Class : BE 6

Roll. No : 42260

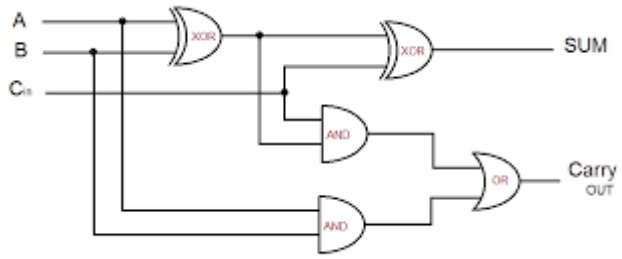
Assignment No. : 1

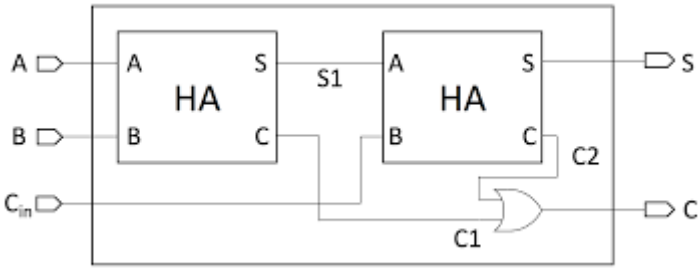
Assignment Name : Full Adder

Date Of Performance : 18/8/21

Assignment: Write a VHDL Program to Model a Full Adder. Synthesize the model for the target PLD. Simulate the model using a TestBench. Implement it on the target PLD

Block Diagram:





Truth Table:

Inputs			Outputs	
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

Main VHDL Program:

Behavioural

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

entity fullbehavior is
   Port ( A : in  STD_LOGIC;
          B : in  STD_LOGIC;
          CIN : in  STD_LOGIC;
          SUM : out  STD_LOGIC;
          COUT : out  STD_LOGIC);
end fullbehavior;

architecture Behavioral of fullbehavior is
signal temp: std_logic_vector(1 downto 0);
 begin
  process(A,B,CIN)
  begin
  temp <= ('0'& A) + ('0'& B) +('0'& CIN) ;
  end process;
  SUM <= temp(0);
  COUT <= temp(1);

end Behavioral;
```

Dataflow

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

entity fulladderlogic is
   Port ( A : in  STD_LOGIC;
          B : in  STD_LOGIC;
          CIN : in  STD_LOGIC;
          SUM : out  STD_LOGIC;
          COUT : out  STD_LOGIC);
end fulladderlogic;

architecture Behavioral_Sampreeti of fulladderlogic is

begin

SUM <= ((A XOR B) XOR CIN);
COUT <= (A AND B) OR (B AND CIN) OR (A AND CIN);

end Behavioral_Sampreeti;
```

Structural

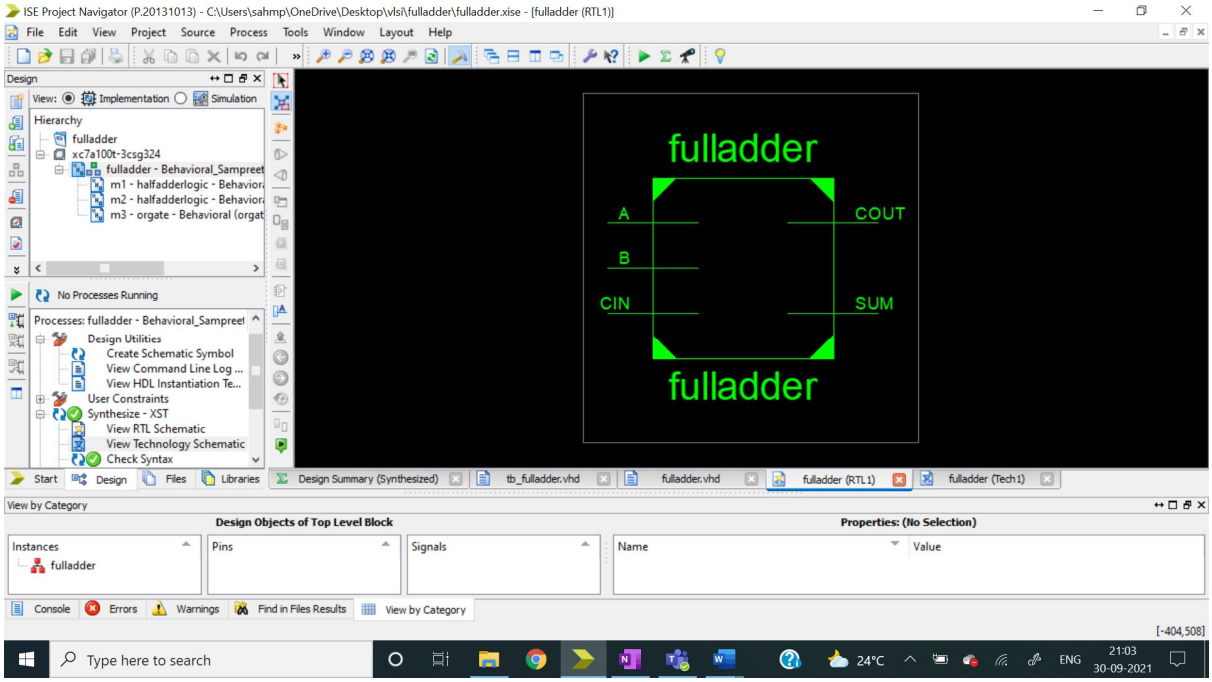
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;



entity halfadderlogic is
   Port ( P : in  STD_LOGIC;
          Q : in  STD_LOGIC;
          S : out  STD_LOGIC;
          C : out  STD_LOGIC);
```

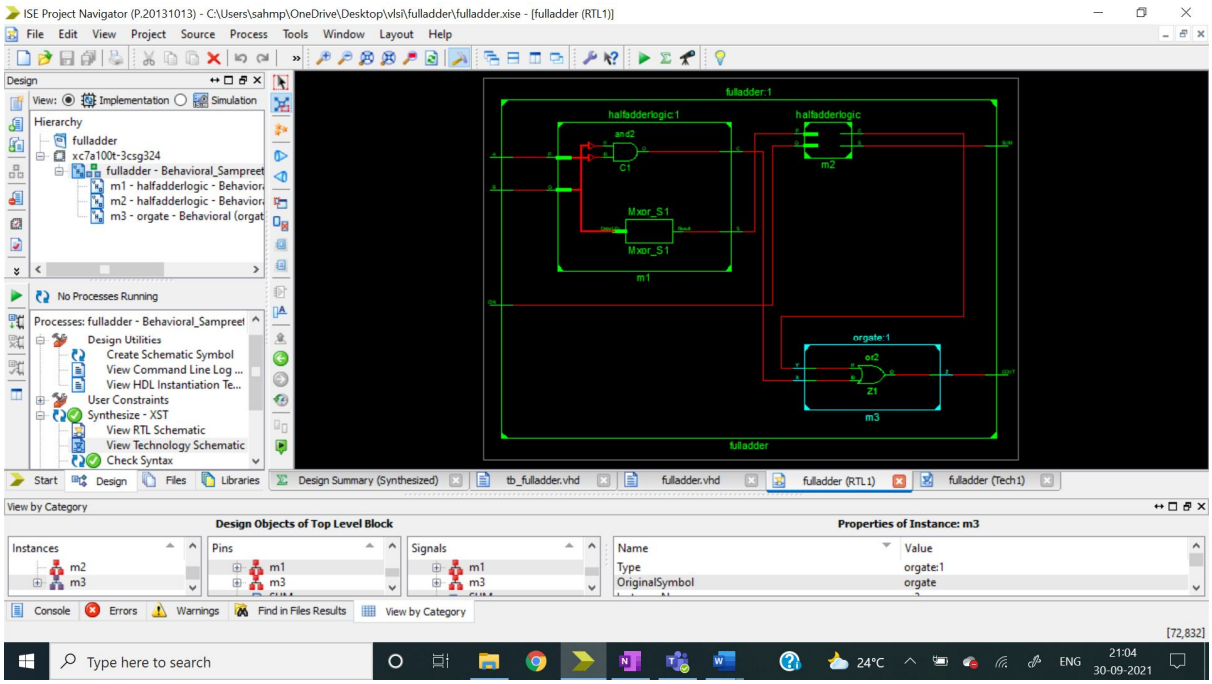
```
end halfadderlogic;

architecture Behavioral_Sampreeti of halfadderlogic is

begin
S <= P xor Q;
C <= P AND Q;

end Behavioral_Sampreeti;




library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity orgate is
   Port ( X : in  STD_LOGIC;
          Y : in  STD_LOGIC;
          Z : out  STD_LOGIC);
end orgate;

architecture Behavioral of orgate is

begin
Z <= X OR Y;

end Behavioral;

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 primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity fulladder is
   Port ( A : in  STD_LOGIC;
          B : in  STD_LOGIC;
          CIN : in  STD_LOGIC;
          SUM : out  STD_LOGIC;
          COUT : out  STD_LOGIC);
end fulladder;

architecture Behavioral_Sampreeti of fulladder is
component halfadderlogic
   Port ( P : in  STD_LOGIC;
          Q : in  STD_LOGIC;
          S : out  STD_LOGIC;
          C : out  STD_LOGIC);
end COMPONENT;

component orgate
   Port ( X : in  STD_LOGIC;
          Y : in  STD_LOGIC;
          Z : out  STD_LOGIC);
end COMPONENT;
```

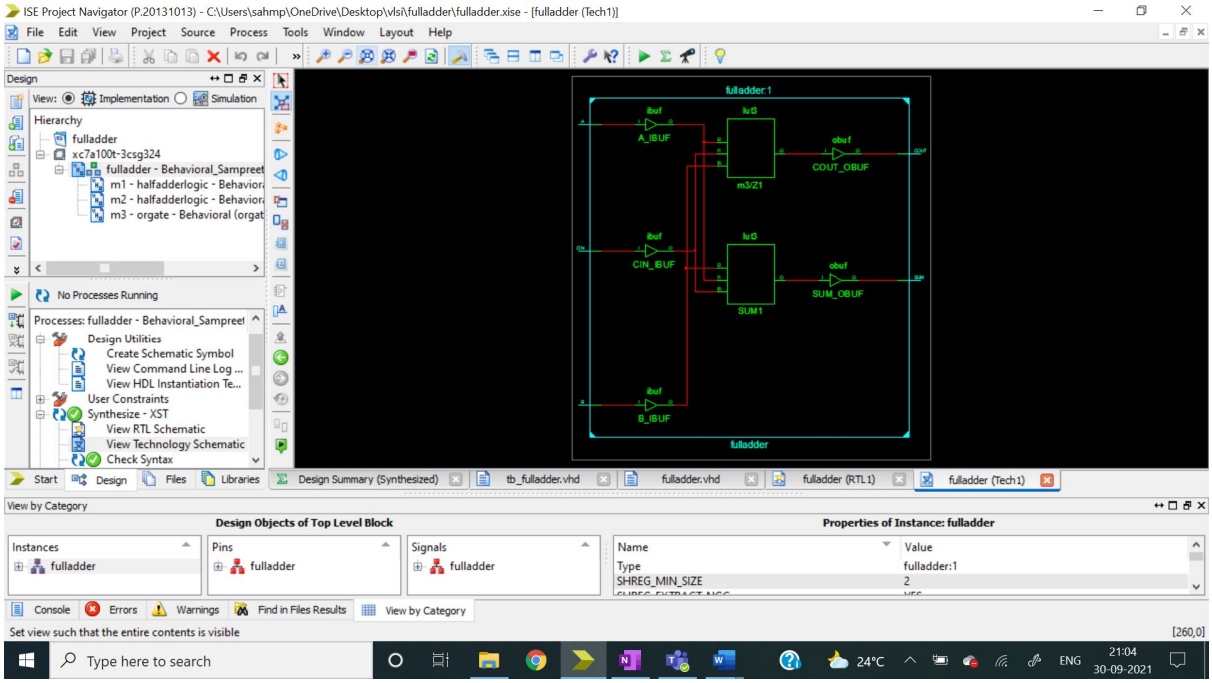
```
signal S1,C1,C2:STD_LOGIC;
begin

m1: halfadderlogic port map(A,B,S1,C1);
m2: halfadderlogic port map(S1,CIN,SUM,C2);
--COUT <= C1 OR C2; mixed
m3: orgate port map(C1,C2,COUT);

end Behavioral_Sampreeti;
```

RTL Schematic:





Technology Schematic:



Synthesis Report

a) Device Utilisation Summary

Top Level Output File Name : fulladder.ngc

Primitive and Black Box Usage:

BELS : 2

```
#      LUT3                  : 2
# IO Buffers                  : 5
#      IBUF                  : 3
#      OBUF                  : 2

Device utilization summary:
---------------------------

Selected Device : 7a100tcsg324-3


Slice Logic Utilization:
 Number of Slice LUTs:                2  out of  63400     0%
   Number used as Logic:              2  out of  63400     0%

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:     2
   Number with an unused Flip Flop:      2  out of     2   100%
   Number with an unused LUT:            0  out of     2     0%
   Number of fully used LUT-FF pairs:    0  out of     2     0%
   Number of unique control sets:        0

IO Utilization:
 Number of IOs:                   5
 Number of bonded IOBs:               5  out of    210     2%

Specific Feature Utilization:
```

b) Timing Report

```
NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:
------------------
No clock signals found in this design

Asynchronous Control Signals Information:
----------------------------------------
No asynchronous control signals found in this design

Timing Summary:
---------------
Speed Grade: -3

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
```

Maximum output required time after clock: No path found
Maximum combinational path delay: 0.893ns

Timing Details:

All values displayed in nanoseconds (ns)

Testbench Program:

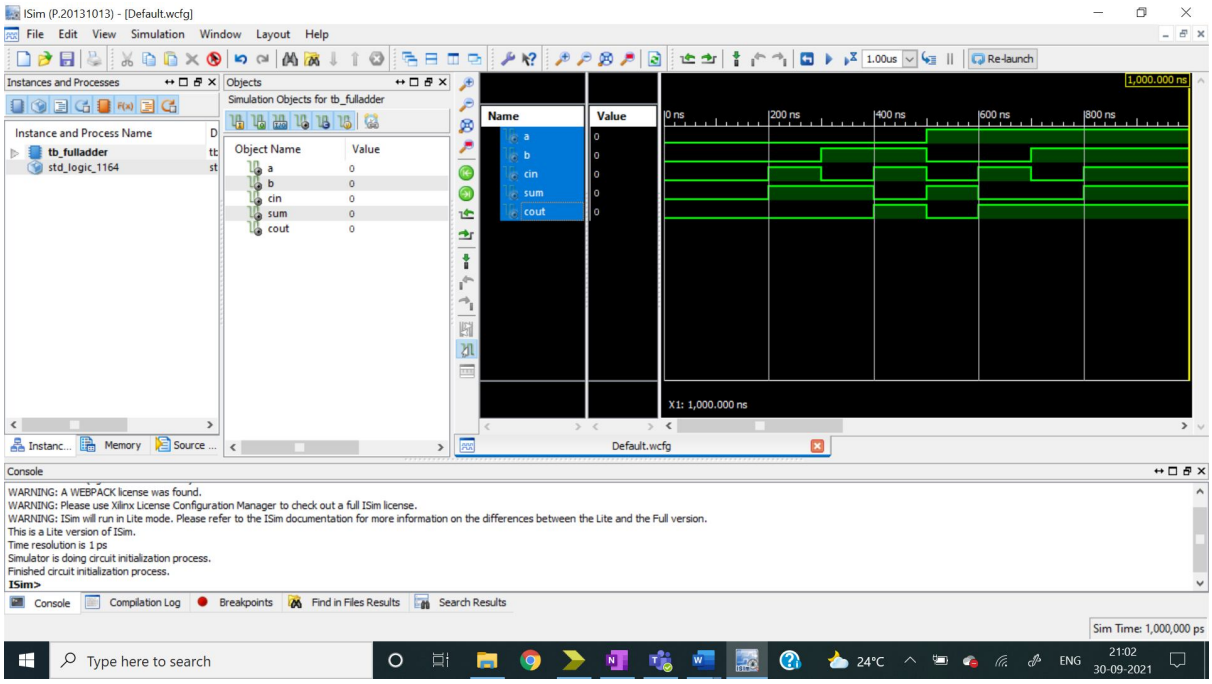
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY tb_fulladder IS
END tb_fulladder;

ARCHITECTURE behavior OF tb_fulladder IS

   -- Component Declaration for the Unit Under Test (UUT)

   COMPONENT fulladder
   PORT(
        A : IN  std_logic;
        B : IN  std_logic;
        CIN : IN  std_logic;
        SUM : OUT  std_logic;
        COUT : OUT  std_logic
       );
   END COMPONENT;


  --Inputs
```

```
  signal A : std_logic := '0';
  signal B : std_logic := '0';
  signal CIN : std_logic := '0';

        --Outputs
  signal SUM : std_logic;
  signal COUT : std_logic;
  -- No clocks detected in port list. Replace <clock> below with
  -- appropriate port name

  --constant <clock>_period : time := 10 ns;

BEGIN

        -- Instantiate the Unit Under Test (UUT)
  uut: fulladder PORT MAP (
       A => A,
       B => B,
       CIN => CIN,
       SUM => SUM,
       COUT => COUT
      );

  -- Clock process definitions
--   <clock>_process :process
--   begin
--                <clock> <= '0';
--                wait for <clock>_period/2;
--                <clock> <= '1';
--                wait for <clock>_period/2;
--   end process;
```

```
-- Stimulus process
stim_proc: process
begin
  -- hold reset state for 100 ns.
  wait for 100 ns;

  --wait for <clock>_period*10;

  -- insert stimulus here
                A <='0';
        B <='0';
        CIN <='0';
         wait for 100 ns;
                A <='0';
        B <='0';
        CIN <='1';
         wait for 100 ns;
                A <='0';
        B <='1';
        CIN <='0';
         wait for 100 ns;
                A <='0';
        B <='1';
        CIN <='1';
         wait for 100 ns;
                A <='1';
        B <='0';
        CIN <='0';
         wait for 100 ns;
```

```
            A <='1';
        B <='0';
        CIN <='1';
         wait for 100 ns;

            A <='1';
        B <='1';
        CIN <='0';
         wait for 100 ns;
            A <='1';
        B <='1';
        CIN <='1';
         wait for 100 ns;


  end process;

END;
```

Isim Waveforms:



Conclusion:

Thus we have :

1) Modeled Full Adder using Dataflow,Behavioral,Structural and Mixed Modeling Styles.

2) Observed following Schematics : RTL & Technology Schematics generated Post-Synthesis.

3) Interpreted Device Utilisation Summary in terms of LUTs , SLICES , IOBs , Multiplexers & D FFs used out of the available device resources.

4) Interpreted the TIMING Report in terms of Maximum combinational delay as indicative of the Maximum Operating Frequency .

5) Written a TESTBENCH to verify the functionality of Full Adder & verified the functionality as per the TRUTH-TABLE ,by observing ISIM Waveforms.

6) Prototyped the FPGA XC3S250EPQ208-5 to realize Full Adder & verified its operation by giving suitable input combinations