



Circuit Board Designer - Final Presentation

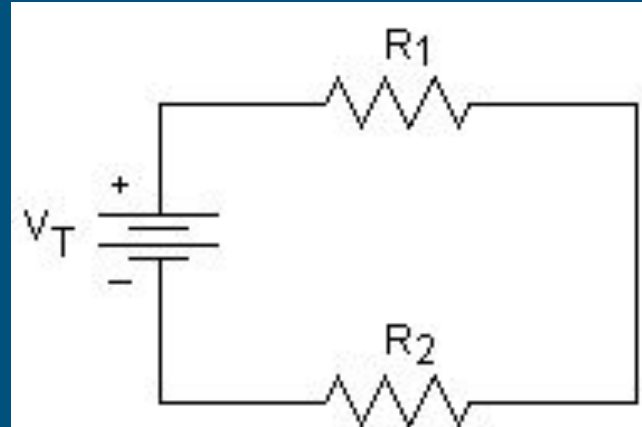


Kenneth Shipley, Jason Rivas, Joseph Spear



Introduction and Purpose

- Today's modern world is founded on the principles of electrical circuits and electronics. Without them, we wouldn't have light bulbs, heating and cooling, computers, and many other luxuries that we have today.
- Developing functional circuits and clean looking diagrams can be a hassle.
- In many cases, there is a huge difference between circuit diagrams and physical PCB layouts, which can cause big problems in converting from one to another.



Software Overview

- This program provides the user with the ability to produce basic circuit diagrams quickly and easily.
- It also provides the user the ability to convert their diagram to a physical PCB layout which is optimized.
 - The program outputs an image of the optimized PCB layout for use in any further electrical application.

Technologies

The technologies included in this program are:

Standalone Programs:

- QT Designer
- Python 3
- VS Code (Not Required for Build)
- GitHub (Not required for build)
- Discord (Not required for build)

Python Libraries:

- PyQt Library
- PySide2 (PyQT fork) Library
- Numpy Library
- XML Library
- JSON Library
- PIL (Python Imaging Library)/Pillow (PIL fork)

Software breakdown

This software is broken up into four main sections:

1. Graphical User Interface
2. PCB Optimization
3. Image Construction
4. File Management

**Graphical
User
Interface**

**Printable
Circuit
Board**

**Image
Conversion**

**File
Management**

Graphical User Interface

- One of the key features of the Circuit Board Designer is the simple yet comprehensive Graphical User Interface.
- The GUI is broken up into three main pages:
 - a. Start Menu / File Menu
 - b. Circuit Building Workspace
 - c. Convert Menu
- All GUI implementation was done using PyQt.

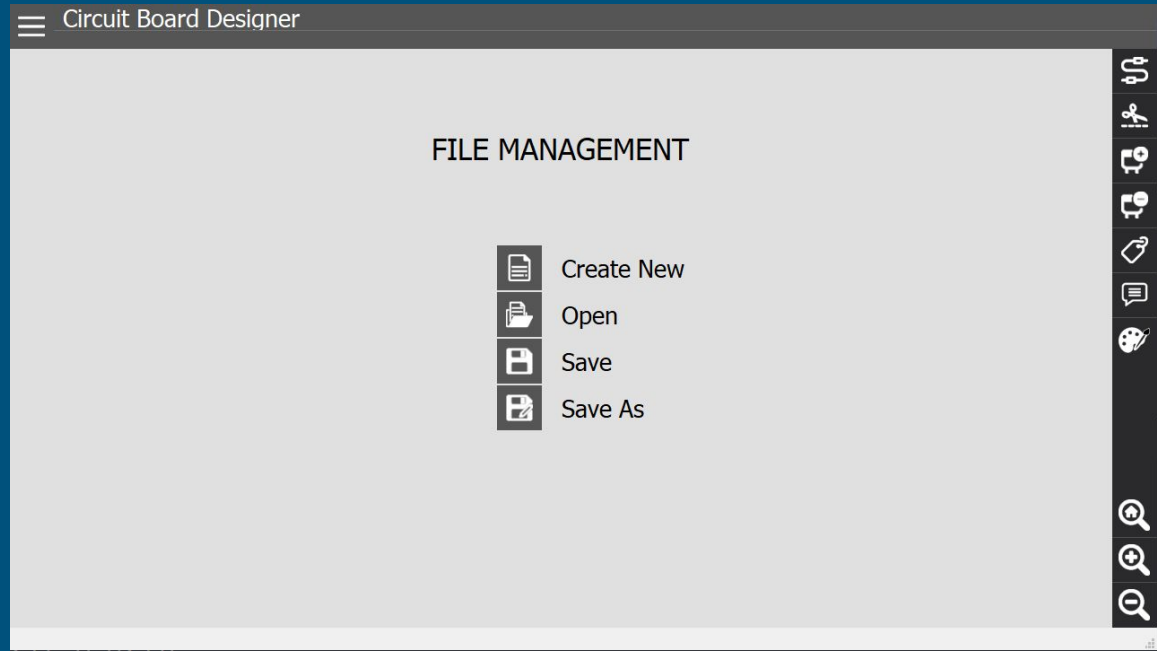


GUI - File Menu

The File Menu page is the initial page which opens when you start the program.

The File Menu includes the following functions:

- Create New
- Open
- Save
- Save As

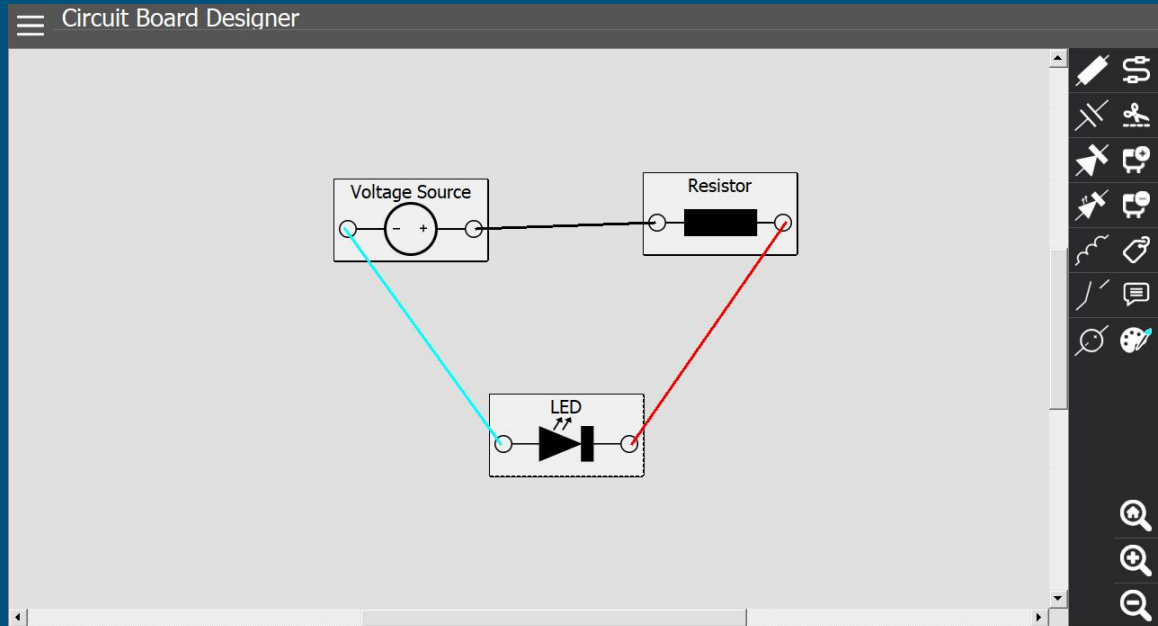


GUI - Circuit Building Workspace

The Circuit Building Workspace is the main GUI page. This is where the actual circuit building takes place.

The functions on the right tab are:

- Wire
- Wire Snipper
- Add Component
 - Subsequent components are shown on a second tab
- Delete Component
- Add Label
- Comment (Not Implemented)
- Color
 - Subsequent Colors are shown on a second tab



GUI - Convert Menu

- The Convert Menu is where the users input will be able to be converted from a circuit diagram into a physical PCB layout.

The Parameters here are:

- Number of Grid Nodes
- Number of Padding Nodes
- Target Layout Score
- Image Scaling
- Background Image Color
- Trace Wire Color

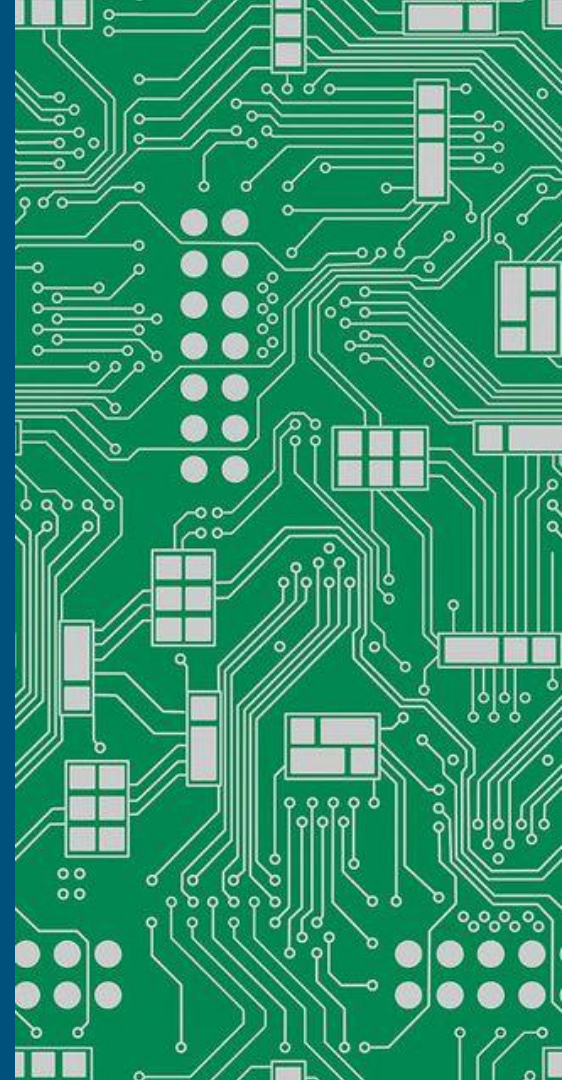
The screenshot displays the 'Circuit Board Designer' application window. The title bar shows the file path: 'C:/Users/rockm/Desktop/CAPSTONE/Circuit-Board-Designer/src/test.circ'. The main panel is titled 'Input Layout Settings' and contains six adjustable parameters:

- Number of Grid Nodes:** A slider ranging from 5 to 15, currently set at 5.
- Number of Padding Nodes:** A slider ranging from 2 to 15, currently set at 2.
- Target Layout Score:** A slider ranging from 0 to 1, currently set at 0.05.
- Image Scaling:** A slider ranging from 50 to 150, currently set at 50.
- Background Image Color:** A selection of color swatches. 'Navy' is selected, and 'Dark Gray' is also visible.
- Trace Wire Color:** A selection of color swatches. 'Yellow' is selected, and 'Red' is also visible.

At the bottom of the panel is a 'Generate Layout' button. A vertical toolbar on the right side of the window contains various icons for file operations and design tools.

PCB Optimization

- The second feature of Circuit Board Designer is converting the circuit diagram input from the user into an optimized physical layout for a PCB.
- This is done by using two common computational techniques:
 - Monte Carlo Algorithm
 - A-Star (A*) Algorithm



PCB Optimization - Monte Carlo

```
paths = []
curr_runs_score = 0
self.initialize_connections_list()
best_layout = [-1, None]
while (best_layout[0] < self.target_score) and (i < max_iters):
    self.paths.clear()
    self.pin_placement_dict.clear()

    self.randomize_layout()
    self.initialize_pin_placement_dict()
    paths = self.run_a_star()
```

PCB Optimization - Monte Carlo

```
j = 0
while paths == [] and j < 4*len(self.connections_list):
    np.random.shuffle(self.connections_list)
    paths = self.run_a_star()
    j += 1
if paths == []:
    i += 1
    continue
curr_runs_score = self.calculate_score(paths)
if curr_runs_score > best_layout[0]:
    best_layout[0] = curr_runs_score
    best_layout[1] = paths
i += 1
self.paths = best_layout[1]
```

PCB Optimization - A*

OPEN

CLOSED

add the start node to OPEN

loop

current = node in OPEN with the lowest f_cost

remove current from OPEN

add current to CLOSED

if (current is the target node)

return

for each neighbour of the current node

if (neighbour is not traversable) OR (neighbour is in CLOSED)

skip to the next neighbour

if (new path to neighbour is shorter) OR (neighbour is not in OPEN)

set f_cost of neighbour

set parent of neighbour to current

if (neighbour is not in OPEN)

add neighbour to OPEN

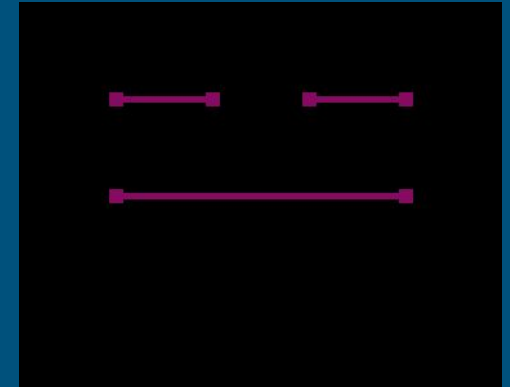
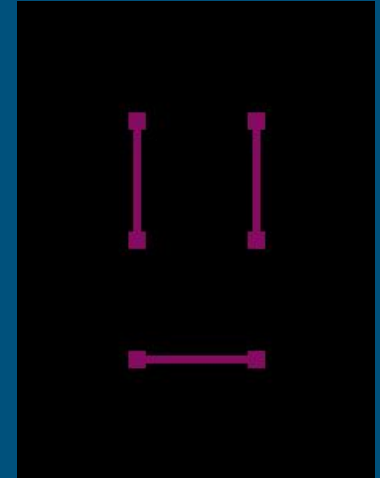
PCB Optimization - A*

```
def get_distance(self, from_node, goal_node):  
    di = abs(goal_node[0] - from_node[0])  
    dj = abs(goal_node[1] - from_node[1])  
  
    if dj > di:  
        return 14 * di + 10 * (dj - di)  
    return 14 * dj + 10 * (di - dj)
```

Image Construction

The final output of the Circuit Board Diagram is a set PCB Layout color coded however the user decides based on the Convert Menu options.

To the right are two different outputs from two circuits where there are three components each.



File Management

- PyQt offers its own way to save and open files.
- Save, Save As, and Open all call the backend's save and load methods as well.
- Create New deletes the schematic instance and creates a new one.
- .circ



Create New



Open



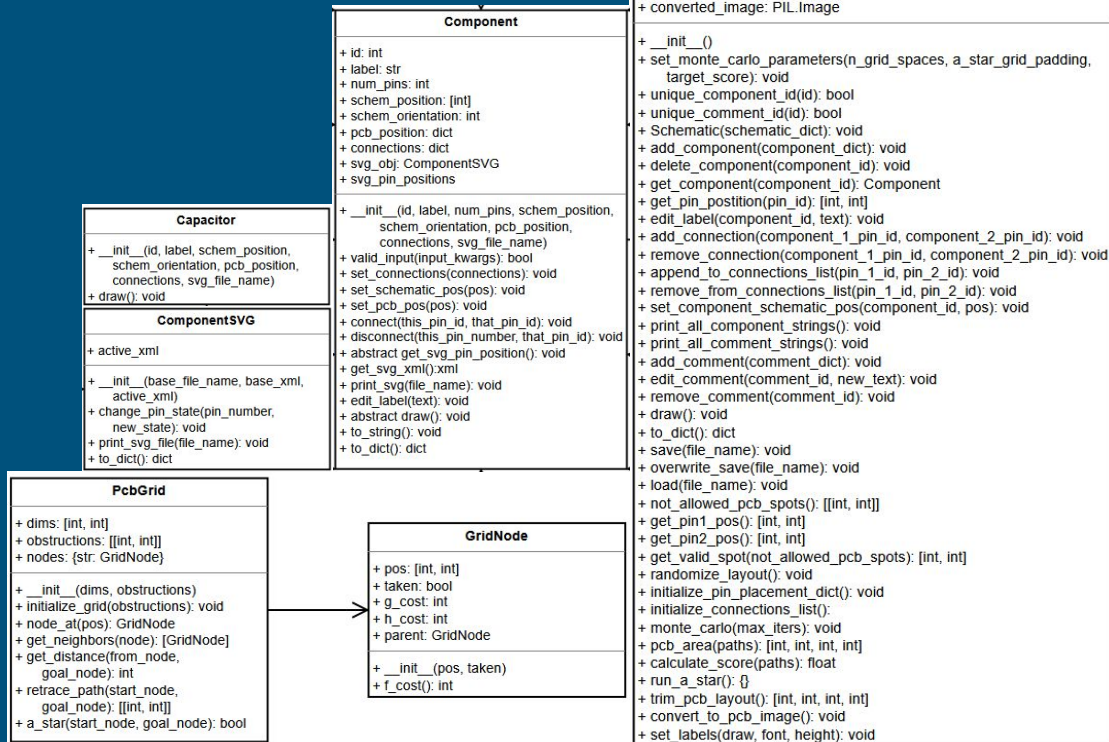
Save



Save As

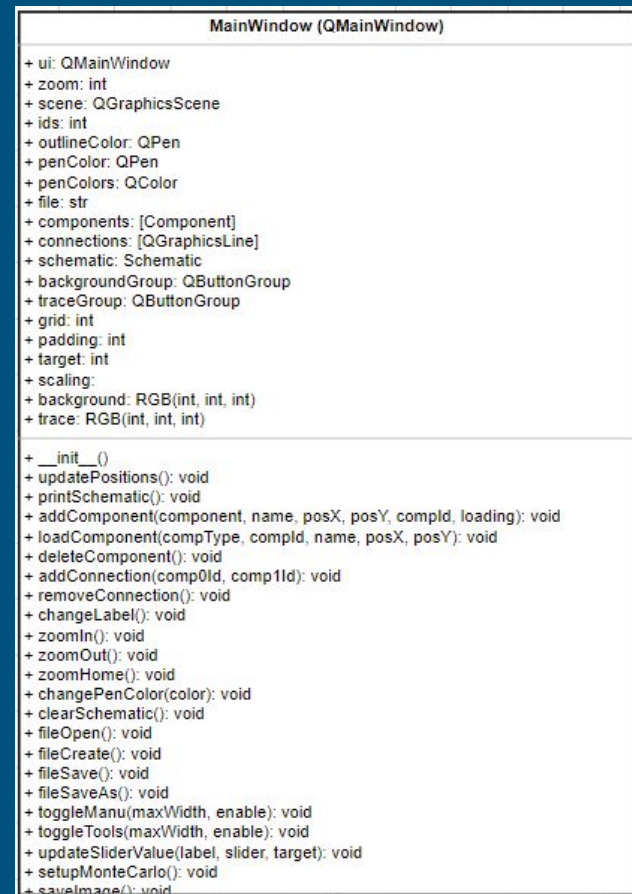
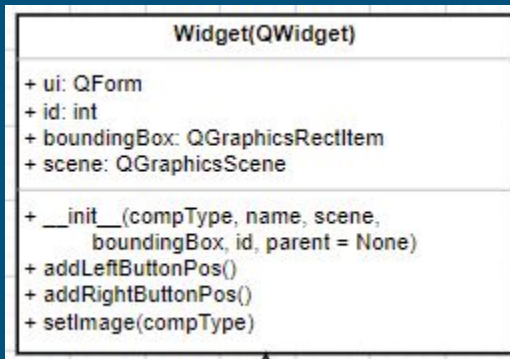
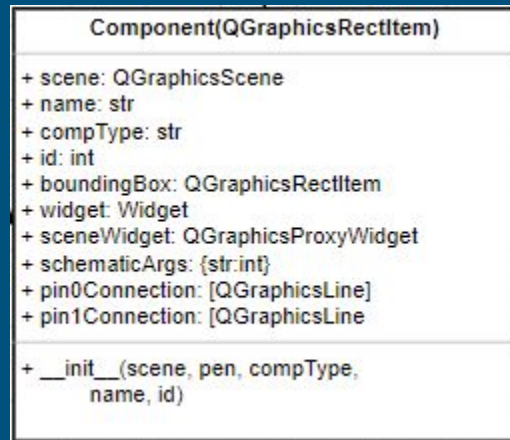
Backend Class Diagram

[Backend Class Diagram Link](#)



GUI Class Diagram

Class Diagram



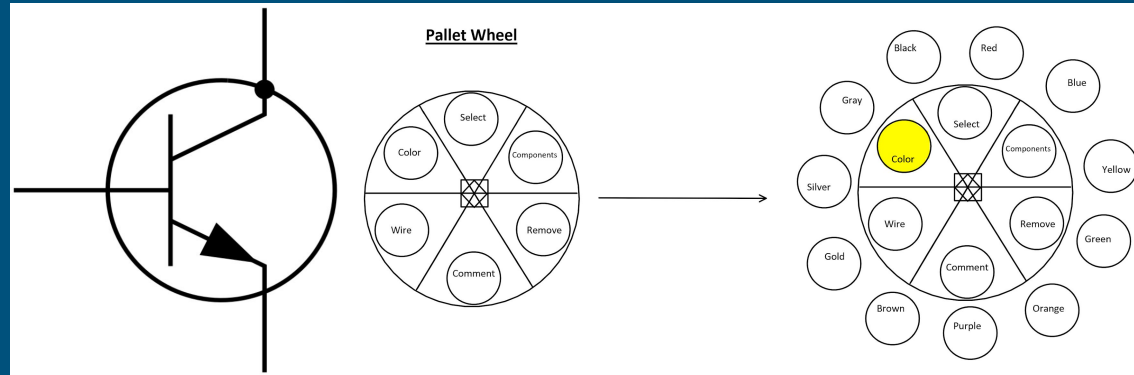
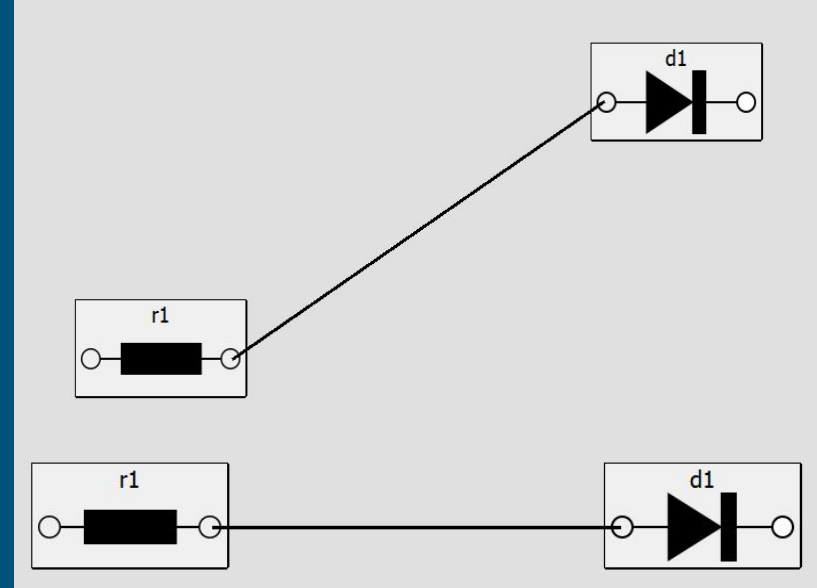


Demonstration

Future Works

Due to time constraints, some of the more complex features had to be omitted from version 1.0. In the event that version 2.0 was to be created, the following features would be implemented:

- Pallet Wheel
- Transistors
- Comments
- Follower Wires





Questions?