LAB NOTE

Subject: Digital Design Principles

Topic: Full Adder

Student: Minh Quan Tran

July 5th, 2024

Table of Contents

TABLE OF FIGURES

1. Objectives

- Design a Full Adder using Quartus software.
- Write a VHDL and Verilog code from the design.
- Push the code and run to the SCEMA5F31C6N board.

2. Theory and Design

2.1 Theory

A full adder is a circuit that forms the arithmetic sum of three inputs bits.

- It has three inputs and two outputs.
- Two inputs are used for the input to the sum.
- The remaining input is the Carry out from the previous stage

2.2 Requirement

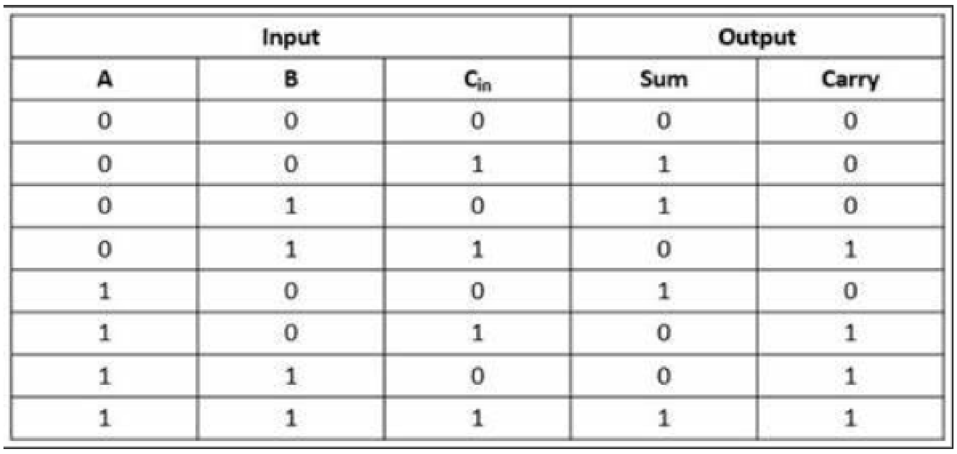
- Use VHDL and Verilog to implement Full Adder and display to LED [0:7].
- Introduction to the VHDL Case statement.
- **Above and beyond:** Adding 1 more modes: Half Subtractor

2.3 Solution

To design this system, first need to identify input and output:

- Input A will be from SW0 to SW3.
- Input B will be from SW4 to SW7.
- Output will be the 7 LED.

Then identify what the output will be using the below truth table,



Input			Output	
A	B	C_{in}	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Figure 2-1: Full Adder Truth Table

3. VHDL and Verilog

3.1 VHDL code

```
library ieee;
use ieee.std_logic_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity MTran_Lab6_VHDL_Adder is
    port
     (
        mode  : in   std_logic_vector (9 downto 8);
        A     : in   std_logic_vector (3 downto 0);
        B     : in   std_logic_vector (7 downto 4);
        LED   : out  std_logic_vector (7 downto 0);
        HEX1  : out  std_logic_vector (6 downto 0);
        HEX0  : out  std_logic_vector (6 downto 0)

      );
end MTran_Lab6_VHDL_Adder;

architecture Behavioral of MTran_Lab6_VHDL_Adder is

-------------------------- Component Declaration--------------------------------

--Half Adder
component MTran_Lab6_VHDL_HalfAdder is
    port
     (
        A     : in   std_logic_vector (3 downto 0);
        B     : in   std_logic_vector (7 downto 4);
        Sum   : out  std_logic_vector (3 downto 0);
        Carry : out  std_logic_vector (3 downto 0)
      );
end component;

--Half Subtractor
component MTran_Lab6_VHDL_HalfSubtractor is
    port
     (
        A            : in   std_logic_vector (3 downto 0);
        B            : in   std_logic_vector (7 downto 4);
        Difference   : out  std_logic_vector (3 downto 0);
        Borrow       : out  std_logic_vector (3 downto 0)
      );
end component;

--Full Adder
component MTran_Lab6_VHDL_FullAdder is
    port
     (
        A            : in   std_logic_vector (3 downto 0);
        B            : in   std_logic_vector (3 downto 0);
        Cin          : in   std_logic;
        Sum          : out  std_logic_vector (3 downto 0);
        CarryOut     : out  std_logic
      );
end component;

-------------------------- End of Component Declaration--------------------------
```

```
-------------------------- Function Declaration-----------------------------------

-- Function to calculate Sum of Half Adder

-- Seven Segment Display
function SevenSegmentDisplay(Number : integer) return std_logic_vector is
begin
    case (Number) is
    when 0 => return "1000000";
    when 1 => return "1111001";
    when 2 => return "0100100";
    when 3 => return "0110000";
    when 4 => return "0011001";
    when 5 => return "0010010";
    when 6 => return "0000010";
    when 7 => return "1111000";
    when 8 => return "0000000";
    when 9 => return "0010000";
    when others => return "1111111";
    end case;
end SevenSegmentDisplay;

--------------------------End of Function Declaration-------------------------

--------------------------Variable and Signal Declaration----------------------

signal Sum_HA : std_logic_vector(3 downto 0);
signal Carry_HA : std_logic_vector(3 downto 0);

signal Difference_HS: std_logic_vector(3 downto 0);
signal Borrow_HS: std_logic_vector(3 downto 0);

signal Sum_FA : std_logic_vector(3 downto 0);
signal Carry_FA : std_logic;
signal Cin_FA: std_logic;

signal TotalSum : integer;
signal tenth    : integer;
signal unit     : integer;
--------------------------End of Variable and Signal Declaration---------------

begin
-- Main code

-- Component
HalfAdder: MTran_Lab6_VHDL_HalfAdder
    port map(
        A => A,
        B => B,
        Sum => Sum_HA,
        Carry => Carry_HA
    );

HalfSubtractor: MTran_Lab6_VHDL_HalfSubtractor
    port map(
        A => A,
        B => B,
        Difference => Difference_HS,
        Borrow => Borrow_HS
    );

FullAdder: MTran_Lab6_VHDL_FullAdder
```

```
    port map(
        A => A,
        B => B,
        Cin => Cin_FA,
        Sum => Sum_FA,
        CarryOut => Carry_FA
    );


process(mode, A, B)
begin
    -- Reset Seven Segment Display and LED
    HEX1 <= "1111111";
    HEX0 <= "1111111";
    LED  <= "00000000";

    -- Change mode
    if mode = "00" then                 -- Half Adder mode
        LED(3 downto 0) <= Sum_HA;
        LED(7 downto 4) <= Carry_HA;
    elsif mode = "01" then          -- Half Subtractor mode
        LED(3 downto 0) <= Difference_HS;
        LED(7 downto 4) <= Borrow_HS;
    elsif mode = "10" then          -- Full Adder with Cin = 0  mode
        Cin_FA <= '0';
        LED(3 downto 0) <= Sum_FA;
        LED(4) <= Carry_FA;

        TotalSum <= to_integer(unsigned(Carry_FA & Sum_FA));
        tenth <= TotalSum / 10;
        unit  <= TotalSum mod 10;

        HEX1 <= SevenSegmentDisplay(tenth);
        HEX0 <= SevenSegmentDisplay(unit);
    elsif mode = "11" then          -- Full Adder with Cin = 1  mode
        Cin_FA <= '1';
        LED(3 downto 0) <= Sum_FA;
        LED(4) <= Carry_FA;

        TotalSum <= to_integer(unsigned(Carry_FA & Sum_FA));
        tenth <= TotalSum / 10;
        unit  <= TotalSum mod 10;

        HEX1 <= SevenSegmentDisplay(tenth);
        HEX0 <= SevenSegmentDisplay(unit);
    end if;
end process;
end Behavioral;
```

3.2 Verilog code

```
module MTran_Lab6_Verilog_Adder(
        input  wire [9:8] mode,
        input  wire [3:0] A,
        input  wire [3:0] B,
        output reg  [7:0] LED,
        output reg  [6:0] HEX1,
        output reg  [6:0] HEX0
);

/*******************Variable and Signal Declaration**************************/
```

```
wire [3:0] Sum_HA;
wire [3:0] Carry_HA;

wire [3:0] Difference_HS;
wire [3:0] Borrow_HS;

wire [3:0] Sum_FA;
reg Cin;
wire Carry_FA;

reg [4:0] TotalSum;
reg [4:0] tenth;
reg [4:0] units;
/********************End of Variable and Signal Declaration******************/

/********************Module Instantiation*************************************/

MTran_Lab6_Verilog_HalfAdder HA (
        .A(A),
        .B(B),
        .Sum(Sum_HA),
        .Carry(Carry_HA)
);

MTran_Lab6_Verilog_HalfSubtractor HS (
        .A(A),
        .B(B),
        .Difference(Difference_HS),
        .Borrow(Borrow_HS)
);

MTran_Lab6_Verilog_FullAdder FA (
        .A(A),
        .B(B),
        .Cin(Cin),
        .Sum(Sum_FA),
        .CarryOut(Carry_FA)
);
/********************End of Module Instantiation******************************/

/********************Function Declaration*************************************/

function [6:0] SevenSegmentDisplay;
    input reg [4:0] Num;
    begin
        case(Num)
            5'd0: SevenSegmentDisplay = 7'b1000000;
         5'd1: SevenSegmentDisplay = 7'b1111001;
         5'd2: SevenSegmentDisplay = 7'b0100100;
         5'd3: SevenSegmentDisplay = 7'b0110000;
         5'd4: SevenSegmentDisplay = 7'b0011001;
         5'd5: SevenSegmentDisplay = 7'b0010010;
         5'd6: SevenSegmentDisplay = 7'b0000010;
         5'd7: SevenSegmentDisplay = 7'b1111000;
         5'd8: SevenSegmentDisplay = 7'b0000000;
         5'd9: SevenSegmentDisplay = 7'b0010000;
            default: SevenSegmentDisplay = 7'b1111111;
        endcase;
    end
endfunction

/********************End of Function Declaration******************************/
```

```
// Main code
always@(mode,A,B)
begin
   // Reset LED and 7 Segment
    HEX1 = 7'b1111111;
    HEX0 = 7'b1111111;
    LED  = 8'b000000000;

    // Change mode
    if (mode == 2'b00)
        begin
            LED[3:0] = Sum_HA;
            LED[7:4] = Carry_HA;
        end
    else if (mode == 2'b01)
        begin
            LED[3:0] = Difference_HS;
            LED[7:4] = Borrow_HS;
        end
    else if (mode == 2'b10)
        begin
            Cin = 1'b0;

            LED[3:0] = Sum_FA;
            LED[7:4] = Carry_FA;

            TotalSum = {Carry_FA, Sum_FA};
            tenth = TotalSum / 10;
            units = TotalSum % 10;

            HEX1 = SevenSegmentDisplay(tenth);
            HEX0 = SevenSegmentDisplay(units);
        end
    else if (mode == 2'b11)
        begin
            Cin = 1'b1;

            LED[3:0] = Sum_FA;
            LED[7:4] = Carry_FA;

            TotalSum = {Carry_FA, Sum_FA};
            tenth = TotalSum / 10;
            units = TotalSum % 10;

            HEX1 = SevenSegmentDisplay(tenth);
            HEX0 = SevenSegmentDisplay(units);
        end

end
endmodule
```

4. Pin Planner

4.1 Input and Output

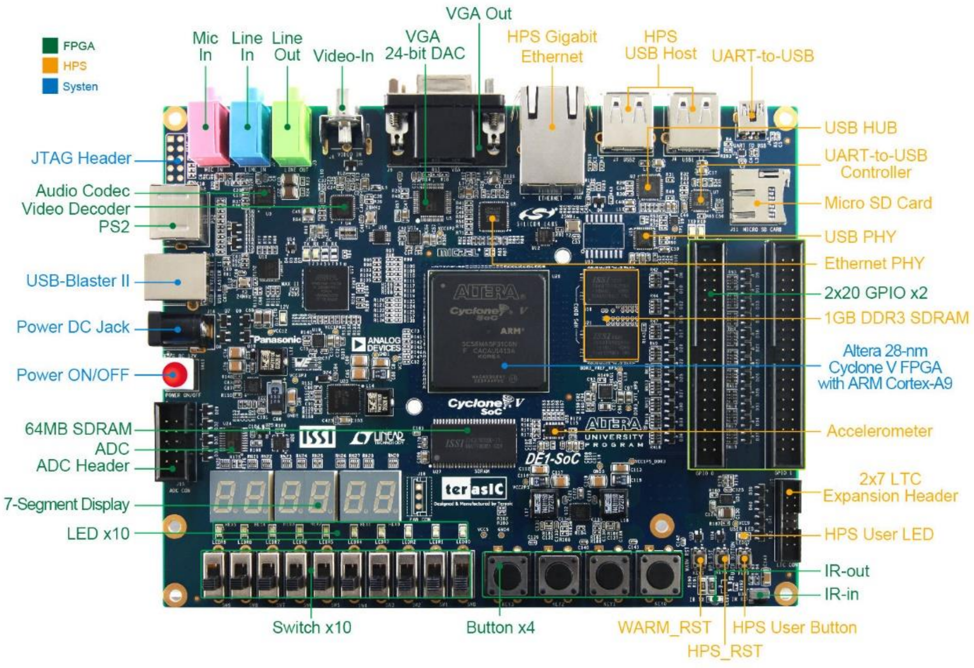


Figure 4-1: SCEMA5F31C6N board

Assigning:

SW[0]:	SW0
SW[1]:	SW1
SW[2]:	SW2

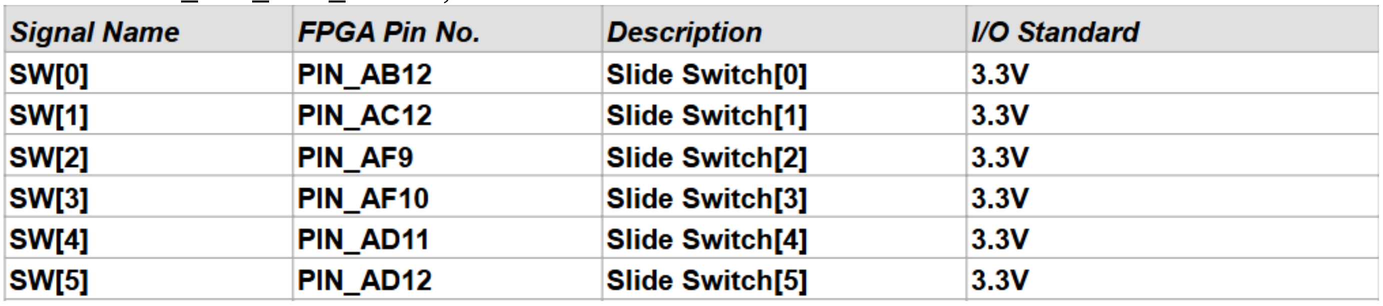
Mode[0]:	SW7
Mode[1]:	SW8
EN:	SW9

LED[0]:	LEDR0
LED[1]:	LEDR1
LED[2]:	LEDR2
LED[3]:	LEDR3
LED[4]:	LEDR4
LED[5]:	LEDR5
LED[6]:	LEDR6
LED[7]:	LEDR7

HEX0[0]:	HEX0[0]
HEX0[1]:	HEX0[1]
HEX0[2]:	HEX0[2]
HEX0[3]:	HEX0[3]
HEX0[4]:	HEX0[4]
HEX0[5]:	HEX0[5]
HEX0[6]:	HEX0[6]

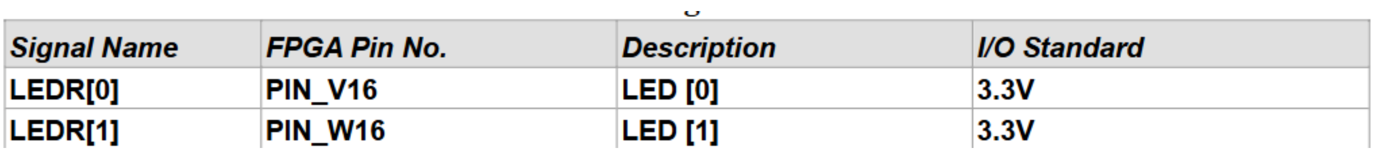
4. Pin Planner

From the DE1_SoC_User_Manual,



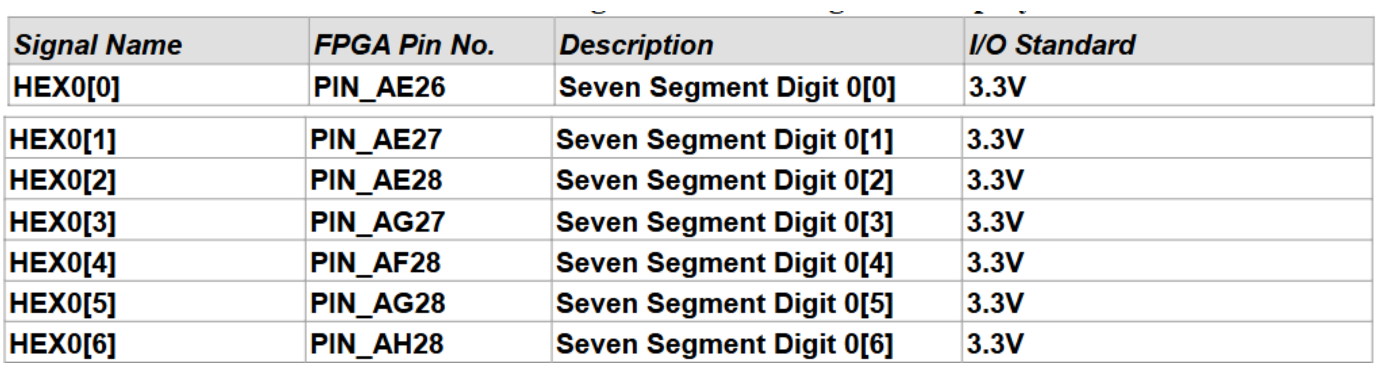
Signal Name	FPGA Pin No.	Description	I/O Standard
SW[0]	PIN_AB12	Slide Switch[0]	3.3V
SW[1]	PIN_AC12	Slide Switch[1]	3.3V
SW[2]	PIN_AF9	Slide Switch[2]	3.3V
SW[3]	PIN_AF10	Slide Switch[3]	3.3V
SW[4]	PIN_AD11	Slide Switch[4]	3.3V
SW[5]	PIN_AD12	Slide Switch[5]	3.3V

Figure 4-2: SW0 and SW1 Pin No



Signal Name	FPGA Pin No.	Description	I/O Standard
LEDR[0]	PIN_V16	LED [0]	3.3V
LEDR[1]	PIN_W16	LED [1]	3.3V

Figure 4-3: LEDR0's Pin No



Signal Name	FPGA Pin No.	Description	I/O Standard
HEX0[0]	PIN_AE26	Seven Segment Digit 0[0]	3.3V
HEX0[1]	PIN_AE27	Seven Segment Digit 0[1]	3.3V
HEX0[2]	PIN_AE28	Seven Segment Digit 0[2]	3.3V
HEX0[3]	PIN_AG27	Seven Segment Digit 0[3]	3.3V
HEX0[4]	PIN_AF28	Seven Segment Digit 0[4]	3.3V
HEX0[5]	PIN_AG28	Seven Segment Digit 0[5]	3.3V
HEX0[6]	PIN_AH28	Seven Segment Digit 0[6]	3.3V

Figure 4-4: HEX's Pin No

5. Result

5.1 Cin = 0

With inputA = “1000” and inputB = “1011” and Cin = 0, Sum should be “0011” and Carry is “1”

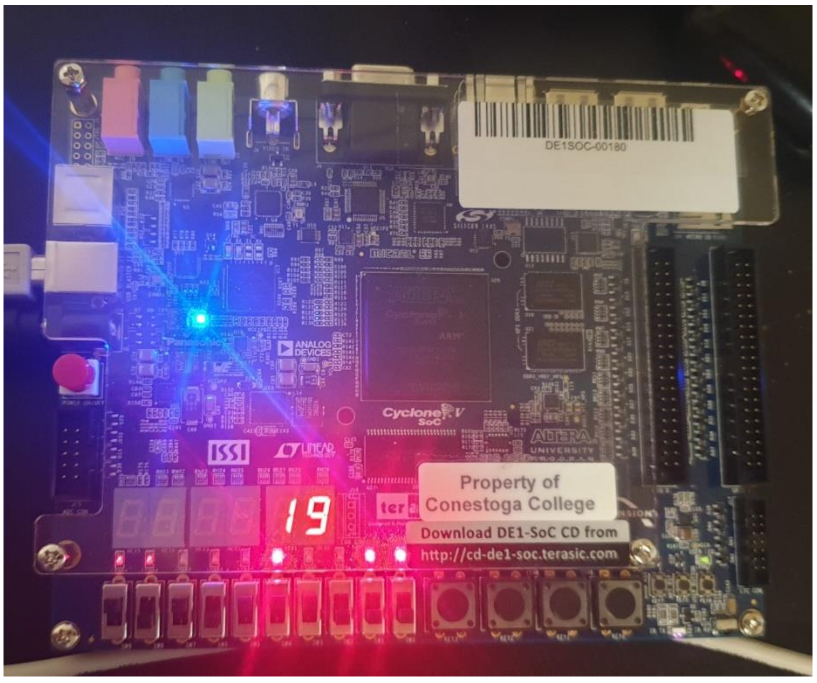


Figure 5-1: Output for Full Adder when Cin = 0.

5.2 Cin = 1

With the same input but Cin = 1. Sum should be “ 0100 ” and Carry is “1”

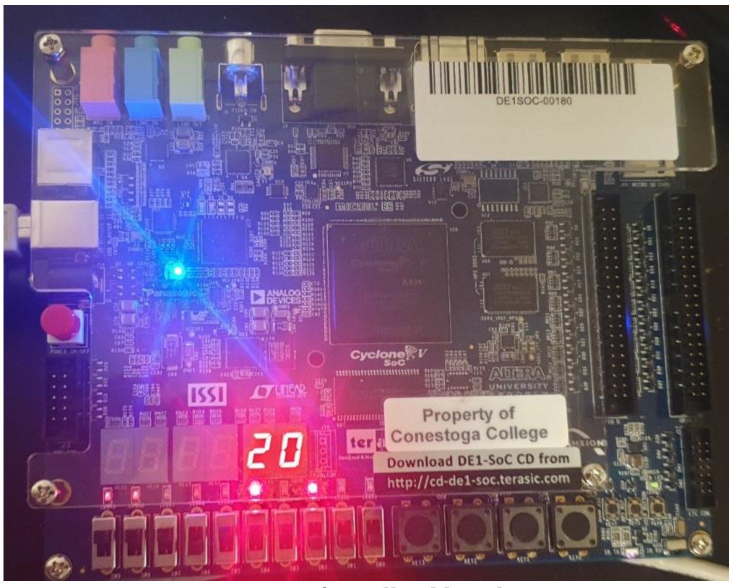


Figure 5-2: Output for Full Adder when Cin = 1

REFERENCES