CONESTOGA
Connect Life and Learning

DIGITAL DESIGN PRINCIPLES

LAB 8 – SHIFT REGISTER

Name & Surname : Sinan KARACA **Date :** 29 / 07 / 2021

Student ID : 8743013

INTRODUCTION

The purpose of the experiment is to design right and left shift register, the design has been done with functions of VHDL and Verilog.

WORKING PRINCIPLE

The design is implemented with the functions of VHDL and Verilog, there are 3 different mode of working princible. When selection is “00” it means NULL, “01” it means 1 bit left shift, “10” it means 1 bit right shift and “11” it means load the 4 bits to output.

Switch case has been implemented for selection left or right, it is working under the clock process. Clock process is being triggered with the positive rising edge of KEY clock.

VHDL MAIN PROGRAM

```
----------------------------------------
----------------------------------------
-- Project Name  : Shift Register
-- File          : skaraca_lab8
-- Creator       : Sinan KARACA
-- Student number: 8743013
-- Date          : 29.07.2021
----------------------------------------
----------------------------------------

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity skaraca_Lab8 is
   port(
      SW       : in STD_LOGIC_VECTOR (7 downto 0);
      Clock    : in STD_LOGIC;
      LEDR     : out STD_LOGIC_VECTOR(3 downto 0));
   end skaraca_Lab8;


architecture Behavioral of skaraca_Lab8 is


function shiftLeft(

   threeBitLeft:in STD_LOGIC_VECTOR(2 downto 0);
   registerBitLeft:in STD_LOGIC)
   return STD_LOGIC_VECTOR is variable output: STD_LOGIC_VECTOR(3 downto 0);

   begin
      output := threeBitLeft & registerBitLeft;
      return output;

end shiftLeft;


function shiftRight(

   threeBitRight:in STD_LOGIC_VECTOR(2 downto 0);
   registerBitRight:in STD_LOGIC)
   return STD_LOGIC_VECTOR is variable output: STD_LOGIC_VECTOR(3 downto 0);

   begin
      output := registerBitRight & threeBitRight ;

      return output;

end shiftRight;
```

```
function Load(

   fourBit:in STD_LOGIC_VECTOR(3 downto 0))
   return STD_LOGIC_VECTOR is variable output: STD_LOGIC_VECTOR(3 downto 0);

   begin
      output := fourBit;
      return output;

end Load;


signal R : STD_LOGIC_VECTOR(3 downto 0);
signal S : STD_LOGIC_VECTOR(1 downto 0);


begin
   process(Clock, S, SW)

   begin

      S <= SW(7 downto 6);
      LEDR <= R;

      if rising_edge(Clock) then

         case S is
               when "00" =>
                  NULL;
               when "01" =>
                  R <= shiftLeft(R(2 downto 0), SW(0));

               when "10" =>
                  R <= shiftRight(R(3 downto 1), SW(5));

               when "11" =>
                  R <= Load (SW(4 downto 1));

               when others =>
                  NULL;
         end case;

      end if;

   end process;


end Behavioral;
```

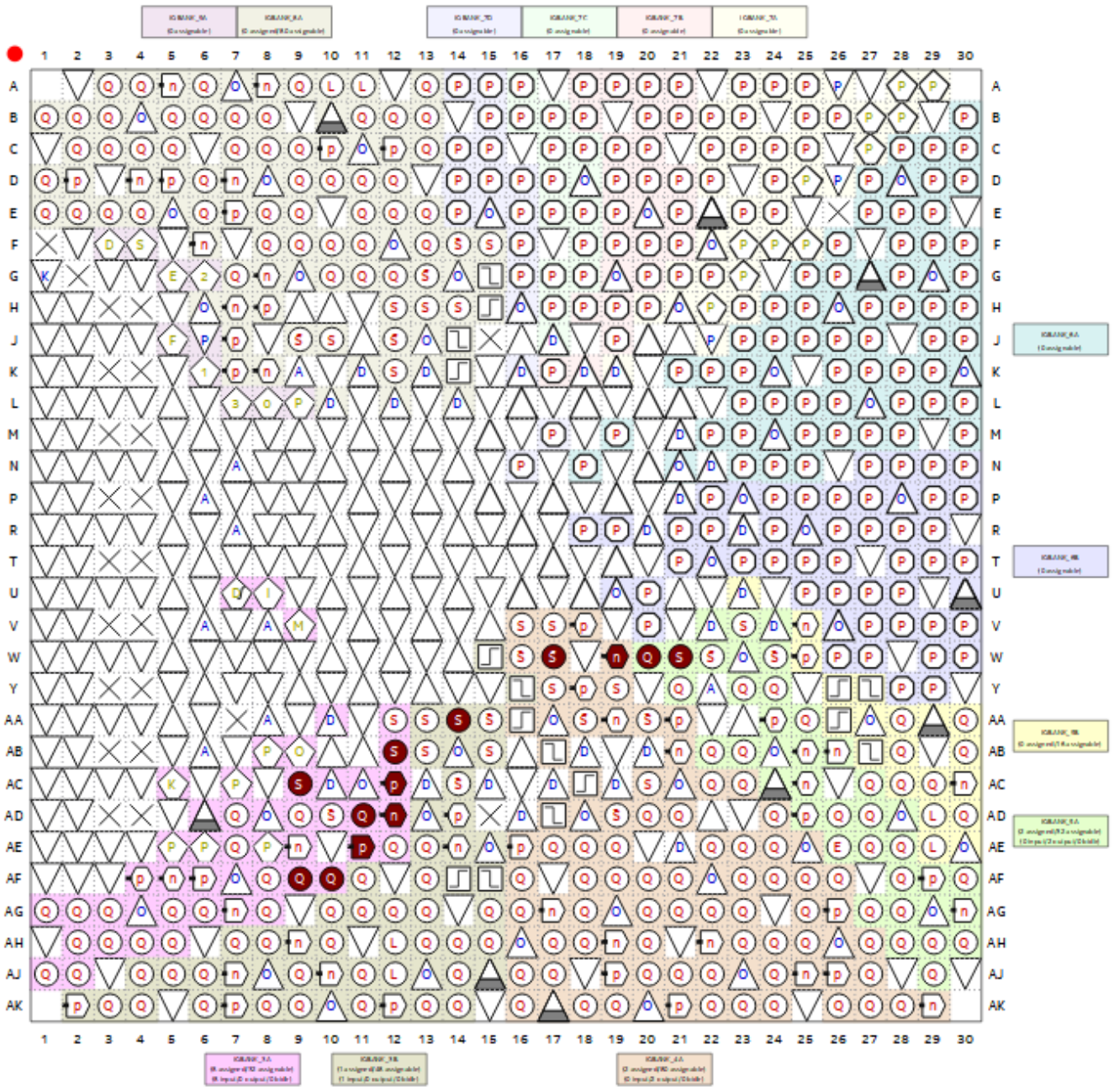
VERILOG MAIN PROGRAM

```
//////////////////////////////////////////////
//////////////////////////////////////////////
// Project Name  : Shift Register
// File          : skaraca_lab8
// Creator       : Sinan KARACA
// Student number: 8743013
// Date          : 29.07.2021
//////////////////////////////////////////////
//////////////////////////////////////////////

module skaraca_Lab8(SW,Clock,LEDR);

    input [7:0]SW;
    input Clock;
    output [3:0]LEDR;


    wire [3:0]R;
    wire [1:0]S;

    function [3:0] shiftLeft;
       input [2:0]threeBitLeft ;
       input registerBitLeft;

       begin

       shiftLeft[3:0] =  { threeBitLeft[2:0], registerBitLeft};

       end
    endfunction

    function [3:0] shiftRight;
       input [2:0] threeBitRight;
       input registerBitRight;

       begin

       shiftRight[3:0] = { registerBitRight, threeBitRight[2:0] };

       end
    endfunction

    function [3:0] Load;
       input [3:0] threeBitRight;


       begin

       Load = threeBitRight;

       end
    endfunction


    // Asynchronous Assignments
    assign S = SW[7:6];
    assign LEDR = R;


    //Clock Assigments
    always @ (posedge Clock) begin
       case(S)

          2'b01     :    R = shiftLeft(R[2:0], SW[0]);
          2'b10     :    R = shiftRight(R[3:1], SW[5]);
          2'b11     :    R = Load(SW[4:1]);

       endcase
    end


endmodule
```

PIN ASSIGNMENT

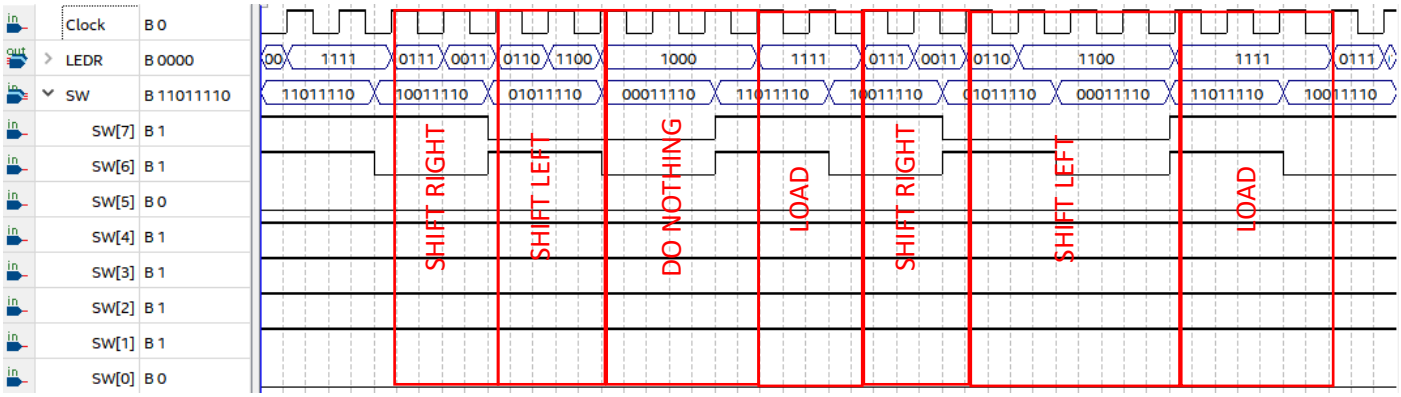
Top View - Wire Bond

Cyclone V - 5CSEMA5F31C6



Clock	Input	PIN_AA14	3B	B3B_N0	PIN_AA14	2.5 V		12mA (default)	
LEDR[3]	Output	PIN_W17	4A	B4A_N0	PIN_W17	2.5 V		12mA (default)	1 (default)
LEDR[2]	Output	PIN_W19	4A	B4A_N0	PIN_W19	2.5 V		12mA (default)	1 (default)
LEDR[1]	Output	PIN_W20	5A	B5A_N0	PIN_W20	2.5 V		12mA (default)	1 (default)
LEDR[0]	Output	PIN_W21	5A	B5A_N0	PIN_W21	2.5 V		12mA (default)	1 (default)
SW[7]	Input	PIN_AB12	3A	B3A_N0	PIN_AB12	2.5 V		12mA (default)	
SW[6]	Input	PIN_AC12	3A	B3A_N0	PIN_AC12	2.5 V		12mA (default)	
SW[5]	Input	PIN_AF9	3A	B3A_N0	PIN_AF9	2.5 V		12mA (default)	
SW[4]	Input	PIN_AF10	3A	B3A_N0	PIN_AF10	2.5 V		12mA (default)	
SW[3]	Input	PIN_AD11	3A	B3A_N0	PIN_AD11	2.5 V		12mA (default)	
SW[2]	Input	PIN_AD12	3A	B3A_N0	PIN_AD12	2.5 V		12mA (default)	
SW[1]	Input	PIN_AE11	3A	B3A_N0	PIN_AE11	2.5 V		12mA (default)	
SW[0]	Input	PIN_AC9	3A	B3A_N0	PIN_AC9	2.5 V		12mA (default)	

SIMALATION WAVEFORM



Simulation waveform shows the all combinations of 4 different scenarios.

SW[1-4] = “1111” , so we are suppose to see “1111” when it is loaded.

And also SW[0] and SW[5] is equal to “0”, so we are suppose to see zeros when it is shifted.

CONCLUSION

Design of the shift register has been doen in this experiment. Each operation has been done with functions. The function is not necessary to use, I used the functions for learning deeply about VHDL and Verilog.