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

Study Programme	: Bachelor of Science Honours in Engineering
Name of the Examination	: Final Examination
Course Code and Title	**: EEX5351 Digital Electronic Systems**
Academic Year	: 2023/24
Date	: 19/03/2025
Time	: 13:30-16:30hrs
Duration	: **3 hours**

General Instructions

1. Read all instructions carefully before answering the questions.

2. This question paper contains one (1) question in SECTION A and two (2) questions in SECTION B on **five (5)** pages.

3. Answer all questions in SECTION A. [60 Marks], and SECTION B. [40 Marks].

4. Answer for 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, if any.

7. This is a Closed Book Test (CBT).

8. Answers should be in clear handwriting.

9. Do not use Red colour pen, and clearly state your assumptions if any

Section A: Answer ALL the questions [60 Marks]

Tunneled Car Wash Conveyor Controller (TCWCC)

The following description gives a tunnel-style car wash system, which operates in multiple sequential sections (pre-soak, scrubbing, rinsing, and drying) using an automated conveyor belt, as depicted in Figure 01. The system is designed for high throughput, efficiently handling multiple vehicles to minimise wait times. Your task is to analyse the given description and design the Tunnelled Car Wash Conveyor Controller (**TCWCC**) unit using digital electronic components.

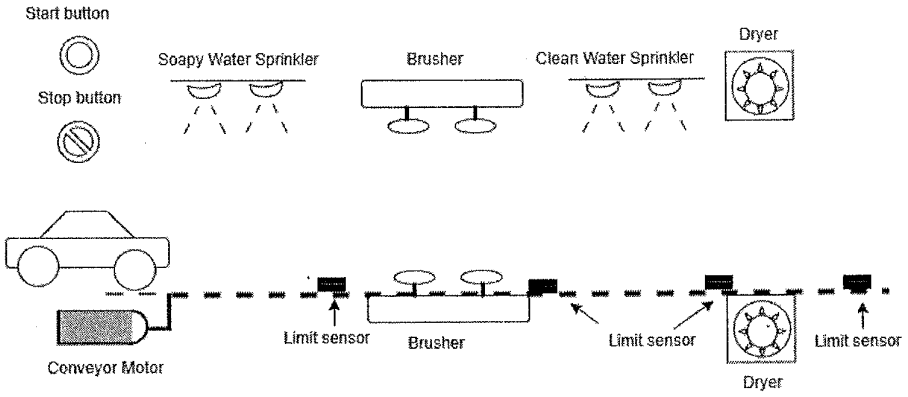


Figure 01: Tunnelled car washing system

Design specification of the control unit,

- When a car arrives on the conveyor, pressing the start button activates the conveyor motor, moving the vehicle to the next section.
- Limit switches are installed at the end of each section to stop the conveyor, triggering the activation of the soap water sprinkler, cotton brushes, clean water sprinkler, and dryer at their respective sections.
- Each section remains active for 20 clock counts before deactivating and allowing the conveyor to move forward to the next section.
- After the drying stage section is completed, the conveyor moves to the exit gate and stops.
- The Stop button pauses the conveyor and the ongoing section operation, while the Start button resumes the process from where it was paused.
- *Reset switch* resets the TCWCC.

[Q1]

(i) Identify the inputs and outputs of the **TCWCC** and draw a block diagram to show how the **TCWCC** is integrated with sensors, actuators, and other subunits.

[05 marks]

(ii) Identify the special digital electronic functional units/circuits and other necessary components in the **TCWCC** and draw a block diagram to show their interconnections. Clearly explain the working procedure of the **TCWCC**, indicating the internal functionality of the controller. [14 marks]

(iii) Draw the Moore-type state transition diagram for the FSM unit in the **TCWCC**. [07 marks]

(iv) Draw the Timing diagram for Q1) (iii) control unit. [07 marks]

(v) Redraw the Q1(iii) FSM without the equivalent state. (*Hint: use implication chart*). [05 marks]

(vi) Draw the circuit diagram using D Flip-flops and other necessary components. [07 marks]

(vii) Write the complete VHDL code for the design **TCWCC** and include comments where necessary. [10 marks]

(viii) Write a VHDL testbench code to illustrate how to perform a functional-level fault model for the **TCWCC**. Generate test cases/sets for testbench by indicating the purpose of each test. [05 marks]

Section B: answer ALL the questions. [40 Marks]

[Q2]

The positive clock edge-triggered sequence detector, as shown in Figure 02, is designed to detect specific sequences based on the given conditions. It operates with a synchronised reset, where input **A** serves as the sequence input and output **B** provides the detection result. When the **Reset** signal is high, output **B** is set to high, initialising the detector. When **Reset** is low, the detector monitors the last three bits of the input sequence. If the sequence **"011"** is detected, output **B** is set to low. However, if the sequence **"110"** is detected, output **B** is locked to **1** permanently. To simplify the circuit and minimise delays, it is assumed that **A** remains high when **Reset** is high.

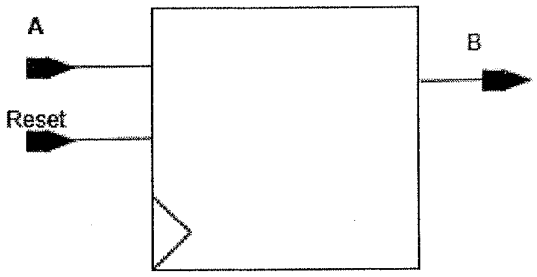


Figure 02: Sequence detection circuit.

(i) Without equivalent states, draw Moore type FSM that produces the output for the sequence detector. [10 marks]

(ii) Draw the Q2(i) sequence detector circuit only using D - flip flops and 2:1 Multiplexers. [10 marks]

[Q3]

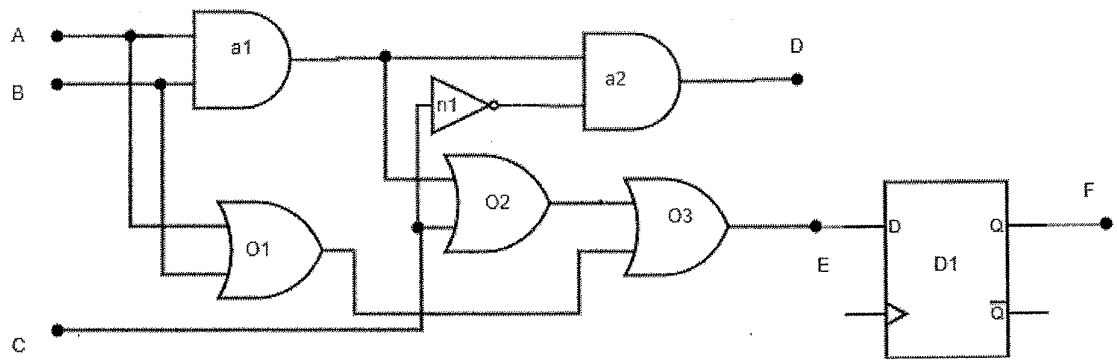


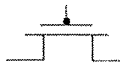
Figure 03: Combinational logic circuit

(For *all gates* T_{PHL} *=5ns and* T_{PLH}*=3ns; D-Flip Flop timing specification are :* T_{PHL}*=25,* T_{PLH}*=30, F=25MHz,* T_{su}*=20 and* T_h*=0*)

(i) Calculate the Controllability and Observability of Figure 03 circuit for input A,B,C and output D,E networks.

[4 marks]

(ii) Draw the CMOS logic gate design for input A,B,C and output D networks using NMOS pulldown and PMOS pullup networks.

(Hint: Use PMOS  and NMOS symbols)

[4 marks]

(iii) Calculate the propagation delay for the input **A,B,C** and output **F**. [5 marks]

(iv) Draw a timing diagram to illustrate the hazard condition in input **A,B,C** and output **E** network and briefly describe how to eliminate the hazard condition.

[7 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;
        …);
ENDCOMPONENT [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;
```