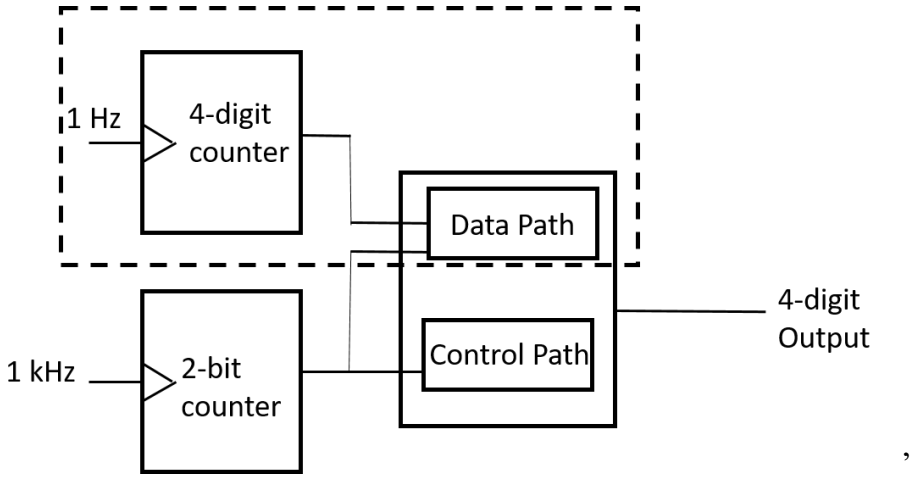
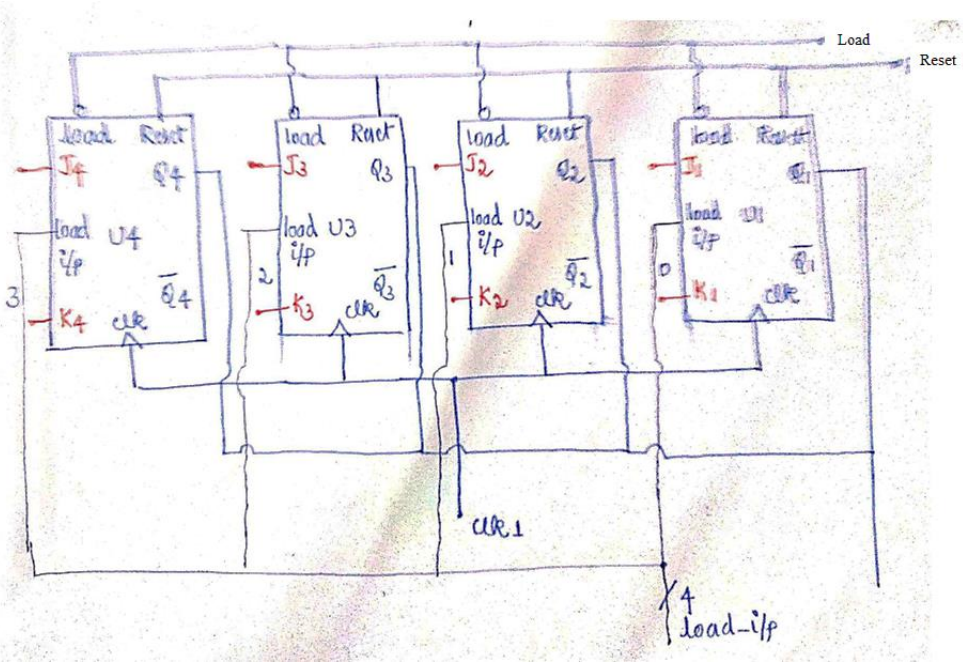
Instructions for this Week's Experiment:

1. This week's experiment is on interfacing array of 7-segment display with the CPLD board.
2. You will be implementing 2-digit counter using the set up
3. The block diagram shown below depicts the logic with which the entire setup will work



4. You have to write the code for the units that are enclosed in the dashed rectangle. The rest of the code is attached with this document.
5. Each digit in the 4-digit counter has four flipflops at the back end as shown in the figure below:



6. Please note that the frequency cannot change within 1-digit and hence each digit counter circuit has to be a synchronous circuit.
7. The LSD circuit (set of 4 flipflops) will get the clock from the outputs of the MSD. Hence, the inter digit circuit is asynchronous.

VHDL Code

```
-- When load = low, data will be loaded
-- when load = high, normal down operation.


-- write the VHDL code for loadable negative edge triggered JK  flip-flop

library IEEE;

use IEEE.STD_LOGIC_1164.ALL;

entity JK_FF is

Port (reset,J,K,clk1,Load,Load_input : in STD_LOGIC;

      q,qbar : out STD_LOGIC);

end JK_FF ;
```

```
--write the VHDL code for synchronous  BCD up/down counter using JK flipflops

library IEEE;

use IEEE.STD_LOGIC_1164.ALL;


entity BCD_up/down_counter is
Port (reset,clk1,Load : in STD_LOGIC;

            Load_input : in std_logic_vector( 3
            downto 0); Q: out std_logic_vector(3
            downto 0));

 end BCD_up/down_counter;
```

```
--write the VHDL code for 4  digit counter. Out of 4 last 2 digits (1,2) will be  loadable
UP/DOWN synchronous BCD counters (3,4 digits=0).



library IEEE;

use IEEE.STD_LOGIC_1164.ALL;


entity FOUR_BCD_counters is

 port (reset,clk1,Load: in std_logic;
```

```
        Load_input : in std_logic_vector( 3 downto 0);
        sel : in STD_LOGIC_vector(1 downto 0);
      Q3,Q2,Q1,Q0 : out STD_LOGIC_vector(3 downto 0));
  end FOUR_BCD_counters ;
--------------------------------------------------------------------------------
```

--write VHDL code for datapath for selecting count values to be displayed on particular SSD.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity data_path is      --data_path
Port ( reset,clk1,Load : in std_logic;
       Load_input : in STD_LOGIC_vector(3 downto 0);
                    sel : in STD_LOGIC_vector(1 downto 0);
        S : in STD_LOGIC_vector(1 downto 0);
         Y_data: out std_logic_vector(3 downto 0));
end data_path;
```

architecture of data_path is to be written by the students

```
---------- Control Path includes decoder 2x4,
  binary_shifting_counter library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Decoder_2x4 is
Port (A,B : in STD_LOGIC;
       Y: out STD_LOGIC_vector(3 downto 0));
  end Decoder_2x4;

architecture Behavioral of Decoder_2x4
  is begin
  Y(0) <= NOT((NOT A) AND (NOT B));
  Y(1) <= NOT((NOT A) AND B);
  Y(2) <= NOT(A AND (NOT B));
    Y(3) <= NOT(A AND B);

end Behavioral;

 ----------------------------------------Control Path----------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity binary_shifting_counter is
  Port (reset,clk2  : in STD_LOGIC;                        -- CLK2 is
        1KHz Q : out STD_LOGIC_vector(1 downto 0);
                    clk_carry : out
  std_LOGIC); end
  binary_shifting_counter;

architecture Behavioral of binary_shifting_counter is

component JK_FF is Port (reset,J,K,clk1,Load,Load_input : in STD_LOGIC;
                                              q,qbar : out
                                             STD_LOGIC);
end component;

signal p0,p1,p0bar,p1bar : std_logic;
begin                                                  Asynchronous 2 bit JK FF
U1: JK_FF port map(reset,p0bar,p0,clk2,'1','0',p0,p0bar);            -- CLK2 is
  1KHz U2: JK_FF port map(reset,p1bar,p1,p0,'1','0',p1,p1bar);
clk_carry <= p0 nor
  p1;
Q(0) <= p0;
Q(1) <= p1;
end Behavioral;
```

```
----------VHDL CODE FOR TOP ENTITY-----------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity top_entity2 is
Port (reset,clk1,Load,clk2: in STD_LOGIC; --add,clk2
Load_input: in STD_LOGIC_vector(3 DOWNTO 0); sel
: in STD_LOGIC_vector(1 downto 0); DP : out
std_LOGIC;
                S,Y_data: out STD_LOGIC_vector(3 DOWNTO
                 0); buzzer_signal: out std_LOGIC);
end top_entity2;

architecture Behavioral of top_entity2 is


component data_path is    --
data_path Port ( reset,clk1,Load :
in std_logic;
       Load_input : in STD_LOGIC_vector(3 downto 0);
       sel : in STD_LOGIC_vector(1 downto 0);
        S : in STD_LOGIC_vector(1 downto 0);         -- From control
                           path
        Y_data: out std_logic_vector(3 downto 0));
end component;

component binary_shifting_counter is
Port (reset,clk2  : in STD_LOGIC;                    -- CLK2 is
     1KHz Q : out STD_LOGIC_vector(1 downto 0);
                clk_carry : out
std_LOGIC); end component;

component Decoder_2x4
is Port (A,B : in
STD_LOGIC;
     Y: out STD_LOGIC_vector(3 downto 0));
end component;
signal s_switching: std_logic_vector(3 downto
0); signal Qx : std_LOGIC_vector(1 downto 0);
signal clk_carry1: std_LOGIC;
begin

U1: Decoder_2x4 port map(Qx(1),Qx(0),s_switching);          -- CLK2 is 1KHz
U2: data_path port map( reset,clk1 and load,Load,Load_input,sel,Qx,Y_data);
U3: binary_shifting_counter port map(reset,clk2 and load,Qx,clk_carry1);

  S <=
s_switching;
  DP <= '1';
buzzer_signal<=’0’:
  end Behavioral;
```