|  |  |  |
| --- | --- | --- |
|  | **BAHRIA UNIVERSITY (Karachi Campus)**  **Department of Software Engineering**  **Final Term Assessment – Spring 2020**  (COURSE TITLE: **Digital Design** COURSE CODE: **CEN-122**) | |
| Class : **BSE-2** **[A]** | | Session : **Morning** |
| Course Instructor : **Engr. Mahawish Fatima** | | Time Allowed : **8 Hours** |
| Date : **3rd July, 2020** | | Max. Marks : **50** |
| Time : **8 a.m - 4 p.m** | |  |
|  | |  |
| Student’s Name: \_\_\_Muhammad Umer Adeem\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | Reg. # : \_\_\_\_65158\_\_\_\_\_ |
| Note:   * All question should be answered in the same file using black font color. * Copying answers from each other is considered as cheating, and such identified cases will be dealt with severely that can also lead to **cancellation of papers**. * Save this file as “**Name-Registration Number**” | | |

|  |  |
| --- | --- |
| Student’ Name: Muhammad\_Umer\_Adeem  Registration #: \_\_65158\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Enrollment #: \_\_\_02-131192-044\_\_\_\_\_\_\_  Semester: \_\_BSE-2-A\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |

**Q: 1 [05 Marks] [CLO 1]**

**Define the following terms.**

**a) Encoder and Decoder. (1.0)**

**Encoder:**

A combinational circuit which is used to convert binary information which are 2^N inputs into N outputs, which represents N bit code for input is called an encoder.

Encoder

n input lines m output lines

**Decoder:**

A decoder is an opposite of the encoder. The decoder converts N inputs into 2^N inputs.

Decoder

n input lines m output lines

**b) Multiplexer and De-multiplexer. (1.0)**

**Multiplexer**

A Multiplexer is a circuit which allows data from multiple sources to be transferred via a single transmission line towards a common source. A multiplexer has multiple inputs but single output. A multiplexer also contains select input which selects one input which will be the output of the multiplexer.

**De-multiplexer**

A De-multiplexer is the opposite of Multiplexer. It is also known as data distributor. It distributes the provided input to the required output. The required output line is selected by the select inputs.

**c) Excitation table and characteristic table. (1.0)**

**Excitation Table**

An excitation table is a truth table which is used to find the inputs which will be required to obtain a particular output of the flip flop.

E.g. Excitation table of SR flip flop

|  |  |  |  |
| --- | --- | --- | --- |
| Qn | Qn+1 | S | R |
| 0 | 0 | 0 | X |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | X | 0 |

**Characteristic Table**

The Characteristic table is used to find what the next state of the flip flop will be, by keeping in view what the current state is.

Both excitation and characteristic table are used in the conversion of flip flop into another flip flop, in making counters etc.

**d) Synchronous sequential circuit and Asynchronous sequential circuits. (1.0)**

There are two types of sequential circuits,

**Asynchronous sequential circuits**

Asynchronous in which the output changes very soon after the input changes. In these circuits the flow and storage of the data is not controlled. In these circuits the some or all outputs don’t change when the clock changes.

**Synchronous sequential circuits**

The second one is synchronous in which the state of the circuit is stored in order to effect the next state. The outputs of the circuit change when the clock changes.

**e) Latches and Flip-flops. (1.0)**

The Latches and flip flops are the basic building blocks of the sequential circuit.

**Latches**

Latches are build up by gates, Latches are asynchronous and they don’t have a clock pulse as the input. Latch is a basic storage element

**Flip Flops.**

The Flip flops are built up by latches, they are synchronous form of latches and they also take a clock pulse as an input. The flip flops samples the inputs and changes the outputs on particular instants.

**Q: 2 [10 Marks] [CLO 2]**

**Simplify the Boolean expressions using K-map.**

**a) F(x,y,z)=∑m(1,3,6) (1.0)**

x yz

00 01 11 10

xyz’

x’z

0

1

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1 | 1 |  |
|  |  |  | 1 |

F = x’z +xyz’

**b) F(x,y,z)=∑m(0,1,2,7) + d(3,6) (1.0)**

x yz

00 01 11 10

y

X’

0

1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1 | X | 1 |
|  |  | 1 | X |

F= x’ + y

**c) F(w,x,y,z)=∑m(0,1,6,9,14,15) (1.0)**

wx yz

x’y’z

00 01 11 10

wxy

xyz’

w’x’y’

00

01

11

10

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1 |  |  |
|  |  |  | 1 |
|  |  | 1 | 1 |
|  | 1 |  |  |

F = w’x’y’ + x’y’z + wxy + xyz’

**d) F(w,x,y,z)=∑m(2,4,5,6,7,9,10) (1.0)**

wx yz

00 01 11 10

wx’y’z

w’x

x’yz

00

01

11

10

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | 1 |
| 1 | 1 | 1 | 1 |
|  |  |  |  |
|  | 1 |  | 1 |

F = w’x + wx’y’z +x’yz

**e) F(a,b,c,d,e)=∑m(16,17,20,21,28,29,24,25,3,7,15,11,2,6,14,10). (1.0)**

ab cde

000 001 011 010 110 111 101 100

a’d

00

01

11

10

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 | 1 | 1 | 1 |  |  |
|  |  | 1 | 1 | 1 | 1 |  |  |
| 1 | 1 |  |  |  |  | 1 | 1 |
| 1 | 1 |  |  |  |  | 1 | 1 |

ad’

F = a’d + ad’

**f) F(x,y,z)=∏M(1,3,6) . D(7,2) (1.0)**

x yz

00 01 11 10

y’

x+z’

0

1

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 | 0 | X |
|  |  | X | 0 |

F = y’.(x+z’)

**g) F(x,y,z)=∏M(0,1,2,7) (1.0)**

x+z

x yz

00 01 11 10

x’+y’+z’

x+y

0

1

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 0 |  | 0 |
|  |  | 0 |  |

F = (x+y).(x’+y’+z’).(x+z)

**h) F(w,x,y,z)=∏M(0,1,6,9,14,15) (1.0)**

wx yz

00 01 11 10

w+x+z

x’+y’+z

w’+x’+y’

x+y+z’

00

01

11

10

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 0 |  |  |
|  |  |  | 0 |
|  |  | 0 | 0 |
|  | 0 |  |  |

F = (w+x+z).(x+y+z’).(x’+y’+z).(w’+x’+y’)

**i) F(w,x,y,z)=∏M(2,4,5,6,7,9,10) . D(1,3,8) (1.0)**

wx yz

00 01 11 10

x+y’+z

w+x’

w’+x+y

00

01

11

10

|  |  |  |  |
| --- | --- | --- | --- |
|  | X | X | 0 |
| 0 | 0 | 0 | 0 |
|  |  |  |  |
| X | 0 |  | 0 |

F = (w’+x+y).(w+x’).(x+y’+z)

**j) F(a,b,c,d,e)=∏M(16,17,20,21,28,29,24,25,3,7,15,11,2,6,14,10). (1.0)**

ab cde

000 001 011 010 110 111 101 100

a+d’

00

01

11

10

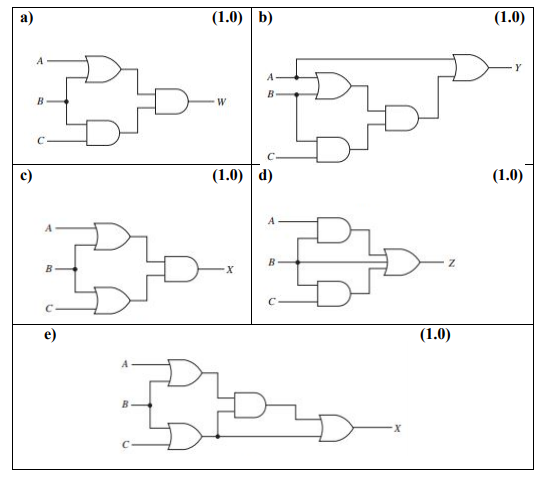
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 0 | 0 | 0 | 0 |  |  |
|  |  | 0 | 0 | 0 | 0 |  |  |
| 0 | 0 |  |  |  |  | 0 | 0 |
| 0 | 0 |  |  |  |  | 0 | 0 |

a’+d

F = (a’+d)+(a+d’)

**Q: 3 [05 Marks] [CLO 3]**

**Write the Boolean equation for the following circuits.**



1. W = (A+B).(B.C)
2. Y = ((A+B).(B.C))+A
3. X = (A+B).(B+C)
4. Z = (A.B)+(B.C)+B
5. X = ((A+B).(B+C))+(B+C)

**Q: 4 [08 Marks] [CLO 3]**

**(a) Design a circuit for an even parity generator, P, for three input variable x, y, z. (4.0)**

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | Even Parity Generator P |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

P = A’B’C + A’BC’ + AB’C’ + ABC

P = A’(B’C+BC’)+ A(B’C’ + BC)

P = A’ (B xor C) + A.(B xor C)’

P = A xor B xor C

Circuit Diagram:



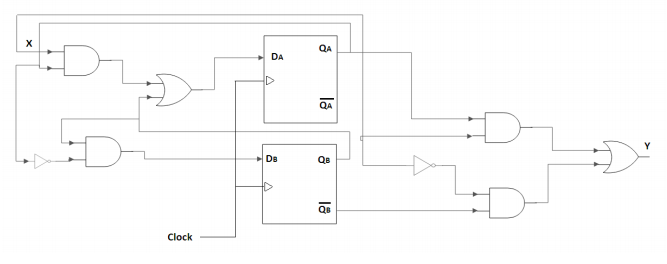
**(b) Design a circuit that has parity checker, C, for an odd parity generator. (4.0)**

**Circuit of 4 bit received Odd Parity Checker:**



**Q: 5 [05 Marks] [CLO 3]**

**Consider the following clock sequential circuit and answer the question given below:**



**a) In your opinion above circuit model is Mealy or Moore? (1.0)**

This circuit is a Moore Machine model because it has input logics as well as output logics

**b) Write flip-flop input equations for the above circuit. (1.0)**

**DA =** (QA.X)+QB

**DB =** (QB.QA)

**c) Write output equations for the above circuit. (3.0)**

**Y=**(QA.X)+(QB’+X’)

**Q: 6 [05 Marks] [CLO 3]**

**Design JK flip flop using SR flip flop and show each and every step in your answer.**

Step 1: Characteristic table of JK Flip Flop

|  |  |  |  |
| --- | --- | --- | --- |
| J | K | Qn | Qn+1 |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

Step 2: Excitation table of SR flip flop

|  |  |  |  |
| --- | --- | --- | --- |
| Qn | Qn+1 | S | R |
| 0 | 0 | 0 | X |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | X | 0 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| J | K | Qn | Qn+1 | S | R |
| 0 | 0 | 0 | 0 | 0 | X |
| 0 | 0 | 1 | 1 | X | 0 |
| 0 | 1 | 0 | 0 | 0 | X |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | X | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 |

Step 3: K maps:

For S:

J K Qn

JQn’

|  |  |  |  |
| --- | --- | --- | --- |
| X | 0 | 0 | 0 |
| 1 | X | 0 | 1 |

S = J.Qn+1

For R:

J K Qn

KQn

|  |  |  |  |
| --- | --- | --- | --- |
| X | 0 | 1 | X |
| 0 | 0 | 1 | 0 |

R = K.Qn



Q

J

S

SR Flip Flop

Clock

K

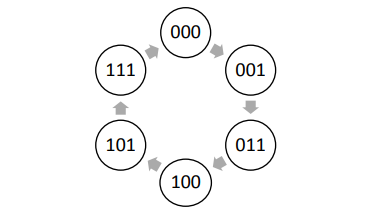


R

Q’

**Q: 7 [06 Marks] [CLO 3]**

**Construct a counter whose counting sequence satisfies the state transition diagram below. Find the “complete” state transition diagram (which includes all states) for the counter. Draw the waveforms of the outputs from the counter you constructed.**



**Step 1:**

Required number of flip flops are 3.

**Step 2:**

We will be using JK flip flops. So, excitation table of JK Flip Flop:

|  |  |  |  |
| --- | --- | --- | --- |
| Qn | Qn+1 | J | K |
| 0 | 0 | 0 | X |
| 0 | 1 | 1 | X |
| 1 | 0 | X | 1 |
| 1 | 1 | X | 0 |

**Step 3:**

Circuit Excitation Table:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Q1 | Q2 | Q3 | Q1\* | Q2\* | Q3\* | J1 | K1 | J2 | K2 | J3 | K3 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | X | 0 | X | 1 | X |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | X | 1 | X | X | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | X | X | 1 | X | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | X | 0 | 0 | X | 1 | X |
| 1 | 0 | 1 | 1 | 1 | 1 | X | 0 | 1 | X | X | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | X | 1 | X | 1 | X | 1 |

**Step 4:**

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **0** | **1** |  |
| **X** | **X** | **X** |  |

**Q1 Q2Q3**

**J1 = Q2.Q3**

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **0** | **1** |  |
| **X** | **X** | **X** |  |

**Q1 Q2Q3**

|  |  |  |  |
| --- | --- | --- | --- |
| **X** | **X** | **X** |  |
| **0** | **0** | **1** |  |

**K1 = Q2.Q3**

**Q1 Q2Q3**

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **1** | **X** |  |
| **0** | **1** | **X** |  |

**J2 = Q3**

**Q1 Q2Q3**

|  |  |  |  |
| --- | --- | --- | --- |
| **X** | **X** | **1** |  |
| **X** | **X** | **1** |  |

**K2 = Q3**

**Q1 Q2Q3**

|  |  |  |  |
| --- | --- | --- | --- |
| **X** | **0** | **1** |  |
| **X** | **0** | **1** |  |

**K3 = Q2.Q3**

**Q1 Q2Q3**

|  |  |  |  |
| --- | --- | --- | --- |
| **1** | **X** | **X** |  |
| **1** | **X** | **X** |  |

**J3 = Q2’**

****

**Q: 8 [06 Marks] [CLO 3]**

1. **Design an 8-to-3 binary encoder with inputs I0-I7 the logic expressions of the outputs Y0-Y2 are given below: (3.0)**

**Y0 = I4 + I5 + I6 +I7**

**Y1= I2 + I3 + I6 + I7**

**Y2 = I1 + I3 + I5 + I7**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| I0 | I1 | I2 | I3 | I4 | I5 | I6 | I7 | Y2 | Y1 | Y0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |



1. **Design 16:1 multiplexer for the function Z = B’C + A’BD + AB’ and also fill the truth-table given below: (3.0)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D** | **Z** |
| **0** | **0** | **0** | **0** | **0** |
| **0** | **0** | **0** | **1** | **0** |
| **0** | **0** | **1** | **0** | **1** |
| **0** | **0** | **1** | **1** | **1** |
| **0** | **1** | **0** | **0** | **0** |
| **0** | **1** | **0** | **1** | **1** |
| **0** | **1** | **1** | **0** | **0** |
| **0** | **1** | **1** | **1** | **1** |
| **1** | **0** | **0** | **0** | **1** |
| **1** | **0** | **0** | **1** | **1** |
| **1** | **0** | **1** | **0** | **1** |
| **1** | **0** | **1** | **1** | **1** |
| **1** | **1** | **0** | **0** | **0** |
| **1** | **1** | **0** | **1** | **0** |
| **1** | **1** | **1** | **0** | **0** |
| **1** | **1** | **1** | **1** | **0** |

Y

S0 S1 S2 S3

0

0

1

1

0

1

0

1

1

1

1

1

0

0

0

0

16 - 1 Multiplexer