



Circuit Solver

SemiVi LLC, Switzerland.

Salient features

The circuit solver takes circuit netlists in spice format and performs desired set of simulations. Salient features –

- Supports non-linear circuit solver,
- Parameterized electronic components,
- Allows user-defined parameters and math functions,
- Python scripting enabled –
 - Parameters editing and processing results in python,
 - Use of data-processing and optimization libraries.
- *User-defined* functional and behavioural models in Python.
- Coupled electro-thermal solver enabled in DC, AC, and transient.

Supported circuit components

The circuit solver supports the following components –

- Linear components – R, L, C, and mutual-inductance,
- Voltage- and current-controlled switches,
- Lossless and lossy transmission lines, RC network,
- Non-linear components – *Diode*, *JFET*, *MOSFET*, *MESFET*, and *BJT*,
- AC, DC, and transient I/V sources,
- *Dependent sources* – VCVS, CCVS, VCCS, CCCS.
- *Subcircuits* – can be parameterized and math functions can be defined.
- *Heat generation* – enabled in passive components.
- *Heat conduction* – use thermal *R* and *C*.

Collaboration

We are open to collaborating with the users to add customized non-linear components.

Functional compact models

Functional modeling interface: A quick way of modeling functionality of a digital chip (e.g. drivers, signal-processors) in the analog circuit.

```
class SimpleDriver (cs.functionalmodel):
    def updateOutputPinVoltages (self,
                                  isOutputPin, inputV, time):
        ...
    return outV
```

Behavioural compact models

Behavioural modeling interface: Define a custom compact model of non-linear devices in Python.

```
class MyDiode (cs.behaviouralmodel):
    def getDerivativesAndPinCurrents
        (self, inputV, time):
        ...
    return np.append (outdI, outI)
```

Python Interface

- Python-interface allows modifying model parameters, analysis, and result extraction.

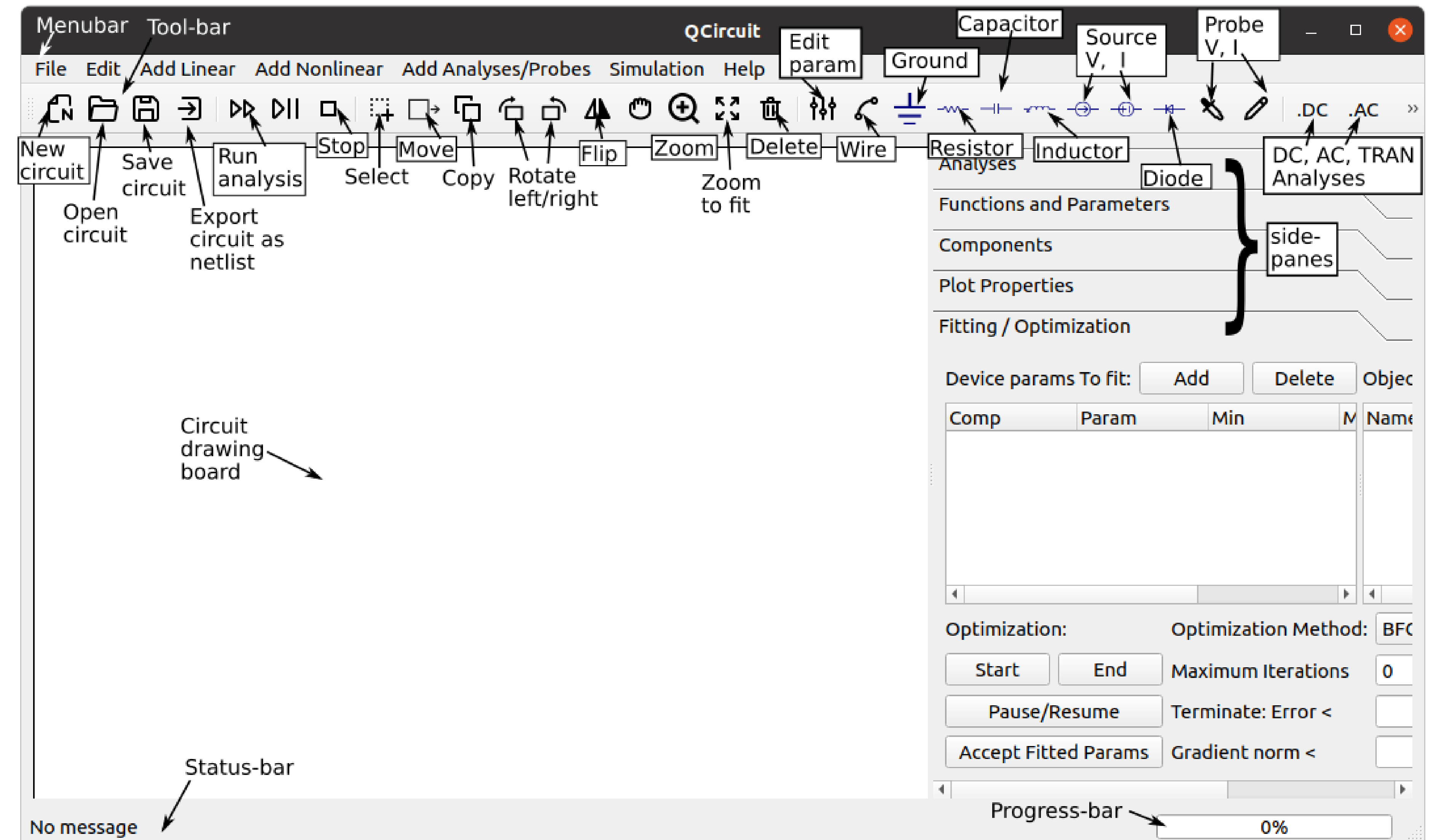
```
import circuitsolver as cs
p = cs.circuit()
p.readSpiceCircuitFile(...)
R3Now = p.getComponentParamVal(...)
p.setComponentParamVal(...)
```

Circuit Optimizer

- Built-in circuit-optimizer: *compact model calibration* and *performance optimization*.

```
f = cs.optimizer(p)
f.setOptimizationAlgorithm(...)
f.addOptimizationParameter(...)
f.addExperimentalDataAndPower(...)
f.optimize()
```

Circuit Draw - GUI



Circuit Draw

Circuit draw provides a user-friendly GUI for –

- Circuit drawing,
- Setting analyses and simulations,
- Results visualization,
- Circuit optimization.

License Purchase

Send the organization details along with –

- Node-locked licenses: mac-id of the machine.
- Server licenses: mac-id of the server.

User-manual and examples

A detailed user-manual provides –

- Equations corresponding to each of the components,
- Parameters of various components and their usage,
- Various analyses and default parameters,
- Python interface and various functions therein.

Contact Information

- Web: <http://www.semivi.ch>
- Email: info@semivi.ch
- Phone: +1 (000) 111 1111