Roll No. 2239114 Total No. of Pages : 02

Total No. of Questions : 09

B.Tech.(ECE) (Sem.–3)

DIGITAL SYSTEM DESIGN

Subject Code : BTEC-302-18

M.Code : 76445

Date of Examination: 12-12-2023

Time : 3 Hrs. Max. Marks : 60

INSTRUCTIONS TO CANDIDATES :

1. SECTION-A is COMPULSORY consisting of TEN questions carrying TWO marks each.
2. SECTION-B contains FIVE questions carrying FIVE marks each and students have to attempt any FOUR questions.
3. SECTION-C contains THREE questions carrying TEN marks each and students have to attempt any TWO questions.

SECTION-A

1. ***Write briefly :***

   a) Why NAND and NOR are called universal gates?

   b) Minimize using Boolean algebra AB'C'D + AC + BC'D + D.

   c) Define propagation delay in logic families.

   d) Implement 4:1 multiplexer using 2:1 multiplexer.

   e) Convert the following expressions to canonical SOP: A'B'C+BC

   f) Prove that AB+A'C+BC = AB+A'C.

   g) Discuss PAL and PLA circuits.

   h) Why priority encoders are preferred?

   i) Discuss the working of BCD adder circuit.

   j) Convert $(0010\ 0110)_{BCD}$ to Excess-3 code.

SECTION-B

2. Differentiate between CPLD and FPGA.
3. Name various modelling styles in VHDL, give example of Structural style of modelling.
4. Describe the working of 2:4 decoder. Can decoder be used as a demultiplexer?
5. Design 2bit magnitude comparator.
6. Define Race Around Condition. How Master slave flip flop is used to avoid Race Around Condition?

SECTION-C

7. Minimize using K Map technique

   $F(w,x,y,z) = \Sigma m\ (0,1,2,7,9,11,14,15) + d(3,5,12,13)$.

8. Describe various characteristics of D/A converter. Also, explain the working of weighted resistor D/A convertor.
9. Design Binary to Gray code convertor.

NOTE : Disclosure of Identity by writing Mobile No. or Making of passing request on any page of Answer Sheet will lead to UMC against the Student.

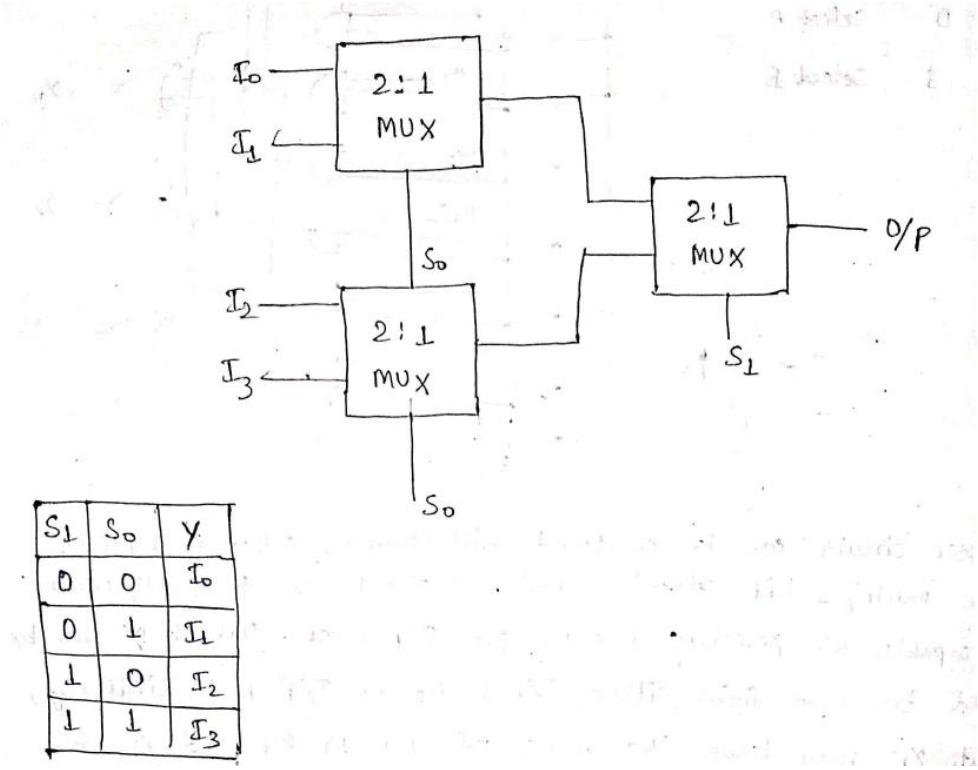
DSD solution PTU paper 2023

Section A

1 a) NAND and NOR gates are called universal gates because they can be used to generate any logical Boolean expression, and can be combined to produce all other basic gates

e) Propagation delay is the time it takes for a logic gate's output to reflect a change in the input. It's also known as gate delay

d). 4:1 using 2:1 mux



S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

g) Programmable Logic Array (PLA) and Programmable Array Logic (PAL) are both types of programmable logic devices (PLDs) that are used to implement logic functions in digital circuits. The main difference between the two is their architecture:

PLA

Has a programmable AND array followed by a programmable OR array. PLAs are more versatile and can support complex logic functions. They are also expensive and work at high speeds.

PAL

Has a programmable AND array followed by a fixed OR array. PALs are simpler, smaller, and faster than PLAs. They are also more budget-friendly and work at low speeds.
When deciding between PLA and PAL, designers must consider factors such as cost, performance requirements, design complexity, and required flexibility.

SECTION-A

2. b)

$$Y = AB'C'D + AC + BC'D + D$$
$$= AB'C'D + AC + D(BC' + 1)$$
$$= AB'C'D + AC + D(\underbrace{B+1}_{1})(\underbrace{C'+1}_{1})$$
$$= AB'C'D + AC + D$$
$$= D[AB'C' + 1] + AC.$$
$$= D(\underbrace{A+1)(B'+1)(C'+1}_{1}) + AC$$
$$= D + AC$$
$$= AC + D \quad - \text{Ans.}$$

2c)
$$Y = A'B'C + BC$$
$$= A'B'C + BC(A + A')$$
$$= A'B'C + ABC + A'BC$$

2. f)
$$AB + A'C + BC = AB + A'C.$$

$$AB + A'C + BC.$$

$$AB + A'C.$$
$$(AB + A')(AB + C).$$
$$(\underbrace{A + A'}_{1})(B + A')(AB + C)$$
$$(B + A')(AB + C)$$
$$= AB + BC + A'.\underbrace{AB}_{=0} + A'C.$$
$$= AB + BC + A'C.$$
$$= \text{LHS.}$$

Proved.

h) Priority encoders are preferred over simple encoders because they can handle multiple inputs that are active at the same time.

i) A BCD (binary coded decimal) adder works by adding two numbers and correcting the result if it is greater than 9:

1. **Add the numbers**: Use a 4-bit binary adder to add the binary representations of the two numbers.

2. Check the sum: Use a logic circuit to determine if the sum is greater than 9 or if the carry is 1.

3. Correct the sum: If the sum is greater than 9, add 6 (0110 in binary) to the result

6	0 1 1 0	←	BCD for 6
+ 8	1 0 0 0	←	BCD for 8
14	1 1 1 0	←	Invalid BCD number

The sum 1 1 1 0 is an invalid BCD number. This has occurred because the sum of the two digits exceeds 9. Whenever this occurs the sum has to be corrected by the addition of six (0110) in the invalid BCD number, as shown below

6	0 1 1 0	←	BCD for 6
+ 8	1 0 0 0	←	BCD for 8
14	1 1 1 0	←	Invalid BCD number
	+ 0 1 1 0	←	Add 6 for correction
0 0 0 1	0 1 0 0		
		←	BCD for 14
1	4		

After addition of 6 carry is produced into the second decimal position.

Sum equals 9 or less with carry 1

Let us consider addition of 8 and 9 in BCD

8	1 0 0 0	←	BCD for 8
+ 9	1 0 0 1	←	BCD for 9
17	0 0 0 1 0 0 0 1	←	Incorrect BCD result

In this, case, result (0001 0001) is valid BCD number, but it is incorrect. To get the correct BCD result correction factor of 6 has to be added to the least significant digit sum, as shown below

8	1 0 0 0	←	BCD for 8
+ 9	1 0 0 1	←	BCD for 9
17	0 0 0 1 0 0 0 1	←	Incorrect BCD result
	+ 0 0 0 0 0 1 1 0	←	Add 6 for correction
	0 0 0 1 0 1 1 1	←	BCD for 17

Section B

2. CPLD and FPGA are programmable logic devices. A Complex Programmable Logic Device (CPLD) is a type of programmable logic device used to create digital circuits. FPGA is an abbreviation for Field Programmable Gate Array, which is a type of programmable logic device used to build digital circuits.

CPLD?

CPLD is an abbreviation for Complex Programmable Logic Device. It is a digital programmable logic device with a programmable logic array (PLA) and a programmable interconnect. CPLDs have been developed to perform a wide range of logic functions and are often used in digital circuits to handle data processing, control, and communication.In a CPLD, the programmable interconnect connects the output of one logic cell to the input of another. The interconnect can be designed to generate a wide range of logic connections, allowing complicated logic functions to be created. The interconnect is often made up of a matrix of programmable switches that connect the logic cells' inputs and outputs.

One of the benefits of CPLDs is their ease of use. Because CPLDs typically have a specified link structure, they can be designed and implemented faster than FPGAs. (Field Programmable Gate Arrays). They are also less expensive than FPGAs, making them an attractive option for low- to medium-complexity systems.

FPGA?

FPGA is an abbreviation for Field Programmable Gate Array, which is a type of programmable logic device used to build digital circuits. FPGAs are constructed from a grid of programmable logic blocks that can be coupled to perform sophisticated logic tasks. FPGAs feature more logic blocks than CPLDs and are therefore better suited for implementing larger and more complicated logic functions.

For implementing simpler logic operations, CPLDs are generally faster and more power-efficient than FPGAs. FPGAs are more flexible and provide greater performance and scalability when it comes to implementing larger and more complicated logic operations. Furthermore, because of their bigger size and better level of integration, FPGAs are often more expensive than CPLDs.The logic cell, also known as a configurable logic block, is the fundamental building block of an FPGA. A CLB is made up of a look-up table (LUT), a flip-flop, and a programmable interconnect. The LUT is a memory block that can be designed to perform a specific logic function. The flip-flop stores the logic function's output, and the programmable interconnect connects the output of one CLB to the input of another.

FPGAs often have a large number of CLBs that can be configured in a two- dimensional array or matrix. An FPGA's connection topology is far more versatile than that of a CPLD, allowing for more complicated designs. The interconnect is often made up of programmable switches that may be set to connect the CLBs' inputs and outputs.

Difference between CPLD and FPGA

The following table highlights the major differences between CPLD and FPGA –

Characteristics	CPLD	FPGA
Logic Cells	It has a small number of logic cells.	It has a large number of logic cells.
Interconnect Structure	It has a fixed interconnect structure.	It has a flexible interconnect structure.
Flexibility	Less Flexible	More Flexible
Cost	Low Cost	High Cost
Power Consumption	Less power consumption	Higher power consumption
Reconfigurability	Less reconfigurability	More reconfigurability
Density	Low to medium	Medium to high
Flip-flop ratios	Less flip-flop ratio	More flip-flop ratio
Applications	Best for simple applications	Best for complex applications

3. An architecture can be written in one of three basic coding styles:(1) Dataflow (2) Behavioral (3) Structural..The difference between these styles is based on the type of concurrent statements used:

- A dataflow architecture uses only concurrent signal assignment statements.
- A behavioral architecture uses only process statements.
- A structural architecture uses only component instantiation statements.

Dataflow Style of Modelling:

1. Dataflow style describes a system in terms of how data flows through the system. Data dependencies in the description match those in a typical hardware implementation.
2. A dataflow description directly implies a corresponding gate-level implementation.
3. Dataflow descriptions consist of one or more concurrent signal assignment statements.

Behavioral Style of Modelling:

1. A behavioral description describes a system’s behavior or function in an algorithmic fashion.
2. Behavioral style is the most abstract style. The description is abstract in the sense that it does not directly imply a particular gate-level implementation.
3. Behavioral style consists of one or more process statements. Each process statement is a single concurrent statement that itself contains one or more sequential statements.
4. Sequential statements are executed sequentially by a simulator, the same as the execution of sequential statements in a conventional programming language.

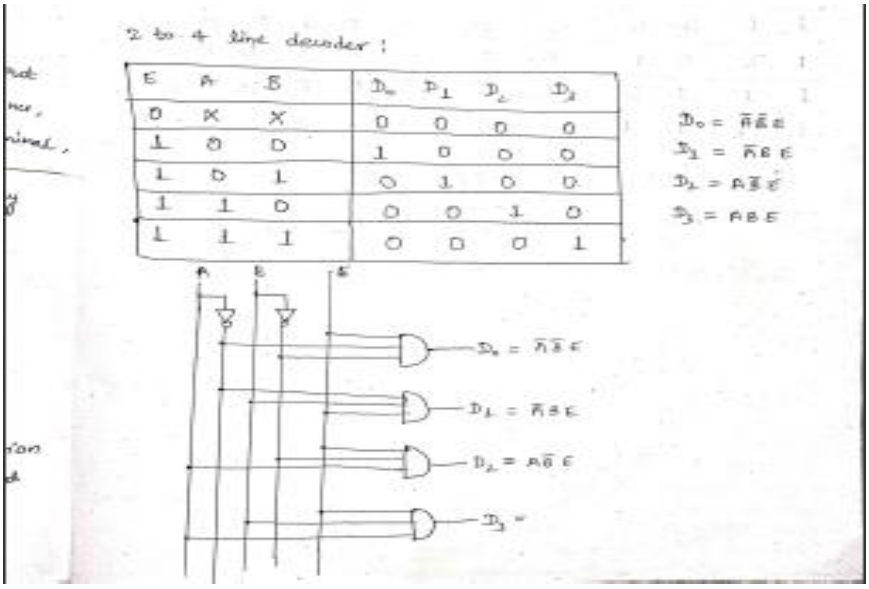
Structural Style of Modelling:

1. In structural style of modelling, an entity is described as a set of interconnected components.
2. The top-level design entity's architecture describes the interconnection of lower-level design entities. Each lower-level design entity can, in turn, be described as an interconnection of design entities at the next-lower level, and so on.
3. Structural style is most useful and efficient when a complex system is described as an interconnection of moderately complex design entities. This approach allows each design entity to be independently designed and verified before being used in the higher-level description.

E.g. Structural style half-adder description.

```
1.  library ieee;
2.  use ieee.std_logic_1164.all;
3.
4.  entity half_adder is                    -- Entity declaration for half
    adder
5.    port (a, b: in std_logic;
6.      sum, carry_out: out std_logic);
7.  end half_adder;
8.
9.  architecture structure of half_adder is     -- Architecture body for half
    adder
10.
11.   component xor_gate                   -- xor component declaration
12.     port (i1, i2: in std_logic;
13.       o1: out std_logic);
14.   end component;
15.
16.   component and_gate                   -- and component declaration
17.     port (i1, i2: in std_logic;
18.       o1: out std_logic);
19.   end component;
20.
21.  begin
22.     u1: xor_gate port map (i1 => a, i2 => b, o1 => sum);
23.     u2: and_gate port map (i1 => a, i2 => b, o1 => carry_out);
24.  -- We can also use Positional Association
25.  --     => u1: xor_gate port map (a, b, sum);
26.  --     => u2: and_gate port map (a, b, carry_out);
27.  end structure;
```

4.



5. 2-Bit Magnitude Comparator

A digital combinational circuit used to compare the magnitudes of two 2-bit binary numbers : **2-bit magnitude comparator**.

Hence, the 2-bit magnitude comparator compares the values represented by two 2-bit binary numbers and then generates an output that indicates whether one number is equal to or greater than or less than the other.

block diagram of a typical 2-bit magnitude comparator is shown in the following
re –

–

Inputs				Outputs		
A_1	A_0	B_1	B_0	L (A < B)	E (A = B)	G (A > B)
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

Case 1: A = B

The comparator produces an output A = B which is E, if $A_0 = B_0$ and $A_1 = B_1$. Therefore, the Boolean expression for the output E will be,

$$\mathbf{E = (A_0 \odot B_0)\,(A_1 \odot B_1)}$$

Case 2: A < B

The comparator produces an output A < B which is L, if

- $A_1 = 0$ and $B_1 = 1$, OR
- $A_1 = B_1$ and $A_0 = 0$ and $B_0 = 1$.

From these statements, we can write the Boolean expression for the output L as follows –

$$\mathbf{L = \overline{A_1}\,B_1 + (A_1 \odot B_1)\,\overline{A_0}\,B_0}$$

Case 3: A > B

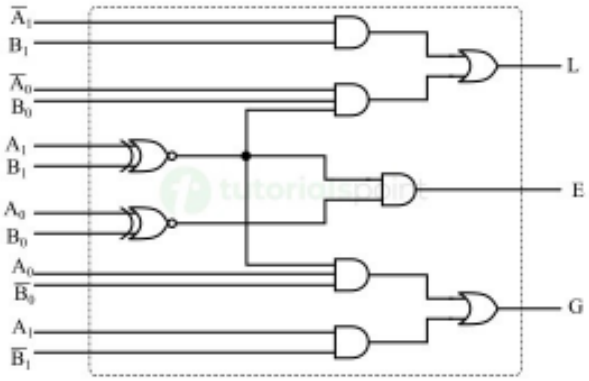
The output of the comparator will be A > B i.e., G, if

- $A_1 = 1$ and $B_1 = 0$, OR
- $A_1 = B_1$ and $A_0 = 1$ and $B_0 = 0$.

From these statements, the Boolean expression for the output G will be,

$$\mathbf{G = A_1\,\overline{B_1} + (A_1 \odot B_1)\,A_0\,\overline{B_0}}$$

The following figure shows the logic circuit diagram of the 2-bit magnitude comparator –



6. The problem of the race-around condition does not exist in the flip-flops where the inputs do not change during the presence of clock pulse. But, in the case of JK flip-flops, the inputs change during the clock pulse due to the feedback path present between inputs and outputs. Hence, in JK flip-flops, the race around condition is a major problem.

The problem of race-around condition and the uncertainty of output can be avoided by increasing the delay of the flip-flops. For that, the delay of the flip-flops must be greater than the duration of the clock signal, i.e. $\Delta t > T$. In another way, the duration of the applied clock signal (T) must be reduced to make it less than the delay of the flip-flops (Δt).

However, the increase in the delay of the flip-flops is not a good practice because it decreases the speed of the system. On the other hand, it is also quite difficult to decrease the duration of the clock pulse (T) beyond the delay of the flip-flops (Δt). This is because, the delay of the JK flip-flops (Δt) is of the order of nanoseconds.

Hence, the most practical way to solve the problem of race-around condition in JK flip-flops is to use the JK flip-flops in the **Master and Slave Mode**. In the master-slave mode of JK flip-flops, two JK flip-flops are cascaded.

This is all about the race-around condition and its remedies in JK flip-flops.

SECTION C

7. Digital-to-analog converters (DACs) have several characteristics, including:

Resolution

The DAC's precision is determined by the number of bits it has. Increasing the number of bits increases the signal's gradation scale.

Reference voltage

The DAC's maximum output voltage is set by the reference voltage, which can be generated within the DAC or applied externally.

Settling time

The time it takes for the output to reach and remain within a specified error band around the final output voltage.

Image signals

An undesirable characteristic of DACs is the appearance of image signals within each Nyquist zone of the output.

Quantization

Quantization is a natural occurrence in DAC operation, but it can be minimized by using high-resolution devices.

- **R2R DAC chips**

  A type of DAC chipset that uses a network of resistors to convert digital signals to analog signals.

- **Audio signal processing**

  DACs are often incorporated into audio codecs, which include signal processing features.
  Other characteristics of DACs include: Speed, Dynamic range, SFDR, ENOB, and Glitch-impulse area

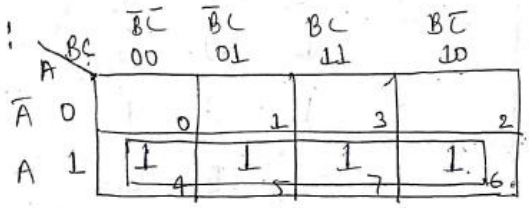
9.

(Q-1) Convert three bit binary to gray code.

Soln:

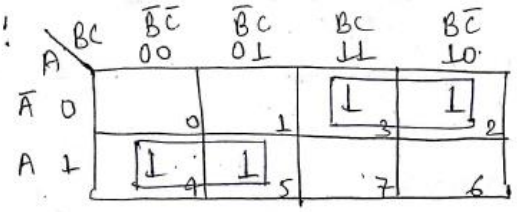
I/P			O/P		
A	B	C	X	Y	Z
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

For – X :



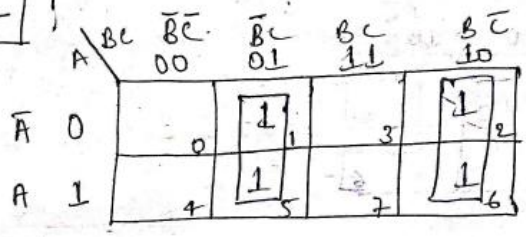
$X = A$

For – Y :



$Y = A\bar{B} + \bar{A}B$

For – Z :



$Z = \bar{B}C + B\bar{C}$