JACOBS
UNIVERSITY

Jacobs University Bremen

CA-ECE-803: Digital design

Title: Intro to the Simulation Tool of Xilinx Software

Lab report: 1

Experiment conducted by:
Betelhem Nebebe
Maria Namachanja
Neeha Hammad

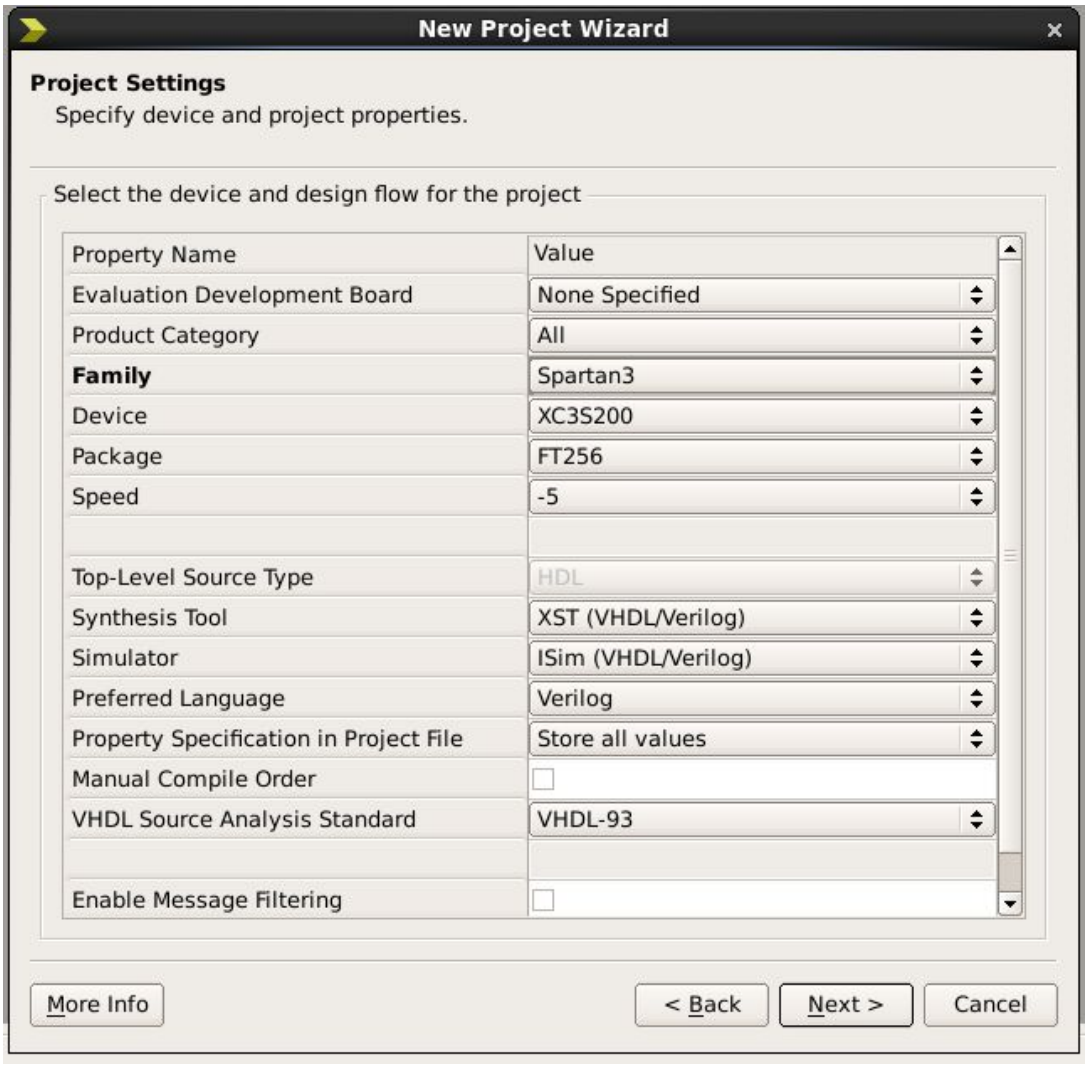
Instructor: Prof. Dr. Fangning Hu
Execution date: February 15, 2021

1. Introduction

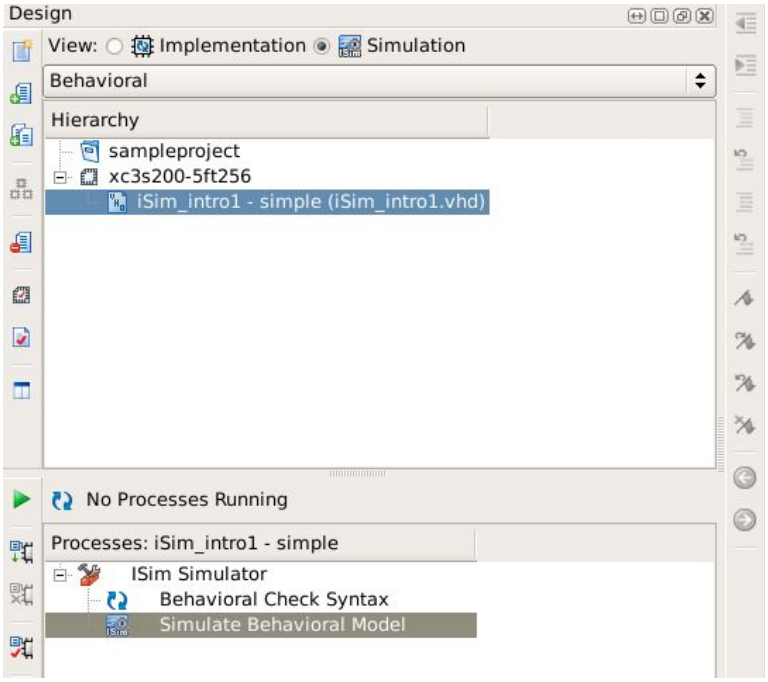
The objective of this lab was to become acquainted with the main simulation tool iSIM in the Xilinx software. The steps followed to create a project in the Project Navigator, add sources and carry out simulations were studied.

2. Execution

After launching the Project Navigation app, the following specifications were set when creating a project to let it determine the resources available and how the output pins should be mapped.



The next step was to add a source to the project and simulate it as shown below



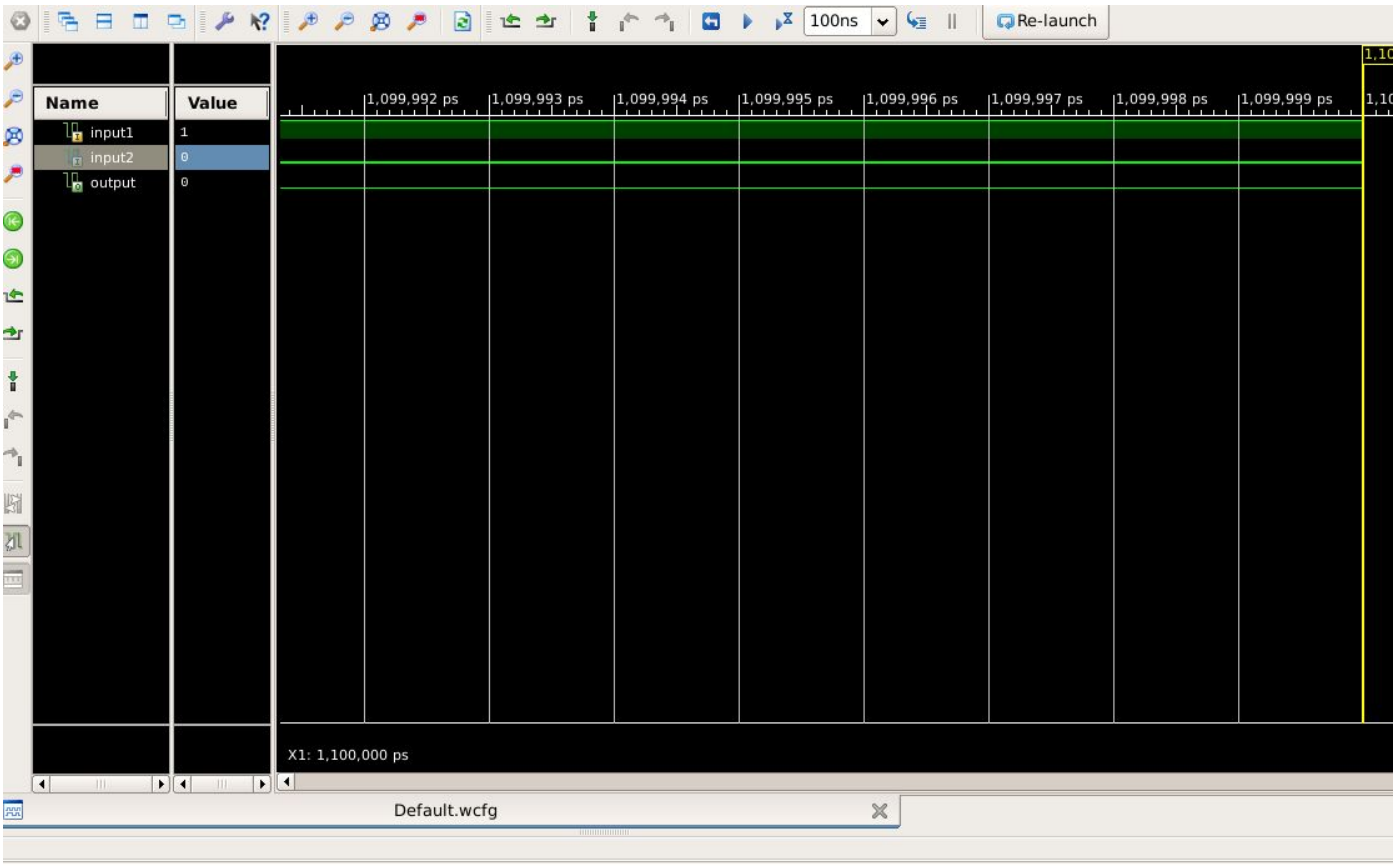
Which will be the same throughout the exercises.

iSim_intro1 Exercise:

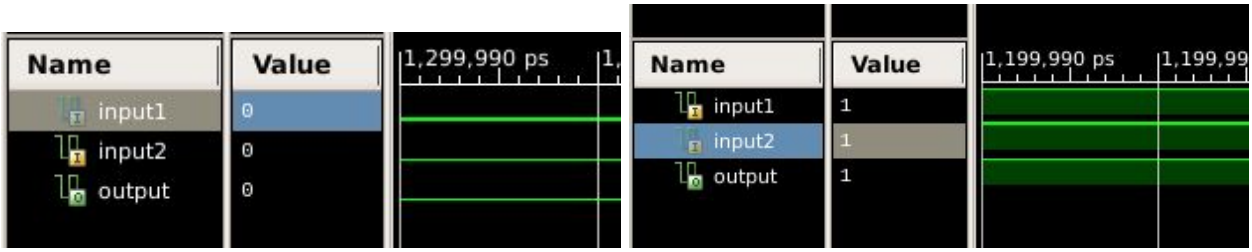
This exercise was meant to demonstrate the implementation of an 'and' gate.

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.all;
3  use IEEE.STD_LOGIC_ARITH.all;
4  use IEEE.STD_LOGIC_UNSIGNED.all;
5
6  entity iSim_intro1 is
7
8    port (
9      input1    : in  std_logic;
10     input2 : in  std_logic;
11     output : out std_logic
12     );
13
14 end iSim_intro1;
15
16 architecture simple of iSim_intro1 is
17
18 begin  -- simple
19
20   output <= input1 and input2;
21
22 end simple;
23
```

The signals of input1&2 were forced to binary values 1 and 0 respectively and the time to run set to 100ns to obtain the waveforms for input and output.



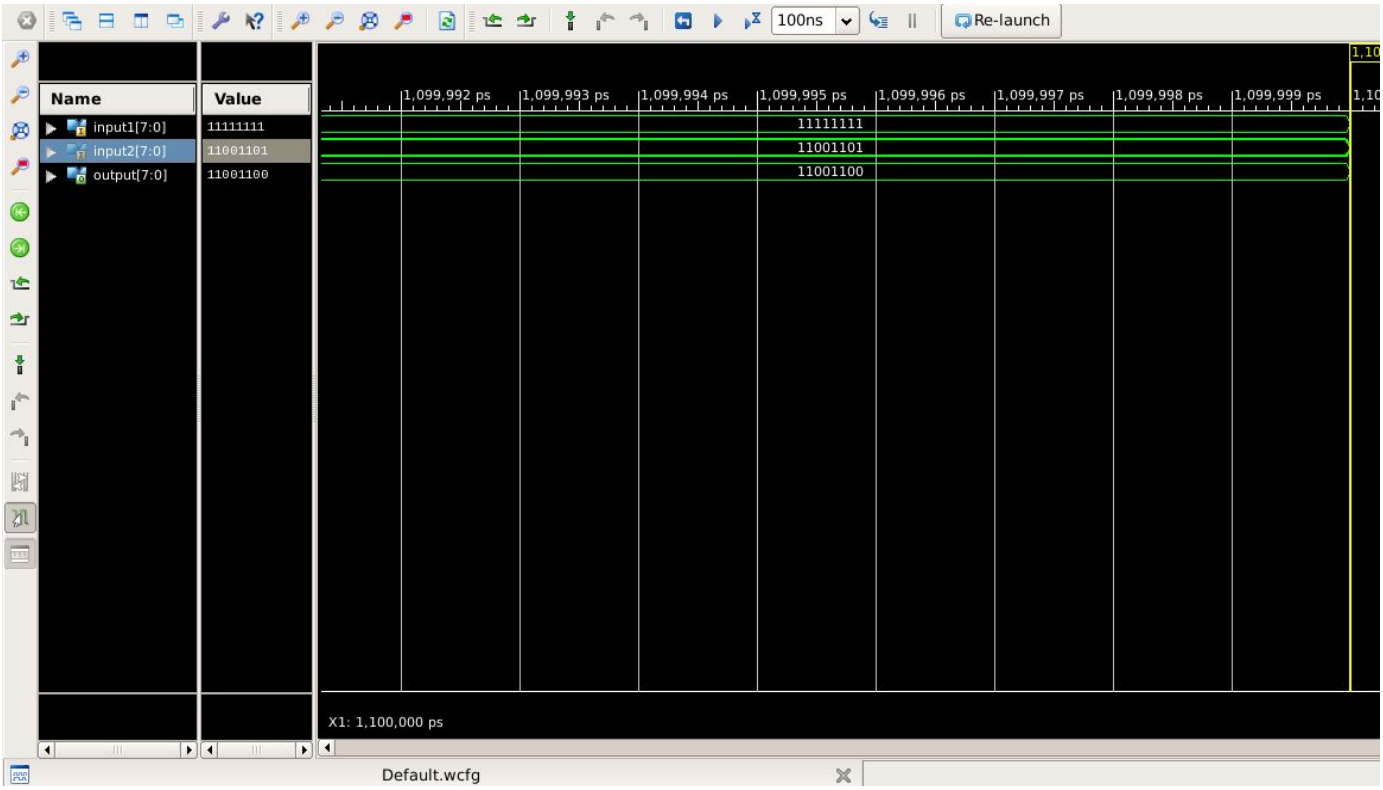
This is indeed an 'and' gate. Different combinations of values were tried and it was seen to work as expected.



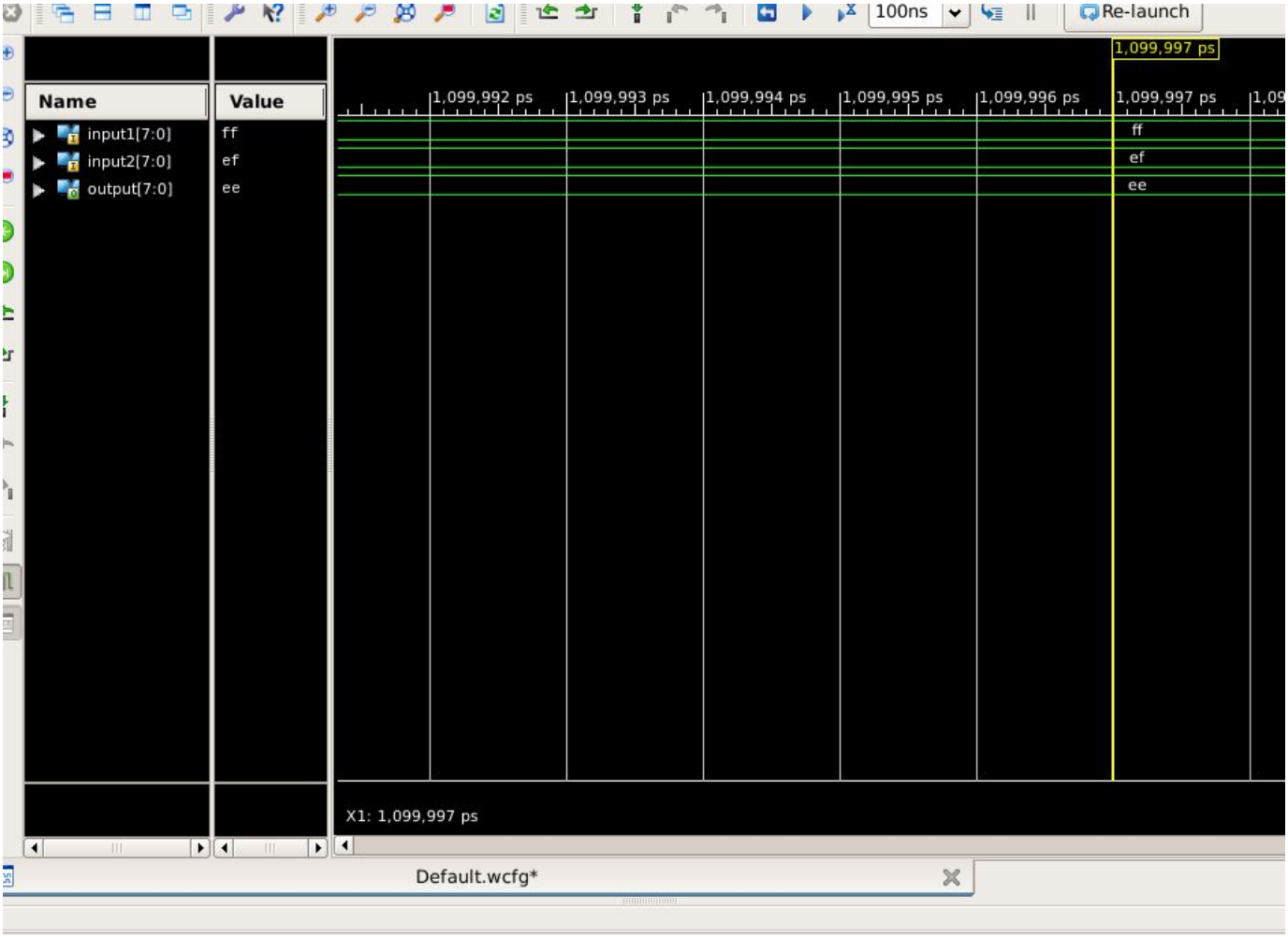
iSim_intro2 Exercise:

Here, an adder for an array of bits was implemented. Instead of using array of bits, we can also use Hexadecimal format or Decimal format.

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.all;
3  use IEEE.STD_LOGIC_ARITH.all;
4  use IEEE.STD_LOGIC_UNSIGNED.all;
5
6  entity iSim_intro2 is
7
8    port (
9      input1 : in  std_logic_vector(7 downto 0);
10     input2 : in  std_logic_vector (7 downto 0);
11     output : out std_logic_vector (7 downto 0)
12     );
13
14 end iSim_intro2;
15
16 architecture simple of iSim_intro2 is
17
18 begin  -- simple
19
20   output <= input1 + input2;
21
22 end simple;
```



Experimenting with a different radix for the waveforms

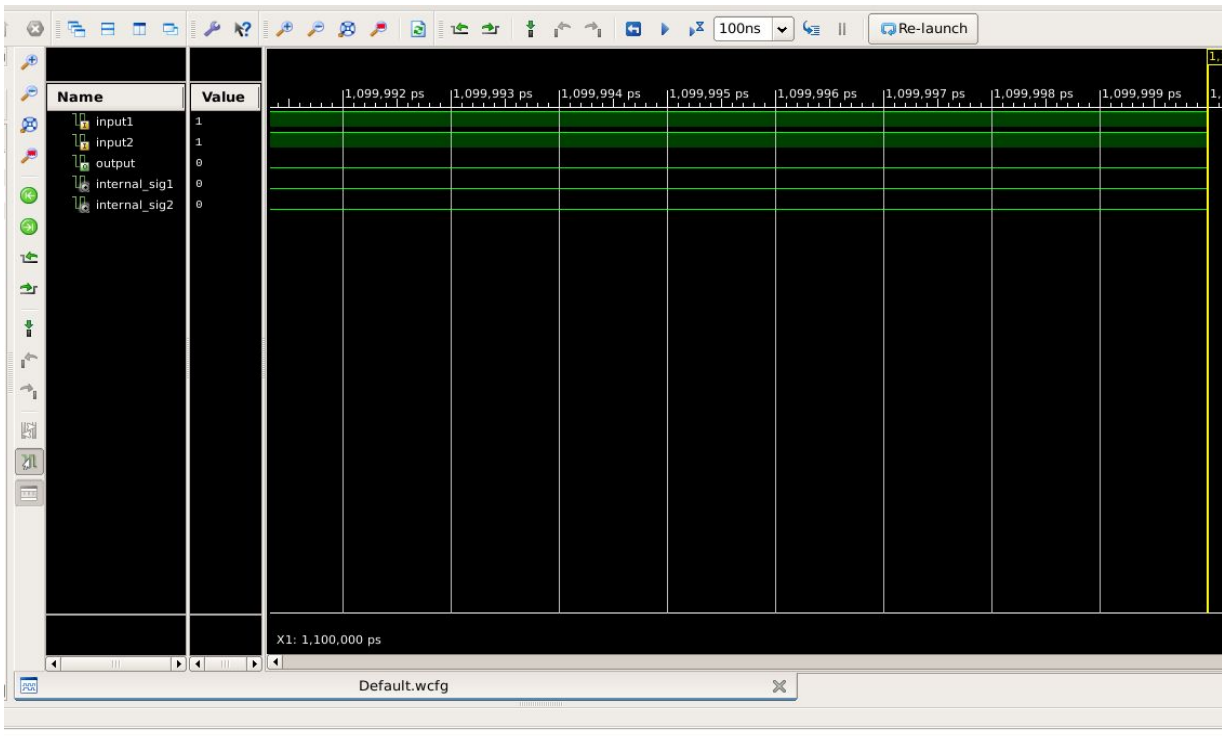


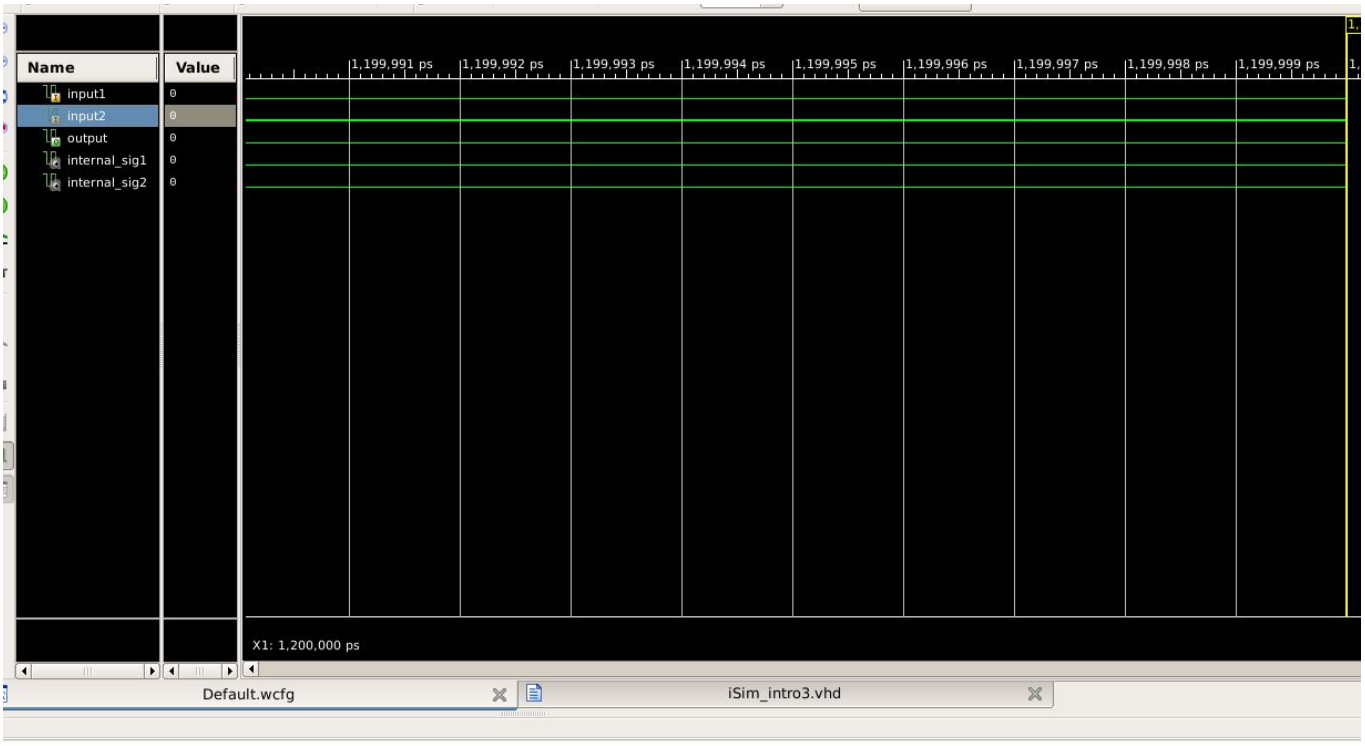
iSim_intro3 Exercise:

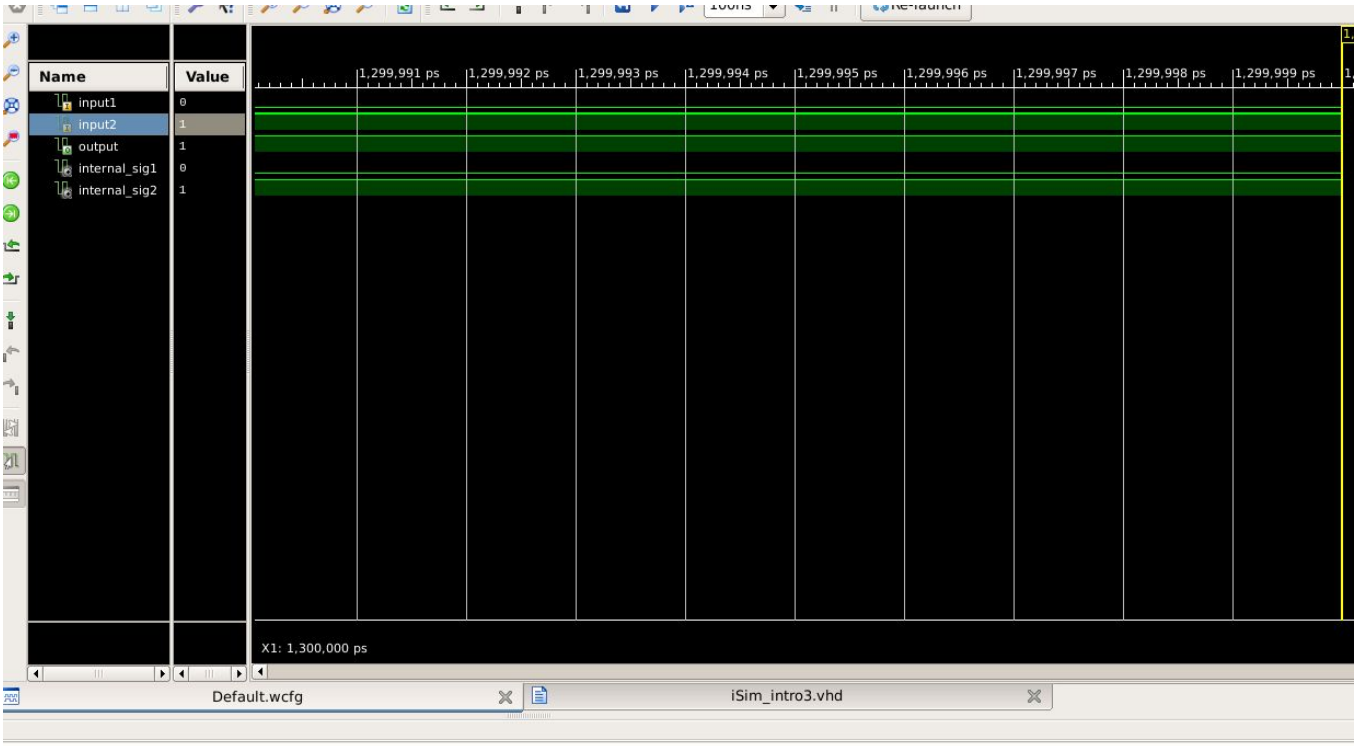
In this exercise, we implemented an "xor" gate using the formula (input1&(!input2)) | (!input1&input2).
In the given code, there was a bug because "not" was missing before input1. This correction is highlighted in the given code below. By setting input1 and input2 to different values, we were able to see that our output is correct.

input1	input2	!input1	!input2	(input1&(!input2))	(!input1&input2)	(input1&(!input2)) (!input1&input2)
0	0	1	1	0	0	0
1	0	0	1	1	0	1
0	1	1	0	0	1	1
1	1	0	0	0	0	0

```vhdl
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;
```




Name
Value
input1
input2
output
internal_sig1
internal_sig2
0
0
0
0
0
1,199,991 ps
1,199,992 ps
1,199,993 ps
1,199,994 ps
1,199,995 ps
1,199,996 ps
1,199,997 ps
1,199,998 ps
1,199,999 ps
X1: 1,200,000 ps
Default.wcfg
iSim_intro3.vhd


Name
Value
input1
input2
output
internal_sig1
internal_sig2
0
1
1
0
1
1,299,991 ps
1,299,992 ps
1,299,993 ps
1,299,994 ps
1,299,995 ps
1,299,996 ps
1,299,997 ps
1,299,998 ps
1,299,999 ps
X1: 1,300,000 ps
Default.wcfg
iSim_intro3.vhd

3. Conclusion

In this lab, we learnt how to work with the main simulation tool iSim. We implemented an "and" gate, an "adder" and "xor" gate. We were about to run these implementations by using different input values and get the desired outcome.

References

http://fpga-fhu.user.jacobs-university.de/?page_id=402