Chandler Griscom
LeTourneau University
COSC4973
27 Oct 2017

CircuitLib Design Specifications Document

This document sets forth design specifications for the initial version of CircuitLib, a graphical user interface that allows users to visually experiment with various common electric circuits. The requirements and feature set may change for subsequent releases.

Environment

CircuitLib is a GUI application that will be developed in Python 2.7 and TkInter. The software shall exist as a cross-platform, portable, standalone set of Python scripts with a launcher. Supporting files contained in the program directory.

Packaging Hierarchy

The program directory shall be arranged in the following format:

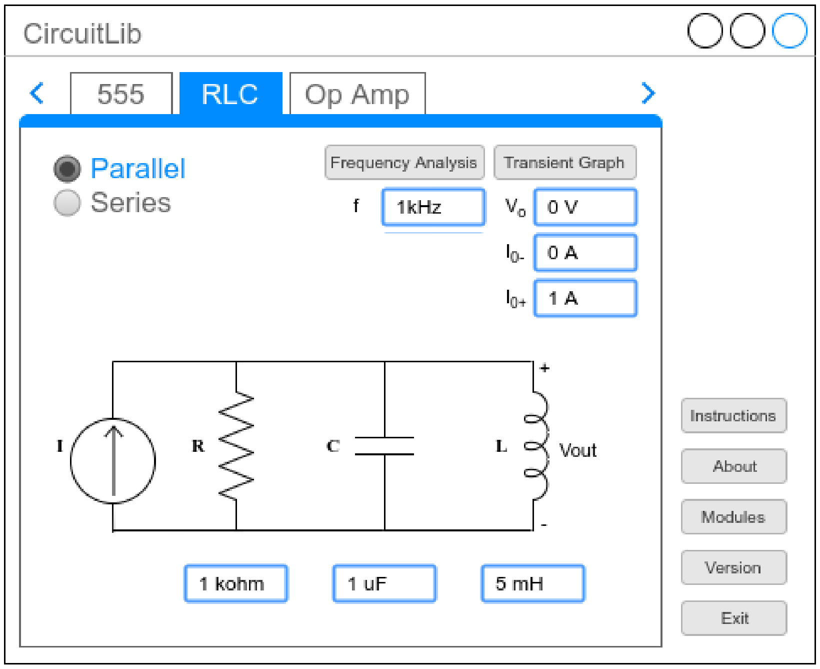
```
CircuitLib/
CircuitLib/CircuitLib.py
CircuitLib/modules/
CircuitLib/modules/{moduleID}/
CircuitLib/modules/{moduleID}/module.py
CircuitLib/modules/{moduleID}/tab.py    (optional)
CircuitLib/modules/{moduleID}/{supporting files}
```

The main Python script is CircuitLib.py. The top level directory could contain some additional executables (.exe or .sh) to launch the script on various platforms such as Windows and Linux.

Graphical User Interface

The Graphical User Interface will comprise a series of tabs nested in the main application window. Next to the tabs box there will be a series of buttons for non-circuit related tasks. Those buttons will initially include:

- Instructions – Open a dialog containing program operation instructions.
- About – Display info about the creator of the program and give a GitHub or website link.
- Modules – Show installed modules (demonstrated later).
- Version – Display the version of the core application.
- Exit – Exit the program.



The tab list above the tab box has two buttons – left and right – for scrolling through the list of tabs in case it ever extends beyond the window boundaries.

Each tab in the tab box shall contain graphics, parameters, and buttons related to a certain circuit arrangement. The exact content will depends on each circuit configuration, but will generally include:

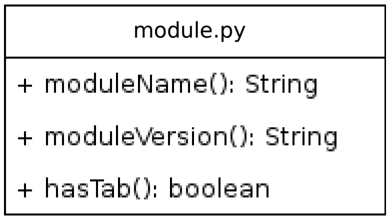
- Tab Title – A short description of the circuit (i.e. RLC).

- Radio Buttons – These will be used to select common variations of the circuit, i.e. series and parallel for RLC.
- Graphic – A graphical (vector or high resolution PNG) representation of the circuit.
- Parameters – Text fields where numbers and units may be entered to specify timing and component values.
- Output Buttons – Buttons to recalculate relevant output shown in Labels, or pop up graphs generated based on the input parameters.

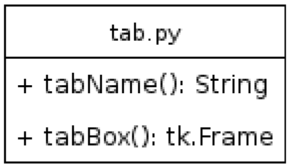
Modules

The application is built in a dynamic modular format so that Pythonistas may be able to quickly add their own circuits and libraries to the application without modifying the main GUI.

When the application loads, it will dynamically search for Python modules in the modules directory. Each module.py file found will be added to a list and will function as a callable library. The UML structure of a module.py file is as follows:

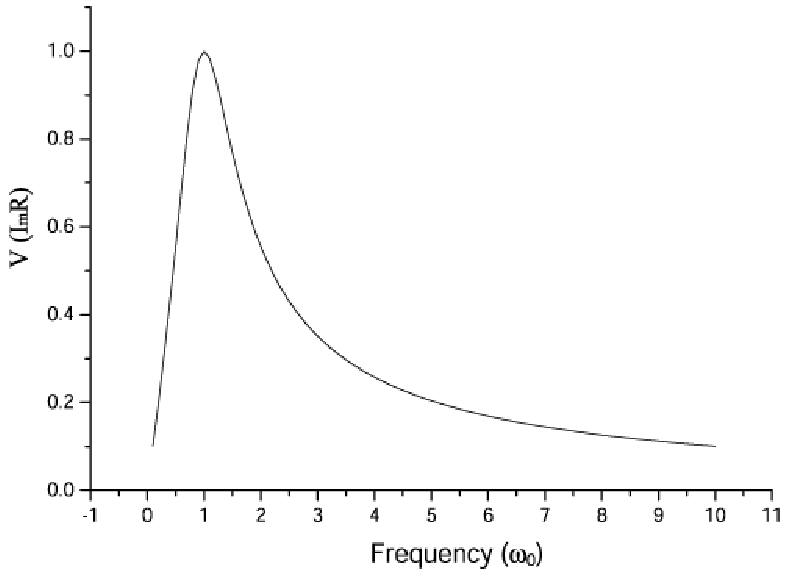


Optionally, a module may include a tab.py file which adds a tab to the tab box in the GUI. The UML structure of a tab.py file is as follows:

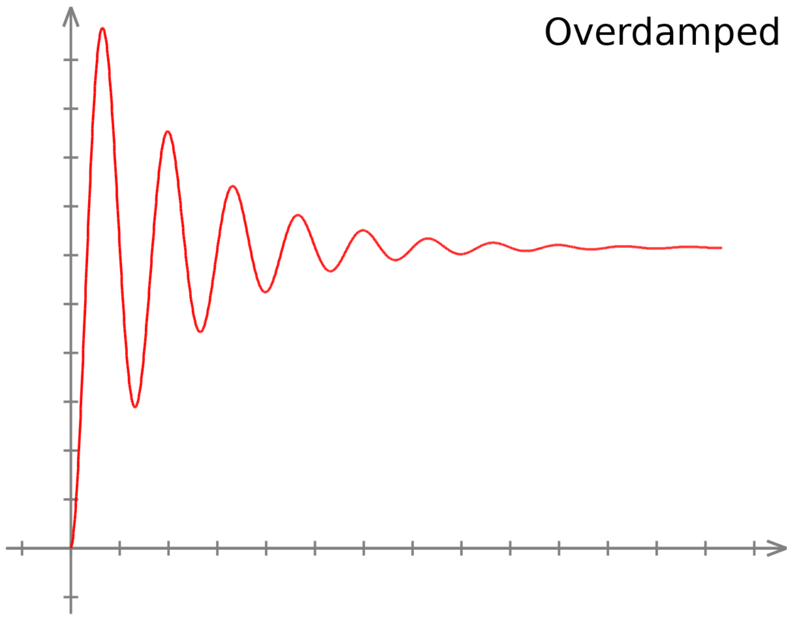


RLC Module Structure

This specific module will have two modes, series and parallel. Using analytical circuit analysis methods, the “Frequency Response” option shall produce a graph:



The “Transient Response” button will produce a graph of the output over time given circuit input at a certain t_{0-} and t_{0+} magnitude. The output window will specify the type of response (overdamped, underdamped, or critical).



As time permits, an additional feature (or alternate version of the module) for calculating numerical solutions for the graph may be added. It would demonstrate the power series solution to ordinary differential equations and provide an input for tweaking the step size used in the approximation.