

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



0084

Study Programme : Bachelor of Science Honours in Engineering
Name of the Examination : Final Examination
Course Code and Title : EEX5351 Digital Electronic Systems
Academic Year : 2024/25
Date : 31st January 2026
Time : 0930-1230h
Duration : 3 hours

General instructions

1. Read all instructions carefully before answering the questions.
2. This question paper consists of **Four (4)** questions in **Five (5)** pages.
3. Answer all questions.
4. Answer for each question should commence from a new page.
5. This is a Closed Book Test (CBT).
6. Refer to the Annexure of the VHDL syntax given on page five (5) to write VHDL code, if any.
7. Show all necessary steps for answers.
8. Answers should be in clear handwriting.
9. Do not use Red colour pen, and clearly state your assumptions if any.

Double-Deck Parking Lift Controller (DDPLC)

The following description gives a Double-Deck Parking Lift Controller (DDPLC), it manages a semi-automatic parking system, as illustrated in Figure Q1. This system utilizes a parking lift to vertically stack one vehicle above another between the ground floor and the basement, lifting both vehicles together to allow direct access to the desired deck from the ground level. This creates two dependent parking spaces arranged one above the other, which is ideal for permanent parking facilities. The system's unique hydraulically driven design offers faster lifting speeds, quieter operation, and superior safety when compared to motorized alternatives. Your task involves analyzing this description and designing the DDPLC unit using digital electronic components.

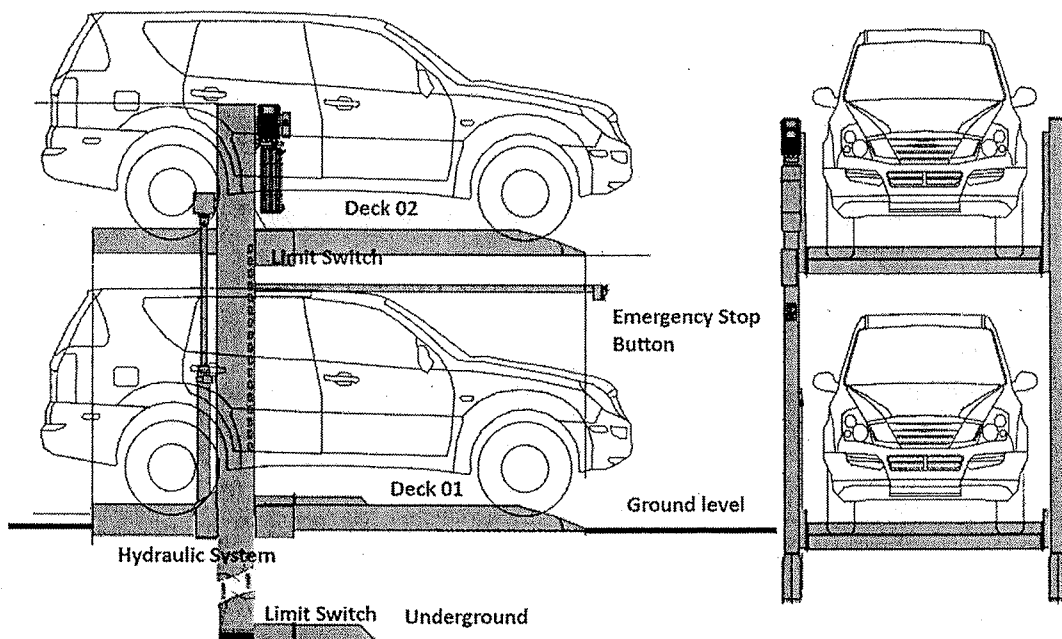


Figure Q1: Double-Deck Home Garage Parking Hydraulic Lift.

Design specification of the control system,

- Upon pressing the Store button, the system initiates a sequence to locate an empty deck and move it to the ground floor. The process involves checking deck occupancy via the deck empty sensors. If, however, both decks are occupied, the system will activate the parking full signal instead.
- The system features two separate Retrieve buttons, one designated for each deck. When a Retrieve button is pressed, the corresponding deck will move to the ground floor, but only if its occupancy sensor confirms it is not empty.

- To ensure precise positioning, limit switches are installed at each designated level. Their function is to automatically stop the lift movement once the platform reaches the correct alignment with either the ground floor or the basement.
- The system is equipped with an Emergency Stop button. When activated, this feature causes the lift to stop its operation promptly.
- *Reset switch* resets the **DDPLC**

Answer the following questions.

- Identify the inputs and outputs of the **DDPLC** and draw a block diagram to show how the **DDPLC** is integrated with sensors, actuators, and other subunits. (05 marks)
- Identify the internal functional units of the **DDPLC** and draw its internal block diagram. Then, clearly explain the controller's working procedure, indicating the functionality of each unit. (Hint: using timing diagrams, truth tables, etc...). (16 marks)
- Draw the mealy-type state transition diagram for the FSMs in the **DDPLC**. (07 marks)
- Draw the circuit diagram using D Flip-flops and other necessary components. (07 marks)
- Write the complete high-level abstract VHDL program for the **DDPLC** and include comments where necessary. (10 marks)
- Write a VHDL testbench code for your **DDPLC** design. Generate test cases/sets for testbench by indicating the purpose of each test. (05 marks)

Question 02

- Design a modulo-7 **synchronous** counter counts from 0 to 6 using D flip flops. The counter has two inputs 'Count' and 'Clear' along with three outputs S0 – S2 as shown in Figure 2. Counter should continue the count when Count signal is high and reset to "0" after count reaches "7". When Clear signal is low (**synchronous** clear), the counter should reset to "0". (9 marks)

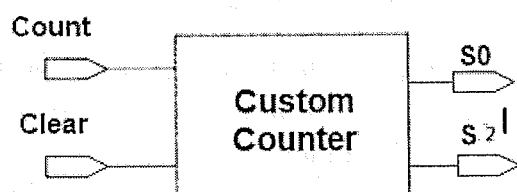


Figure Q2: 3 Bit- Up Counter

- b. Hence, derive modulo-10 up counter using 02 (two) modulo-7 counters and incorporate 8:1 multiplexers for necessary logic combinations and draw the circuit diagram. (11 marks)

Question 03

Figure Q3 depicts a combinational circuit that has input (A,B,C) and output (D,E) as below.

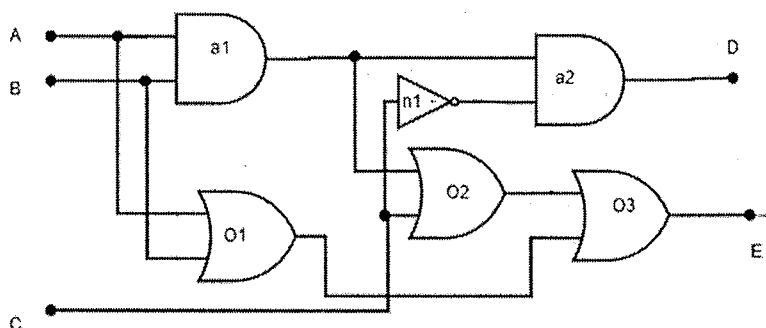


Figure Q3: Combinational Circuit

- Re-draw the simplified "Figure Q3" circuit using 2 input 1 output logic gates. (4 marks)
- Draw the path Sensitization table for "Question 03,a." circuit and identify the complete test set $TS=\{\}$ (8 marks)
- Calculate the Controllability and Observability of "Question 03,a." circuit for input A,B,C and output D,E networks. (8 marks)

Question 04

Figure Q4 depicts a circuit that has input (Y, clk) and output ($\bar{Q}$) as below. (10marks)

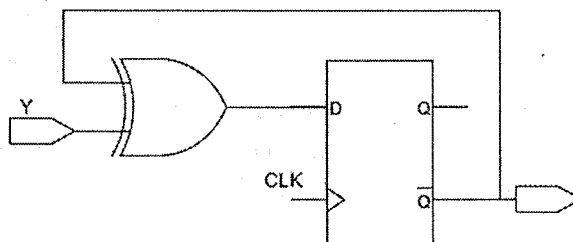


Figure Q4: Sequential Logic Circuit

Draw a timing diagram for the circuit showing Y, D, $\bar{Q}$ and clk. Assume that initially Y is 1, D is 1 and clk is 0 and after Y become 0 for 10 ns (before the clk edge), then Y becomes 1 for 150 ns. Assume that the clock frequency (f_{CLK}) is 10 MHz, XOR gate propagation delay is 10ns. The D flip-flop has High to Low propagation delay (t_{PHL})=30ns, low to high propagation delay (t_{PLH})=25ns, set-up time (t_{su})=20ns, hold time (t_h)=0ns and maximum clock frequency (f_{MAX})=25MHz.

All Rights Reserved

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 (*formall=>actuell*,
 formall=> actuell,
 formall=> actuell, ...);
- ⊗ [*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.inclusion*;
- ⊗ WAIT FOR *time_expression*;
- ⊗ WAIT ON *signal1*, *signal2*, ...;
- ⊗ WAIT UNTIL *condition*;
- ⊗ WHILE *condition* LOOP
 Sequential statements
 END LOOP;