Laboratory Session 4

Introduction to VHDL: Additional

Objective:

- Understanding the basics of VHDL descriptions
- Use Quartus IDE
- Use Modelist Simulator

Software Packages

- Altera Quartus II Software
- ModelSim Altera Software

1. Introduction

In VHDL, sequential statements are implemented using processes. A process block can have an optional label and it takes a sensitivity input list. The sensitivity list can include inputs and signals that the activates the process block upon changing. In item declaration, variables can be defined. Note that when a value is written to a signal, the updated value is read in the next cycle whereas variables are updated immediately.

```
my_label: process(sensitivity_list) is
   <item_declaration>
begin
   <sequential_statements>
end process my_label;
```

A function in VHDL is used to simplify code development. Statements in a function are executed sequentially.

```
function function_name (parameter_list) return type is
        declarations
begin
        sequential statements
end function_name;
```

2. Samples

2.1.FSM under VHDL

Use the following VHDL code in a new project:

```
library ieee;
use ieee.std_logic_1164.all;
entity FSM is
port (
        clk, rst: in std_logic;
        output: out std_logic_vector(1 downto 0)
);
end entity;

architecture beh of FSM is
type state is (S0, S1, S2, S3);
signal current_state, next_state : state;
begin
        process(clk, rst)
        begin
                if (rst = '1') then
                        current_state <= S0;
                elsif (rising_edge(clk)) then
                        current_state <= next_state;
                else
                        null;
                end if;
        end process;

        process(current_state)
        begin
                case current_state is
                        when S0 => next_state <= S1;
                                        output <= “00”;
                        when S1 => next_state <= S2;
                                        output <= “01”;
                        when S2 => next_state <= S3;
                                        output <= “10”;
                        when S3 => next_state <= S3;
                                        output <= “11”;
                end case;
        end process;
end architecture;
```

After compiling, check the RTL, power consumption and maximum frequency:

- Check the **maximum frequency** as follows:

1. Go to the compilation report tab
2. On the left pane, under table of contents expand TimeQuest Timing Analyzer.
3. Expand Slow Model. Note: in case there are several Slow Models expand the one with the highest temperature as this analysis are device based and we are not specifying our device so far.
4. Select Fmax Summary

- To **verify the FSM**, go to Tools → Netlist Viewer → State Machine Viewer to view the available states, make sure that there are 4 states S0, S1, S2 and S3. Verify using ModelSim.

2.2. Lookup Tables under VHDL

The following lookup table implementations converts decimals to binary; create a VHDL file and paste the following code:

```
library ieee;
use ieee.std_logic_1164.all;
entity Lookup_Table is
port(
        a: in integer;
        b: out std_logic_vector(2 downto 0)
);
end entity;
architecture beh of Lookup_Table is
        function LUT (X: integer) return std_logic_vector is
        begin
                case X is
                        when 0 => return "000";
                        when 1 => return "001";
                        when 2 => return "010";
                        when 3 => return "011";
                        when 4 => return "100";
                        when 5 => return "101";
                        when 6 => return "110";
                        when 7 => return "111";
                        when others => return "000";
                end case;
        end function;
begin
        b <= LUT(a);
end architecture;
```

Compile and check the RTL, Power Consumption, Fmax and State Machine Viewer. Verify using ModelSim

3. Deliverables

Practical demonstration of the sample exercises will be requested during the lab. Keep in mind that a combined report that presents the analysis and testing of the sample exercises as well as the practice exercises of lab 1 and 2 will be requested, so maintain your project file for the snapshots of the work done.