00056

THE OPEN UNIVERSITY OF SRI LANKA
BACHELOR OF TECHNOLOGY - LEVEL 6
ECX6236 – PROCESSOR DESIGN
FINAL EXAMINATION 2011

DURATION: THREE HOURS

Date : 15th March 2012 **Time : 1400 - 1700 Hrs**

Answer **three** questions including the question in Section A and selecting two from Section B.

Section A

The following question is compulsory. It carries 70 marks.

1. In Sri Lanka, Electronic Ticket Issuing Machines (eTIM) are not widely used in the public transport sector. Though electronic handheld machines are recently provided for private bus conductors, in SLTB buses, mechanical machines are generally used for issuing tickets. Moreover, at railway stations, no eTIM are used for issuing tickets. On the other hand passenger-operated eTIM are not installed anywhere in the country, although it is common in other countries.

   In order to manufacture different kind of eTIM a special purpose processor is required. This processor which is an Application Specific Integrated Circuit (ASIC) will help to simplify manufacturing and maintaining eTIM.

   Your task is to design a processor for eTIM. As the processor is used only for this purpose all inputs and outputs should be relevant to any kind of eTIM. 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 instruction format). According to the command received, the processor must perform the relevant task and output signals required for eTIM.

   The processor that you are going to design should be able to perform at least the tasks given here. It should issue signals for displaying the information of a ticket on a display, signals for printing a ticket and signals for refunding/returning the balance money in appropriate denominators (for passenger-operated eTIM). The processor should also do the necessary calculations whenever needed. In addition to that, there should be a mode to input and update necessary information such as sectors of travelling and their fares, denominations available for returning the balance (for passenger-operated eTIM) and etc.

   As this is a special purpose processor, your design may differ from general purpose processors. You may use internal memory/registers to keep all the information and may include special functional units/ components along with a description. Clearly state any other assumptions you made (if any).

   a) Draw a diagram to show how to deploy the processor you designed for eTIM (once it is fabricated) when manufacturing eTIM.
   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 needed for this processor.
   d) Using your instructions write sequential steps for printing a ticket for a given destination and returning the balance with predefined denominations.

e) Draw a block diagram for the processor. Clearly state all functions of each block inside the processor and show the data path. Indicate all input and output signals of the processor.

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. (Refer the Annexure for syntax of VHDL instructions).

Section B

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

2.

a) Name and describe the modelling methods available in VHDL. Give examples for each modelling method selecting from the code in *Question (1.g)*.

b) Integrate the VHDL codes for different entities in *Question (1.g)* of *Section A* to obtain a complete VHDL code for the processor for eTIM.

3.

a) What are the steps that you follow to design a hardwired control unit of a processor?

b) Construct the state diagram of the Control Unit of the processor you designed in *Question 1.*

4.

a) Briefly, describe the tasks of a Computer Designer.

b) Describe how you are estimating the performance of a processor.

c) Estimate the performance of the processor for eTIM.

5.

a) Write a Behavioural VHDL code for the *Full Adder* of Fig 5.1, which adds two binary digits (X and Y), and a carry (C_{in}) to give a sum (Sum) and a carry out (C_{out}).

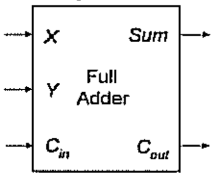

Fig 5.1

b) Draw a schematic diagram for a *6-bit Comparator* (Fig 5.2), which sets the signal S to 1 when the inputs A and B are equal, using the *Full Adder* given in Fig 5.1.

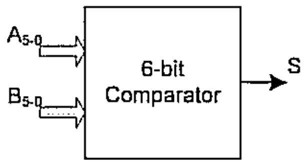

Fig 5.2

c) Write a Structural VHDL code for the *6-bit Comparator* 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;
```