THE OPEN UNIVERSITY OF SRI LANKA
BACHELOR OF TECHNOLOGY - LEVEL 6
ECX6236 – PROCESSOR DESIGN
FINAL EXAMINATION 2013

DURATION: THREE HOURS

Date : **28th August 2014** **Time :** **09:30 – 12:30 Hrs**

Answer **three** questions including the question in Section A and selecting two from Section B. Refer the Annexure for syntax of VHDL instructions

Section A

The following question is compulsory. It carries 70 marks.

1. Some universities have already set up Remote Laboratories (RL) for their students. Remote laboratories are used for conducting real experiments remotely by using the internet. Therefore students will be able to do their experiments from home or any other physical location without attending to the laboratory premises.

   In RL usually all necessary resources such as measuring instruments, components, and other hardware boards/ units must be allocated for a particular laboratory experiment. In addition to that, these resources should be connected through the internet to the student so as to control the resources by interconnecting the resources, giving inputs (controlling signals, data etc.) and reading/ viewing outputs (data, video streams, etc.). On the other hand there can be more than one student simultaneously connected to the RL for doing their experiments depending on the availability of the resources.

   Your task is to design a special purpose processor for Resource Allocation and Controlling (PRAC), which can be used for developing a switching unit (SU) for RL. The main function of the SU is to establish interconnections among the resources (if necessary) and to provide necessary resources to the end user. For that reason, the SU can be connected to the internet through a server. Consequently, RL can be easily set up by using the SU. Moreover, experiments and resources can be scaled up by just programming the processor.

   The processor should receive all commands in a standard instruction format similar to an instruction set in general purpose processors (in the way of Opcode and Operands in an instruction format). According to the command received, the processor must perform the relevant task. Since students can be simultaneously connected to RL, the PRAC should be able to assign input and/or output ports to each student as soon as it receives a request. Accordingly, the processor will automatically allocate necessary resources for the experiment by switching them appropriately.

   In addition to the functions mentioned above, the processor must do the resource management according to the given set of instructions. Therefore necessary instructions should be provided for this purpose. For simplicity you may assume that any resource will not be shared by two or more experiments at a time. However some resources may not be allowed for a longer duration. You may use internal memory/registers to keep all the information and may include special functional units/components with a description.

   As this is a special purpose processor, your design may differ from general purpose processors. Clearly state any other assumptions you made (if any).

a) Draw a diagram to show how to deploy the processor, PRAC you designed (once it is fabricated) when manufacturing SU.
b) Write a short description on the working procedure of the complete system indicating the internal functionality of your processor.
c) Accordingly, identify the necessary instructions and design an ISA for this processor.
d) Using your ISA write a program which can handle two experiments at the same time.
e) Draw a block diagram for the processor indicating all input and output signals. Clearly state all functions of each block inside the processor and show the data path.
f) Identify entities for which you need to write VHDL codes to synthesise the processor.
g) Write the behavioural/ structural VHDL codes for each entity except for the Control Unit of the processor. You may define the Control Unit as a component.

Section B

*Answer **two** questions from this section. Each question carries 15 marks.*

2.

a) Integrate the VHDL codes for different entities in *Question (1.g)* of *Section A* to obtain a complete VHDL code for the PRAC.
b) Briefly describe how you are estimating the performance of a processor. Estimate the performance of the PRAC you designed in *Question 1.*

3.

a) Construct the state diagram of the Control Unit of the processor you designed in *Question 1.*
b) Which factors do you need to consider estimating the cost of the PRAC?

4.

a) Propose a method to share the resources concurrently using PRAC. Accordingly, what modifications do you need for ISA?
b) Briefly explain the steps that you have to follow to implement PRAC on an FPGA.

5.

a) Write a Behavioural VHDL code for the *8-bit Full Adder* as given in the Fig 1.

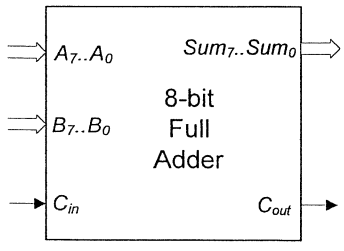

Fig 1

b) Construct a unit (draw a schematic diagram) to find the sum (y) of all elements of an array (A) as given in the following equation using the *8-bit Full Adder.*

$$y=\sum_{i=1}^{n} A_i \text{ ; assume } y<256.$$

c) Write a Structural VHDL code for the *unit* according to your schematic diagram drawn for the *Question (5.b).*

Annexure

Syntax of selected instructions of the VHDL

⊠
```
ARCHITECTURE architecture_name OF entity_name IS
  [declaration part]
BEGIN
  Concurrent statements part
END architecture_name
```

⊠
```
CASE expression IS
  WHEN value=> statements;
  WHEN value=> statements;
  WHEN OTHERS statements;
END CASE;
```

⊠
```
COMPONENT component_name
  PORT (port1_name : port1_type;
        port2_name : port2_type;
        …);
END COMPONENT [component_name];
```

⊠
```
ENTITY entity_name IS
  PORT (port1 : port1_type;
        port2 : port2_type;
        …);
END entity_name;
```

⊠
```
IF condition THEN
  Sequence of statements
  {ELSIF condition THEN
     Sequence of statements}
[ELSE
  Sequence of statements]
END IF;
```

⊠
```
LIBRARY library_name;
```

⊠
```
Instance_label: component_name PORT MAP (first_port, second_port,
                                         third_port, …);
Instance_label: component_name PORT MAP (formal1=> actual1,
                                         formal1=> actual1,
                                         formal1=> actual1, …);
```

⊠
```
[process_label:] PROCESS (signal1, signal2, …)
                   [declaration part]
                 BEGIN
                   Sequential statements part
                 END PROCESS;
```

⊠
```
SIGNAL signal_name : signal_type;
```

⊠
```
TYPE type_name;
```

⊠
```
USE library_name.type_expression.inclussion;
```

⊠
```
WAIT FOR time_expression;
```
WAIT ON **signal1, signal2, …;**
WAIT UNTIL ***condition;***

⊠
```
WHILE condition LOOP
  Sequential statements
END LOOP;
```