Lab 6 - 2

Date: 11th march, 2020

Lab Title : Full Adder

Recorded and Submitted by : Bhesaniya Abhishek

Objective : To write VHDL and Vlog program for full Adder and test it using DESoC - I FBGA Board.

Implementation :

full adder is a combinational logic circuit which has 3 input and 2 output.

VHDL procedure :

- Write a code for 4 full adder.
- we will use signal for both input and output

```
Signal A : std_logic_vector(3 downto 0);
Signal B : std_logic_vector(7 downto 4);
signal carry : std_logic_vector(6 downto 4);
Signal cout : std_logic;
signal cin : std_logic;

where   A <= SW(3 downto 0);
        B <= SW(7 downto 4);
        CIN <= C;

Sum(0) <= (A(0) XOR B(4)) XOR CIN;
Carry(4) <= (A(0) AND B(4)) OR (CIN AND (A(0) XOR B(4)));
```

and like wise for all 3 full adder.

- At the end we will get 4 sum bit and 1 carry bit.
- Sum and carry will be shown on sevensegment as well as on LED.
- Then do pin assignment and compile your program
- After compiling download your program on board and Test it for different case.

Vlog procedure:

This time Again we are going to use reg for output and wire for input.

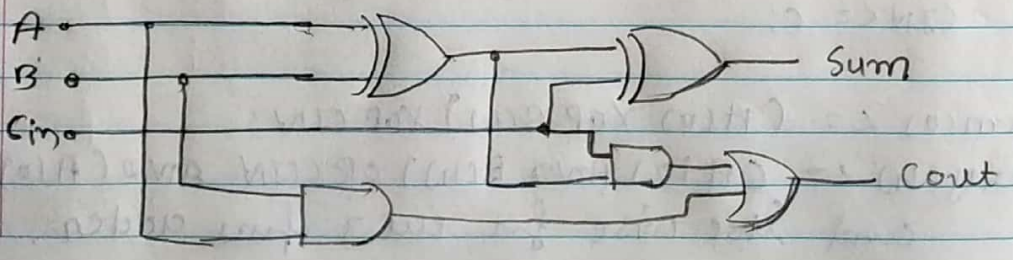
```
wire [3:0] A = SW[3:0];
wire [7:4] B = SW[7:4];
reg [3:0] sum;
reg [6:4] carry;
reg cout;
```

After writing your logic compile your code and go for pin assignment.

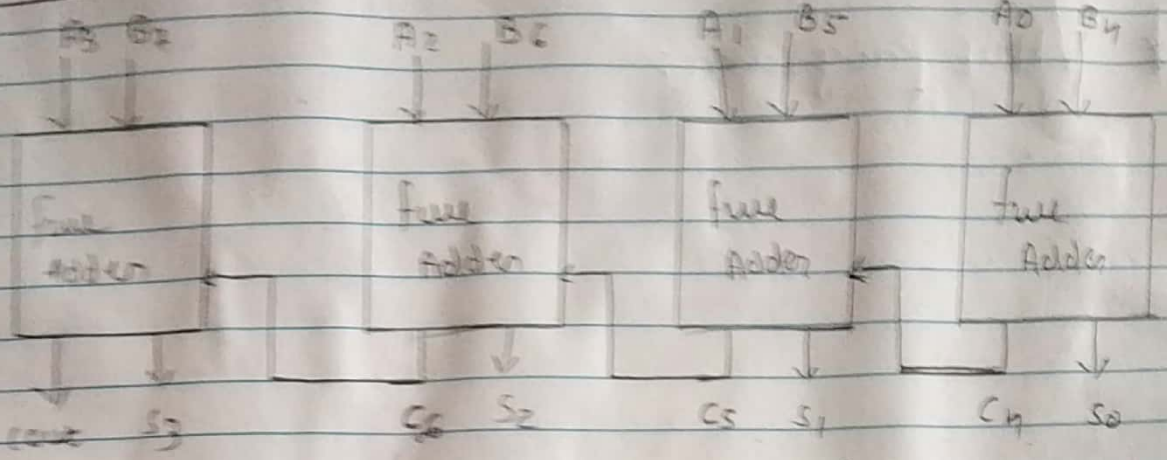
in this case also sum and carry will be shown on sevensegement and LEDs both.

- Load your code on board and test it for different cases.

Full Adder:



Circuit diagram :



Truth Table

A	B	Cin	Sum	Cout	A	B	Cin	Sum	Cout
0	0	0	0	0	1	0	1	0	1
0	0	1	1	0	1	1	0	0	1
0	1	0	1	0	1	1	1	1	1
0	1	1	0	1					
1	0	0	1	0					

Pin Assignment :

in	CIN	Input	PIN_AD10
in	FLIP	Input	PIN_AE12
out	LED[4]	Output	PIN_W17
out	LED[3]	Output	PIN_V18
out	LED[2]	Output	PIN_V17
out	LED[1]	Output	PIN_W16
out	LED[0]	Output	PIN_V16
out	SLED0[6]	Output	PIN_AE26
out	SLED0[5]	Output	PIN_AE27
out	SLED0[4]	Output	PIN_AE28
out	SLED0[3]	Output	PIN_AG27
out	SLED0[2]	Output	PIN_AF28
out	SLED0[1]	Output	PIN_AG28
out	SLED0[0]	Output	PIN_AH28
out	SLED1[6]	Output	PIN_AJ29
out	SLED1[5]	Output	PIN_AH29
out	SLED1[4]	Output	PIN_AH30
out	SLED1[3]	Output	PIN_AG30
out	SLED1[2]	Output	PIN_AF29
out	SLED1[1]	Output	PIN_AF30
out	SLED1[0]	Output	PIN_AD27
out	SLED2[6]	Output	PIN_AB23
out	SLED2[5]	Output	PIN_AE29
out	SLED2[4]	Output	PIN_AD29
out	SLED2[3]	Output	PIN_AC28
out	SLED2[2]	Output	PIN_AD30
out	SLED2[1]	Output	PIN_AC29
out	SLED2[0]	Output	PIN_AC30
out	SLED3[6]	Output	PIN_AD26
out	SLED3[5]	Output	PIN_AC27
out	SLED3[4]	Output	PIN_AD25
out	SLED3[3]	Output	PIN_AC25
out	SLED3[2]	Output	PIN_AB28
out	SLED3[1]	Output	PIN_AB25
out	SLED3[0]	Output	PIN_AB22
out	SLED4[6]	Output	PIN_AA24
out	SLED4[5]	Output	PIN_Y23
out	SLED4[4]	Output	PIN_Y24
out	SLED4[3]	Output	PIN_W22
out	SLED4[2]	Output	PIN_W24
out	SLED4[1]	Output	PIN_V23
out	SLED4[0]	Output	PIN_W25
in	SW[7]	Input	PIN_AC9
in	SW[6]	Input	PIN_AE11
in	SW[5]	Input	PIN_AD12
in	SW[4]	Input	PIN_AD11
in	SW[3]	Input	PIN_AF10
in	SW[2]	Input	PIN_AF9
in	SW[1]	Input	PIN_AC12
in	SW[0]	Input	PIN_AB12

Programs

* VHDL Program:

```
1
2    library ieee;
3    use ieee.std_logic_1164.all;
4
5
6    entity abhesaniyalab62 is
7    port(
8      SW : in std_logic_vector(7 downto 0);
9      FLIP : in std_logic;
10     C : in std_logic;
11   LED : out std_logic_vector(4 downto 0);
12   SLED0: out std_logic_vector(6 downto 0);
13   SLED1: out std_logic_vector(6 downto 0);
14   SLED2: out std_logic_vector(6 downto 0);
15   SLED3: out std_logic_vector(6 downto 0);
16   SLED4: out std_logic_vector(6 downto 0)
17
18   );
19   end abhesaniyalab62;
20
21   architecture fulladder of abhesaniyalab62 is
22
23   signal A : std_logic_vector(3 downto 0) ;
24   signal B : std_logic_vector(7 downto 4) ;
25   signal SUM : std_logic_vector(3 downto 0) ;
26   signal CARRY : std_logic_vector(6 downto 4) ;
27   signal COUT : std_logic;
28   signal CIN : std_logic;
29
30
31   begin
32
33      A <= SW(3 downto 0);
34      B <= SW(7 downto 4);
35      CIN <= C;
36
37      SUM(0) <= (A(0) XOR B(4)) XOR CIN;
38      CARRY(4) <= (A(0) AND B(4))  OR (CIN AND (A(0) XOR B(4)));
39
40      SUM(1) <= (A(1) XOR B(5)) XOR CARRY(4);
41      CARRY(5) <= (A(1) AND B(5))  OR (CARRY(4) AND (A(1) XOR B(5)));
42
43      SUM(2) <= (A(2) XOR B(6)) XOR CARRY(5);
44      CARRY(6) <= (A(2) AND B(6))  OR (CARRY(5) AND (A(2) XOR B(6)));
45
46      SUM(3) <= (A(3) XOR B(7)) XOR CARRY(6);
47      COUT <= (A(3) AND B(7))  OR (CARRY(6) AND (A(3) XOR B(7)));
48
49
50      LED(0) <= SUM(0);
51      LED(1) <= SUM(1);
52      LED(2) <= SUM(2);
53      LED(3) <= SUM(3);
54      LED(4) <= COUT;

55
56
57      process (SUM , CARRY, FLIP)
58      begin
59
60        if ( FLIP = '0') then
61           SLED4 <= "0100100";
62
63           if (SUM(0) = '0') then
64              SLED0 <="0000001";
65           else if (SUM(0) = '1') then
66              SLED0 <="1001111";
67           end if;
68           end if;
69
70           if (SUM(1) = '0') then
71            SLED1 <="0000001";
72           else if (SUM(1) = '1') then
73            SLED1 <="1001111";
74           end if;
75           end if;
76
77           if (SUM(2) = '0') then
78            SLED2 <="0000001";
79           else if (SUM(0) = '1') then
80            SLED2 <="1001111";
81           end if;
82           end if;
```

```
        if (SUM(3) = '0') then
         SLED3 <="0000001";
        else if (SUM(3) = '1') then
         SLED3 <="1001111";
        end if;
        end if;

        else if(FLIP = '1')then

            SLED4 <="0110001";
            SLED1 <="1111111";
            SLED2 <="1111111";
            SLED3 <="1111111";
            if (COUT = '0') then
              SLED0 <="0000001";
            else if (COUT = '1') then
              SLED0 <="1001111";
            end if;
            end if;

        end if;
        end if;

    end process;
```

* Vlog Program

VLOG Program

```
1
2
3   module abhesaniyalab662 (SW,FLIP,CIN,LED,SLED0,SLED1,SLED2,SLED3, SLED4);
4
5   input [7:0]SW;
6   input FLIP;
7   input CIN;
8   output[4:0]LED;
9   output[6:0]SLED0;
10  output[6:0]SLED1;
11  output[6:0]SLED2;
12  output[6:0]SLED3;
13  output[6:0]SLED4;
14
15  reg [4:0]LED;
16  reg [6:0]SLED0;
17  reg [6:0]SLED1;
18  reg [6:0]SLED2;
19  reg [6:0]SLED3;
20  reg [6:0]SLED4;
21
22  wire [3:0] A = SW[3:0];
23  wire [7:4] B = SW[7:4];
24  reg [3:0] SUM;
25  reg [6:4]CARRY;
26  reg COUT;
27
28
29  always @(A or CIN or B or FLIP)
30  begin
31
32          SUM[0] = (A[0] ^ B[4]) ^ CIN;
33          CARRY[4] = (A[0] & B[4]) | ((A[0] ^ B[4]) & CIN);
34
35          SUM[1] = (A[1] ^ B[5]) ^ CARRY[4];
36          CARRY[5] = (A[1] & B[5]) | ((A[1] ^ B[5]) & CARRY[4]);
37
38          SUM[2] = (A[2] ^ B[6]) ^ CARRY[5];
39          CARRY[6] = (A[2] & B[6]) | ((A[2] ^ B[6]) & CARRY[5]);
40
41          SUM[3] = (A[3] ^ B[7]) ^ CARRY[6];
42          COUT = (A[3] & B[7]) | ((A[3] ^ B[7]) & CARRY[6]);
43
44
45  LED[3:0] = SUM;
46  LED[4] = COUT;
47
48      if (FLIP == 0)
49      begin
50          SLED4 = 7'b0100100;
51          if (SUM[0] == 0)
52            SLED0 = 7'b0000001;
53          else if (SUM[0] == 1)
54            SLED0 = 7'b1001111;
55
56
57          if (SUM[1] == 0)
58            SLED1 = 7'b0000001;
59          else if (SUM[1] == 1)
60            SLED1 = 7'b1001111;
61
62          if (SUM[2] == 0)
63            SLED2 = 7'b0000001;
64          else if (SUM[2] == 1)
65            SLED2 = 7'b1001111;
66
67          if (SUM[3] == 0)
68            SLED3 = 7'b0000001;
69          else if (SUM[3] == 1)
70            SLED3 = 7'b1001111;
71      end
72
73      else if (FLIP == 1)
74      begin
75
76          SLED4 = 7'b0110001;
77          SLED1 = 7'b1111111;
78          SLED2 = 7'b1111111;
79          SLED3 = 7'b1111111;
80
81          if (COUT == 0)
82            SLED0 = 7'b0000001;
83          else if (COUT == 1)
84            SLED0 = 7'b1001111;
85
86
87      end
88
89  end
90
91  endmodule
```

Discussion:

LSB A MSB
SW0 SW3

```
  1 1 1
A 1010
B 1110
  11001
```