DEF/LEF To SVG

Presented By:

Ahmed Abdelrahman 900161054

Zeyad Zaki 900160268

Nouran Flaifel 900160793

GITHUB:
https://github.com/nouranadell/DEF-LEF-Webviewer

Agenda

Overview

Web Page Features

Design Logic

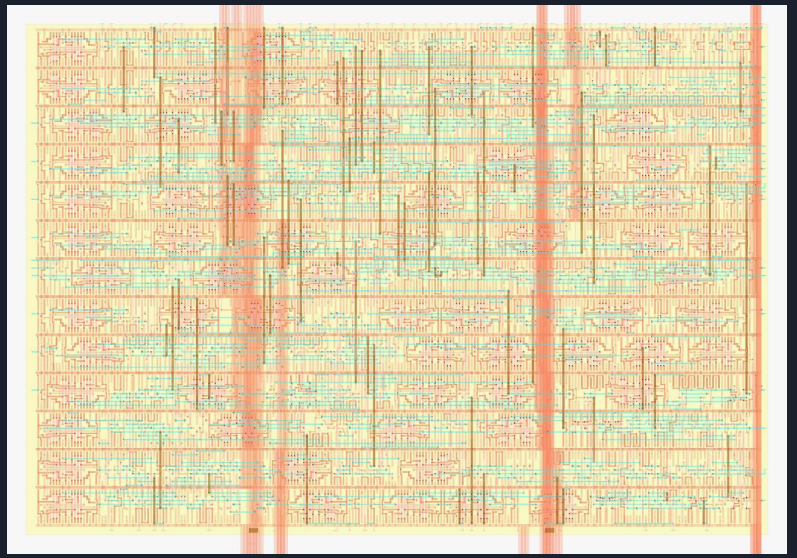
Libraries Used and Resources

Tests

Limitations and further improvements

Overview

- The Objective was to create a web page view that receives the DEF and LEF file paths and outputs the layout as an SVG image like the one shown here, with features that will help the user fully understand the design

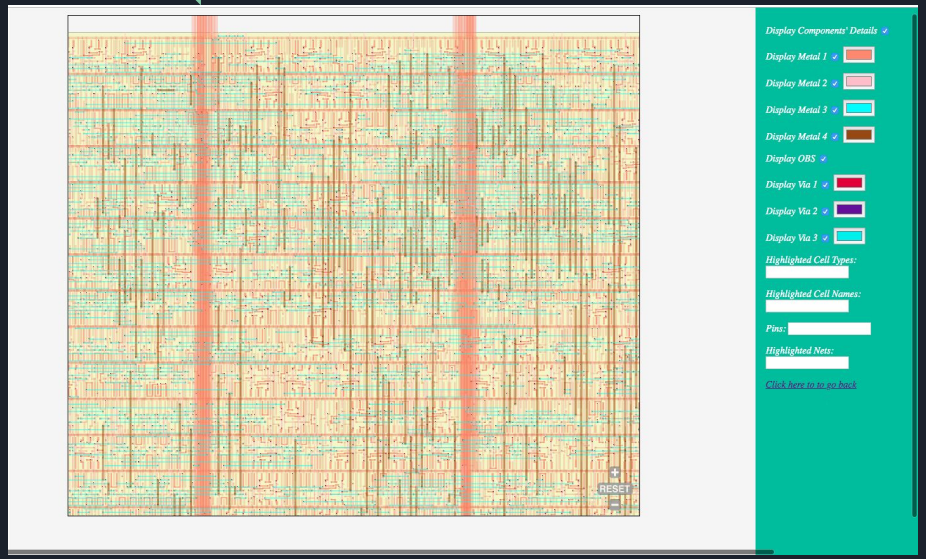


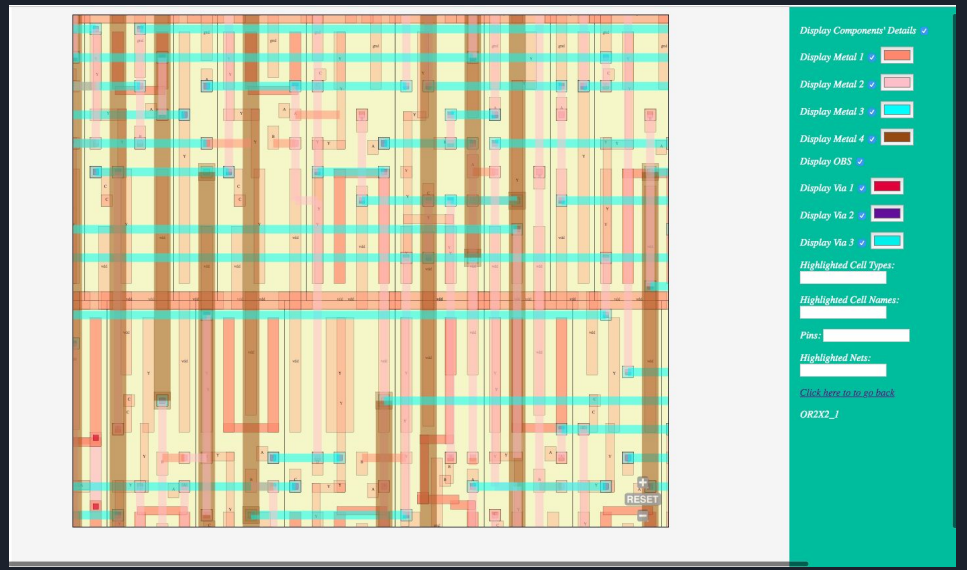
Web Page Viewer Features:

- Zooming and Panning
- Hiding/Showing cell components, metals and/or via
- A searchable drop down list to highlight Nets
- A searchable drop down list to highlight Cells by Cell Names
- A searchable drop down list to highlight Cells by Cell Types
- A searchable drop down list to highlight Pins
- Display the cell name when you hover over the cell
- Changing each metal layer and via colors

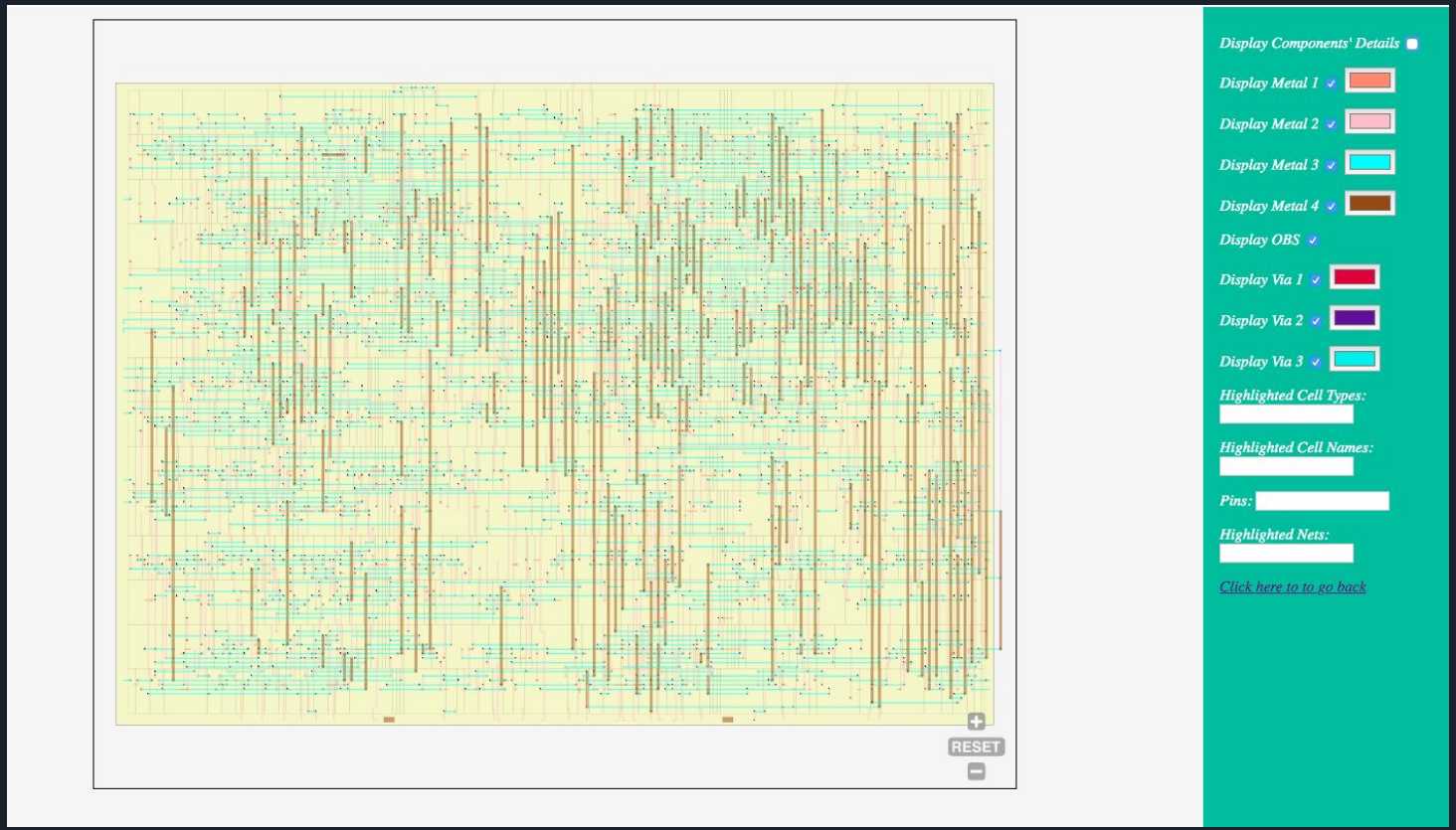
Zooming And Panning

The user can zoom and pan using the buttons or using the trackpad and drag to pan.





Hiding/Showing cell components

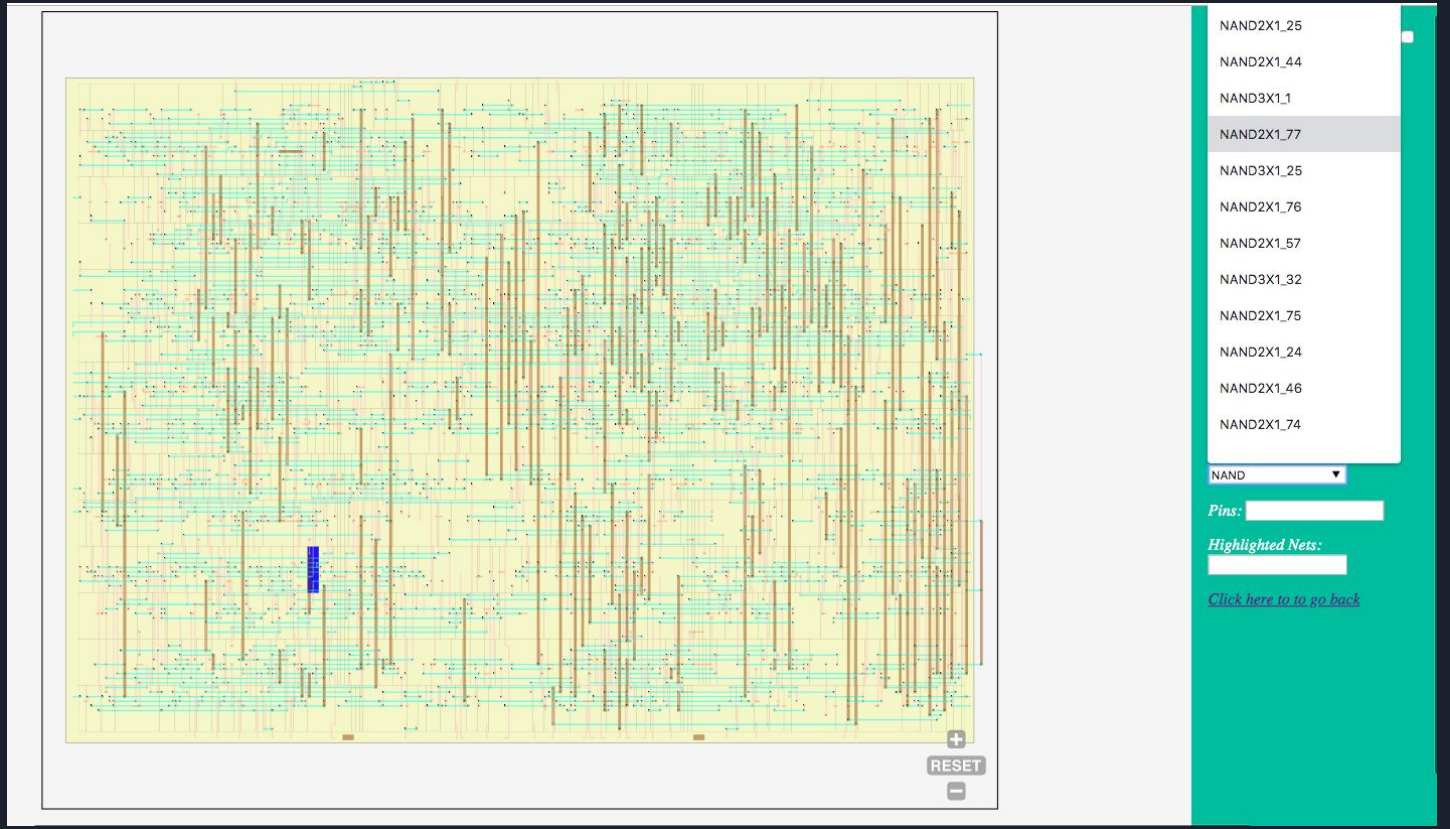


Above is an example hiding cell components and only displaying all the metals and vias

A searchable drop down list to highlight Cell Types

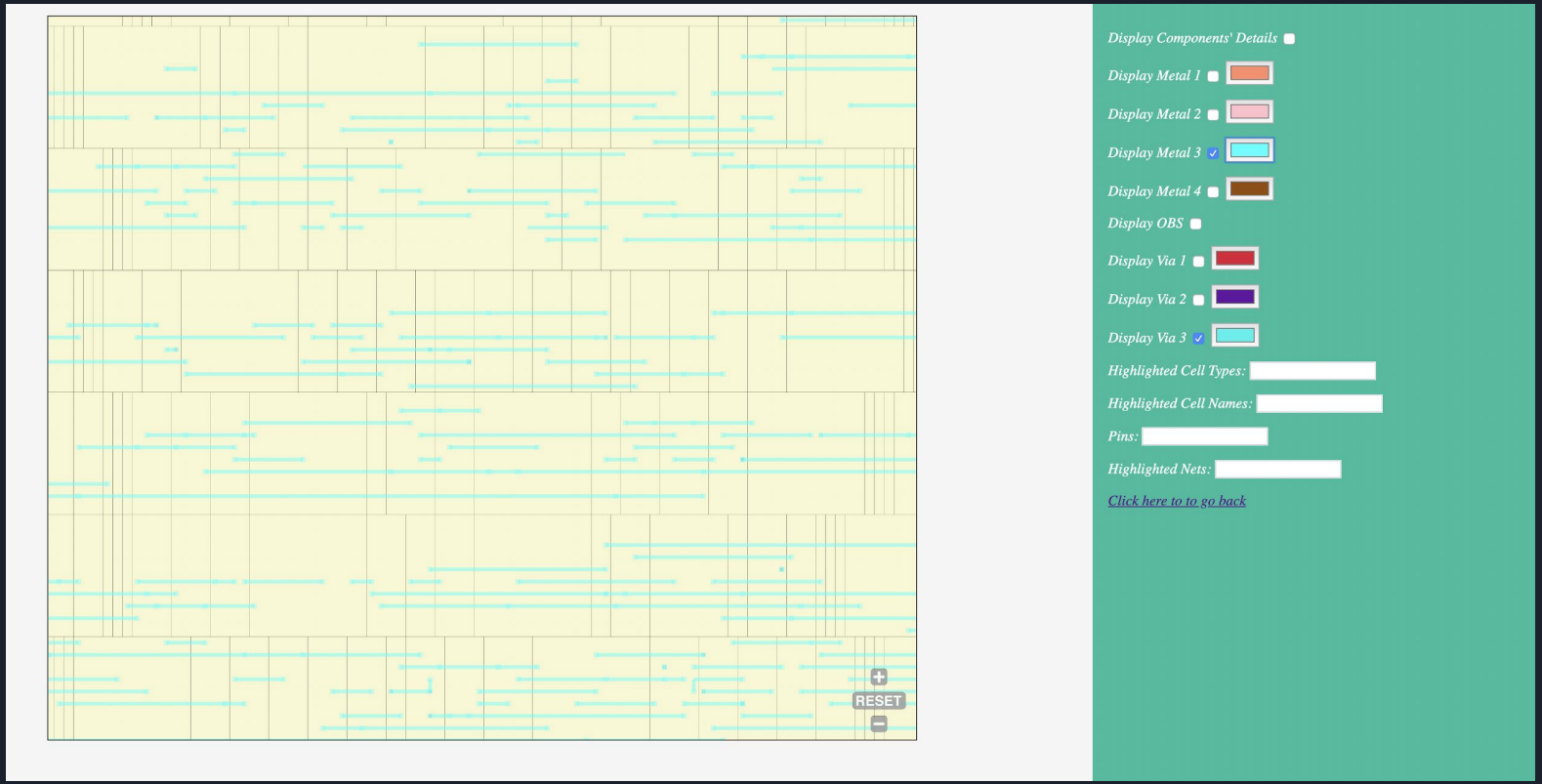
Above is an example of searching for DFF in the cell types and highlighting all cells of that type

A searchable drop down list to highlight by Cell Name



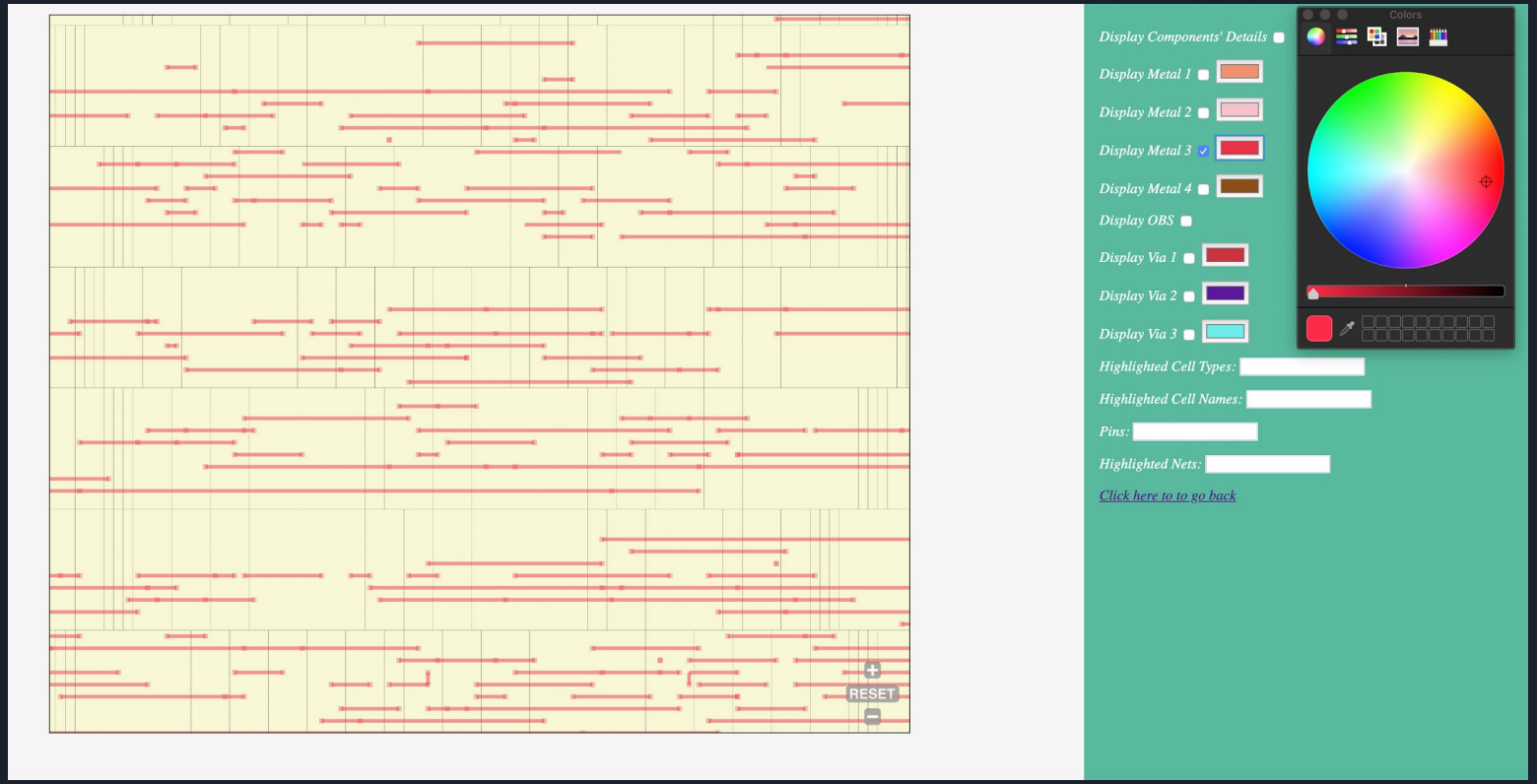
Above is an example of searching for NAND in the cell names and highlighting NAND2X1_77

Hiding/Showing metal and via layers



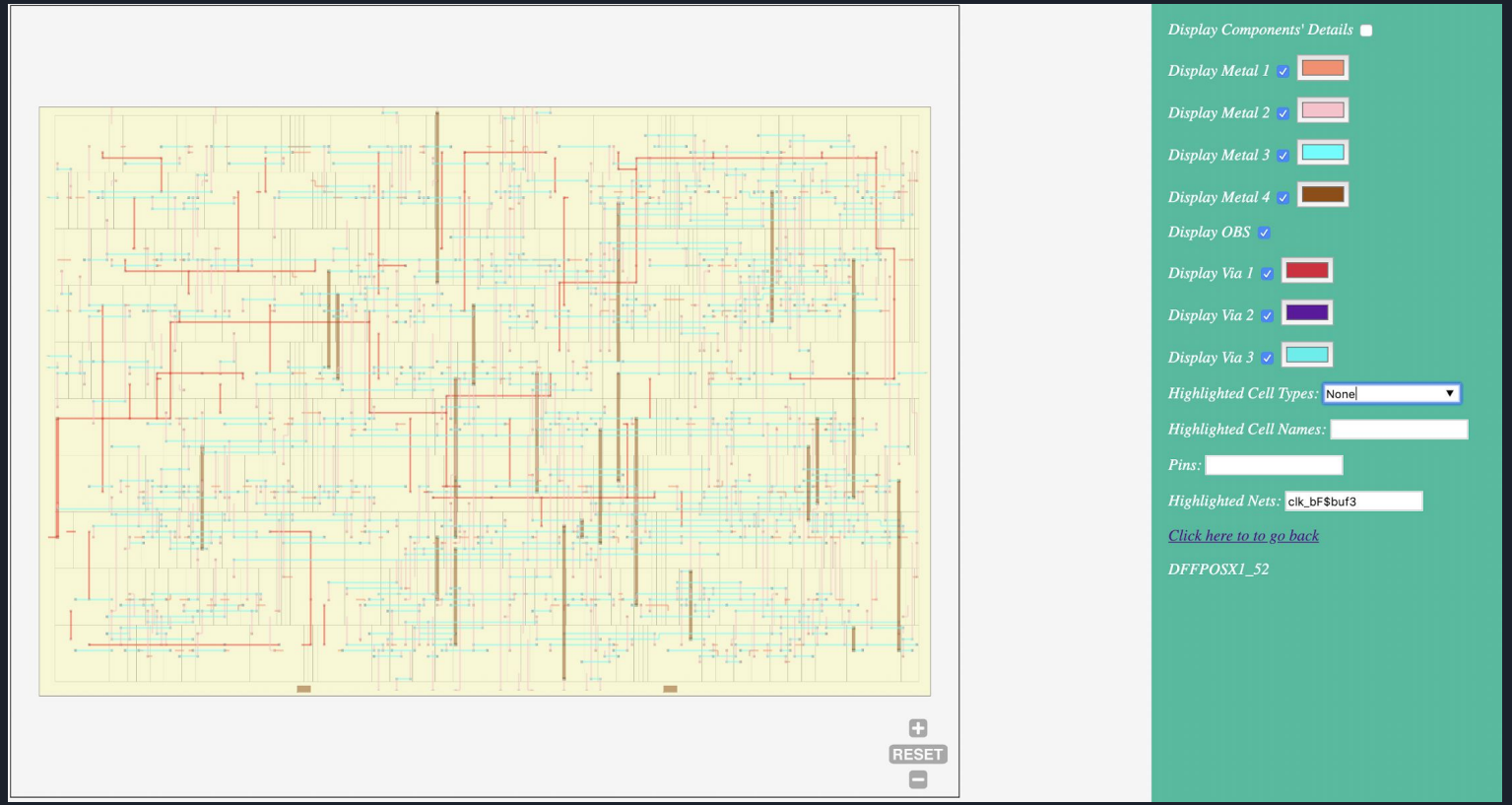
Above is an example of showing only metal3 and via3 layers

Changing metals and vias Colors



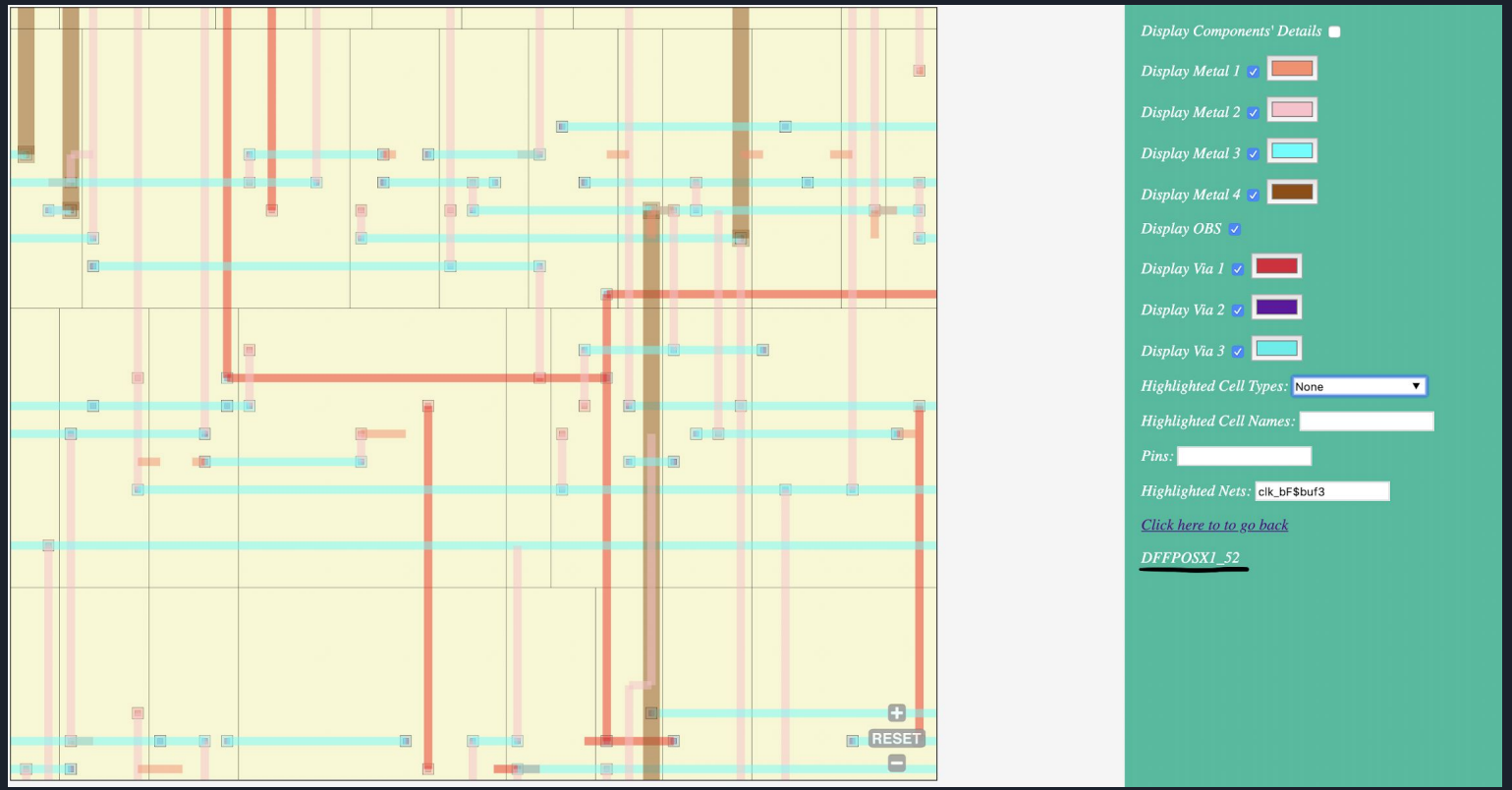
Above is an example of changing metal3 color to red

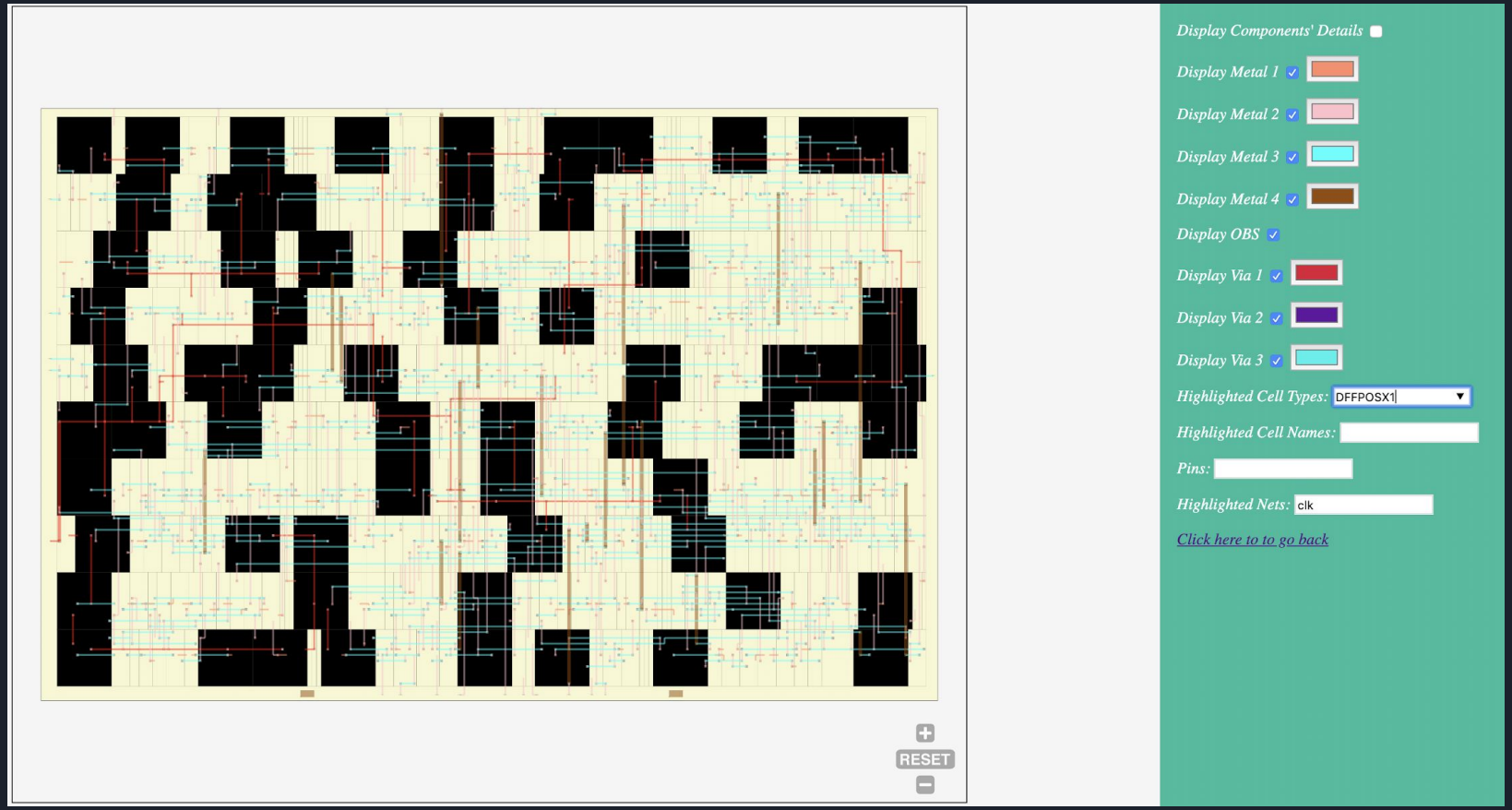
A searchable drop down list to highlight Nets



Above is an example of highlighting all clk nets

Displaying Cell names upon hovering





Above is an example of highlighting all clk nets and all cells of type DFFPOSX1

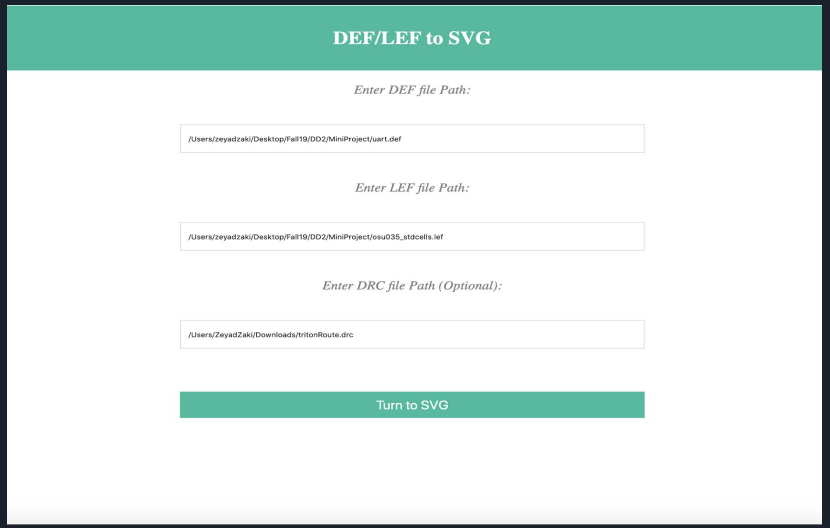
DRC Violations

- The web app gives the user the choice to add a DRC file if available.
- Since there were no associated DRC files for the DEF files we ran the tests on the highlighting of the DRC violations was not implemented
- However the backend now parses the DRC file (if it was provided) for future implementations.
- Below is an example of how the sample DRC file was parsed; the file was parsed into 5 main structs; violation_pos1, v_pos2, v_types, v_srcs, v_layers

```
['(352.01,271.695)', '(352.01,271.695)', '(352.01,271.695)', '(352.49,271.695)', '(352.01,271.695)', '(352.49,271.695)', '(0.33,33.515)',
['(352.15,271.835)', '(352.15,271.835)', '(352.63,271.835)', '(352.63,271.835)', '(352.63,271.835)', '(352.63,271.835)', '(0.483,33.655)',
['0', '0', '0', '0', '0', '0', '2', '2', '2', '2', '2', '0', '0']
[' _04984_ _06208_', ' _04984_ _06208_', ' _04984_ _06208_', ' _04984_ _06208_', ' _04984_ _06208_', ' _04984_ _06208_', ' vram_din[7]', '
['met1', 'met1', 'met1', 'met1', 'met1', 'met1', 'li1', 'li1', 'li1', 'li1', 'li1', 'met2', 'met2']
```

How to Build & Run using github desktop

1. Clone repository and choose the path where you want to download the repo.
2. Open the downloaded folder using Pycharm
3. Configure python interpreter to python 3.7
4. Import Libraries: Flask, Matplotlib, drawSvg
5. Add configuration - choose flask server - target type and choose app.py as your target
6. Run program and open in local host
7. Provide DEF, LEF and DRC (optionally) file paths respectively in the Home screen



Design Logic

- DEF/LEF file paths are inputted by the user
- Python program parses and processes the files to generate the svg file
- Flask Framework used to act a server between the webpage files and python backend code.
- HTML /CSS used to create the website and style
- Front-end scripting was done using Javascript

Libraries Used and Resources

- Languages Used:
  - Python
  - HTML
  - CSS
  - Javascript
- Python Libraries:
  - drawSvg
  - matplotlib.pyplot
  - numpy
- Resources:
  - https://github.com/trimcao/lef-parser
  - LEF/DEF Language Reference
  - File formats for Benchmarks and Contest Deliverables
  - https://github.com/ariutta/svg-pan-zoom

Tests

- The app ran 9 different .def files but all were using the same library (.lef file)
- All tests gave positive results and all features were working normally in all cases
- The app was tested on two different browsers (Google Chrome and Safari)
- All the test files are provided in the GH repository in the tests folder
- No testing for the drc violations highlighting (implemented in future modifications)

Limitations and Further Improvements

- Different algorithms could be used to implement the web features other than brute-force algorithms to enhance the app's speed
- Highlighting DRC violations on the viewer
- Running more tests with different .lef files
- Allowing the user to download the .svg file from the web app

Thank you!