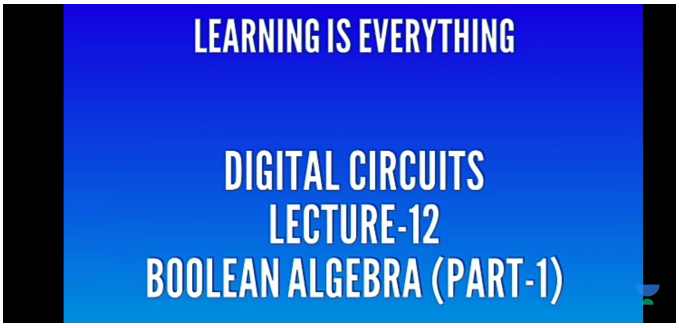
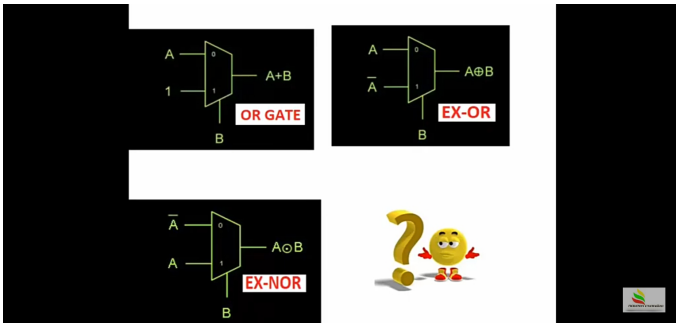
Day 9 report

Date:	28/05/2020	Name:	Ankitha c c
Course:	LOGIC DESIGN	USN:	4al16ec004
Topic:	1. BOOLEAN EQUATIONS FOR DIGITAL CIRCUIT 2. COMBINATIONAL CIRCUIT 3. BCD TO 7 SEGMENT DISPLAY CONVERSATION	Semester & Section:	8th & "A" section
Github Repository:	ankitha-c-c		

FORENOON SESSION DETAILS

Image of session





.

Report – Report can be typed or hand written for up to two pages.

1. Boolean equations for digital circuit

Boolean algebra

The logical symbol 0 and 1 are used for representing the digital input or output. The symbols "1" and "0" can also be used for a permanently open and closed digital circuit. The digital circuit can be made up of several logic gates. To perform the logical operation with minimum logic gates, a set of rules were invented, known as the Laws of Boolean Algebra. These rules are used to reduce the number of logic gates for performing logic operations.

The Boolean algebra is mainly used for simplifying and analyzing the complex Boolean expression. It is also known as Binary algebra because we only use binary numbers in this. George Boole developed the binary algebra in 1854.

Rules in Boolean algebra

Only two values(1 for high and 0 for low) are possible for the variable used in Boolean algebra.

The overbar(-) is used for representing the complement variable. So, the complement of variable C is represented as .

The plus(+) operator is used to represent the ORing of the variables.

The dot(.) operator is used to represent the ANDing of the variables.

Properties of Boolean algebra

1. Annulment Law

When the variable is AND with 0, it will give the result 0, and when the variable is OR with 1, it will give the result 1, i.e.,

$B.0 = 0$

$B+1=1$

2. Identity Law

When the variable is AND with 1 and OR with 0, the variable remains the same, i.e.,

$B.1 = B$

$B+0 = B$

3. Idempotent Law

When the variable is AND and OR with itself, the variable remains same or unchanged, i.e.,

4. Complement Law

When the variable is AND and OR with its complement, it will give the result 0 and 1 respectively.

$B.B' = 0$

$B+B' = 1$

5. Double Negation Law

This law states that, when the variable comes with two negations, the symbol gets removed and the original variable is obtained.

((A)')' = A

6. Commutative Law

This law states that no matter in which order we use the variables. It means that the order of variables doesn't matter in this law.

A.B = B.A

A+B = B+A

7.Associative Law

This law states that the operation can be performed in any order when the variables priority is of same as '*' and '/'.

(A.B).C = A.(B.C)

(A+B)+C = A+(B+C)

8.Distributive Law

This law allows us to open up of brackets. Simply, we can open the brackets in the Boolean expressions.

A+(B.C) = (A+B).(A+C

A.(B+C) = (A.B)+(A.C)

9.Absorption Law

This law allows us for absorbing the similar variables.

B+(B.A) = B

B.(B+A) = B

10.De Morgan Law

The operation of an OR and AND logic circuit will remain same if we invert all the inputs, change operators from AND to OR and OR to AND, and invert the output.

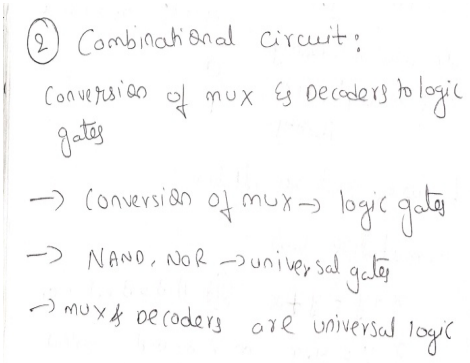
(A.B)' = A'+B'

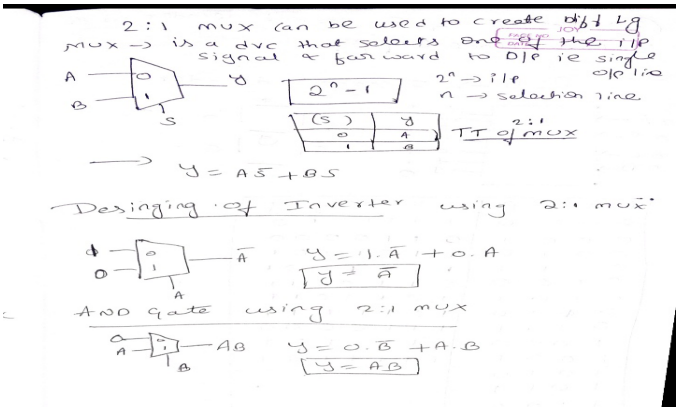
(A+B)' = A'.B'

2. Combinational circuits

Combinational Circuits (CC) are circuits made up of different types of logic gates. A logic gate is a basic building block of any electronic circuit. The output of the combinational circuit depends on the values at the input at any

given time. The circuits do not make use of any memory or storage device.





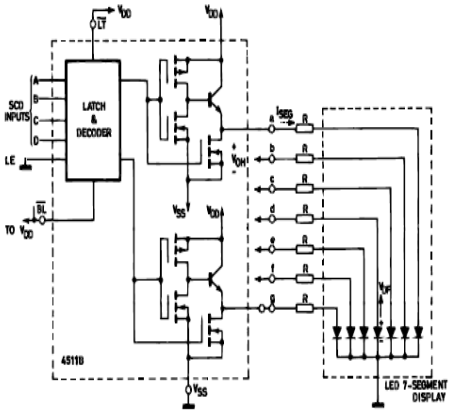
3. BCD TO 7-SEGMENT DISPLAY CONVERSATION

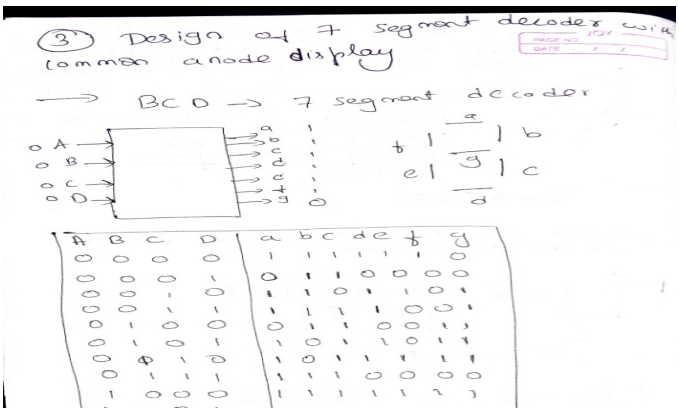
A seven-segment display is an electronic display device for displaying decimal numerals. Seven-segment displays are widely used in digital clocks, electronic meters and other electronic devices that display numerical information.

The schematic shows a BCD to 7 Segment Display for one of the digits of a digital clock. The following components are used.

CD4511 - BCD to 7 Segment Display Decoder IC

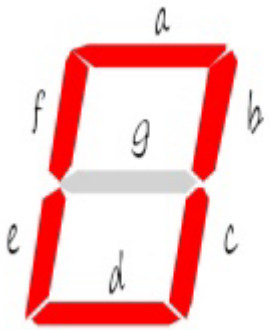
LSHD-A103 - 7 Segment LED Common Cathode Display





7 Segment Display

A 7 Segment LED display generally has 8 input connections, one for each LED segment and one that acts as a common terminal. There are 2 types of 7 Segment LED digital display.



a

b

c

d

e

f

g

d

Common Cathode Display – all the cathode connections of the LEDs are connected to ground. A logic '1' applied to the anode terminal of the individual segment illuminates it.

Common Anode Display – all the anode connections of the LEDs are connected to VCC. A logic '0' applied to the cathode terminal of the individual segment illuminates it.

BCD to 7 Segment Display Decoder

A BCD to Seven Segment decoder is a combinational logic circuit that accepts a decimal digit in BCD (input) and generates appropriate outputs for the segments to display the input decimal digit.The truth table is extracted from the CD4511 IC datasheet. This truth table is interactive. Click on any row to see the respective 7 segment display output.

Truth Table

Display	D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1
Blank	1	0	1	0	0	0	0	0	0	0	0
Blank	1	0	1	1	0	0	0	0	0	0	0
Blank	1	1	0	0	0	0	0	0	0	0	0
Blank	1	1	0	1	0	0	0	0	0	0	0
Blank	1	1	1	0	0	0	0	0	0	0	0
Blank	1	1	1	1	0	0	0	0	0	0	0

Date: 28/05/2020

Name: Ankitha c c

Course: Phython

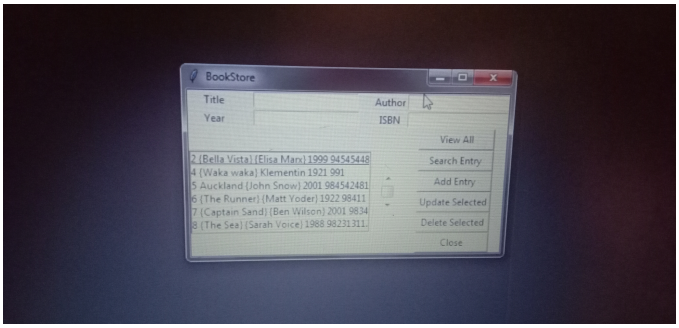
USN: 4al16ec004

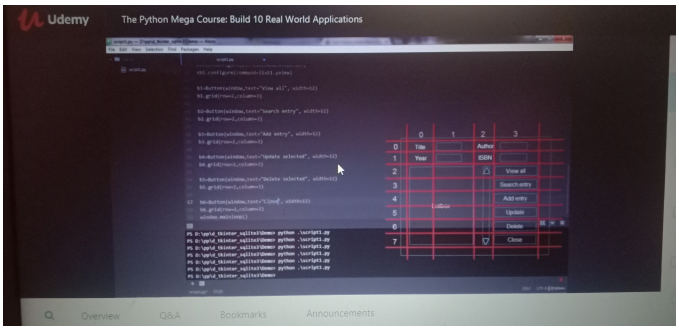
Topic:

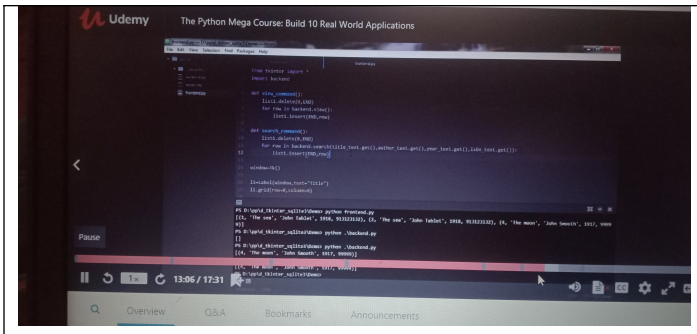
Semester & Section: 8th & a section

AFTERNOON SESSION DETAILS

Image of session







Report – Report can be typed or hand written for up to two pages.

Fixing the Bug (Practice)

Exercise

If you haven't already noticed, the program has a bug. When the listbox is empty and the user clicks the listbox, an IndexError is generated in the terminal:

Why does this error happen?

Well, everything starts with the user clicking on the listbox. Clicking the listbox executes the following code:

```
list1.bind('<<ListboxSelect>>',get_selected_row)
```

That code calls the get_selected_row function:

```
def get_selected_row(event):
    global selected_tuple
    index=list1.curselection()[0]
    selected_tuple=list1.get(index)
    e1.delete(0,END)
    e1.insert(END,selected_tuple[1])
    e2.delete(0,END)
    e2.insert(END,selected_tuple[2])
    e3.delete(0,END)
```

```
    e3.insert(END,selected_tuple[3])
    e4.delete(0,END)
    e4.insert(END,selected_tuple[4])
```

Since the listbox is empty, list1.curselection() will be an empty list with no items. Trying to access the first item on the list with [0] in line 3 will throw an error, because there is no first item in the list.

Try to fix that bug. The next lecture contains the solution.

Solution

Solution

```
def get_selected_row(event):
    try:
        global selected_tuple
        index=list1.curselection()[0]
        selected_tuple=list1.get(index)
        e1.delete(0,END)
        e1.insert(END,selected_tuple[1])
        e2.delete(0,END)
        e2.insert(END,selected_tuple[2])
        e3.delete(0,END)
        e3.insert(END,selected_tuple[3])
        e4.delete(0,END)
        e4.insert(END,selected_tuple[4])
    except IndexError:
        pass
```

Explanation

The error was fixed by simply implementing a try and except block. When the get_selected_row function is called, Python will execute the indented block under try . If there is an IndexError, none of the lines under try will be executed; the line under except will be executed, which is pass. The

pass statement means "do nothing". Therefore the function will do nothing when there's an empty listbox.