**‘One platform for drawing circuit diagrams, evaluating Boolean expressions and generating truth tables’**

**Abstract:**

Boolean algebra was named after British mathematician George Boole and is an algebra of logical values. All computers today perform their functions utilizing two-value Boolean logic. Their electrical circuits are a physical manifestation of two-value Boolean logic. Logic can be represented in three ways, truth tables, it is a chart of 1s and 0s arranged to indicate the results (or outputs) of all possible inputs, a logic diagram which is a pictoral description of logic gates in combination to represent a logic expression and Boolean expressions used for writing a logic expression in equation form. Boolean circuits also provide a model for many digital components used in computer engineering like multiplexers, adders, and arithmetic logic units. It would be helpful to study these digital components with truth tables, logic circuit diagrams and Boolean expressions on one platform. This project attempts to do so. I will be developing this project in Java with its inbuilt packages like java.io.File java.util.Scanner and graphics package for jpanels for drawing circuit diagrams for NAND gate. For evaluation of Boolean expression, for loops with operators and operands will be used in traverse with AND, OR, XOR operands. Once I have a functional platform, further improvisations on the existing code for incorporating AND, NOR, OR, XOR gates in jpanel will be necessary. Options to export the circuits as JPEG/PNG format files can be useful for publications. The truth table usage will be only for 2 variables; it can be extended for 4 or 8 or more variables in future.

**Background and Motivation:**

All computers today perform their functions utilizing two-value Boolean logic. Their electrical circuits are a physical manifestation of two-value Boolean logic. This can be achieved by voltages on wires in high-speed circuits or as an orientations of a magnetic domain in ferromagnetic storage devices [1]. This is the reason for them to be two-value Boolean circuits. Computers also use ordered sequences of Boolean values or bits (32 or 64) values. An example of a Boolean values can be 01101000110101100101010101001011. Another usage of Boolean values could be in low-level digital structure, where programmers work with the low-level digital structure of the data registers in machine code assembly language [1]. The voltages for these registers operates with Boolean values with 0 Boolean values representing zero volts while 1 representing a reference voltage (often +5V, +3.3V, +1.8V). Such assembly languages can support both numeric operations (executes arithmetic operations like add, subtract, multiply, or divide) and logical operations (Boolean logical operations). Boolean algebra was named for British mathematician George Boole and is an algebra of logical values (true and false). It gives a postulates and theorems that provides ways to simplify logic expressions [2].

Logic can be represented in following three ways:

1. Truth Tables:

It is a chart of 1s and 0s arranged to indicate the results (or outputs) of all possible inputs. The list of all possible inputs are arranged in columns on the left and the resulting outputs are listed in columns on the right. There are 2 to the power n possible states [3]. For three inputs there are 2^3=8 possible combination of inputs. Following is an example of truth table for 3 inputs:



2. Logic Circuit Diagram:

A logic diagram is a pictoral description of logic gates in combination to represent a logic expression. Following is an example that shows a logic diagram with three inputs (A, B, and C) and one output (Y) [3].



3. Boolean Expression:

Boolean Algebra can be used to write a logic expression in equation form.  Following are some examples of writing Boolean expressions [3].



Boolean circuits also provide a model for many digital components used in computer engineering like multiplexers, adders, and arithmetic logic units. It would be helpful to study these digital components with truth tables, logic circuit diagrams and Boolean expressions on one platform. This project attempts to do so. It can create truth tables for two input variables. It can also evaluate Boolean expressions and can generate circuit design application for NAND gates.

**Materials and Methods:**

I have used inbuilt packages, java.io.File for inputting data, java.io.FileNotFoundException for exception handler, java.util.Scanner to scan new files, graphics packages like java.awt.BorderLayout, java.awt.Color, java.awt.Dimension, java.awt.Graphics, java.awt.Graphics2, java.awt.Point, java.awt.Polygon, java.awt.RenderingHints, java.awt.Shape, event listener packages like java.awt.event.ActionEvent, java.awt.event.ActionListener, java.awt.event.MouseAdapter, java.awt.event.MouseEvent, java.awt.event.MouseListener, java.awt.event.MouseMotionListener, java.util.ArrayList, jframe packages like javax.swing.JButton, javax.swing.JFrame, javax.swing.JLabel, javax.swing.JPanel, javax.swing.SwingConstants and javax.swing.border.LineBorder.

Following classes were used for:

1. class Connection: a connection connects an output to an Input.

2. class DrawPanel: contains components, draws lines (connecting pairs of components)

3. class Input: Triangle, left facing, at LHS of component (points into the component)

4. method contains(Point p): used to check for mouse clicks on this Connector

5. class NANDgate: used to create the dimensions of nand box display. The box has 2 blue colored inputs, and a red colored output indicating a nand gate. All the gates are numbered serially as you make new additions.

6. methods mouseDragged(MouseEvent e), mouseEntered(MouseEvent e), mouseExited(MouseEvent e), mousePressed(MouseEvent e), mouseReleased(MouseEvent e) for interacting jpanel and actions.

Evaluation of boolean expression:

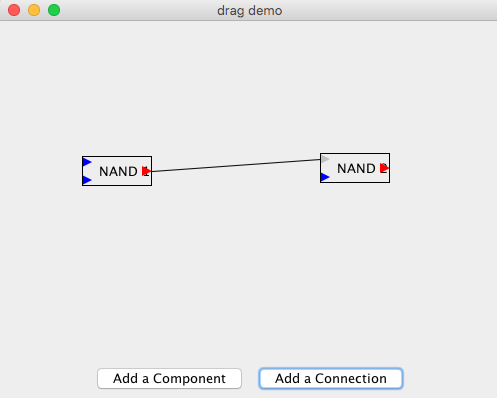
For evaluation of boolean expression, for loops with operators and operands are used in traverse with AND. OR, XOR operands. String should consist of only 0, 1, A, B, C where A = AND, B = OR, C = XOR. The software asks for entering your string to evaluate boolean expression in turn to return output on same interface.

Generation of truth tables:

For generating truth table with two variables, interface asks to enter values of variables 'a' and ‘b’ to get the truth table and the output in form of a table for each of the AND, OR, XOR and NOT gates are generated on same interface.

**Results:**

1. On starting the jpanel (Figure 1), two options for making a new NAND component and adding connections is provided. The user can select ‘Add new component’ button to generate a new component with mouse click. ‘Add a connection’ button can create a new connection between two components. The components are numbered serially to keep track of them. The user can create as many components and connections as needed to generate logic circuits.



**Figure 1: Jpanel with two options for adding a NAND component and a connection between components.**

1. After entering values of variable ‘a’ and 'b' from the user interface, a table is generated as follows which is the 2 variable truth table:

A B AND OR XOR NOT

1 1 1 1 0 1

1 0 0 1 1 1

1 1 1 1 0 0

1. A Boolean expression can be evaluated on a string and outputted on the same interface in terms of a 0 or 1 value indicating true or false.

**Conclusions and Future Scope:**

The aim of this project was to evaluate Boolean expression, generate a truth table and logic circuits for Boolean expressions in one place. Some further improvisations on the existing code of incorporating AND, NOR, OR, XOR gates in jpanel are necessary. Further options to export the circuits as JPEG/PNG format files can be useful for result publications. The truth table usage is only for 2 variables; it can be extended for 4 or 8 or more variables. Further the Boolean expressions are evaluated without precedence in the current software, precedence rules should be taken into consideration in future.

**References:**

1. J. Michael Dunn; Gary M. Hardegree (2001). Algebraic methods in philosophical logic. Oxford University Press US. p. 2. ISBN 978-0-19-853192-0.
2. Boolean Expressions. URL, 2017. retrieved from: https://docs.oracle.com/cd/B12037\_01/olap.101/b10339/expression006.htm
3. Logic Circuits, Boolean Algebra, and Truth Tables. URL, 2017. retrieved from: <https://drstienecker.com/tech-332/3-logic-circuits-boolean-algebra-and-truth-tables/>