Computer Logic 1 – Practical 1

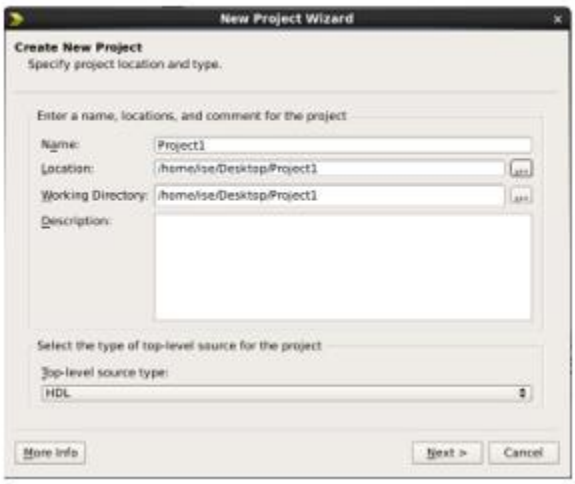
Objective:

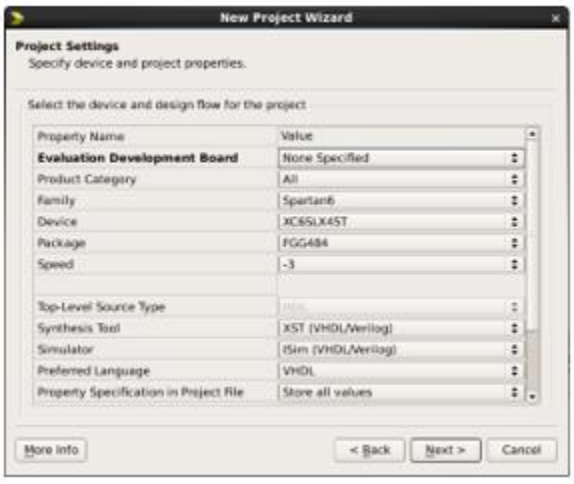
To use the *Xilinx ISE Design Suite* to simulate simple combinational circuits.

Tasks:

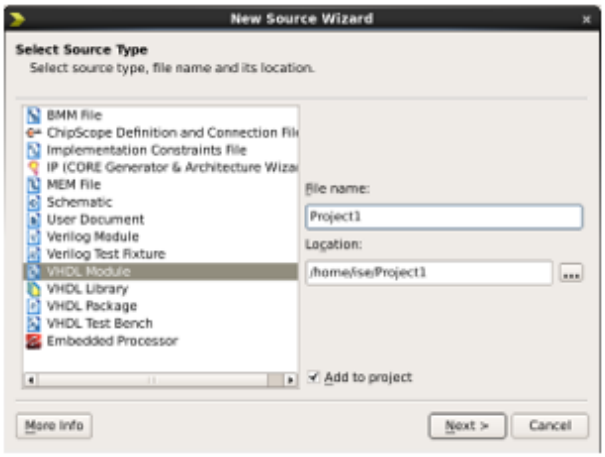
1) The *Xilinx ISE environment* was opened on the Linux virtual machine and a new project was created using the menu item *File ~> New Project*. The project was then called "*Project1*" and it was set to be stored in a particular location. Then, the following options were chosen (whilst others were left unchanged):

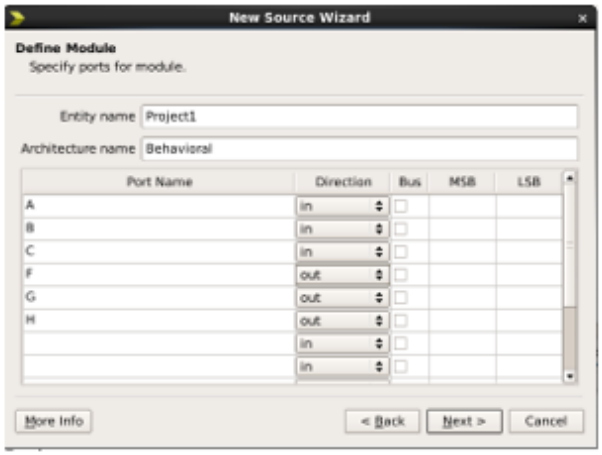
   - Top-level source type: HDL
   - Family: Spartan6
   - Device: XC6SLX45T
   - Package: FGG484
   - Synthesis tool: XST (VHDL/Verilog)
   - Simulator: Isim (VHDL/Verilog)
   - Preferred language: VHDL





2) A new source file was then created using the menu item *Project ~> New Source*. The new source was of type VHDL Module and 3 input(A, B, C) and another 3 output(F, G, H) ports were assigned as shown.





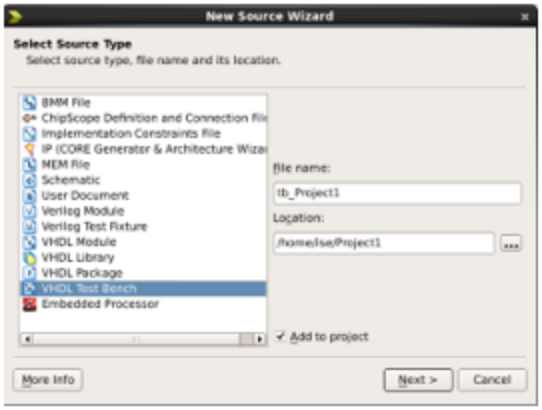
3) In the architecture body, VHDL code was added to set the inputs to correspond to Truth Table 1. The truth table and code inputted are shown below:

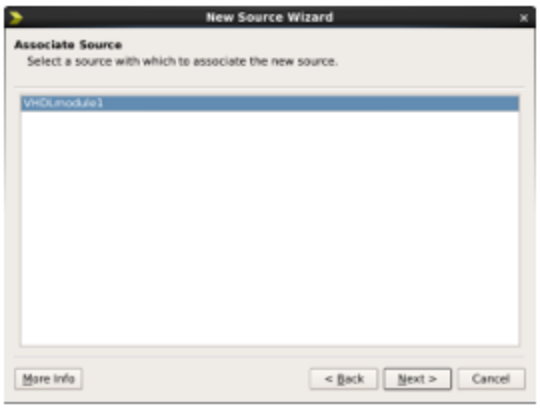
TABLE 1: TRUTH TABLE.

A	*B*	*C*	*F*	*G*	*H*
0	0	0	1	0	1
0	0	1	0	1	1
0	1	0	0	1	0
0	1	1	1	0	1
1	0	0	0	1	1
1	0	1	1	0	1
1	1	0	0	0	0
1	1	1	0	1	0

```
architecture Behavioral of VHDLmodule1 is

begin
F <= (not A and not B and not C)
      or (not A and B and C)
      or (A and not B and C);

G <= (not A and not B and C)
      or (not A and B and not C)
      or (A and not B and not C)
      or (A and B and C);

H <= (not A and not B and not C)
      or (not A and not B and C)
      or (not A and B and C)
      or (A and not B and not C)
      or (A and not B and C);
end Behavioral;
```

4) After the VHDL code was entered, functional simulation was performed. This was done by generating a test bench:
   a) A new file called *tb_Project1* was created from menu item *Project ~> New Source*. The source type was set to *VHDL Test Bench* and the original VHDL file *Project1* was associated with this newly created one.





b) Parts of the test bench usually used for clock were eliminated as they were irrelevant to this particular design.

c) The stimulus process was replaced with the following code:

```
a_proc: process
  begin
     A <= '0';
     wait for 200 ns;

     A <= '1';
     wait for 200 ns;
  end process;
```

d) Other processes were then written to stimulate B and C as shown:

```
b_proc: process
  begin
     B <= '0';
     wait for 100 ns;

     B <= '1';
     wait for 100 ns;
  end process;
```

```
c_proc: process
  begin
     C <= '0';
     wait for 50 ns;

     C <= '1';
     wait for 50 ns;
  end process;
```

5) The *Simulation* view in the top left panel was then selected, followed by the *Behavioral* simulation. The test bench was then selected and the *Simulate Behavioral Model* option in the process panel was chosen.
6) The outputs were checked to correspond to *Truth Table 1*.
7) The test bench file was then modified with some artificial glitches and saved.

```
B <= '0';   to:   B <= '0' after 3 ns;
B <= '1';   to:   B <= '1' after 3 ns;
C <= '0';   to:   C <= '0' after 5 ns;
C <= '1';   to:   C <= '1' after 5 ns;
```

8) The design was simulated once more.
9) In this case, some glitches known as hazards were introduced. There were 2 types:
   a) Static hazards: Those having the same levels before and after (initial value ‘1’, followed by short ‘0’ and final value is ‘1’ again)

   b) Dynamic hazards: Those having different levels before and after (initial value ‘0’, followed by a short ‘1’ and another short ‘0’ and final value is ‘1’)

These hazards were caused by the artificial glitches introduced in step 7.

An example of a static hazard in this Project would be: Outputs *F* and *G* between 100 – 110 ns.

An example of a static hazard in this Project would be: Outputs *F* and *G* between 190 – 210 ns.

Appendix (Code):

VHDL Module (Project1):

```
----------------------------------------------------------------------------------
-- Company:         UOM
-- Engineer:        Tristan Oa Galea
--
-- Create Date:     22:28:58 11/08/2018
-- Design Name:     Project 1
-- Module Name:     VHDLmodule1 - Behavioral
-- Project Name:    Project 1
-- 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 primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity VHDLmodule1 is
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           C : in  STD_LOGIC;
           F : out  STD_LOGIC;
           G : out  STD_LOGIC;
           H : out  STD_LOGIC);
end VHDLmodule1;

architecture Behavioral of VHDLmodule1 is

begin
F <= (not A and not B and not C)
      or (not A and B and C)
      or (A and not B and C);
```

```
47
48  G <= (not A and not B and C)
49        or (not A and B and not C)
50        or (A and not B and not C)
51        or (A and B and C);
52
53  H <= (not A and not B and not C)
54        or (not A and not B and C)
55        or (not A and B and C)
56        or (A and not B and not C)
57        or (A and not B and C);
58  end Behavioral;
```

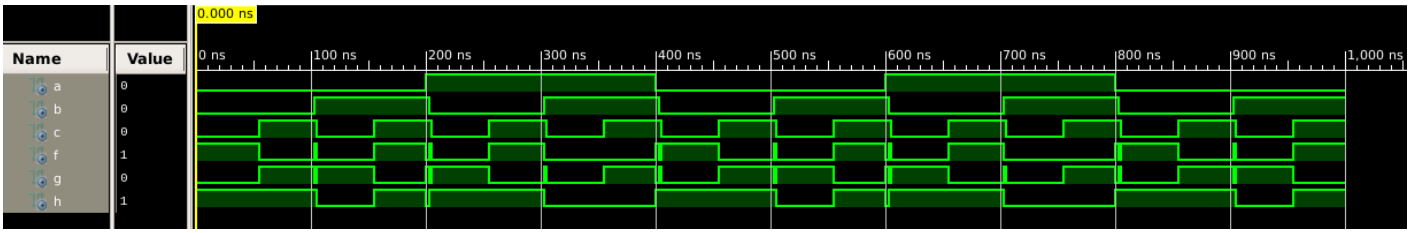
VHDL Test Bench (tb_Project1):

```
--------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:   22:40:25 11/08/2018
-- Design Name:
-- Module Name:   /home/ise/Project1/tb_Project1.vhd
-- Project Name:  Project1
-- Target Device:
-- Tool versions:
-- Description:
--
-- VHDL Test Bench Created by ISE for module: VHDLmodule1
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
-- Notes:
-- This testbench has been automatically generated using types std_logic and
-- std_logic_vector for the ports of the unit under test.  Xilinx recommends
-- that these types always be used for the top-level I/O of a design in order
-- to guarantee that the testbench will bind correctly to the post-implementation
-- simulation model.
--------------------------------------------------------------------------------
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_Project1 IS
END tb_Project1;

ARCHITECTURE behavior OF tb_Project1 IS

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

    COMPONENT VHDLmodule1
    PORT(
         A : IN  std_logic;
         B : IN  std_logic;
         C : IN  std_logic;
         F : OUT  std_logic;
         G : OUT  std_logic;
         H : OUT  std_logic
        );
    END COMPONENT;

   --Inputs
   signal A : std_logic := '0';
   signal B : std_logic := '0';
   signal C : std_logic := '0';
```

```
57
58      --Outputs
59      signal F : std_logic;
60      signal G : std_logic;
61      signal H : std_logic;
62      -- No clocks detected in port list. Replace <clock> below with
63      -- appropriate port name
64
65  BEGIN
66
67      -- Instantiate the Unit Under Test (UUT)
68      uut: VHDLmodule1 PORT MAP (
69              A => A,
70              B => B,
71              C => C,
72              F => F,
73              G => G,
74              H => H
75            );
76
77   a_proc: process
78     begin
79        A <= '0';
80        wait for 200 ns;
81
82        A <= '1';
83        wait for 200 ns;
84     end process;
85
86   b_proc: process
87     begin
88        B <= '0' after 3 ns;
89        wait for 100 ns;
90
91        B <= '1' after 3 ns;
92        wait for 100 ns;
93     end process;
94
95   c_proc: process
96     begin
97        C <= '0' after 5 ns;
98        wait for 50 ns;
99
100       C <= '1' after 5 ns;
101       wait for 50 ns;
102    end process;
103
104 END;
```

Simulated Behavioral Model:

(With glitches)



Conclusion:

Xilinx ISE Design Suite was successfully used to simulate simple combinational circuits.