**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **29/may/2020** | **Name:** | **Yashaswini R** |
| **Course:** | **Logic design** | **USN:** | **4al17ec098** |
| **Topic:** | * **Analysis of clocked sequential circuit** * **Digital clock design** | **Semester & Section:** | **6th b** |
| **Github Repository:** | **yashaswini** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report – Report can be typed or hand written for up to two pages.**  ANALYSIS OF CLOCKED SEQUENTIAL CIRCUITS  **Positive Edge Triggered D Flip-flop**     * A circuit diagram of a Positive edge triggered D Flip-flop is shown as below. It has an **additional reset input** connected to the three NAND gates.      * When the **reset input is 0 it forces output Q' to Stay at 1** which clears output Q to 0 thus resetting the flip-flop.      * Two other connections from the reset input ensure that the S input of the third **SR latch stays at logic 1** while the reset input is at 0 regardless of the values of D and Clk.      * Function table suggests that:      * **When R = 0, the output is set to 0 (independent of D and Clk).**      * The clock at Clk is shown with an upward arrow to indicate that the flip-flop triggers on the positive edge of the clock.      * The value in D is transferred to Q with every positive-edge clock signal provided that R = 1.     Analysis with D Flip-Flops     * The input equation of a D Flip-flop is given by **DA = A ⊕ x ⊕ y.** DA means a D Flip-flop with output A.      * The x and y variables are the inputs to the circuit. No output equations are given, which implies that the output comes from the output of the flip-flop.      * The state table has one column for the present state of flip-flop 'A' two columns for the two inputs, and one column for the next state of A.      * The next-state values are obtained from the state equation **A(t + 1) = A ⊕ x ⊕ y.**      * The expression specifies an odd function and is equal to 1 when only one variable is 1 or when all three variables are 1.       Analysis with JK Flip-Flops     * The circuit can be specified by the flip-flop input equations:      * **JA = B; KA = Bx'**      * **JB = x'; KB = A'x + Ax' = A ⊕ x** * The next state of each flip-flop is evaluated from the corresponding J and K inputs and the characteristic table of the JK flip-flop listed as:      * **When J = 1 and K = 0 the next state is 1**      * **When J = 0 and K = 1 the next state is 0**      * **When J = 0 and K = 0 there is no change of state and the next-state value is the same as that of the present state.**      * **When J = K = 1, the next-state bit is the complement of the present-state bit.**      * The characteristic equations for the flip-flops are      * **A(t + 1) = JA' + K'A**      * **B(t + 1) = JB' + K'B** * This gives us the state equation of A by substituting the values of JA, KA        * **A(t + 1) = BA' + (Bx')'A = A'B + AB' + Ax** * The state equation provides the bit values for the column headed "Next State" for A in the state table. Similarly, the state equation for flip-flop B can be derived from the characteristic equation by substituting the values of JB and KB.:      * **B(t + 1) = x'B' + (A ⊕ x)'B = B'x' + ABx + A'Bx'**       Analysis with T Flip-Flops     * The circuit can be specified by the characteristic equations:      * **Q(t+1) = T ⊕ Q = T'Q + TQ'** * The sequential circuit has two flip-flops A and B, one input x, and one output y and can be described algebraically by two input equations and an output equation:      * **TA = Bx**      * **TB = x**      * **y = AB** * The state table for the circuit is listed below. The values for y are obtained from the output equation. The values for the next state can be derived from the state equations by substituting TA and TB in the characteristic equations yielding:      * **A(t + 1) = (Bx)' A + (Bx)A' = AB' + Ax' + A'Bx**      * **B(t + 1) = x ⊕ B**       STATE REDUCTION AND ASSIGNMENT     * Two sequential circuits may exhibit the same input-output behavior but have a different number of internal states in their state diagram.      * Certain properties of sequential circuits may simplify a design by reducing the number of gates and flip-flops it uses. Reducing the number of flip-flops reduces the cost of a circuit.      * The reduction in the number of flip-flops in a sequential circuit is referred to as the state reduction problem. State-reduction algorithms are concerned with procedures for reducing the number of states in a state table while keeping the external input-output requirements unchanged     **Example of State Reduction**     * First we need the state table: it is more convenient to apply procedures for state reduction with the use of a table rather than a diagram.      * Then we apply the reduction algorithms *"Two states are said to be equivalent if for each member of the set of inputs they give exactly the same output and send the circuit either to the same state or to an equivalent state."*      * Similarly, states f and d are equivalent, and state f can be removed and replaced by d.      * In general reducing the number of states in a state table may result in a circuit with less equipents. But it does not guarantee a saving in the number of flip-flops or the number of gates. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:** | **29/may/2020** | **Name:** | **Yashaswini R** | |
| **Course:** | **python** | **USN:** | **4al17ec098** | |
| **Topic:** | **Object Oriented Programming** | **Semester&Section:** | **6th b** | |
| **Git hub repository** | **yashaswini** |  |  | |
| **AFTERNOON SESSION DETAILS** | | | |
| **Image of session** | | | |
| **Report – Report can be typed or hand written for up to two pages.**  **GUI in OOP Design**  *frontend.py*and *backend.py*scripts in OOP style.   * #frontend.py * from tkinter import \* * from backend import Database * database=Database("books.db") * class Window(object): * def \_\_init\_\_(self,window): * self. Window = window * self.window.wm\_title("Bookstore") * l1=Label(window,text="Title") * l1.grid(row=0,column=0) * l2=Label(window,text="Author") * l2.grid(row=0,column=2) * l3=Label(window,text="Year") * l3.grid(row=1,column=0) * l4=Label(window,text="ISBN") * l4.grid(row=1,column=2) * self.title\_text=StringVar() * self.e1=Entry(window,textvariable=self.title\_text) * self.e1.grid(row=0,column=1) * self.author\_text=StringVar() * self.e2=Entry(window,textvariable=self.author\_text) * self.e2.grid(row=0,column=3) * self.year\_text=StringVar() * self.e3=Entry(window,textvariable=self.year\_text) * self.e3.grid(row=1,column=1) * self.isbn\_text=StringVar() * self.e4=Entry(window,textvariable=self.isbn\_text) * self.e4.grid(row=1,column=3) * self.list1=Listbox(window, height=6,width=35) * self.list1.grid(row=2,column=0,rowspan=6,columnspan=2) * sb1=Scrollbar(window) * sb1.grid(row=2,column=2,rowspan=6) * self.list1.configure(yscrollcommand=sb1.set) * sb1.configure(command=self.list1.yview) * self.list1.bind('<<ListboxSelect>>',self.get\_selected\_row) * b1=Button(window,text="View all", width=12,command=self.view\_command) * b1.grid(row=2,column=3) * b2=Button(window,text="Search entry", width=12,command=self.search\_command) * b2.grid(row=3,column=3) * b3=Button(window,text="Add entry", width=12,command=self.add\_command) * b3.grid(row=4,column=3) * b4=Button(window,text="Update selected", width=12,command=self.update\_command) * b4.grid(row=5,column=3) * b5=Button(window,text="Delete selected", width=12,command=self.delete\_command) * b5.grid(row=6,column=3) * b6=Button(window,text="Close", width=12,command=window.destroy) * b6.grid(row=7,column=3) * def get\_selected\_row(self,event): * index=self.list1.curselection()[0] * self.selected\_tuple=self.list1.get(index) * self.e1.delete(0,END) * self.e1.insert(END,self.selected\_tuple[1]) * self.e2.delete(0,END) * self.e2.insert(END,self.selected\_tuple[2]) * self.e3.delete(0,END) * self.e3.insert(END,self.selected\_tuple[3]) * self.e4.delete(0,END) * self.e4.insert(END,self.selected\_tuple[4]) * def view\_command(self): * self.list1.delete(0,END) * for row in database.view(): * self.list1.insert(END,row) * def search\_command(self): * self.list1.delete(0,END) * for row in database.search(self.title\_text.get(),self.author\_text.get(),self.year\_text.get(),self.isbn\_text.get()): * self.list1.insert(END,row) * def add\_command(self): * database.insert(self.title\_text.get(),self.author\_text.get(),self.year\_text.get(),self.isbn\_text.get()) * self.list1.delete(0,END) * self.list1.insert(END,(self.title\_text.get(),self.author\_text.get(),self.year\_text.get(),self.isbn\_text.get())) * def delete\_command(self): * database.delete(self. selected\_tuple[0]) * def update\_command(self): * database. update(self.selected\_tuple[0],self.title\_text.get(),self.author\_text.get(),self.year\_text.get(),self.isbn\_text.get()) * window=Tk() * Window(window) * window.mainloop()   **backend.py script in OOP:**   * #backend.py * import sqlite3 * class Database: * def \_\_init\_\_(self, db): * self.conn=sqlite3.connect(db) * self.cur=self.conn.cursor() * self.cur.execute("CREATE TABLE IF NOT EXISTS book (id INTEGER PRIMARY KEY, title text, author text, year integer, isbn integer)") * self.conn.commit() * def insert(self,title,author,year,isbn): * self.cur.execute("INSERT INTO book VALUES (NULL,?,?,?,?)",(title,author,year,isbn)) * self.conn.commit() * def view(self): * self.cur.execute("SELECT \* FROM book") * rows=self.cur.fetchall() * return rows * def search(self,title="",author="",year="",isbn=""): * self.cur.execute("SELECT \* FROM book WHERE title=? OR author=? OR year=? OR isbn=?", (title,author,year,isbn)) * rows=self.cur.fetchall() * return rows * def delete(self,id): * self.cur.execute("DELETE FROM book WHERE id=?",(id,)) * self.conn.commit() * def update(self,id,title,author,year,isbn): * self.cur.execute("UPDATE book SET title=?, author=?, year=?, isbn=? WHERE id=?",(title,author,year,isbn,id)) * self.conn.commit() * def \_\_del\_\_(self): * self.conn.close()   **Report of Webinar on Preparing for Next Normal:by Mr.Mohan Kumar**  **Case Study**    **Business Impact:**    **Trends**      How Organisations React to the Covid 19?  Trust->Compassion->Stability->Hope  “We cannot solve problems in same Thinking “-Einstein  ●Resilient Dynamism  ●Digital Transmission  ●Economic Crisis  Digital Transmission in Education  ●Institution  ●Forcefully | | | |