CS1050- Lab 3

Index No: 210670N | Name : Vidushini.A.O

Part 1

The assigned task was first to design a circuit for a half adder , then to implement a full adder circuit using half adders using the software Vivado. The second task was to create a symbol for the full adder so that it can be utilized to design new circuits in new projects. The final task was to design a four bit adder circuit using four full adders and simulate the circuit using Xsim

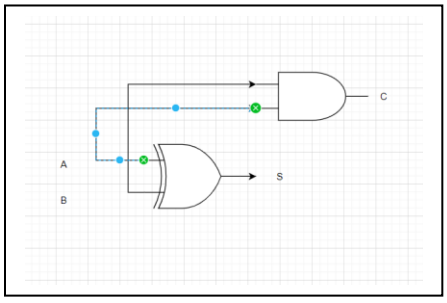
Part 2

- Half Adder

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

S = A XOR B

C = A AND B



- Full Adder

A	B	C_in	S	C_out
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

Using SoP

$S = \overline{A}\overline{B}C\_in + \overline{A}B\overline{C\_in} + A\overline{B}\,\overline{C\_in} + ABC$

$S = \overline{A}(\overline{B}C_in + B\overline{C_in}) + A(\overline{B}\,\overline{C\_in} + BC)$

$S = \overline{A}(B \oplus C) + A(\overline{B \oplus C})$

$S = A \oplus B \oplus C$

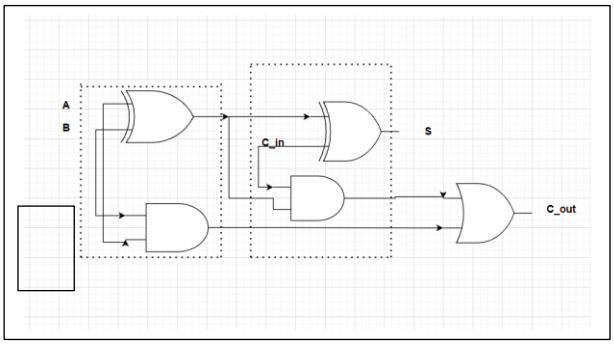
Using SoP

$C = \overline{A}BC\_in + A\overline{B}C\_in + AB\,\overline{C\_in} + ABC$

$S = \overline{A}(\overline{B}C_in + B\overline{C_in}) + A(\overline{B}\,\overline{C\_in} + BC)$

$S = \overline{A}(B \oplus C) + A(\overline{B \oplus C})$

$S = A \oplus B \oplus C$



Part 3- All VHDL files

HALF ADDER

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

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity HA is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           S : out STD_LOGIC;
           C : out STD_LOGIC);
end HA;

architecture Behavioral of HA is

begin
S <= A XOR B;
C <= A AND B;

end Behavioral;
```

FULL ADDER

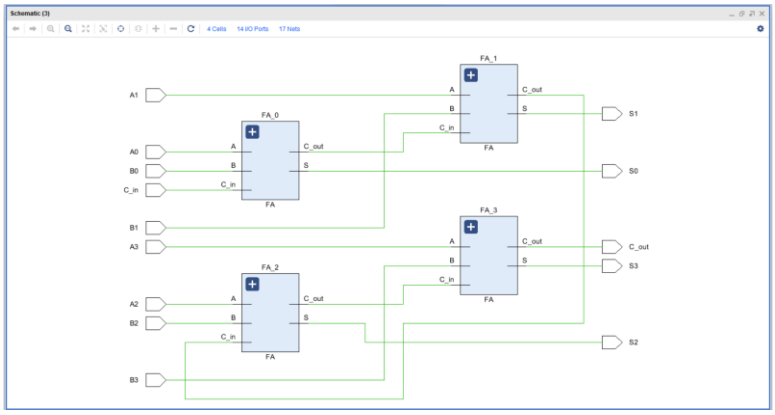
```
entity FA is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C_in : in STD_LOGIC;
           S : out STD_LOGIC;
           C_out : out STD_LOGIC);
end FA;

architecture Behavioral of FA is
component HA
 port (
 A: in std_logic;
 B: in std_logic;
 S: out std_logic;
 C: out std_logic);
end component;
SIGNAL HA0_S, HA0_C, HA1_S, HA1_C : std_logic;
begin

HA_0 : HA
 port map (
 A => A,
 B => B,
 S => HA0_S,
 C => HA0_C);
 HA_1 : HA
 port map (
 A => HA0_S,
 B => C_in,
 S => HA1_S,
 C => HA1_C);

 S <= HA1_S;
 C_out <= HA1_C OR HA0_C;

end Behavioral;
```

RCA_4

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 03/17/2023 12:46:24 AM
-- Design Name:
-- Module Name: RCA_4 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity RCA_4 is
```

```
entity RCA_4 is
    Port ( A0 : in STD_LOGIC;
           A1 : in STD_LOGIC;
           A2 : in STD_LOGIC;
           A3 : in STD_LOGIC;
           B0 : in STD_LOGIC;
           B1 : in STD_LOGIC;
           B2 : in STD_LOGIC;
           B3 : in STD_LOGIC;
           C_in : in STD_LOGIC;
           S0 : out STD_LOGIC;
           S1 : out STD_LOGIC;
           S2 : out STD_LOGIC;
           S3 : out STD_LOGIC;
           C_out : out STD_LOGIC);
end RCA_4;

architecture Behavioral of RCA_4 is
component FA
    port(
    A: in std_logic;
    B: in std_logic;
    C_in: in std_logic;
    S: out std_logic;
    C_out: out std_logic);
end component;

SIGNAL FA_0S, FA0_C,FA_1S, FA1_C,FA_2S, FA2_C,FA_3S, FA3_C: std_logic;


begin
    FA_0 :FA
        port map(
        A => A0,
        B=> B0,
        C_in => C_in,
        S => S0,
        C_out => FA0_C
        );
```

```
        );
    FA_1 :FA
        port map(
        A => A1,
        B=> B1,
        C_in => FA0_C,
        S => S1,
        C_out => FA1_C
        );
    FA_2 :FA
        port map(
        A => A2,
        B=> B2,
        C_in => FA1_C,
        S => S2,
        C_out => FA2_C
        );
     FA_3 :FA
        port map(
        A => A3,
        B=> B3,
        C_in => FA2_C,
        S => S3,
        C_out => C_out
        );


end Behavioral;
```

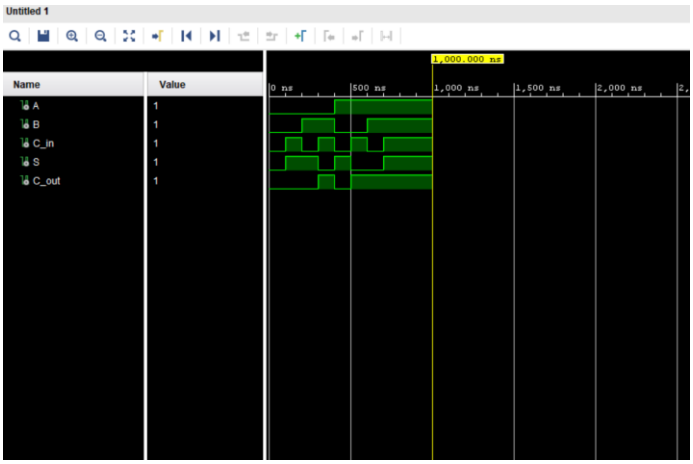


Schematic Diagram of the 4 bit Ripple Adder

Part 4- All Timing Graphs

HALF ADDER

FULL ADDER



RCA_4

Part 4- why some of the input combinations results in outputs that cannot be represented using LED LD0-LD3. Discuss the role of LD15?

Some of the input combinations results more than 4 bits which is five bits where it cannot be represented using 4 LED s

As an Example:

1111
+1110
11101

In this case the result cannot be represented alone with LED LD0-LD3 , To represent the overhead bit (circled in red in this case), the C_out signal which is represented by LD15 is used.

Part 5 – Other Conclusions

- With the hierarchical design, complex components can be designed using many basic components.
- A macro symbol can be built (Block) of a certain component to make it able to used in future designs in other Vivado projects.
- A full adder can be designed using two half adders and even more complex bit adders like the 4 bit ripple adder can be designed out of several full adder circuits.