00041

THE OPEN UNIVERSITY OF SRI LANKA
BACHELOR OF TECHNOLOGY - LEVEL 6
ECX6236 – PROCESSOR DESIGN
FINAL EXAMINATION 2010

DURATION: THREE HOURS

Date : 24th March 2011 **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. You may have noticed in cricket grounds scoreboards are operated manually. Someone has to attend all the time during a cricket match to make any change to the scoreboard. Though there are digital scoreboards full details are not displayed. Moreover the official scorers most of the time use printed scorebooks and scoring is also done manually. Therefore it is suggested to manufacture a special unit – eScorebook which can be used for a scorebook and as well as to send data for an electronic scoreboard. That unit will be based on a special purpose processor – Cricket Scorebook and Scoreboard (CSS) processor which is an Application Specific Integrated Circuit (ASIC).

   Your task is to design this CSS processor. As the processor is used to build the eScorebook you need to provide facility to input details which a scorer usually do when scoring during a match. There should be provision to select the batsman, bowler, fielder who took the catch or involved in a run out, etc. It also needs to input ball by ball descriptions such as whether it is a dot ball, no ball, wide and so on and runs scored. This information can be sent to CSS processor as an instruction using the keyboard/ keypad of the eScorebook. Additionally it is necessary to input the teams at the beginning before starting the match.

   In addition to those inputs eScorebook should continuously update the electronic scoreboard with full details i.e. score and runs of the batsman after each delivery, number of wickets, current batsman and the bowler.

   As the CSS processor is used only for the mentioned purpose your design may differ from general purpose processors. You may use internal memory to keep all the information for one match 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 CSS processor (once it is fabricated) for developing the eScorebook.
   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 a match at least for the first over which is having the following events:
      Dot ball, 2 runs, 1 run, wide ball, no ball and a boundary, wicket (bowled), dot ball, dot ball

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 the modelling methods available in VHDL. Which of the modelling methods is suitable for CSS processor? Justify your answer.
b) Integrate the VHDL codes for different entities in *Question (1.g)* of *Section A* to obtain a complete VHDL code for the CSS processor.

3.

a) How do you estimate the cost of a packaged integrated circuit?
b) Construct the state diagram of the Control Unit of the CSS processor in *Question 1*.

4.

a) Name three applications where you can use systolic architecture and briefly describe one of them.
b) When you implement the CSS processor on an FPGA you will not be able to check all internal signals of the processor other than input/output signals. Propose a technique that you can use in order to check all signals inside the FPGA. You are free to add new components into your design.

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}).

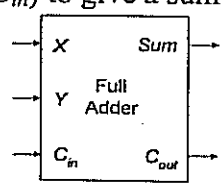

Fig 5.1

b) Draw a schematic diagram for a *4-bit Full Adder* (Fig 5.2), which adds two 4-bit values (A and B), using the *Full Adder* given in Fig 5.1.

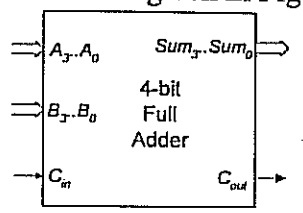

Fig 5.2

c) Write a Structural VHDL code for the *4-bit Full Adder* 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;
```