The Open University of Sri Lanka
Faculty of Engineering Technology 044
Department of Electrical and Computer Engineering

Study Programme	: Bachelor of Technology Honours in Engineering Bachelor of Science Honours in Engineering
Name of the Examination	: Final Examination
Course Code and Title	**: EEX7436 Processor Design**
Academic Year	: 2022/2023
Date	: 09th February 2024
Time	: 0930-1230 hrs
Duration	: **3 hours**

General Instructions

1. Read all instructions carefully before answering the questions.
2. This question paper contains eight (8) questions on three (3) 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 three (3) 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

Neural Processing Unit (NPU) is a specialised hardware designed to accelerate artificial intelligence and machine learning applications. They execute neural network operations. One of the principal operations is matrix multiplication. Further, making the matrix operations faster will improve the execution time of these applications. Therefore, having a specialised matrix operation unit (MOU) will increase the performance of NPUs.

Your task is to design an MOU to support NPUs. However, MoUs can also be integrated into microprocessors as well. There should be special instructions to handle all matrix operations such as addition, subtraction, multiplication, and inversion. Moreover, it should support different matrix multiplications for neural networks. Additionally, you must provide other supporting instructions for matrix operations, such as defining the size of a matrix, moving elements of a matrix from memory to registers and from registers to memory and so on.

It is advisable to do all calculations through buffers/registers to obtain maximum performance. Since the number of registers is limited, you must adopt an appropriate approach to handle large matrices.

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, state any other assumptions you made (if any) clearly.

1. Write a short description of the working procedure of the MOU, indicating the internal functionality. *(10 marks)*
2. List all instructions and supporting instructions for the MOU. Define instruction formats and give a brief description of each instruction. *(25 marks)*
3. Write a sample program for different matrix operations using your ISA. *(12 marks)*
4. Draw a block diagram for the MOU. Clearly, give all functions of each block inside the unit and show the data path. Indicate all input and output signals. *(15 marks)*
5. Identify entities for which you need to write VHDL codes to synthesise the processor. *(8 marks)*
6. Write the behavioural/ structural VHDL codes for each entity except for the Control Unit and obtain a complete VHDL code for the processor. You may define the Control Unit as a component. *(15 marks)*
7. Briefly describe how you would design the Control Unit of the MOU. *(5 marks)*
8. Briefly describe how you estimate the cost of the MOU. Calculate the performance in CPI. *(10 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;
```