Anish Ghosh

Lab Report for Lab 1 : Intro to the Simulation Tool of Xilinx Software

Digital Design

Instructor: Dr.Fanging Hu

2021

Abstract

Tasks performed in Lab 1

- Implementation of AND gate
- Implementation of 8-bit adder
- Finding the bug iSIm_intro3.vhd and fixing it to implement XOR gate

Xilinx ISE is being used to simulate the hardware designs and the main simulation tool used is iSim.All the vhdl codes were provided.No important circuits to be displayed because all of them are very fundamental.

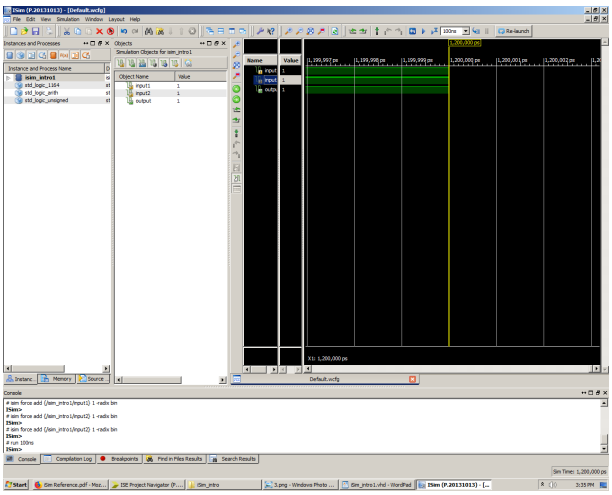
1 Implementation of AND gate

Code given :

```
    library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.STD_LOGIC_ARITH.all;
use IEEE.STD_LOGIC_UNSIGNED.all;

entity iSim_intro1 is

  port (
    input1    : in  std_logic;
    input2 : in  std_logic;
    output : out std_logic
    );

end iSim_intro1;

architecture simple of iSim_intro1 is

begin  -- simple

  output <= input1 and input2;

end simple;
```

The code consists of two major units: entity declaration and architecture body.The entity declaration is specifying the inputs: input1 , input2 and output.The architecture body is specifying the internal organization of the gate.(which is very simple because the 'and' operation is directly used rather than getting a DNF form of the truth table and then implementing it)

Output :



The graph shows the timing diagram of the AND gate where the inputs are 1 and 1 so the output is also 1.

2 Implementation of 8-bit adder

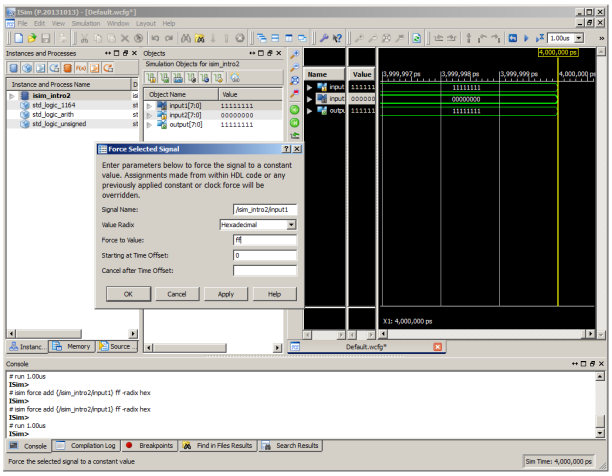
Code Given :

```
    library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.STD_LOGIC_ARITH.all;
use IEEE.STD_LOGIC_UNSIGNED.all;

entity iSim_intro2 is

  port (
    input1 : in  std_logic_vector(7 downto 0);
    input2 : in  std_logic_vector (7 downto 0);
    output : out std_logic_vector (7 downto 0)
    );

end iSim_intro2;

architecture simple of iSim_intro2 is

begin  -- simple

  output <= input1 + input2;

end simple;

--unsigned("0000"&input1)
```

The 8 bit adder adds two vectors input1 and input2 according to the code in the architecture.It takes two input vectors and gives the output as the timing

diagram in the simulation tool.According to the given code,the hexadecimal and decimal inputs are converted to binary and they are added to obtain the 8-bit output.Since 1+1 = 10 we take the 0 as the output forward and carry forward 1 to the next operation.The output shows the results when we add FF to 00000000.

Output :



3 Implementation of XOR gate

Code Given(with bug) :

```
    library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.STD_LOGIC_ARITH.all;
use IEEE.STD_LOGIC_UNSIGNED.all;

entity iSim_intro3 is

  port (
    input1    : in  std_logic;
    input2 : in  std_logic;
    output : out std_logic
    );

end iSim_intro3;

architecture simple of iSim_intro3 is

  signal internal_sig1 : std_logic;
  signal internal_sig2 : std_logic;

begin  -- simple
```

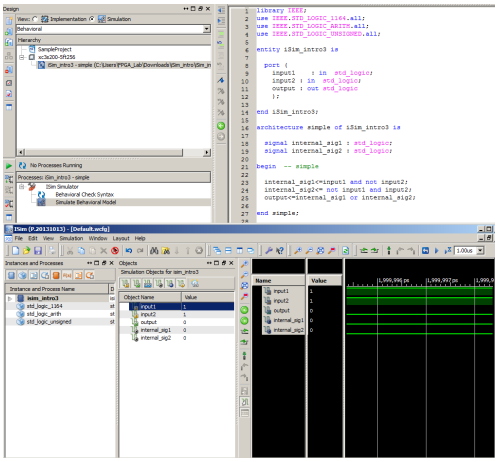
```
  internal_sig1<=input1 and not input2;
  internal_sig2<=input1 and input2;
  output<=internal_sig1 or internal_sig2;

end simple;
```

The code given above has a bug which is not syntactical but logical. The DNF of the XOR gate is $(A \oplus B) = \bar{A} \cdot B + A \cdot \bar{B}$ but in the given code the expression implemented is $(A \oplus B) = A \cdot B + A \cdot \bar{B}$ which is wrong.So the corrected version of the code is given below.
The results for the timing diagrams of the XOR gate is correct after rectifying the code.

Code (corrected) :

```
    library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.STD_LOGIC_ARITH.all;
use IEEE.STD_LOGIC_UNSIGNED.all;

entity iSim_intro3 is

  port (
    input1    : in  std_logic;
    input2 : in  std_logic;
    output : out std_logic
    );

end iSim_intro3;

architecture simple of iSim_intro3 is

  signal internal_sig1 : std_logic;
  signal internal_sig2 : std_logic;

begin  -- simple

  internal_sig1<=input1 and not input2;
  internal_sig2<= not input1 and input2;
  output<=internal_sig1 or internal_sig2;

end simple;
```

Output(Simulation) :



4 Conclusion :

We used Xillinx to simulate some fundamental gates and a 8-bit adder.We learned how to implement basic VHDL code using Xillinx to see the timing diagrams of the gates and the adder.

5 References

1. $http://fpga-fhu.user.jacobs-university.de/?page_id=402$
2. $https://onlinelibrary.wiley.com/doi/pdf/10.1002/0471786411.ch2$
3. $https://www.electronics-tutorials.ws/logic/logic_7.html$
4. $https://web.itu.edu.tr/\ ateserd/VHDL.pdf$