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

(24) 00061

Study Programme	: Bachelor of Technology Honours in Engineering
Name of the Examination	: Final Examination
Course Code and Title	**: EEX6351/EEX6151 Digital Electronic Systems**
Academic Year	: 2017/18
Date	: 31st January 2019
Time	: 0930-1230hrs
Duration	: **3 hours**

General Instructions

1. Read all instructions carefully before answering the questions.

2. This question paper contains three (3) questions in SECTION A and three (3) questions in SECTION B on five (5) pages.

3. Answer ALL questions in SECTION A. [60 Marks], and answer any TWO questions from SECTION B. [40 Marks]

4. The answer for each question should commence from a new page.

5. Refer the Annexure of the VHDL syntax given in page five (5) 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

SECTION A: Answer ALL questions. [60 Marks]

Numerical Processing Unit (NPU)

The following description is about a 4-bit Numerical Processing Unit depicts in Figure Q3 which is used for computing arithmetic operations such as addition, subtraction, etc. The NPU consists of a 4-bit Carry Look Ahead (CLA) Adder and the other digital electronic components.

The relationship between present carry (C_{i+1}) and previous carry (C_i) of a 4-bit CLA Adder is given by the Boolean equation 1 below.

$$C_{i+1} = G_i + P_iC_i \quad \text{--- (equation 1)}$$

Where carry generation $G_i = A_iB_i$, carry propagation $P_i = (A_i \oplus B_i)$. A_i and B_i are the i^{th} input bit of the CLA adder. The limit of the integer **i** is $0 \leq i \leq 3$

[Q1]

(i) Draw a block diagram for 4-bit CLA Adder using 2-bit Full Adders and a CLA Logic. Clearly, show the output (sum) of the CLA Adder.

[05 Marks]

(ii) If S_i is i^{th} output (sum) ($S_i = P_i \oplus C_i$), draw a circuit diagram for the 4-bit CLA Adder where expressed by the Boolean equation 1 above using 2-inputs logic gates. Clearly show the inputs (A_i, B_i, and C_i), and outputs (C_{i+1}, P_i, G_i, and S_i).

[05 Marks]

(iii) Draw a circuit diagram for the CLA Logic denoted by G_0, G_1, G_2, G_3, P_0, P_1, P_2, P_3, C_0, C_1, C_2, C_3, and C_4 using logic gates.

[10 Marks]

(iv) Draw a complete block diagram for 4-bit CLA Adder using CLA Logic and suitable components. Clearly, show the inputs and the outputs.

[05 Marks]

[Q2]

(i) Write behavioural VHDL code for the CLA Logic drawn in Q1 (iii).

[05 Marks]

(ii) Write structural VHDL code for the 4-bit CLA Adder drawn in Q1 (iv) using behavioural model of the CLA Logic in Q2 (i).

[10 Marks]

[Q3] The 4-bit NPU depicts in Figure Q3 to perform the arithmetic operations listed in Table Q3. (Assume that the Initial carry is zero, and the final carry can be ignored.)

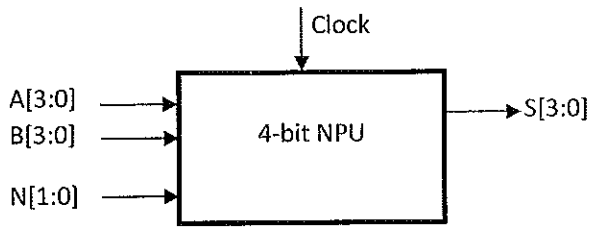


Figure Q3: Block diagram of the 4-bit NPU

(i) Draw a circuit diagram for the 4-bit NPU, using the *4-bit CLA Adder* along with the other suitable components.

[10 Marks]

(ii) Write a behavioral VHDL code for the circuit diagram for the above question Q3 (i).

[10 Marks]

Table Q3: Arithmetic Operations

N	Function	Output (S)
00	Addition	**A+B**
01	Subtraction	**A-B**
10	Multiplication	**A*B**

SECTION B: Answer any TWO questions. [40 Marks]

[Q4]

(i) Draw an Algorithmic State Machine (ASM) chart for the subunit which produces the timing diagram shown in Figure Q4. [10 Marks]

(ii) Construct the circuit which produces the following timing diagram shown in Figure Q4 by using digital logic gates and a D-Flip Flop. [10 Marks]

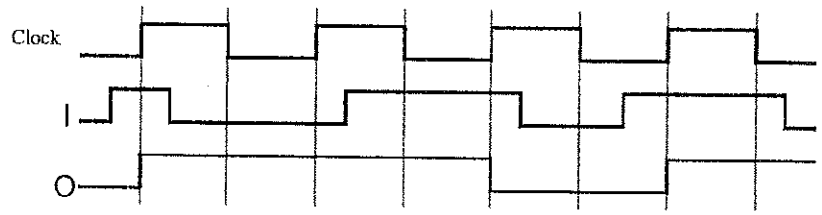


Figure Q4: Timing Diagram (I: Input, O: Output)

[Q5] Design a Finite State Machine (FSM) that counts the decimal sequence 2, 1, 0, 3, 2, 1, 0, 3,... by using D-flip-flops and the digital logic gates. The counting starts when the control input C=1 and continues until C=0 and remains in the same state when C=0. Whenever C=1 then counter will continue its counting from the state where it has stopped.

(i) Draw a state diagram for the FSM. [05 Marks]
(ii) Draw a state transition table for FSM. [05 Marks]
(iii) Draw a digital circuit diagram for the FSM [05 Marks]
(iv) Write behavioral VHDL program for the FSM. [05 Marks]

[Q6] Figure Q6 depicts a circuit which has inputs (X, CLK) and output ($\bar{Q}_A$).

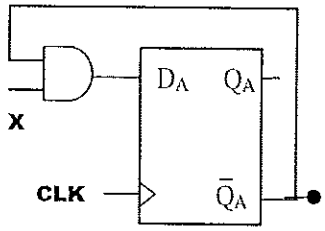


Figure Q6

(a) Draw a timing diagram for the circuit showing X, D_A, $\bar{Q}_A$ and CLK. Assume that X is initially 1 and after X becomes 0 for 10 ns, then X becomes 1 for 200 ns. Assume that the clock frequency (f_{CLK}) is 20MHz. [08 Marks]

(b) Modify the timing diagram drawn in above Q6 (a) for the following parameters

AND Gate propagation delay is 10ns and D flip-flop has HIGH to LOW propagation delay (t_{PHL}) = 100ns, LOW to HIGH propagation delay (t_{PLH}) = 60ns, set-up time (t_{SU}) = 20ns, hold time (t_H) = 0ns, maximum clock frequency (f_{CLKMAX}) = 20MHz. [12 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;
```