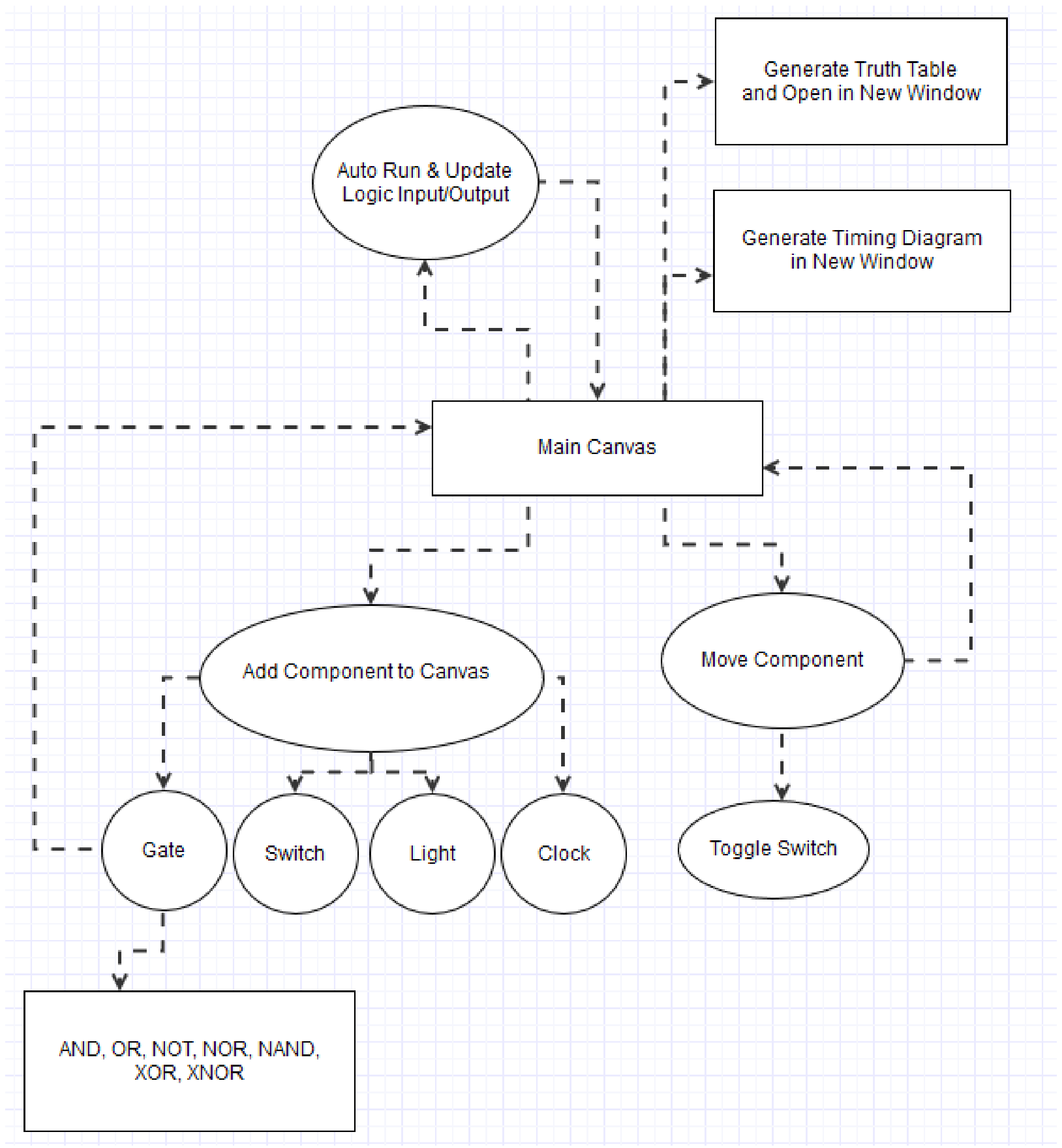
5.1 Introduction

When learning the basics of electrical engineering, it can often be difficult to format information in a way that is easy to teach and learn. This difficulty is especially apparent when it comes to choosing a software for digital logic simulation. Although there are a variety of options, many are lacking good design and easy usability. Furthermore, many softwares do not have all the features needed to teach and learn. My project is to create a standalone desktop application that combines intuitive design, straightforward interactions, and a suite of all the basic features and functionality required for teachers and students to demonstrate digital logic circuits effectively. The specifics of this project will be outlined in the remainder of this document.

This UML state diagram describes the current functionality, but will need to be updated to reflect all future functionality


Generate Truth Table and Open in New Window
Auto Run & Update Logic Input/Output
Generate Timing Diagram in New Window
Main Canvas
Add Component to Canvas
Move Component
Gate
Switch
Light
Clock
Toggle Switch
AND, OR, NOT, NOR, NAND, XOR, XNOR

5.2 Functional Requirements

The completed software will include: Digital logic gate simulation from the seven basic gates, interactive input components (switches), clocks, and simple output lights. My software will also have tools for describing the circuit, including: truth table generation and timing diagram generation. The user will be able to save and load circuits, and export images. The user will also be able to take smaller circuits and create "black box" representations for use in larger circuits.

5.2.1 Graphical User Interface

5.2.1.1 The GUI shall provide a set of buttons or draggables to access all functions of the application.

Functions Overview:

- Drag gate to canvas
- Drag switch, clock, or light to canvas
- Draw lines between components
- Generate truth table
- Generate timing diagram
- Create black box from current circuit
- Save current circuit to file
- Load circuit from file
- Export circuit as image

The GUI will be intuitive and easy to use and learn, based on user feedback.

5.2.1.2 All components placed on the canvas shall be movable by dragging.

The user should be able to move multiple components at once.

5.2.1.3 Lines between components shall auto adjust to new locations.

The user should be able to make lines neat with right angles.

5.2.1.4 Right clicking or holding right click over any component shall delete it.

5.2.2 Logic Simulation

5.2.2.1 The user shall be able to visually create circuits using the seven basic logic gates.

5.2.2.2 The user shall be able to add clocks components to the circuit.

The user should be able to change the frequency of clocks.

5.2.2.3 The user shall be able to add interactive inputs (switches) to the circuit.

The user will be able to toggle switches by left clicking on them.

5.2.2.4 The user shall be able to add outputs (lights) to the circuit.

5.2.2.5 The user shall be able to use a line tool to draw connections between circuits.

5.2.2.6 Logic simulation for all circuits shall auto update, and be accurate.

5.2.3 Truth Tables

5.2.3.1 The user shall be able to generate truth tables for applicable circuits.

Circuits with clocks are not applicable for truth table generation.

Circuits with loops are not applicable for truth table generation.

5.2.3.2 Truth tables will be labeled and show all possible input and output combinations.

5.2.3.3 Truth tables shall open in a new window.

Only one truth table window will be displayed at a time.

5.2.4 Timing Diagrams

5.2.4.1 The user shall be able to generate interactive timing diagrams for all circuits.

5.2.4.2 Timing diagrams shall show high and low states for inputs, outputs and clocks as a function of time.

5.2.4.3 The user shall be able to pause the timing diagram with a pause button.

Dragging windows and pausing should not affect timing diagram accuracy.

5.2.4.4 Timing diagrams shall open in a new window

Only one timing diagram window will be displayed at a time.

5.2.5 Black Box Components

5.2.5.1 The user shall be able to convert components into “black boxes”, to be used in the current or other project.

5.2.5.2 The user shall be able to build a component library with descriptions of inputs, outputs, and function of many different black box objects.

The user should be able to import/export components from other users.

5.2.6 Exporting, Saving, and Loading

5.2.6.1 The user shall be able to save a circuit as a namable local file.

5.2.6.2 The user shall be able to load a circuit from a saved or downloaded file.

5.2.6.3 The user shall be able to export circuits as images.

The user should also be able to export truth tables and timing diagrams as images.

5.3 Performance Requirements

5.3.1 The full application should be able to run well (Above 30 fps average)

on most (Medium-low and up hardware spec) systems, regardless of use case.

5.4 Environment Requirements

5.4.1 Development Environment Requirements

5.4.1.1 The program is being written with Python 2.7

5.4.1.2 Graphics are handled by the PyGame graphics library

5.4.1.3 Atom and Notepad++ are used to edit code

5.4.1.4 Photoshop is used to create icons and buttons

5.4.2 Execution Environment Requirements

5.4.2.1 The application shall be built as a standalone program for all environments

The application should have no external software requirements.

5.4.2.2 The hardware requirements shall be determined via testing once the program is deemed to be in beta.