040

The Open University of Sri Lanka
Faculty of Engineering Technology
Department of Electrical Engineering

Study Programme	:	Bachelor of Science Honours in Engineering
Name of the Examination	:	Final Examination
Course Code and Title	:	**EEX7436 – Processor Design**
Academic Year	:	2023/2024
Date	:	15th February 2025
Time	:	1.30 pm - 4.30 pm
Duration	:	3 hours

General Instructions

1. Read all the instructions carefully before answering the questions.
2. This question paper contains ten (10) questions on four (4) pages.
3. Answer ALL questions, and they are based on the Scenario.
4. The answer to each question should commence from a new page.
5. Refer to the Annexure of the VHDL syntax given on page four (4) to write VHDL code.
6. This is a Closed Book Test (CBT).
7. Answers should be in clear handwriting.
8. Do not use Red colour pen and clearly state your assumptions, if any.

Answer all questions. All questions are based on the Scenario given below.
Refer to the Annexure for the syntax of VHDL instructions.

Scenario

Fuzzy Matrix operations are extensively used today in many applications. Medical diagnosis, Decision-making, Learning techniques, Appointment procedures, Yoga are some of those applications. Standard microprocessors take a considerable number of operations to do any fuzzy matrix operation. Moreover, when the matrix becomes larger, the time taken for any matrix operation will increase exponentially. Therefore, making these operations faster will improve the execution time of those applications. Your task is to design a Fuzzy Matrix Unit (FMU). Accordingly, anyone can use FMU for supporting general microprocessors or integrating FMU into special purpose processors.

We can define fuzzy addition (+), fuzzy multiplication (·), and fuzzy subtraction (−) as follows:

$$a + b = \max(a, b),$$
$$a \cdot b = \min(a, b), \quad and$$
$$a - b = \begin{cases} a \text{ if } a > b \\ 0 \text{ if } a \leq b. \end{cases}$$

There should be special instructions to handle fuzzy matrix operations such as addition, subtraction, and multiplication. Moreover, you must provide other supporting instructions for matrix operations. For example, instructions for defining the size of a matrix, moving elements of a matrix from memory to registers and vice versa and creating submatrix and any other branching instructions.

You have to do all calculations through registers to obtain maximum performance. You may assume that the number of registers is enough to hold three matrices, each with dimension of 3 x 3 at a given time.

You are free to include any special functional units. However, you must describe any specific unit or component included in your design. When answering the questions, clearly state any other assumptions you made, if any.

1. Write a short description of the working procedure of the FMU, indicating the internal functionality of your coprocessor. (8 marks)
2. List all instructions and supporting instructions for matrix operations. Briefly describe them. (12 marks)
3. Define instruction formats for all instructions listed above. (12 marks)
4. Write a sample program for adding and multiplying two 3x3 matrices using your ISA. (12 marks)
5. Draw a diagram for Fuzzy Logic Unit (FLU) that processes fuzzy operations for 2 inputs and give an output. (8 marks)
6. Draw a block diagram for the FMU. You can use FLU for design FMU. Clearly, list function of each block inside the processor and show the data path. Indicate all input and output signals of the processor. (15 marks)
7. Identify entities for which you need to write VHDL codes to synthesise the processor. (5 marks)

8. Write the behavioural/ structural VHDL codes for each entity except for the Control Unit and obtain a complete VHDL code for the FMU. You may define the Control Unit as a component. (15 marks)

9. Briefly describe how you can estimate the cost and the performance of the FMU. (8 marks)

10. Estimate the performance of the FMU in CPI. (5 marks)

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;
```