00177

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

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	: 2019/20
Date	: 21st January 2021
Time	: 1330-1630hrs
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 to each question should commence from a new page.
5. Refer to the Annexure of the VHDL syntax given on 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]

Digital Pattern Recognizer (DPR)

The following description is about the Digital Pattern Recognizer (DPR), which can be used for comparing two bit-streams where one bit-stream is user data, and the other bit-stream is reference (template) data. Your task is to analyze the following specifications and design the DPR using digital electronic components and logic gates. Figure Q1 depicts the typical view of the DPR.

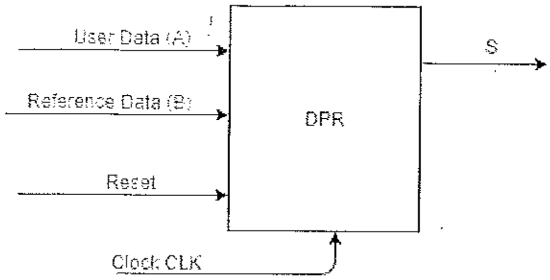

Figure Q1: Typical view of the DPR

Specifications of the DPR

1. The two bit-streams user data and the reference data serially feed to the DPR.
2. User data and reference data are considered equal if three (3) consecutive bit pairs are equal.
3. If two bit-streams are equal for three (3) consecutive clocks, then the output s = user data.
4. If two bit-streams are NOT equal, then the output s remains as it is (no change).

[Q1]

(i) Draw a truth table for 1-bit magnitude comparator and draw a circuit diagram using logic gates.

[5 Marks]

(ii) Write the structural and behavioural VHDL code for 1-bit magnitude comparator.

[5 Marks]

[Q2]

(i) Draw a state diagram for counter that can count up to 4 and produce output when it reaches 4 with a reset option.

[5 Marks]

(ii) Draw a circuit diagram for the counter to represent the state diagram in Q2.i

[5 Marks]

(iii) Write the structural or behavioural VHDL code for the counter.

[5 Marks]

[Q3]

(i) Draw a state diagram for the DPR. [05 Marks]

(ii) Draw a circuit diagram by integrating above 1-bit magnitude comparator and the counter with other digital logic components and gates to perform the DPR operation. Clearly explain the working procedure of the DPR, indicating the internal functionality of the pattern recognizer. You must show clearly how the pattern comparison is made according to the given specifications and show the data paths.

[10 Marks]

(iii) Write complete VHDL code for DPR. Clearly show the port mappings of each entity inside DPR.

[10 Marks]

[Q4]

(i) Briefly explain how you perform functional, and structural testing with the fault model for DPR. [6 Marks]

(ii) List commonly used algorithms for testing digital circuits.

[4 Marks]

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

[Q5]

(i) Draw an Algorithmic State Machine (ASM) chart for the state diagram shown in Figure Q5. [10 Marks]

(ii) Construct the circuit for the state diagram shown in Figure Q5 using digital logic gates and D-Flip Flops. [10 Marks]

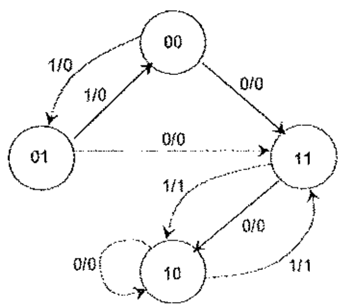

Figure Q5: State Diagram

[Q6] Design a Finite State Machine (FSM) that detect the binary sequence 1011,... by using D-flip-flops and the digital logic gates. The FSM starts when the control input C=1 continues until C=0 and remains in the same state when C=0. Whenever C=1 then FSM will continue its operation 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]

[Q7] Figure Q7 depicts a circuit which has inputs (X, CLK) and output ($\bar{Q}_A$) as below.

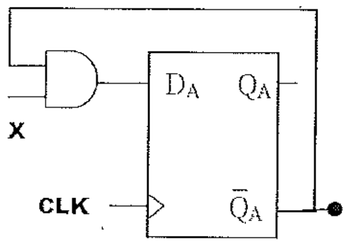

Figure Q7: A Circuit Diagram

(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 180 ns. Assume that the clock frequency (f_{CLK}) is 10MHz. [08 Marks]

(b) Modify the timing diagram drawn in above Q7 (a) for the following parameters

AND Gate propagation delay is 10ns and D flip-flop has HIGH to LOW propagation delay (t_{PHL}) = 80ns, 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;
```