the pins as shown and tested the circuit .

	Signal Name	Package Pin	I/O Std
Up Button	BTNU	T18	LVCMOS18
Right Button	BTNR	R18	LVCMOS18
Down Button	BTND	R16	LVCMOS18
Center Button	BTNC	P16	LVCMOS18
Left Button	BTNL	N15	LVCMOS18

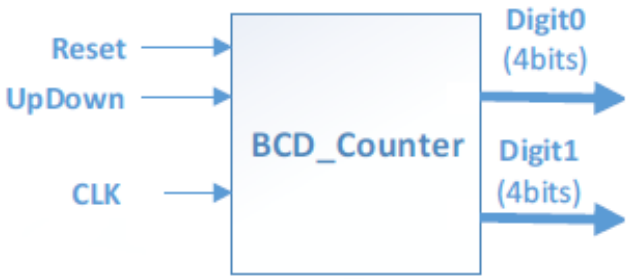
Part 4: Two-Digit BCD Counter.

In this part, I built a two-digit BCD counter that counts from 00 to 99. And the result displayed on a two-seven-segment display common cathode module connected to the kit. The circuit diagram for this module is shown in Fig1.



The circuit contain four blocks :

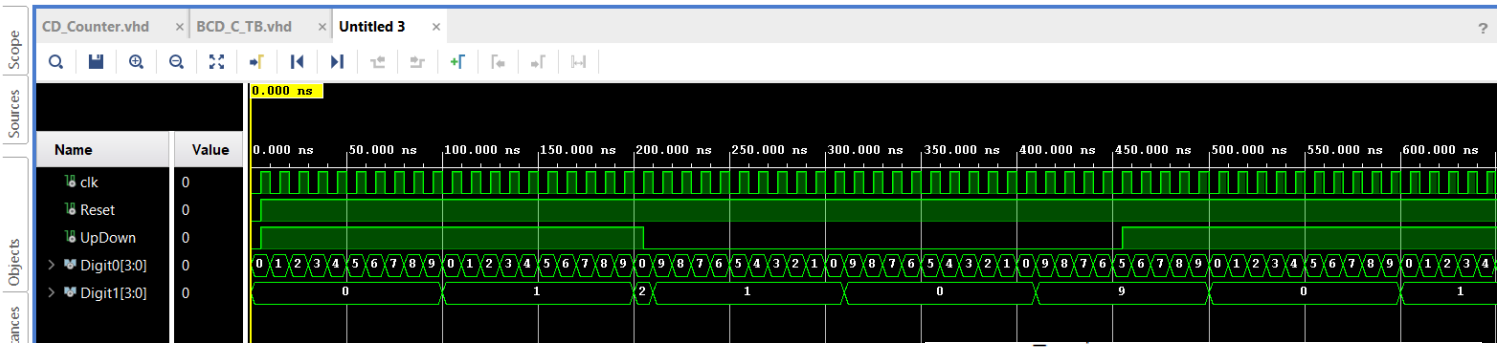
1- Divider : the same as the Divider in Part 2
2- BCD_Counter: It is a two-digit (two-decade) BCD counter counts in decimal from 00 decimal to 99.



BCD_Counter VHDL CODE :

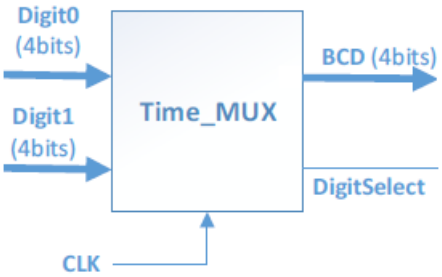
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.numeric_std.all;
use ieee.std_logic_unsigned.all;
entity BCD_Counter is
Port (
    clk : in STD_LOGIC;
    Reset : in STD_LOGIC;
    UpDown: in STD_LOGIC;
    Digit0: out STD_LOGIC_VECTOR (3 downto 0);
    Digit1: out STD_LOGIC_VECTOR (3 downto 0));
end BCD_Counter;
architecture Behavioral of BCD_Counter is

signal temp0,temp1:std_logic_vector(3 downto 0):="0000";

begin
process (clk,Reset,UpDown)
begin
    if (Reset = '0') then
        temp0<="0000";
        temp1<="0000";
    else
        if (clk'event and clk='1')then
            if (UpDown='1')then
                if (temp0 < "1001")then
                    temp0<=temp0+'1';
                else
                    temp0<="0000";
                    if (temp1<"1001")then
                        temp1<=temp1+'1';
                    else
                        temp1<="0000";
                    end if;
                end if;
            elsif (UpDown='0')then
                if (temp0 > "0000")then
                    temp0<=temp0-'1';
                else
                    temp0<="1001";
                    if (temp1>"0000")then
                        temp1<=temp1-'1';
                    else
                        temp1<="1001";
                    end if;
                end if;
            end if;
        else
            null;
        end if;--2
    end if;--1
        Digit0<=temp0;
        Digit1<=temp1;
end process;
end Behavioral;
```

BCD_Counter simulation :



3- Time_MUX:

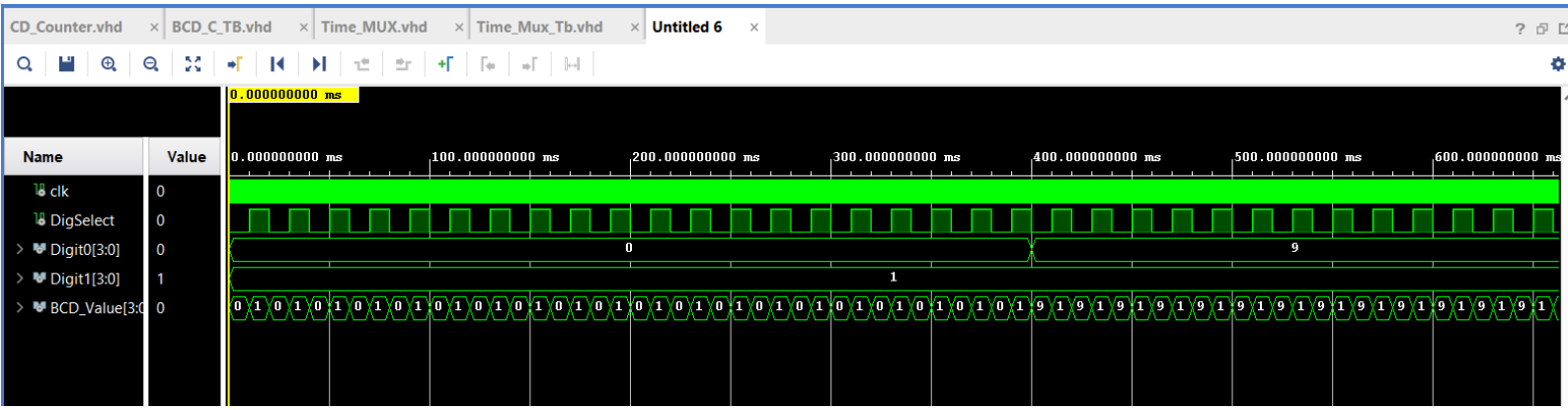
To continuously display a digit on each display faster than the human eye (10 ms) can respond .so the time-mux will change the digit to display it .




```vhdl
architecture Behavioral of Time_MUX is
signal count:integer range 0 to 1000000:=0;
signal CLK1 : std_logic := '0';
begin
process(clk)
begin
    if (clk'event and clk='1') then
        if (count=1000000)then
            CLK1<=not CLK1;
            count<=0;

        else
            count<= count +1;
        end if;
    end if;
 if (CLK1='0')then
     BCD_Value<=Digit0;
 else
     BCD_Value<=Digit1;
 end if;


end process;
          DigSelect <= CLK1;
end Behavioral;
```

Time_MUX simulation :

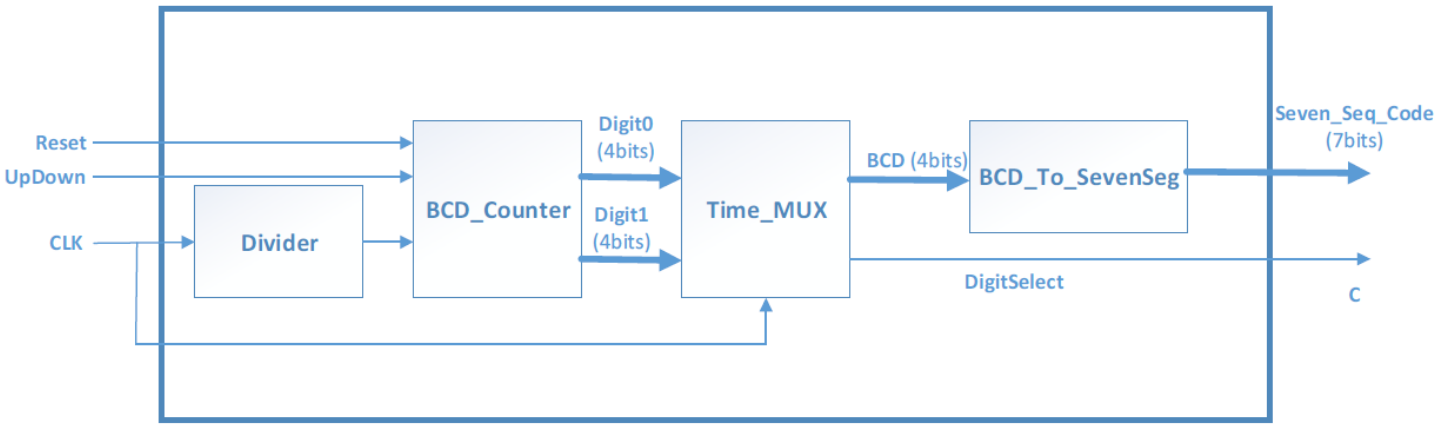
Assume two bcd digit (10) and (19)



4- BCD_To_SevenSeg:

To decode the binary value and prepped it to display

And finally we got the top level component of Two-Digit BCD Counter:



Synthesis, Implementation, and Bitstream Generation:

I connected the pins as shown and tested the circuit .

	Inputs	Package Pin	I/O Std
Reset =>	SW0	F22	LVCMOS18
	SW1	G22	LVCMOS18
	CLK	Y9	LVCMOS33

Signal Name	Package Pin	I/O Std
JA1	Y11	LVCMOS33
JA2	AA11	LVCMOS33
JA3	Y10	LVCMOS33
JA4	AA9	LVCMOS33
JB1	W12	LVCMOS33
JB2	W11	LVCMOS33
JB3	V10	LVCMOS33
Dig select => JB4	W8	LVCMOS33

Conclusion:

In this experiment I learned how to build 4 bit ruble counter using T-Flip flops , how to build Auto Up Down Synchronous Counter (behavior), how to Debounce a Push Button ,and i become familiar with Two-Digit BCD Counter.