EL323//El324 Digital System Laboratory II

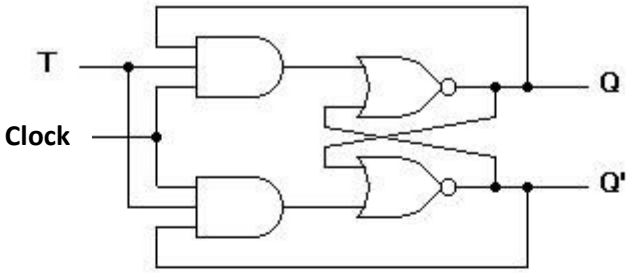
LAB 7 : Sequential Design and Hierarchy

Objective:

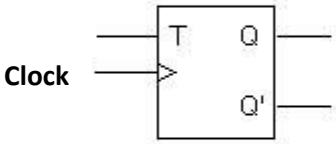
1. Can use the Quartus II and create the new project using VHDL
2. To know the assign pin of DE-0 in Quartus.
3. Design using VHDL
4. To know the counter and shift register

LAB 7.1 Counter

1. Create new Project name “Counter4bit”.
2. Create new entity for T Flip-flop in this below.



(a) Logic diagram



(b) Graphical symbol

Q	T	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

(c) Transition table

Clocked T flip-flop

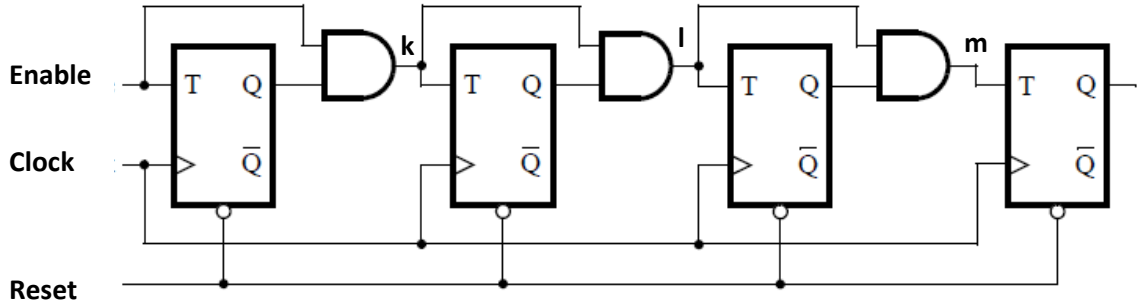
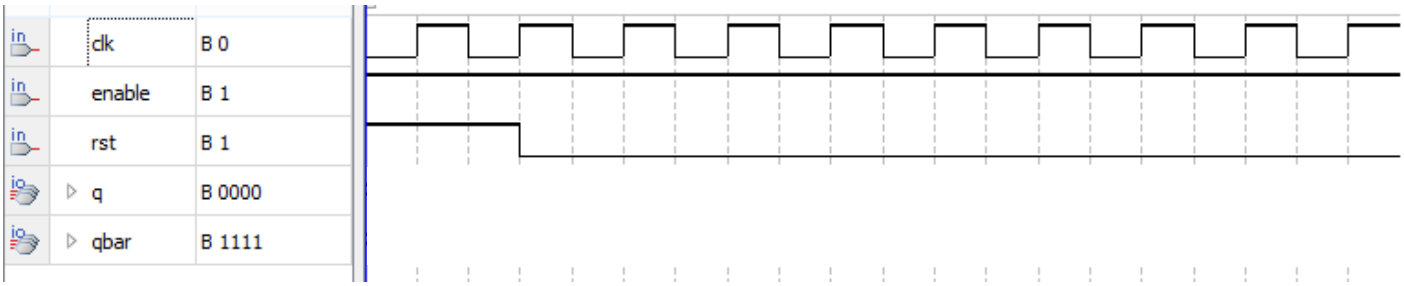


Figure 1: A 4-bit counter.

3. The pushbutton KEY_0 as the Clock input, switches SW_1 and SW_0 as Enable and Reset inputs. The Hex_0 as the hexadecimal output.
4. Use the DE0 User Manual to define the DE0 pin.
5. Compile the code and program to DE0.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity Counter4bit IS
        port(   enable, clk, rst : In std_logic;
                q, qbar : Inout std_logic_vector(3 DOWNTO 0);
                Hex, Out std_logic_vector(6 DOWNTO 0));
End Counter4bit;

Architecture Behavioral OF Counter4bit is
        Component TFF1 is
                Port(   t, rst, clk : In std_logic;
                        q, qb : Out std_logic);
                End component;
        Component seg7display is
                PORT(S : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
                  H : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
                End component;

        Signal k, l, m : std_logic;
                Begin
                        k <= _____________________________
                        l <= _____________________________
                        m <= _____________________________
                        Counter1:_____________________________________
                        Counter2:_____________________________________
                        Counter3:_____________________________________
                        Counter4:_____________________________________

                        HEX1: _____________________________________

        End Behavioral;
```

```
Library IEEE;
Use IEEE.STD_LOGIC_1164.ALL;
Entity TFF1 IS
Port(     t,rst,clk : In std_logic;
                  q,qb : Out std_logic);
End TFF1;

Architecture Behavioral OF TFF1 Is
Begin
          Process (clk,rst)
                    Variable x : std_logic:='0';
                    Begin
                              If (clk' Event and clk=____) Then
                                        If _____ Then
                                                  x:=____;
                                        Elsif t=____ Then
                                                  x:=_____;
                                        Else
                                                  x:=_____;
                                        End if;
                              End if;
                    q<=x;
                    qb<=not x;
          End Process;
End Behavioral;

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY seg7display IS
          PORT(S : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
                    H : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
END seg7display;

ARCHITECTURE behavior OF seg7display IS
BEGIN
          -- // NEED CODE of Seg7display

END behavior;
```



Conclusion

……

……

……

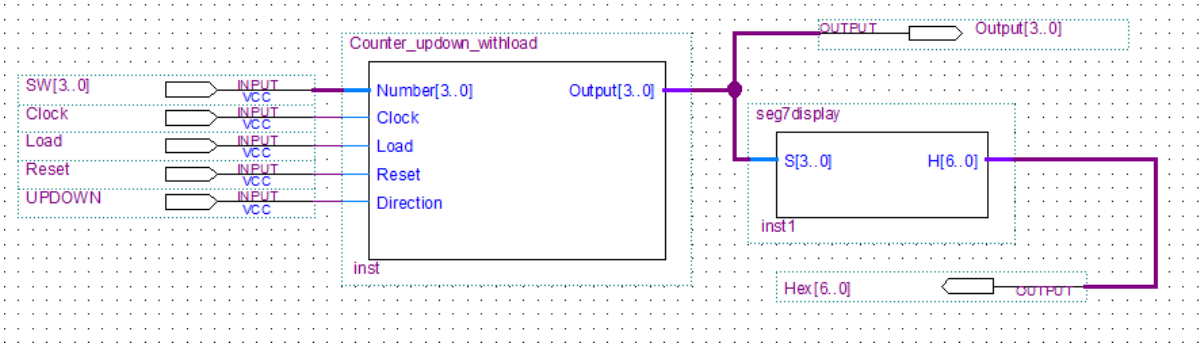
……

……

……

<u>LAB7.2</u> Counter Up/Down With load

1. Create new Project name “Counter_updown_withload”.
2. Create new entity for counter in this below.



The Diagram of counter with load.

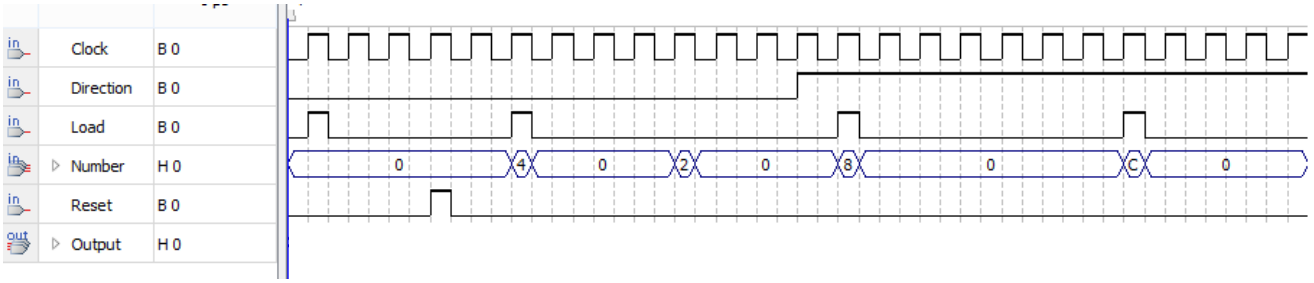
Input	Input FPGA
Number[3..0]	SW_3-SW_0
Clock	$Button_2$
UPDOWN(direction)	SW_9
Load	SW_8
Reset	SW_7

Output	Output FPGA
Hex[6..0]	Hex1
Out[3..0]	LED_3-LED_0

The input type	Description
Number	The input is start with this number
Load	The Load = 1 ,Load Number to start The Load = 0, Ready to UPDOWN.
Reset	The Reset = 0 , Clear the all bits The Reset = 1, Do nothing.
UPDOWN(direction)	The UPDOWN = 0 , number = number +1 The UPDOWN = 1 , number = number -1

3. Use the DE0 User Manual to define the DE0 pin.
4. Compile the code and program to DE0.

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

entity Counter_updown_withload is
  port( Number: in std_logic_vector(3 downto 0);
        Clock: in std_logic;
        Load: in std_logic;
        Reset: in std_logic;
        Direction: in std_logic;
        Hex: out std_logic_vector(6 downto 0) );
        Output: out std_logic_vector(3 downto 0) );
end Counter_updown_withload;

architecture Behavioral of Counter_updown_withload is
        Component seg7display is
                PORT(S : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
                  H : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
        End component;
        signal temp: std_logic_vector(3 downto 0);
begin
  process(Clock,Reset)
  begin
    if Reset='1' then
      temp <= _________________;
    elsif ( Clock'event and Clock='1') then
          if Load='1' then
            temp <= ____________________;
          elsif (Load='0' and Direction='0') then
            temp <= ____________________;
          elsif (Load='0' and Direction='1') then
            temp <= ____________________;
          end if;
    end if;
  end process;
        Output <= _____________________
        HEX1: _________________________
end Behavioral;

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY seg7display IS
        PORT(S : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
                H : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
END seg7display;

ARCHITECTURE behavior OF seg7display IS
BEGIN
        -- // NEED CODE of Seg7display
END behavior;
```



Number[3..0]	Clock	UPDOWN(direction)	Load	Reset	Out[3..0]	HEX
0 1 0 0	1 pulse	0	1	0		
0 0 0 0	2 pulse	0	0	0		
0 0 0 0	3 pulse	0	0	1		
0 0 0 0	1 pulse	1	0	0		
0 0 0 0	1 pulse	1	0	0		
1 0 0 0	1 pulse	1	1	0		
1 0 0 0	2 pulse	1	0	0		
1 0 0 0	3 pulse	1	0	0		

Conclusion

……

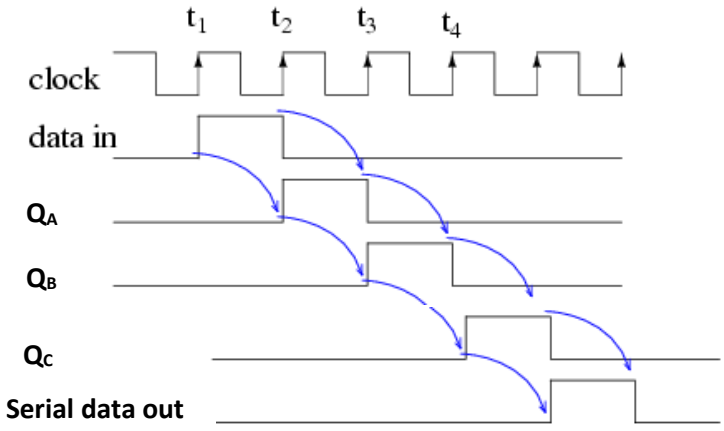
……

……

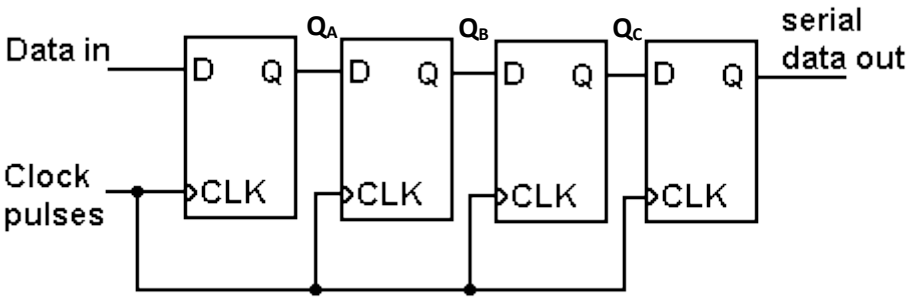
……

<u>LAB7.3</u> Serial in and Serial out Shift Register

1. Create new Project name “Shift_register”.
2. Create new entity for shift register in this below.



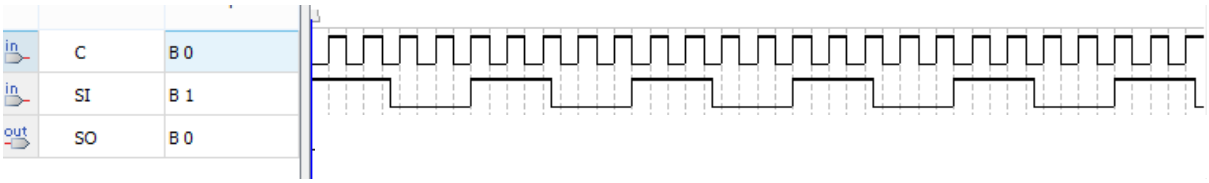
The time diagram of shift register 4 bit



The shift register 4 bit using D-flipflop.

```
Library ieee;
Use ieee.std_logic_1164.all;

Entity S_Left is
  Port(C, SI : in  std_logic;
       SO : out std_logic);
End S_Left;
Architecture behavior of S_Left is
  Signal tmp: std_logic_vector(3 downto 0);
  Begin
    Process (C)
      Begin
        if (C'event and C='1') then
          for i in 0 to 2 loop
            tmp(i+1) <= tmp(i);
          end loop;
          tmp(0) <= SI;
        end if;
    end Process;
    SO <= tmp(3);
End behavior;
```

3. Compile the code and program to DE0.



Conclusion

……

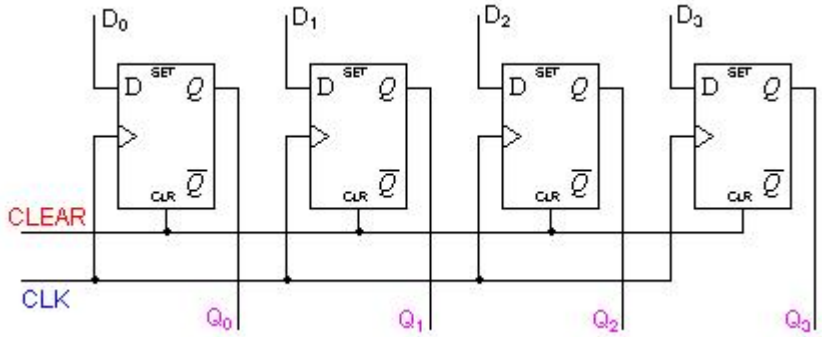
……

……

……

LAB7.4 Parallel in- Parallel out Shift Register

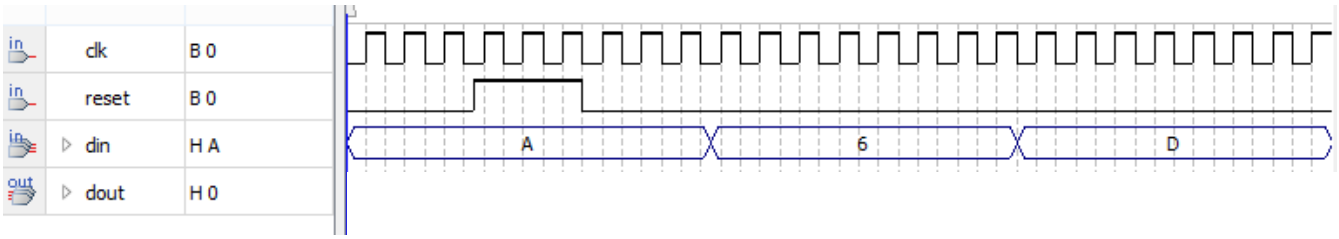
1. Create new Project name “Pin_Pout”.
2. Create new entity for shift register in this below.



The diagram of Parallel in- Parallel out 4 bits.

```
Library IEEE;
Use IEEE.STD_LOGIC_1164.all;

Entity Pin_Pout is
    Port(
       din : in STD_LOGIC_VECTOR (3 downto 0);
       clk : in STD_LOGIC;
       reset : in STD_LOGIC;
       dout : out STD_LOGIC_VECTOR(3 downto 0)
       );
End Pin_Pout;

Architecture behavior of Pin_Pout is

Component D_Flip_Flop is
    PORT (          Clk, D, Rst         : IN       STD_LOGIC;
                                        Q                   : OUT    STD_LOGIC       );
End Component D_Flip_Flop;


Begin

   u0 : D_Flip_Flop port map (clk,din(0),reset,dout(0));
   u1 : D_Flip_Flop port map (clk,din(1),reset,dout(1));
   u2 : D_Flip_Flop port map (clk,din(2),reset,dout(2));
   u3 : D_Flip_Flop port map (clk,din(3),reset,dout(3));

End behavior;

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY D_Flip_Flop IS
          PORT (  Clk, D, Rst         : IN       STD_LOGIC;
                                        Q                   : OUT    STD_LOGIC       );
```

```
END D_Flip_Flop;

ARCHITECTURE Structural OF D_Flip_Flop IS
	SIGNAL Qa: STD_LOGIC;
BEGIN
	PROCESS (D, Clk, Rst)
	BEGIN
		IF (RST = '1') THEN
			Qa <= '0';
		ELSIF (Clk = '1' and Clk'event)
			Qa <= D;
		END IF;
	END PROCESS;
	Q <= Qa;
END Structural;
```



Conclusion

……

……

……

……