Research 1-71 EE Laboratory

CO-521-B Digital Signal Processing Lab

Spring 2023

Friday, 10^{th} of April 2023

LAB 5: Digital_Down_Converter

Report by: Prince Lee Muhera

1) Introduction

In this lab, we use a simulation tool developed by Xilinx to simulate our hardware design. To begin, we quickly acquaint ourselves with the main simulation tool iSim. The objective is to introduce ourselves with Xilinx and understand how it works.

2) Circuit Design

There are no important circuit designs to be presented for this lab.

3) VHD files used

We used the following VHD files in this lab:

1. iSim_intro1.vhd
2. iSim_intro2.vhd
3. iSim_intro3.vhd

These files were provided to us in the lab. Each of the files are coded with different implementations of logic gates in VHD. The iSim_intro1.vhd is an implementation of an "AND" gate. iSim_intro2.vhd is the implementation of an "Adder". And iSim_intro3.vhd is the implementation of an "XOR" gate using the formula (input1&(!input2))|((!input1)&input2).

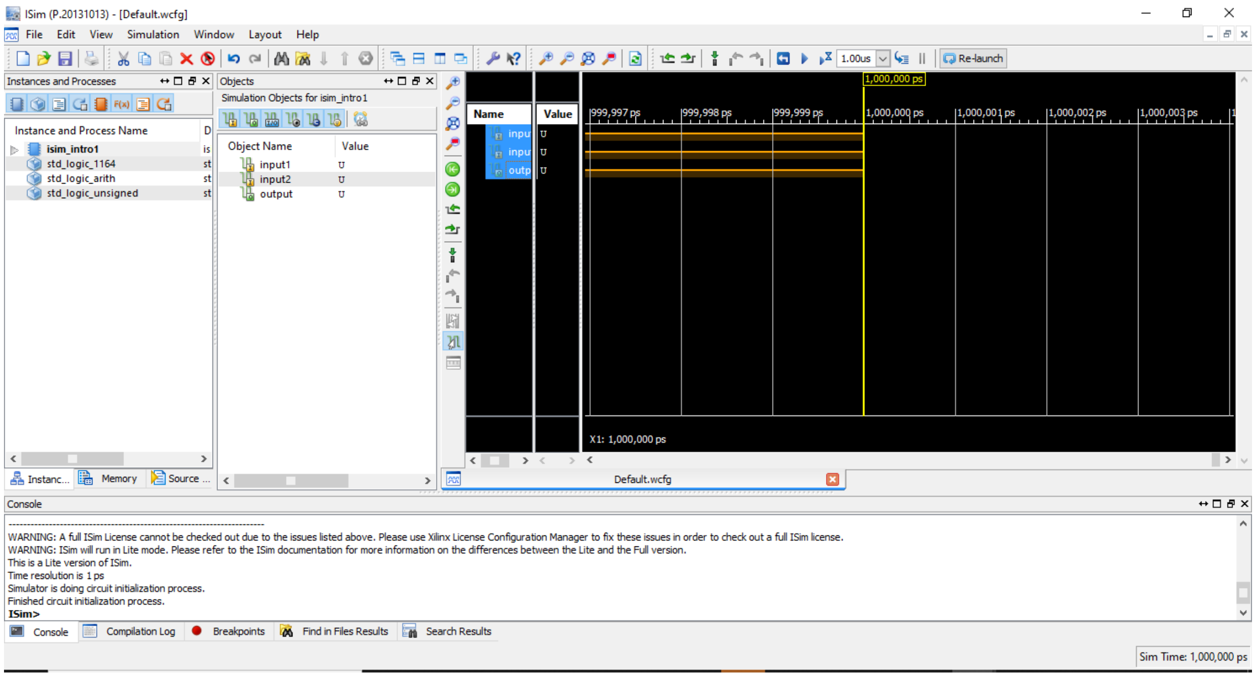
A summary can be found in the table below:

Nr.	Name	Description
1	iSim_intro1.vhd	AND gate
2	iSim_intro2.vhd	8 Bit Adder
3	iSim_intro3.vhd	XOR gate

4) Exercises

We begin this exercise by learning how to use Xilinx, and how to implement an AND gate. We do this by first opening the Xilinx ISE Project Navigator, and opening a new project. Then we add the source files to the Project Navigator, which are given to us.

4.1) We select the iSim_intro1.vhd file, select the simulation option, right click on the “Simulate Behavioral Model” tag in the Processes window, and click run. The following window appears:


ISim (P.20131013) - [Default.wcfg]

Instance and Process Name	D
isim_intro1	is
std_logic_1164	st
std_logic_arith	st
std_logic_unsigned	st

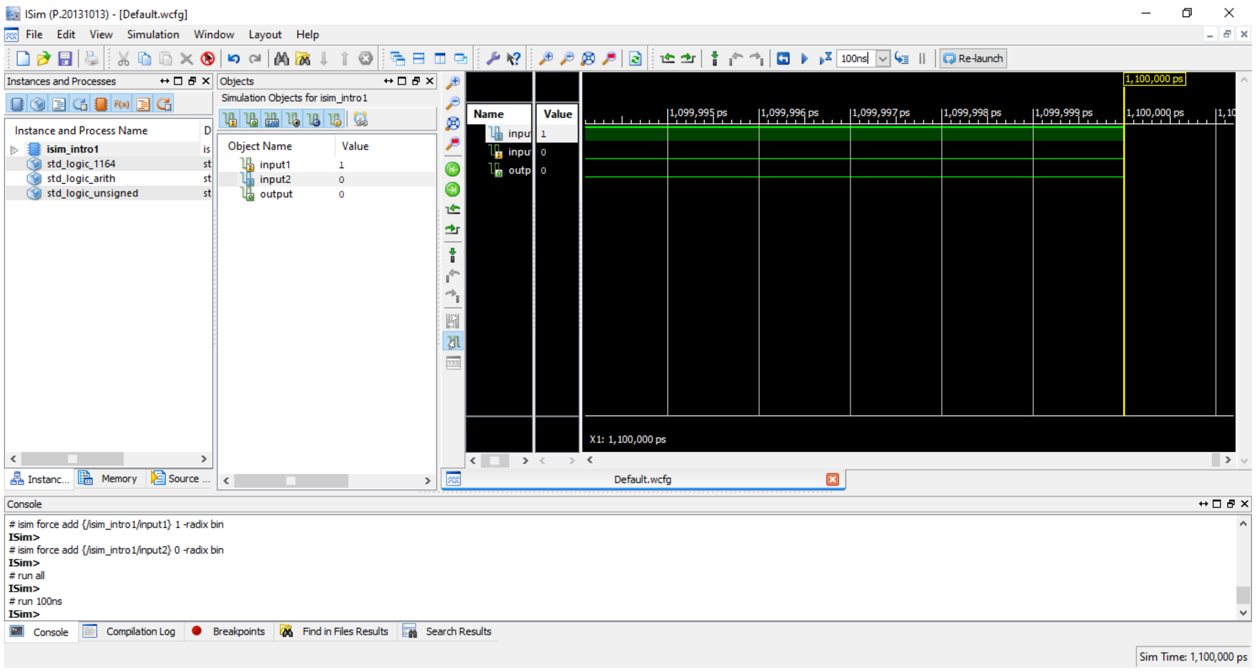
Simulation Objects for isim_intro1

Object Name	Value
input1	U
input2	U
output	U

```
----------------------------------------------------------------------
WARNING: A full ISim License cannot be checked out due to the issues listed above. Please use Xilinx License Configuration Manager to fix these issues in order to check out a full ISim license.
WARNING: ISim will run in Lite mode. Please refer to the ISim documentation for more information on the differences between the Lite and the Full version.
This is a Lite version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim>
```

Sim Time: 1,000,000 ps

As per the problem statement, we click on input1 and change force constant to 1. We repeat the same for input2 and change force constant to 0. Then, we change the time to 100ns and press “run”. We obtain the following results:


ISim (P.20131013) - [Default.wcfg]

Instance and Process Name	D
isim_intro1	is
std_logic_1164	st
std_logic_arith	st
std_logic_unsigned	st

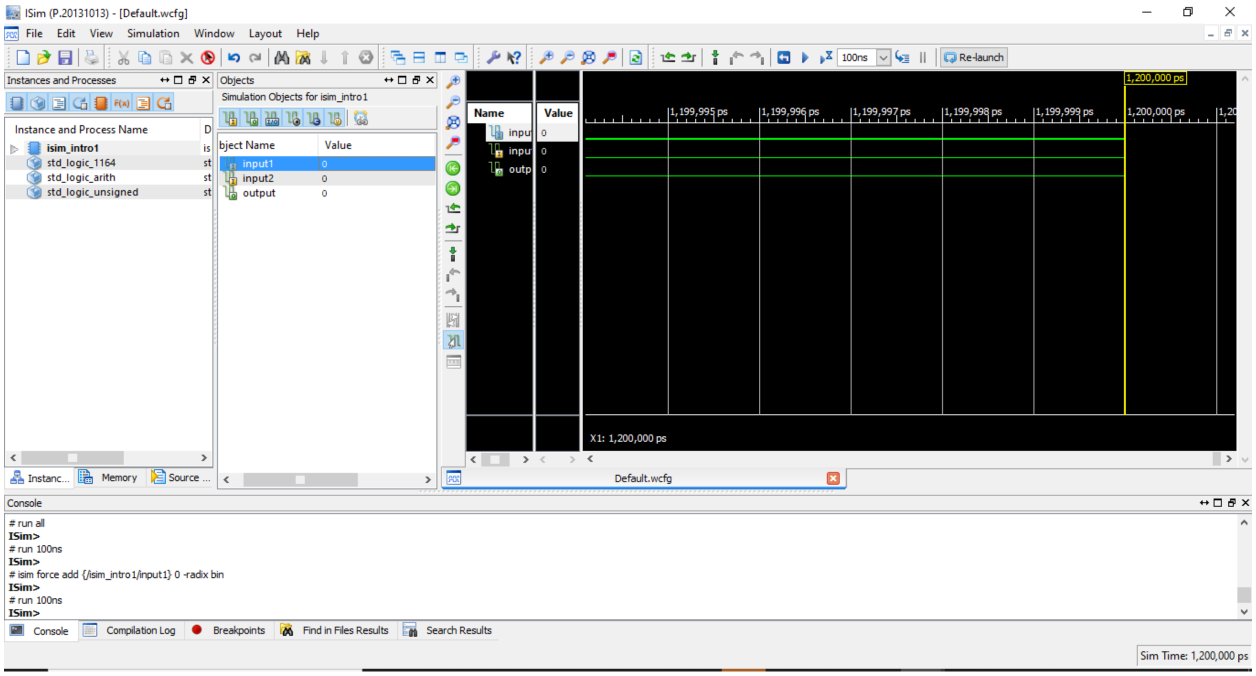
Simulation Objects for isim_intro1

Object Name	Value
input1	1
input2	0
output	0

```
# isim force add {/isim_intro1/input1} 1 -radix bin
ISim>
# isim force add {/isim_intro1/input2} 0 -radix bin
ISim>
# run all
ISim>
# run 100ns
ISim>
```

Sim Time: 1,100,000 ps

We can see that the output to the inputs 1 and 0 result to 0. Test runs with inputs 0 and 0 provide the following:



ISim (P.20131013) - [Default.wcfg]

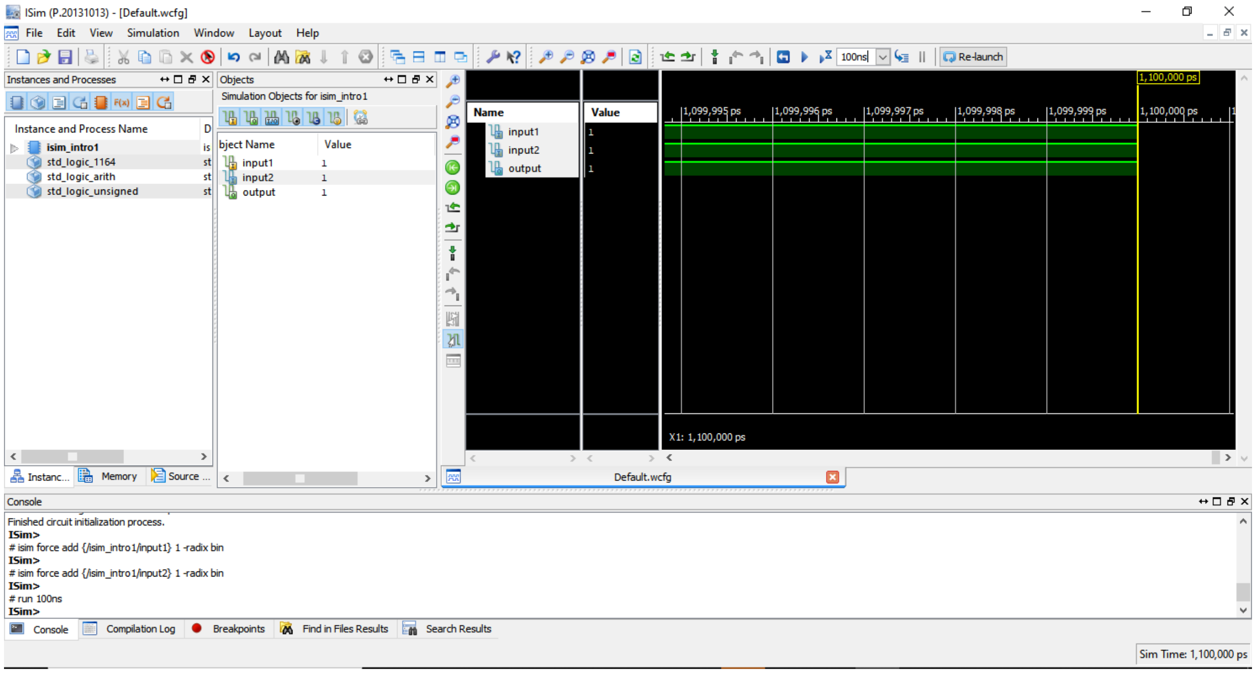
Object Name	Value
input1	0
input2	0
output	0

X1: 1,200,000 ps

```
# run all
ISim>
# run 100ns
ISim>
# isim force add {/isim_intro1/input1} 0 -radix bin
ISim>
# run 100ns
ISim>
```

Sim Time: 1,200,000 ps

Testing with inputs 1 and 1 provide the following results:



ISim (P.20131013) - [Default.wcfg]

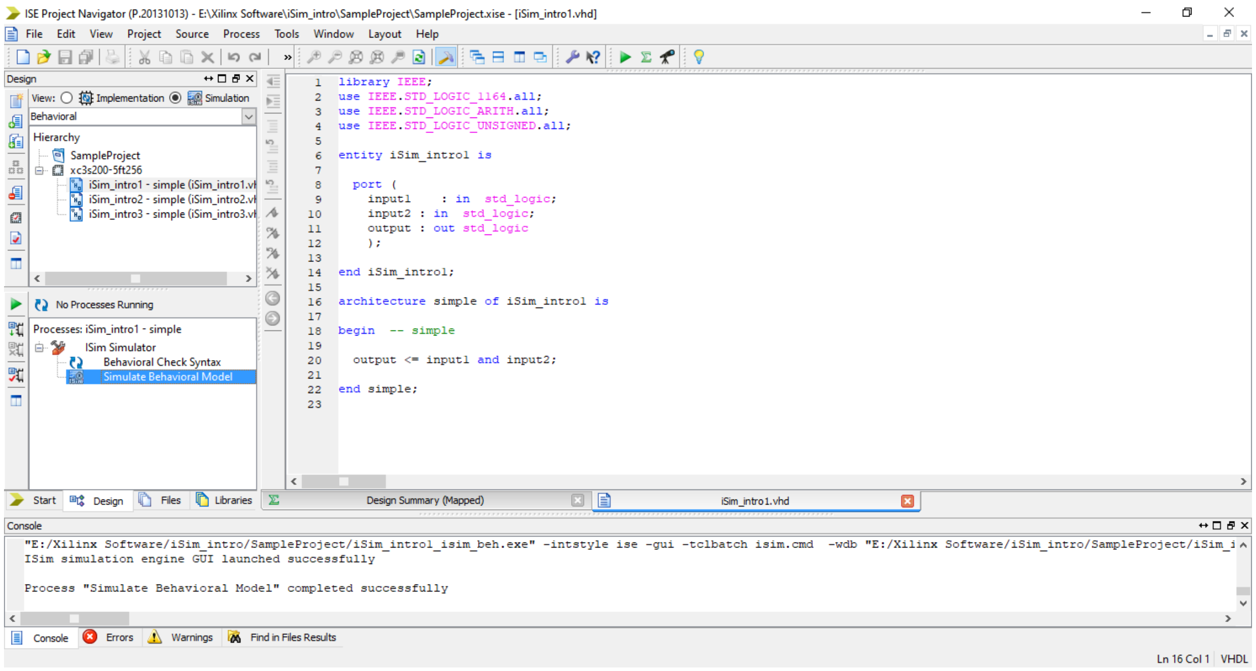
Object Name	Value
input1	1
input2	1
output	1

X1: 1,100,000 ps

```
Finished circuit initialization process.
ISim>
# isim force add {/isim_intro1/input1} 1 -radix bin
ISim>
# isim force add {/isim_intro1/input2} 1 -radix bin
ISim>
# run 100ns
ISim>
```

Sim Time: 1,100,000 ps

These results originate from the following code:


ISE Project Navigator (P.20131013) - E:\Xilinx Software\iSim_intro\SampleProject\SampleProject.xise - [iSim_intro1.vhd]
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Behavioral
Hierarchy
SampleProject
xc3s200-5ft256
iSim_intro1 - simple (iSim_intro1.vhd)
iSim_intro2 - simple (iSim_intro2.vhd)
iSim_intro3 - simple (iSim_intro3.vhd)
No Processes Running
Processes: iSim_intro1 - simple
ISim Simulator
Behavioral Check Syntax
Simulate Behavioral Model
Start Design Files Libraries
Design Summary (Mapped)
iSim_intro1.vhd
Console
Console Errors Warnings Find in Files Results
Ln 16 Col 1 VHDL

```vhdl
 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
```

```
"E:/Xilinx Software/iSim_intro/SampleProject/iSim_intro1_isim_beh.exe" -intstyle ise -gui -tclbatch isim.cmd  -wdb "E:/Xilinx Software/iSim_intro/SampleProject/iSim_i
ISim simulation engine GUI launched successfully

Process "Simulate Behavioral Model" completed successfully
```

We see the following line in line 20:

“output <= input1 and input2;”

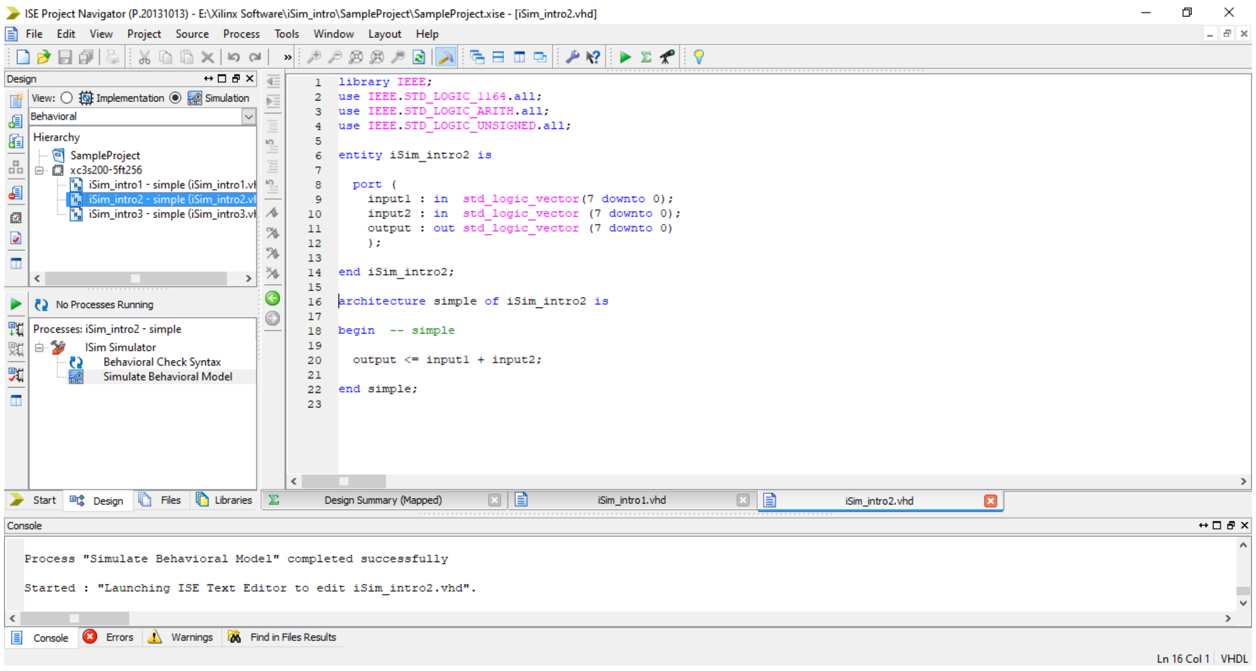
Hence, we can see from the code that this is the implementation of an “AND” gate, which explains the results we see in the simulation:

1 AND 0 => 0

0 AND 0 => 0

1 AND 1 => 1

4.2) Now we move on to the file iSim_intro2.vhd. We are given the following code:



ISE Project Navigator (P.20131013) - E:\Xilinx Software\iSim_intro\SampleProject\SampleProject.xise - [iSim_intro2.vhd]

```
 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;
23
```

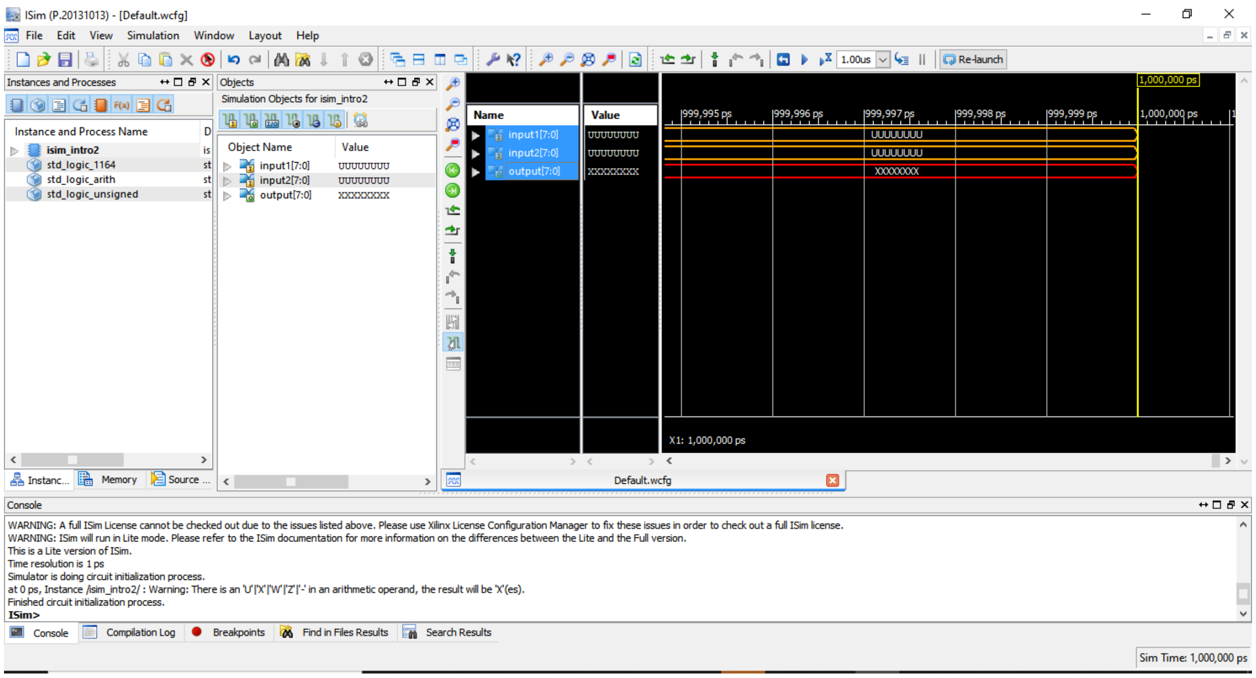
Console

```
Process "Simulate Behavioral Model" completed successfully

Started : "Launching ISE Text Editor to edit iSim_intro2.vhd".
```

In line 20, we see the following line of code: “output <= input1 + input2”. Therefore, we can see that we are dealing with an “Adder”.

We follow the same steps as before and run “Simulate Behavioral Model”, and obtain the following window initially:



ISim (P.20131013) - [Default.wcfg]

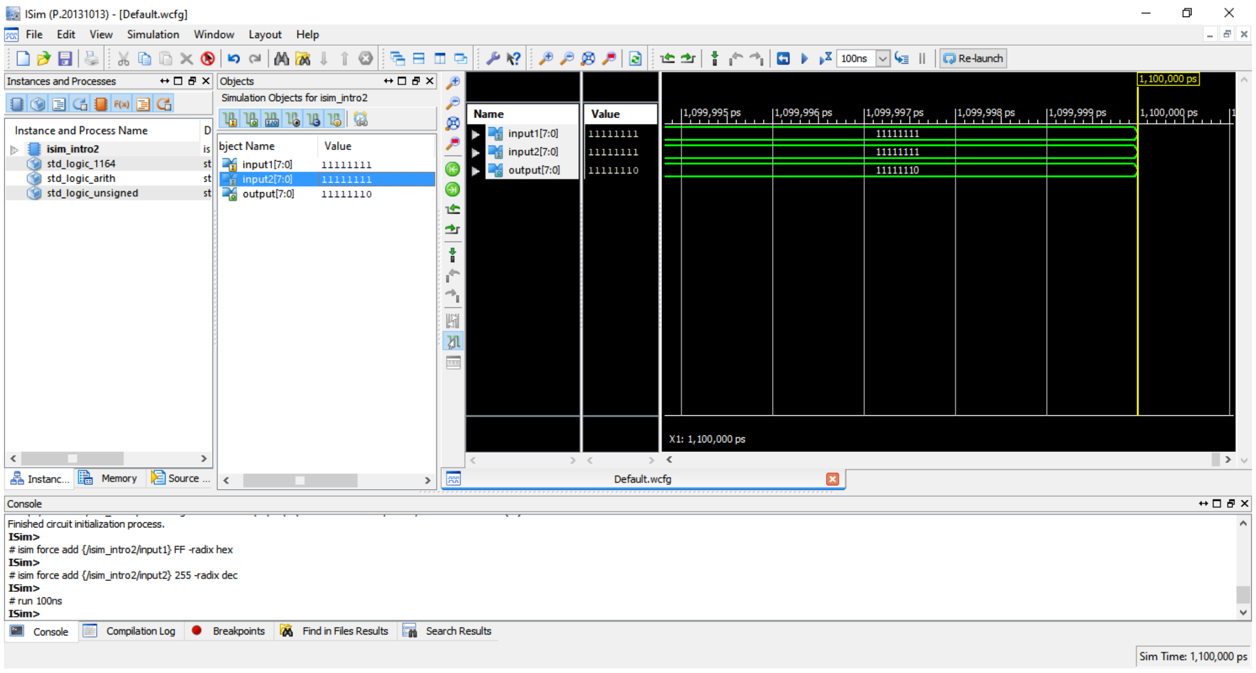
Name	Value
input1[7:0]	UUUUUUUU
input2[7:0]	UUUUUUUU
output[7:0]	XXXXXXXX

Console

```
WARNING: A full ISim License cannot be checked out due to the issues listed above. Please use Xilinx License Configuration Manager to fix these issues in order to check out a full ISim license.
WARNING: ISim will run in Lite mode. Please refer to the ISim documentation for more information on the differences between the Lite and the Full version.
This is a Lite version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
at 0 ps, Instance /isim_intro2/ : Warning: There is an 'U'|'X'|'W'|'Z'|'-' in an arithmetic operand, the result will be 'X'(es).
Finished circuit initialization process.
ISim>
```

Sim Time: 1,000,000 ps

As per the problem statement, “FF” is entered into Force constant dialog box of input1 (value radix set to hexadecimal) and “255” is entered in Force dialog box of input2 (value radix set to signed decimal). Time is set to 100ns and the code is run. We obtain the following results:


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Objects
Simulation Objects for isim_intro2
Default.wcfg
Console Compilation Log Breakpoints Find in Files Results Search Results

```
Instance and Process Name
isim_intro2
std_logic_1164
std_logic_arith
std_logic_unsigned
```

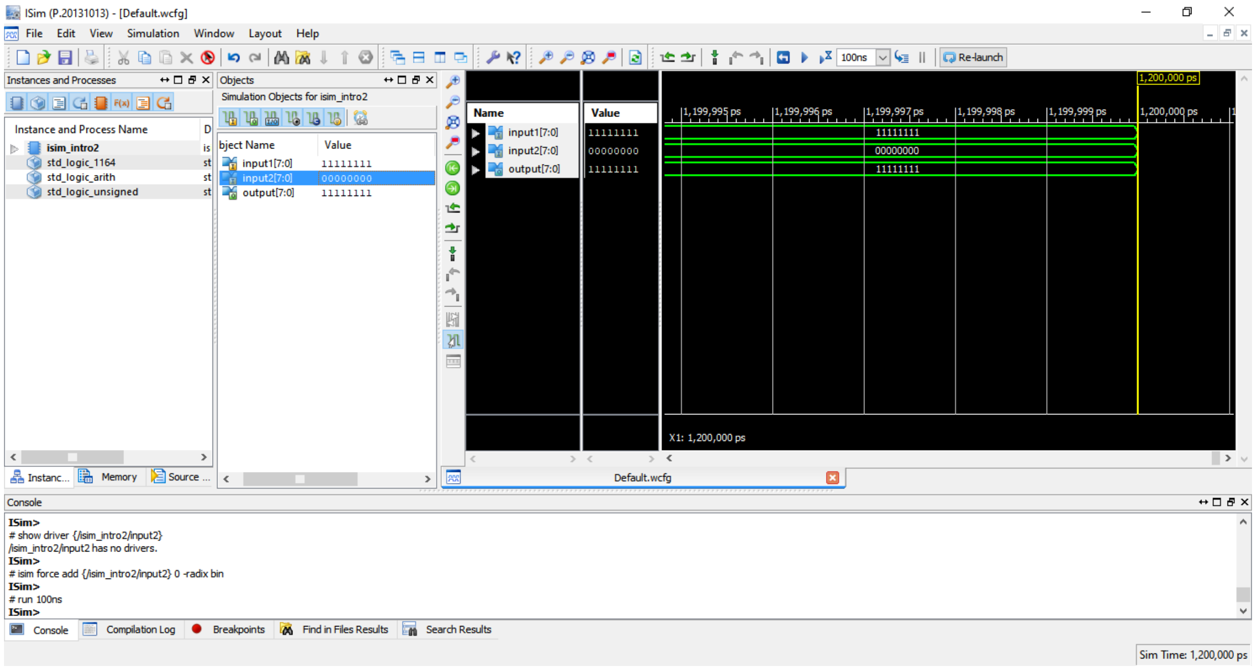
```
Object Name    Value
input1[7:0]    11111111
input2[7:0]    11111111
output[7:0]    11111110
```

```
Name           Value
input1[7:0]    11111111
input2[7:0]    11111111
output[7:0]    11111110
```

```
X1: 1,100,000 ps
Sim Time: 1,100,000 ps
```

```
Finished circuit initialization process.
ISim>
# isim force add {/isim_intro2/input1} FF -radix hex
ISim>
# isim force add {/isim_intro2/input2} 255 -radix dec
ISim>
# run 100ns
ISim>
```

As per the code, the hexadecimal and decimal inputs are converted to binary and the 8-bit inputs are being added to obtain the 8-bit output. Since, in bit operations, 1+1 = 10, we take 0 as output of operation and carry forward the 1 to the next bit operation. The result of continuing this operation is 111111110. However, out result is 8-bit long so the last bit is truncated, and the final output is 11111110. When we add “FF” with the 8-bit array 00000000, we find the following:


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Objects
Simulation Objects for isim_intro2
Default.wcfg
Console Compilation Log Breakpoints Find in Files Results Search Results

```
Object Name    Value
input1[7:0]    11111111
input2[7:0]    00000000
output[7:0]    11111111
```

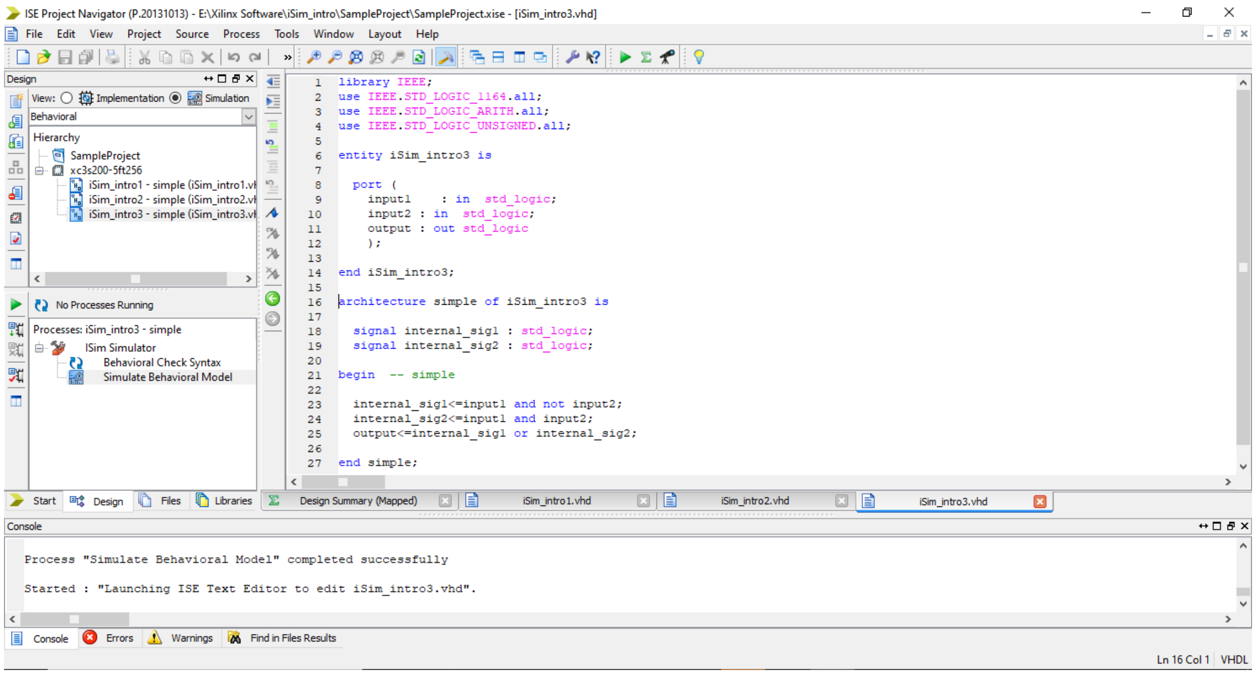
```
Name           Value
input1[7:0]    11111111
input2[7:0]    00000000
output[7:0]    11111111
```

```
X1: 1,200,000 ps
Sim Time: 1,200,000 ps
```

```
ISim>
# show driver {/isim_intro2/input2}
/isim_intro2/input2 has no drivers.
ISim>
# isim force add {/isim_intro2/input2} 0 -radix bin
ISim>
# run 100ns
ISim>
```

For each operation, 1+0 = 1. Hence, the result of 11111111 and 00000000 is 11111111.

4.3) iSim_intro3.vhd contains the implementation of an “XOR” gate using the formula “(input1&(!input2))|((!input1)&input2)”. We are provided with the following code:


ISE Project Navigator (P.20131013) - E:\Xilinx Software\iSim_intro\SampleProject\SampleProject.xise - [iSim_intro3.vhd]
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Behavioral
Hierarchy
SampleProject
xc3s200-5ft256
iSim_intro1 - simple (iSim_intro1.vh
iSim_intro2 - simple (iSim_intro2.vh
iSim_intro3 - simple (iSim_intro3.vh
No Processes Running
Processes: iSim_intro3 - simple
ISim Simulator
Behavioral Check Syntax
Simulate Behavioral Model
Start Design Files Libraries
Design Summary (Mapped)
iSim_intro1.vhd
iSim_intro2.vhd
iSim_intro3.vhd
Console
Console Errors Warnings Find in Files Results
Ln 16 Col 1 VHDL

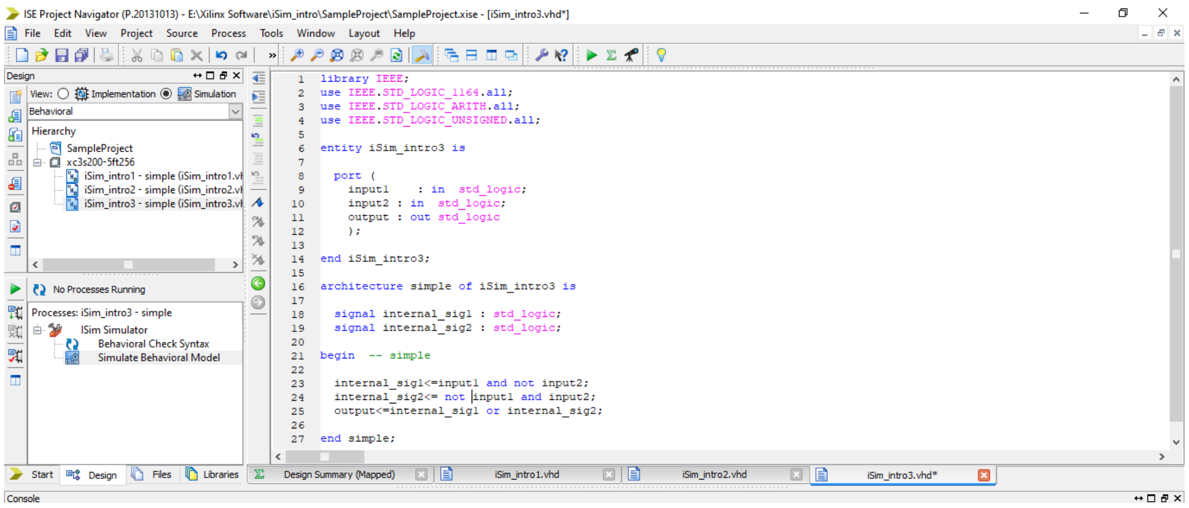
```
1   library IEEE;
2   use IEEE.STD_LOGIC_1164.all;
3   use IEEE.STD_LOGIC_ARITH.all;
4   use IEEE.STD_LOGIC_UNSIGNED.all;
5
6   entity iSim_intro3 is
7
8     port (
9       input1    : in  std_logic;
10      input2 : in  std_logic;
11      output : out std_logic
12      );
13
14  end iSim_intro3;
15
16  architecture simple of iSim_intro3 is
17
18    signal internal_sig1 : std_logic;
19    signal internal_sig2 : std_logic;
20
21  begin  -- simple
22
23    internal_sig1<=input1 and not input2;
24    internal_sig2<=input1 and input2;
25    output<=internal_sig1 or internal_sig2;
26
27  end simple;
```

```
Process "Simulate Behavioral Model" completed successfully

Started : "Launching ISE Text Editor to edit iSim_intro3.vhd".
```

However, there is a bug in the code. We can spot this bug in line 24. The keyword “not” is missing in front of input1. Consequently, we change “internal_sig2<=input1 and input2;” to “not internal_sig2<=input1 and input2;” If we left line 24 as before, the implementation would be “(input1&(!input2))|((input1)&input2)”, which is not the “XOR” gate. This circuit would create high output for 2 high input signals, or high input1 and low input2. Instead, we require high output for high input1 and low input2 or vice versa, and low output when both inputs are simultaneously high or simultaneously low.

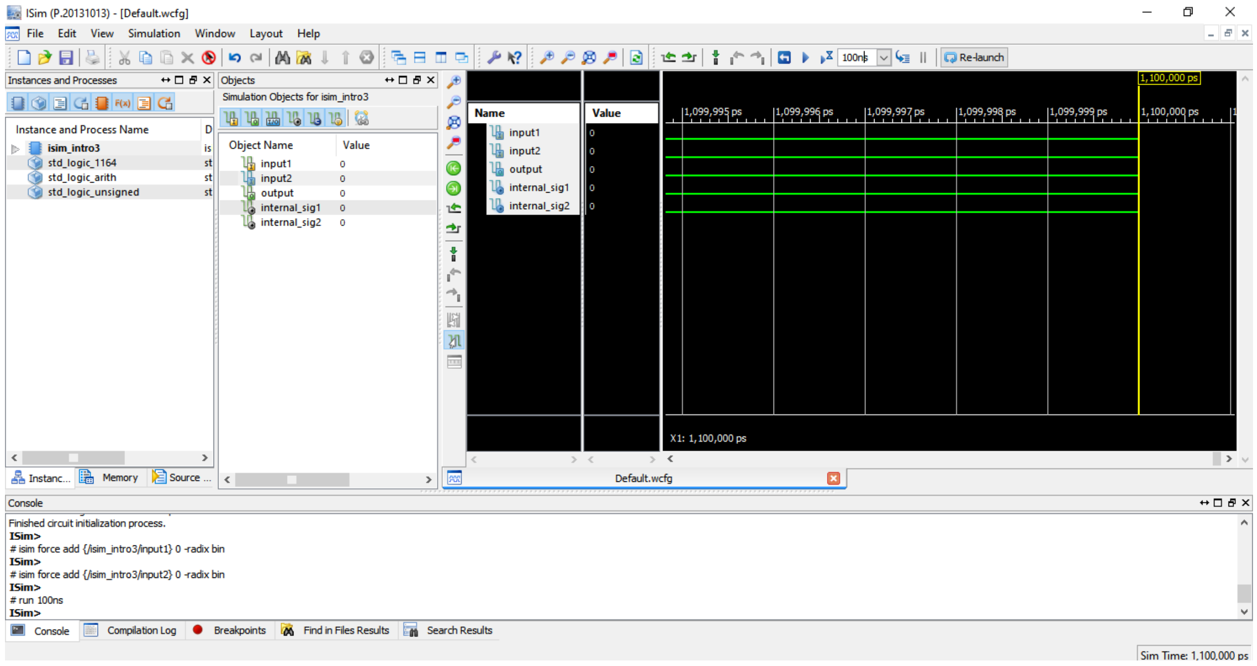
The code looks like this after the changes:


ISE Project Navigator (P.20131013) - E:\Xilinx Software\iSim_intro\SampleProject\SampleProject.xise - [iSim_intro3.vhd*]
File Edit View Project Source Process Tools Window Layout Help
Design
View: Implementation Simulation
Behavioral
Hierarchy
SampleProject
xc3s200-5ft256
iSim_intro1 - simple (iSim_intro1.vh
iSim_intro2 - simple (iSim_intro2.vh
iSim_intro3 - simple (iSim_intro3.vh
No Processes Running
Processes: iSim_intro3 - simple
ISim Simulator
Behavioral Check Syntax
Simulate Behavioral Model
Start Design Files Libraries
Design Summary (Mapped)
iSim_intro1.vhd
iSim_intro2.vhd
iSim_intro3.vhd*
Console

```
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_intro3 is
7
8     port (
9       input1    : in  std_logic;
10      input2 : in  std_logic;
11      output : out std_logic
12      );
13
14  end iSim_intro3;
15
16  architecture simple of iSim_intro3 is
17
18    signal internal_sig1 : std_logic;
19    signal internal_sig2 : std_logic;
20
21  begin  -- simple
22
23    internal_sig1<=input1 and not input2;
24    internal_sig2<= not input1 and input2;
25    output<=internal_sig1 or internal_sig2;
26
27  end simple;
```

After running the simulation, we obtain the following results for provided inputs:

Inputs: 0 and 0, output: 0



ISim (P.20131013) - [Default.wcfg]

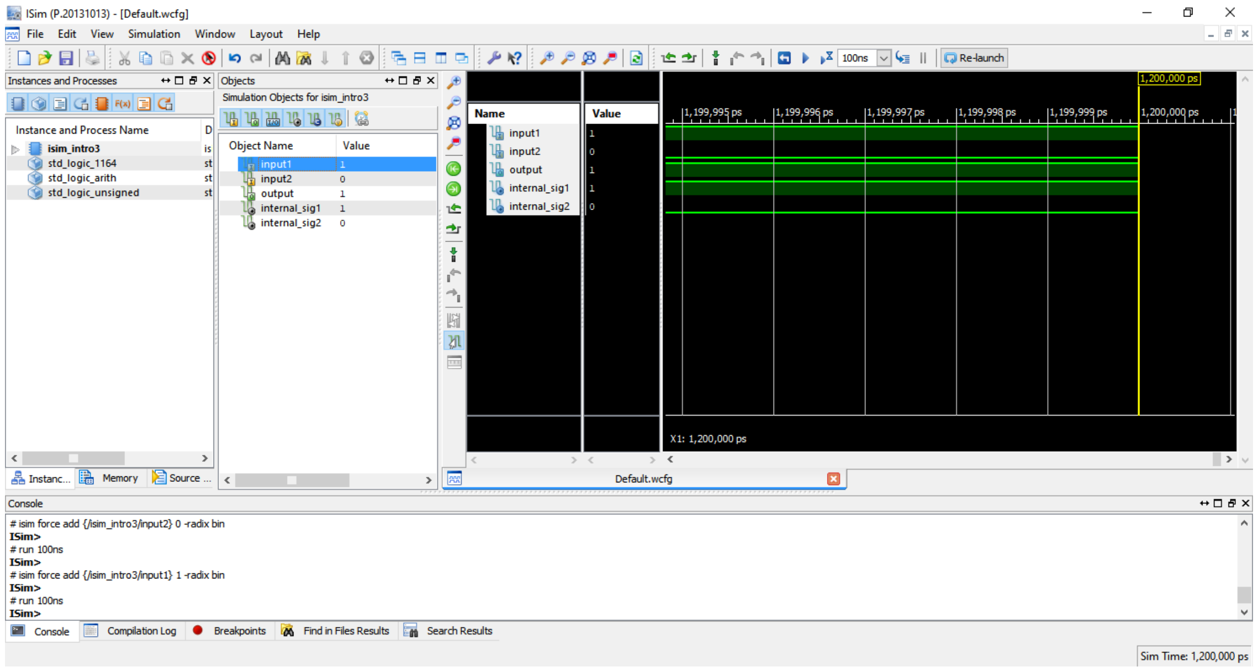
Object Name	Value
input1	0
input2	0
output	0
internal_sig1	0
internal_sig2	0

X1: 1,100,000 ps

```
Finished circuit initialization process.
ISim>
# isim force add {/isim_intro3/input1} 0 -radix bin
ISim>
# isim force add {/isim_intro3/input2} 0 -radix bin
ISim>
# run 100ns
ISim>
```

Sim Time: 1,100,000 ps

Inputs: 1 and 0, output: 1



ISim (P.20131013) - [Default.wcfg]

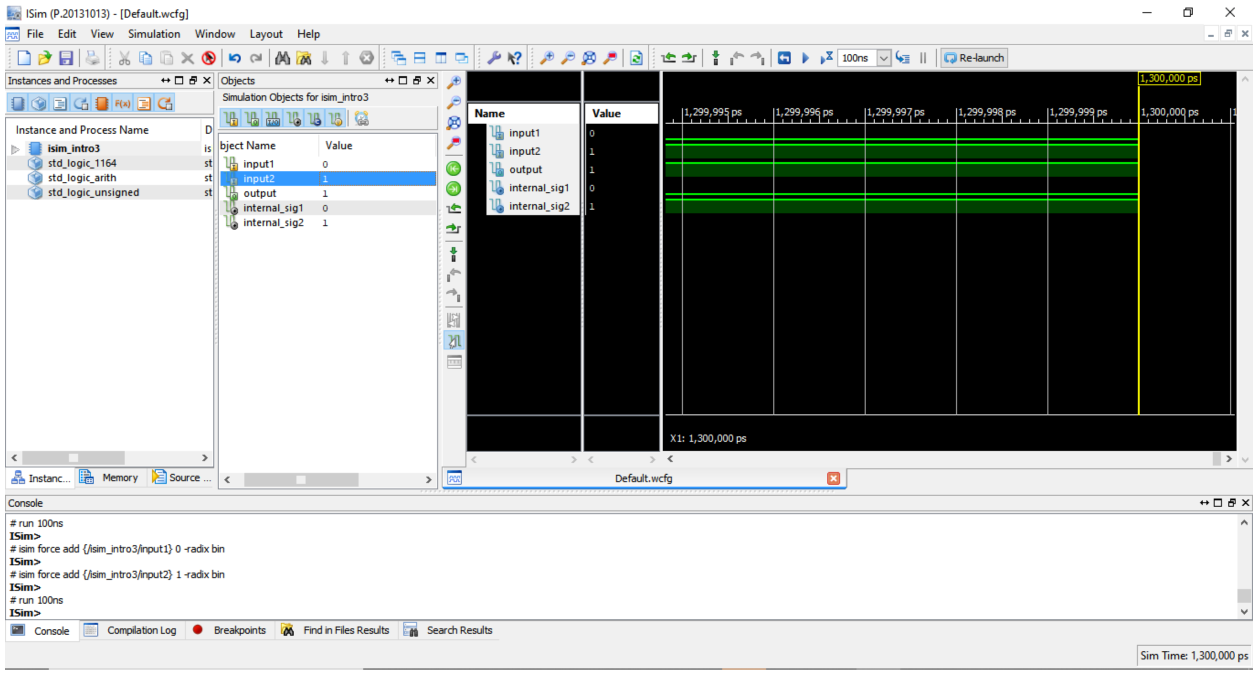
Object Name	Value
input1	1
input2	0
output	1
internal_sig1	1
internal_sig2	0

X1: 1,200,000 ps

```
# isim force add {/isim_intro3/input2} 0 -radix bin
ISim>
# run 100ns
ISim>
# isim force add {/isim_intro3/input1} 1 -radix bin
ISim>
# run 100ns
ISim>
```

Sim Time: 1,200,000 ps

Inputs: 0 and 1, output: 1

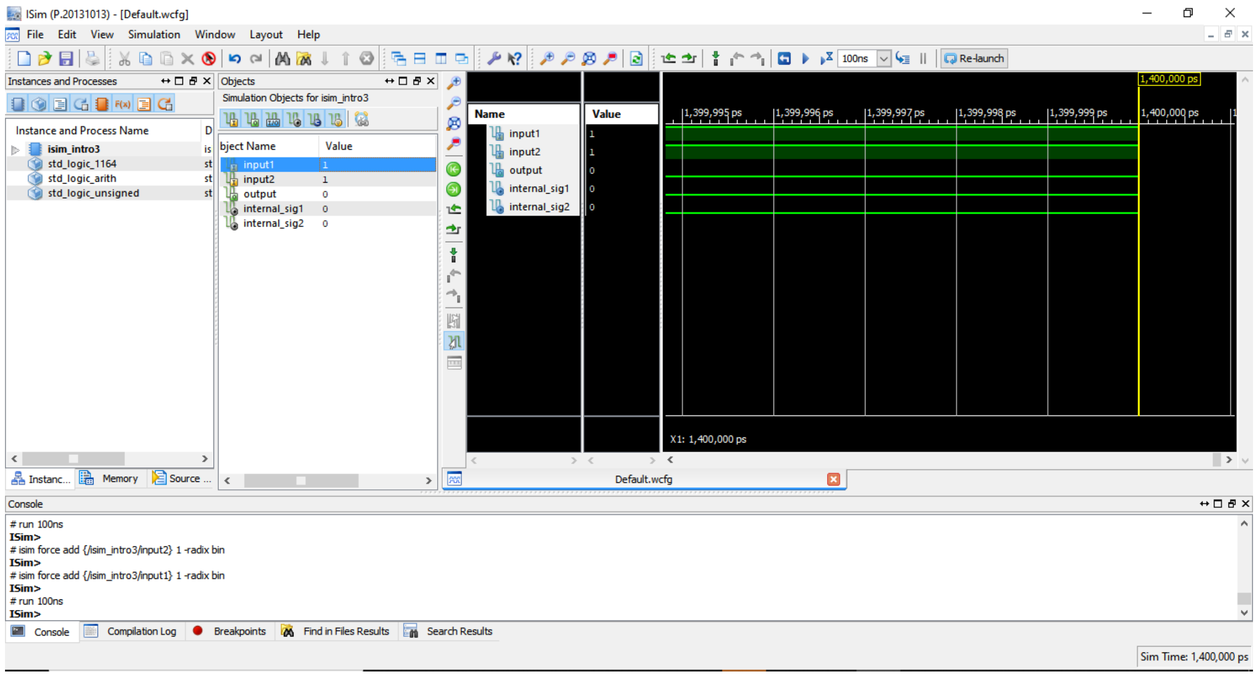


bject Name	Value
input1	0
input2	1
output	1
internal_sig1	0
internal_sig2	1

```
# run 100ns
ISim>
# isim force add {/isim_intro3/input1} 0 -radix bin
ISim>
# isim force add {/isim_intro3/input2} 1 -radix bin
ISim>
# run 100ns
ISim>
```

Sim Time: 1,300,000 ps

Inputs: 1 and 1, output: 0



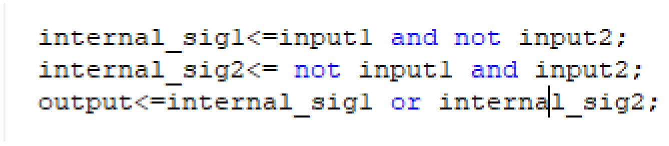
bject Name	Value
input1	1
input2	1
output	0
internal_sig1	0
internal_sig2	0

```
# run 100ns
ISim>
# isim force add {/isim_intro3/input2} 1 -radix bin
ISim>
# isim force add {/isim_intro3/input1} 1 -radix bin
ISim>
# run 100ns
ISim>
```

Sim Time: 1,400,000 ps

These results are consistent with the implementation of an XOR logic gate.

To see what is actually happening, let us focus on the following snippet of code:



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

Let’s say we have inputs input1 = 1 and input2 = 0. We can break down the process as follows:

not input2 = 1

input1 and not input2 = 1 and 1 = 1 = internal_sig1

not input1 = 0

not input1 and input2 = 0 and 0 = 0 = internal_sig2

output = internal_sig1 or internal_sig2 = 1 or 0 = 1

The same process as the one above takes place internally for all the inputs. As a result, we obtain the outputs shown in the images provided above.

Conclusion

In this lab, we obtained a basic introduction to Xilinx ISE, VHDL code and logic circuit simulation. We learned how to create projects and add source files. We also learned how to implement and run code in Xilinx that mimic specific logic circuits or logical units such as AND gate, XOR gate and Adder, and how to analyze code and debug it for semantic errors. We also learned how to provide inputs to the logical unit simulation and analyze the outputs.

References

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

http://fpga-fhu.user.jacobs-university.de/docs/iSim%20Reference.pdf