Qucs Help Documentation

Release 0.0.18

Qucs Team (2014)

20.01.2015

Inhaltsverzeichnis

Contents:

KAPITEL 1

Erste Schritte

Qucs (gesprochen: kju:ks) ist ein Schaltungssimulator mit graphischer Benutzeroberfläche. Das Programm dient der Durchführung verschiedenartiger Schaltungssimulationen, wie z.B. Gleichspannung, S-Parameter, etc. Dieses Dokument soll einen kurzen Überblick über die Nutzung von Qucs geben.

Wenn Qucs zum ersten Mal gestartet wird, wird ein Verzeichnis mit dem Namen ".qucs" innerhalb des eigenen Heimatverzeichnisses erzeugt. Jede Datei wird in dieses Verzeichnis oder eines der darin enthaltenen Unterverzeichnisse gespeichert. Nachdem Qucs gestartet ist, erscheint das Hauptfenster, das so aussieht wie in Abbildung 1 dargestellt. Auf der rechten Seite befindet sich die Hauptarbeitsfläche (6), der die zu simulierende Schaltung, die Graphen, etc. beeinhaltet. Mit den Tabulatorschaltflächen (5) überhalb dieses Bereiches kann man schnell zu allen weiteren geöffneten Schaltungen, etc. umschalten. Auf der linken Seite neben dem Qucs Hauptfensters befindet sich eine weiterer Bereich (1), dessen Inhalt sich in Abhängigkeit vom ausgewählten Tabulator-Reiter auf der linken Seite ändert: "Projekte" (2), "Inhalt" (3) und "Komponenten" (4). Nachdem Qucs gestartet wurde, ist die Tabulatorschaltfläche "Projekte" vorselektiert. Wenn es das erste Mal ist, dass Qucs benutzt wird, so ist dieser Bereich leer, da noch kein Projekt existiert. Durch Betätigung des "Neu"-Knopfes überhalb dieses Bereiches (1) erscheint ein Dialog. In diesem Dialog wird der Name des Projektes definiert. Für das erste Projekt wird der Name "ErstesProjekt" eingegeben und der Dialog mit Drücken des Knopfes "Ok" verlassen. Daraufhin erzeugt Qucs ein Projekt-Verzeichnis innerhalb des Verzeichnisses "~/.qucs", für dieses Beispiel "ErstesProjekt_prj". Jede Datei, die zu diesem Projekt gehört, wird in diesem Verzeichnis gespeichert. Das neue Projekt wird unverzüglich geöffnet, was man aus dem Eintrag des Titels im Hauptfenster entnehmen kann. Die Tabulatorschaltfläche auf der linken Seite wird auf "Inhalt" (3) umgeschaltet, wo der Inhalt des gegenwärtig geöffneten Projekts angezeigt wird. Bis jetzt ist noch kein Dokument erzeugt worden. Deshalb wird der "Speichern"-Kopf in der Knopfleiste betätigt oder aus der Menüzeile die Auswahl "Datei->Speichern", um das gegenwärtig noch unbenannte Dokument zu speichern, welches im Hauptfenster angezeigt wird (6). Im nachfolgenden Dialog wird der Name für das Dokument festgelegt. Als Beispielname wird "ersteSchaltung" eingegeben und der Kopf "Speichern" betätigt.

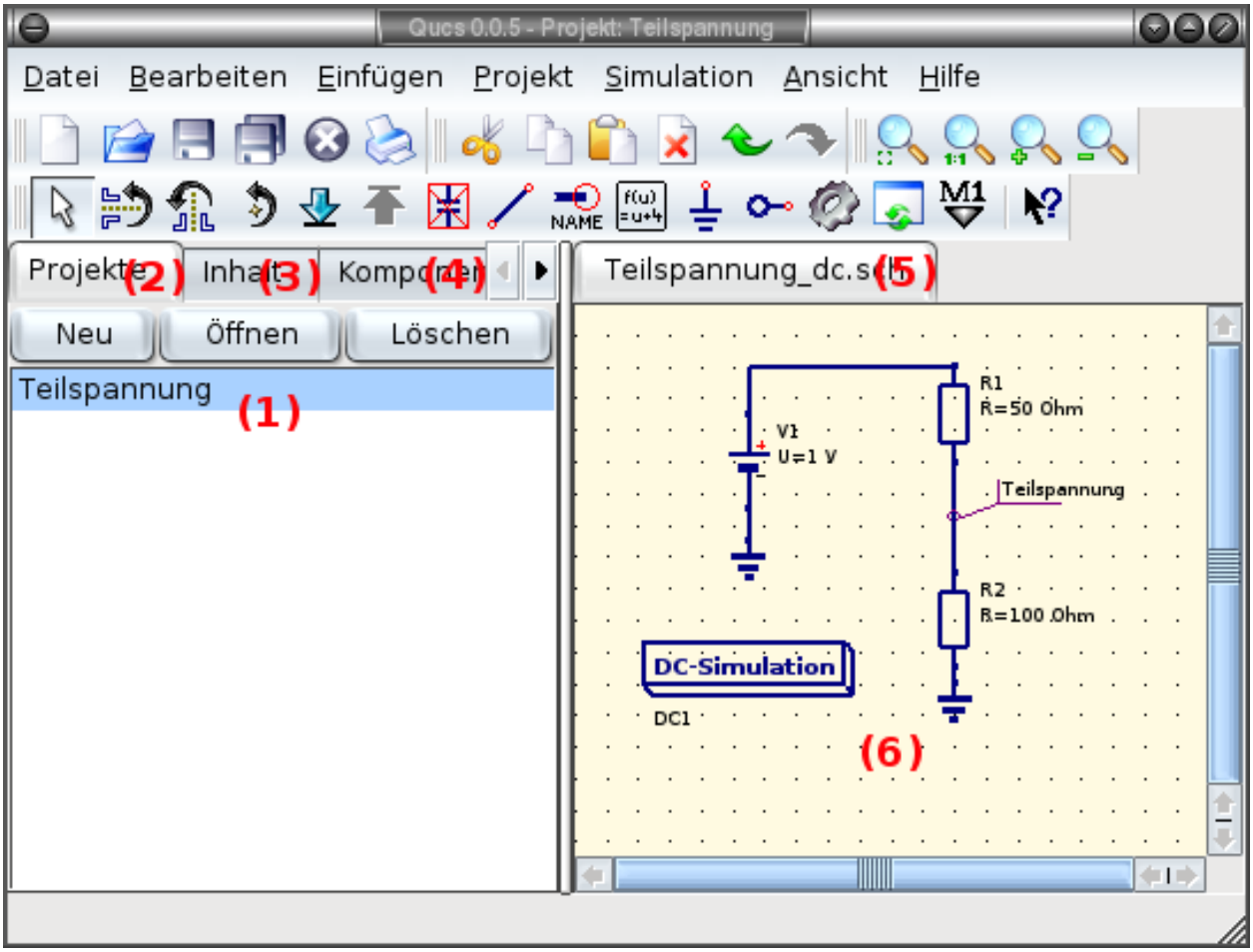


Abbildung 1 - Qucs Hauptfenster

Es soll eine einfache Gleichspannungssimulation durchgeführt werden, z.B. soll die Schaltung aus Abbildung 1 analysiert werden. Dazu wird die Komponenten-Schaltfläche ((4) in Abbildung 1) aktiviert. Dort kann man einen Blättertext erkennen, aus dem die Komponentengruppe ausgewählt wird. Unterhalb des Blättertextes befinden sich dann die in dieser Gruppe zusammengefassten Komponenten. Hier wird “diskrete Komponenten” als Blättertext für das Beispiel ausgewählt und das Bauteil “Widerstand” ausgewählt. Durch Bewegen des Mauszeigers in die Hauptarbeitsfläche (6) kann man die Komponente (hier Widerstand) platzieren. Das Drücken der rechten Maustaste bewirkt ein Drehen um jeweils 90° der ausgewählten Komponente. Das Betätigen der rechten Maustaste hingegen bewirkt ein Absetzen der ausgewählten Komponente auf der Hauptarbeitsfläche. Der Vorgang wird anschließend für alle weiteren in Abbildung 1 dargestellten Komponenten durchgeführt. Die DC-Spannungsquelle kann unter dem Blättertexteintrag “Quellen” gefunden werden, das Massesymbol unter “diskrete Komponenten” und die gewünschte Simulationsart wird über die Auswahl des Blättertextes “Simulationen” und Hinzufügen der Komponente “DC-Simulation” zur Schaltung gewährleistet. Um die Parameter des Widerstandes zu verändern erfolgt ein Doppelklick auf diesen. Es erscheint ein Dialog, in dem der Widerstandswert verändert werden kann. Auf der rechten Seite wird der Wert 100 Ohm eingegeben und mit “Ok” bestätigt.

Um die Komponenten miteinander zu verbinden, im Beispiel die Widerstände und die Spannungsquelle, wird der Kopf “Draht einfügen” betätigt oder der Menüeintrag Einfügen->Draht (STRG-E) ausgewählt. Der Cursor wird auf einen Port (bei unverbundenen Ports markiert durch kleine rote Kreise) bewegt. Durch einen Klick auf den Port wird mit einem Draht begonnen. Jetzt wird der Mauszeiger auf den Port einer weiteren Komponente bewegt und durch einen Klick wird der Draht verbunden. Die zwei Komponenten sind nun verbunden. Falls man die Richtung des Drahtverlaufs ändern möchte, kann man, bevor man einen weiteren Port anklickt, noch auf die Hauptarbeitsfläche klicken, um so eine Richtungsänderung bzw. Ecke im Drahtverlauf zu bewirken. Außerdem kann man einen Draht auch unverbunden enden lassen. Dazu wird einfach im Hauptarbeitsfenster ein Doppelklick mit der linken Maustaste

durchgeführt.

Zum Schluss, doch nicht weniger wichtig, muss man noch den Knoten beschriften, an dem Qucs die Spannung berechnen soll. Aus der Knopf-Leiste wird der Knopf Verbindung-/Knoten-Bezeichnug einfügen (STRG-L) (oder man benutzt den Menüeintrag: Einfügen->Verbindungsbezeichnung). Der zu beschriftende Draht wird angeklickt. Es öffnet sich ein Dialog, der Name wird festgelegt. Hier wird "Teilspannung" für das vorliegende Beispiel eingegeben und die Dialogbox mit "Ok" verlassen. Nun sollte die Schaltung wie in Abbildung 1 dargestellt, aussehen.

Um die Simulation zu starten, wird der "Simulieren"-Knopf betätigt (oder Simulation->Simulieren (F2-Taste)). Es öffnet sich ein Fenster, das den Fortschritt bei der Simulation anzeigt. Nachdem die Simulation erfolgreich beendet wurde, wird die Datenanzeige geöffnet. Normalerweise geschieht das so schnell, dass man nur ein kleines flackern erkennt. Nun muß man ein Diagramm platzieren, um sich die Simulationsergebnisse graphisch darstellen zu lassen. Auf der linken Seite befindet sich die Komponenten-Gruppe "Diagramme", welche automatisch ausgewählt wurde. Durch die Auswahl des Eintrags "Tabelle" und das Bewegen des Mauszeigers über die Hauptarbeitsfläche, kann durch einen Klick mit der linken Maustaste in diese, ein Tabelle plaziert werden. Es erscheint ein Dialog, in dem ausgewählt werden kann, was aus der Simulation dargestellt werden soll. In dem linken Bereich kann man den Namen "Teilspannung" erkennen, den man zuvor definiert hat. Ein Doppelklick auf den Namen plaziert diesen auf der rechten Seite des Fensters. Der Dialog wird über die Betätigung des "Ok"-Knopfes verlassen. Nun kann man das Simulationsergebnis begutachten: 0,666667 Volt. Wunderbar, die erste Simulations ist vollbracht. Man darf sich nun selber beglückwünschen.

KAPITEL 2

Erste digitale Schritte

Qucs ist auch eine grafische Benutzeroberfläche für die Durchführung von digitalen Simulationen. Dieses Dokument enthält einen kurze Beschreibung, wie dies vonstatten geht.

Qucs verwendet das Programm FreeHDL (http://www.freehdl.seul.org), um digitale Simulationen durchzuführen. Das bedeutet, dass sowohl das FreeHDL-Packet, als auch der GNU C++ Compiler auf dem Computer installiert sein müssen.

Der Unterschied zwischen analogen und digitalen Simulationen ist nicht sehr groß. Falls also Erste Schritte bereits gelesen wurde, ist es jetzt recht einfach, auch eine digitale Simulation zum Laufen zu bringen. Es soll beispielhaft die Lokigtabelle eines einfachen logischen UNDs berechnet werden. Mit Hilfe der digitalen Komponenten in der Kombobox des Komponenten-Reiters auf der linken Seite sollte es gelingen, den Schaltplan, der in Abbildung 1 zu sehen ist, nachzubauen. Der Digitalsimulations-Block kann unter den anderen Simulationen gefunden werden.

Die Digitalquellen S1 und S2 sind die Eingänge, der Knoten mit der Bezeichnung Ausgang ist der Augang. Nach dem Starten der Simulation öffnet sich die Seite für die Datenvisualisierung. Das Diagramm Logiktabelle wird darauf platziert und die Variable Ausgang anschließend eingefügt. Jetzt wird die Logiktabelle eines UNDs mit zwei Eingängen angezeigt. Gratulation, die erste Digitalsimulation ist fertig!

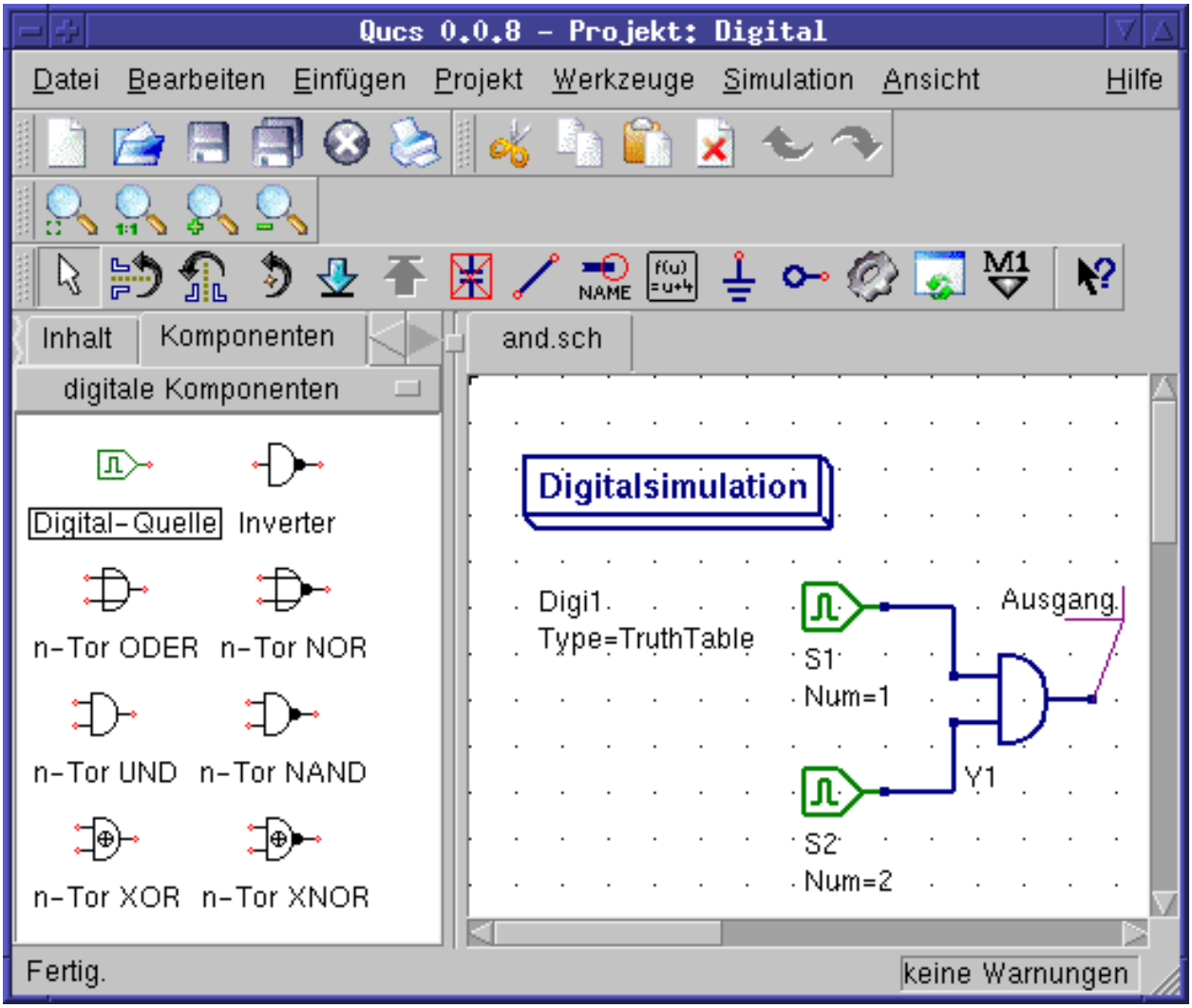


Abbildung 1 - Qucs Hauptfenster

Die Logiktabelle ist nicht die einzige Digitalsimulation, die Qucs durchführen kann. Es ist weiterhin möglich, ein beliebiges Signal in einen Schaltkreis einzuspeisen. Die Ausgangssignale können dann mit Hilfe eines Zeitdiagramms angezeigt werden. Um das zu erreichen, muss der Parameter Type des Simulations-Blocks auf TimeList gesetzt werden und die Zeitdauer der Simulation in den darauf folgenden Parameter eingetragen werden. Die Digitalquellen haben jetzt eine andere Bedeutung: Sie geben eine beliebige Bitsequenz aus, indem ein Ausgangszustand, das erste Bit, (low oder high) und eine Liste von Zeiten, an denen der Zustand gewechselt werden soll, definiert wird. Es ist zu beachten, dass diese Liste nach derem Ende wiederholt wird. Um ein 1GHz Taktsignal mit einem Tastverhältnis von 1:1 zu erzeugen, sieht die erwähnte Liste so aus: 0.5ns; 0.5ns

Um sich die Ergebnisse dieser Simulationsart anzusehen, gibt es den Diagrammtyp Zeitverlaufsdiagramm. Darin können die Ergebnisse aller Knoten Zeile für Zeile angezeigt werden. Viel Spaß dabei...

2.1 VHDL File Component

More complex and more universal simulations can be performed using the component “VHDL file”. This component can be picked up from the component list view (section “digital components”). Nevertheless the recommended usage is the following: The VHDL file should be a member of the project. Then go to the content list view and click on the file name. After entering the schematic area, the VHDL component can be placed.

The last entity block in the VHDL file defines the interface, that is, all input and output ports must be declared here. These ports are also shown by the schematic symbol and can be connected to the rest of the circuit. During simulation the source code of the VHDL file is placed into the top-level VHDL file. This must be considered as it causes some limitations. For example, the entity names within the VHDL file must differ from names already given to subcircuits. (After a simulation, the complete source code can be seen by pressing F6. Use it to get a feeling for this procedure.)

Einführung in die Optimierung

Für die Schaltkreisoptimierung verwendet Qucs ein Programm namens ASCO (http://asco.sourceforge.net/). Es folgt eine kurze Beschreibung, wie man ein Schaltplan dafür vorbereitet, die Optimierung ausführt und die Ergebnisse interpretieren kann. Bevor man diese Funktionalität benutzen kann, muss ASCO auf Ihrem Computer installiert sein.

Die Optimierung eines Schaltkreises ist nicht mehr als die Minimierung einer Kostenfunktion. Das kann entweder die Verzögerungszeit oder die Anstiegszeit in einer digitalen Schaltung, oder die Leistungsverstärkung einer analogen Schaltung sein. Eine andere Möglichkeit ist die Definition des Optimierungsproblems als eine Zusammensetzung von Funktionen, was in diesem Fall zu einem Gütefaktor führt.

Um einen Schaltplan für ein Optimierung vorzubereiten, müssen zwei Dinge hinzugefügt werden: Gleichungen und die Optimierungskomponente. Nehmen Sie den Schaltplan aus Abbildung 1 und verändern Sie es solange, bis Sie den Schaltplan in Abbildung 2 erhalten.

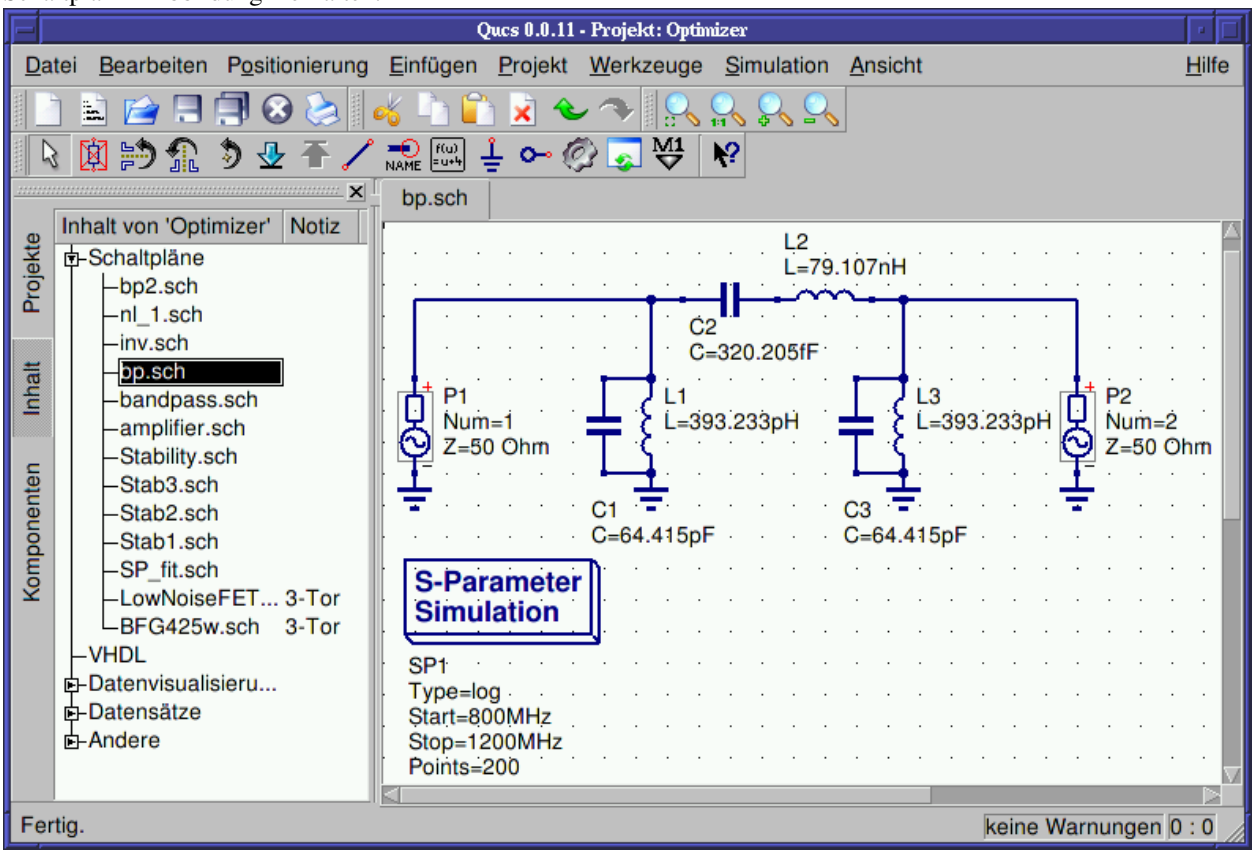


Abbildung 1 - Ursprünglicher Schaltplan.

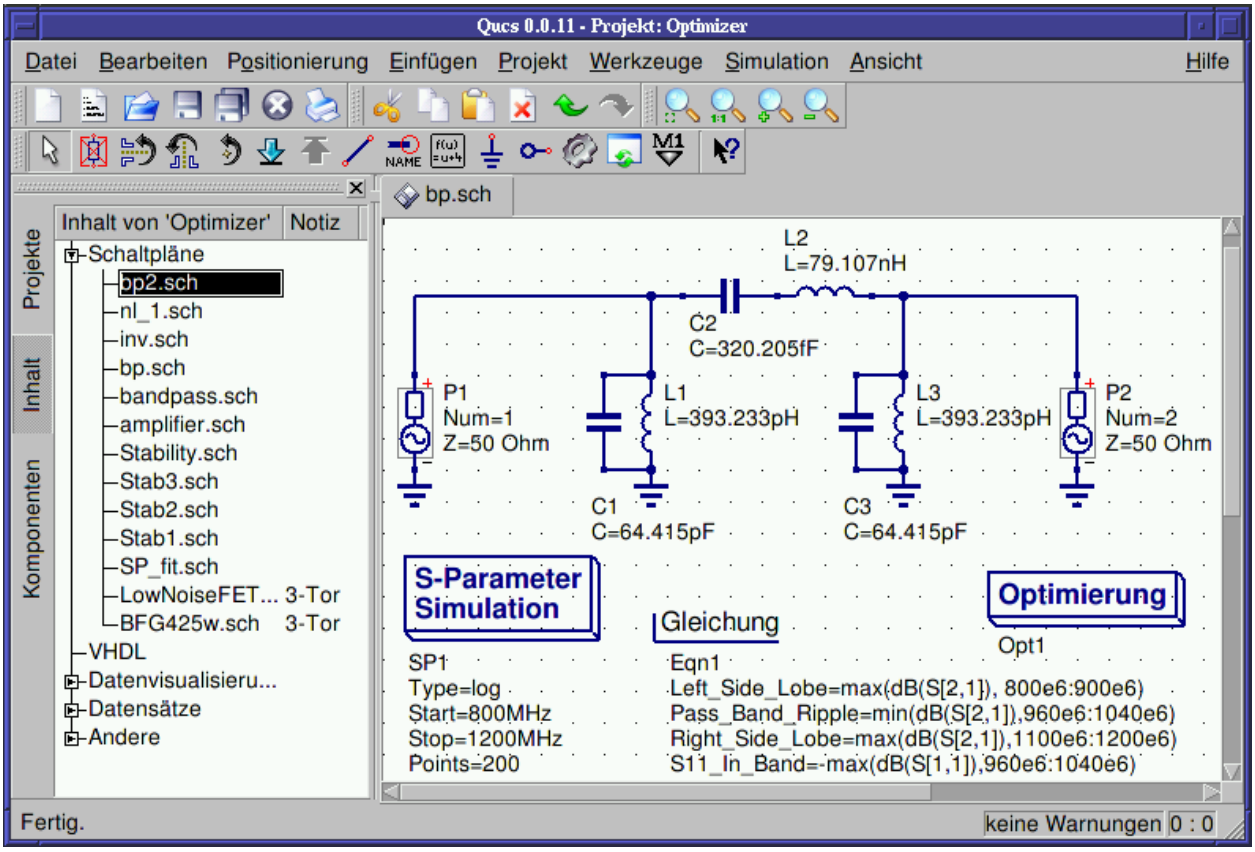


Abbildung 2 - Vorbereiteter Schaltplan.

Jetzt öffnen Sie die Optimierungskomponente und wählen die Algorithmusschaltfläche an. Aus den existierenden Parametern sollte besonders auf ‘Maximale Anzahl der Iterationen’, ‘Constant F’ und ‘Crossing over factor’ geachtet werden. Über- oder Unterschätzung kann zur vorzeitigen Konvergenz des Optimierers in einem lokalen Optimum führen oder auch zu sehr langen Optimierungszeiten.

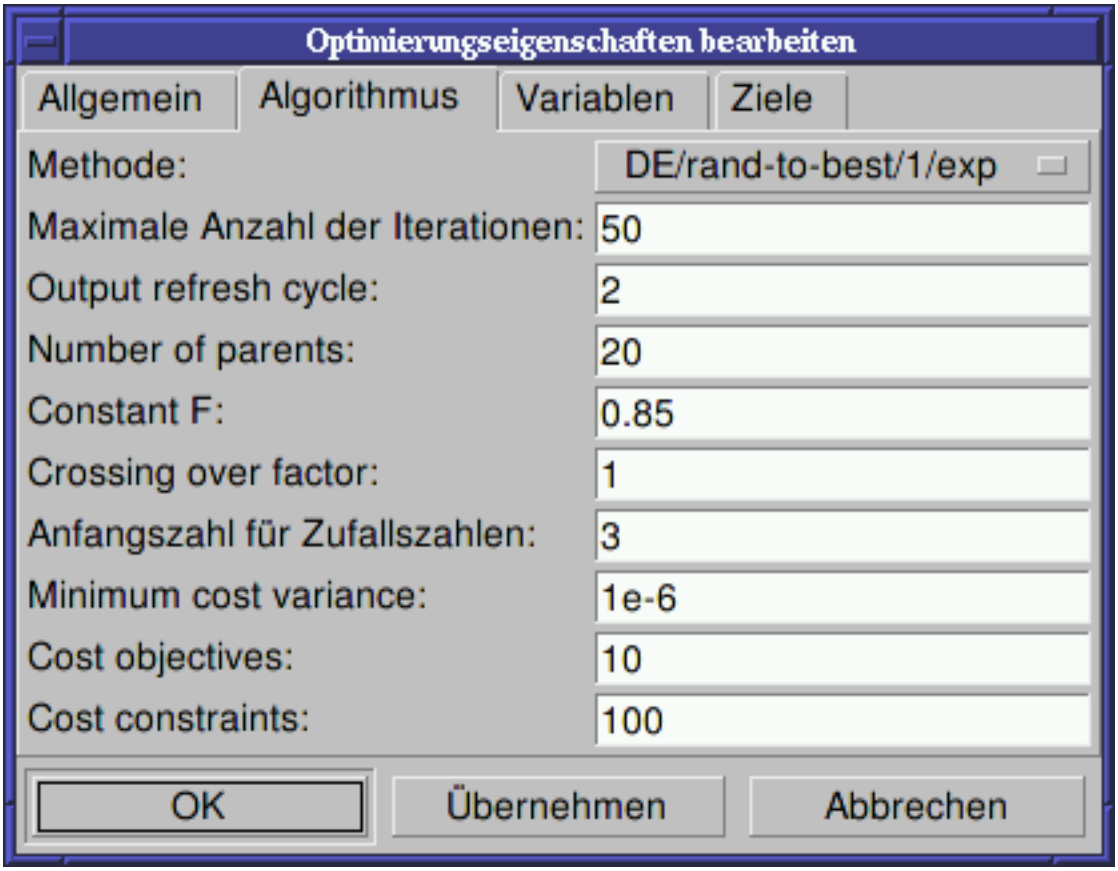


Abbildung 3 - Optimierungsdialog, Algorithmusoptionen.

Die Variablenschaltfläche, wo die Schaltkreiselemente definiert werden, die in einem bestimmten Interval optimiert werden können, ist in Abbildung 4 dargestellt. Die Variablennamen korrespondieren zu den Namen, die in die Komponenteneigenschaften platziert wurden und **nicht** zu den Namen der Komponenten.

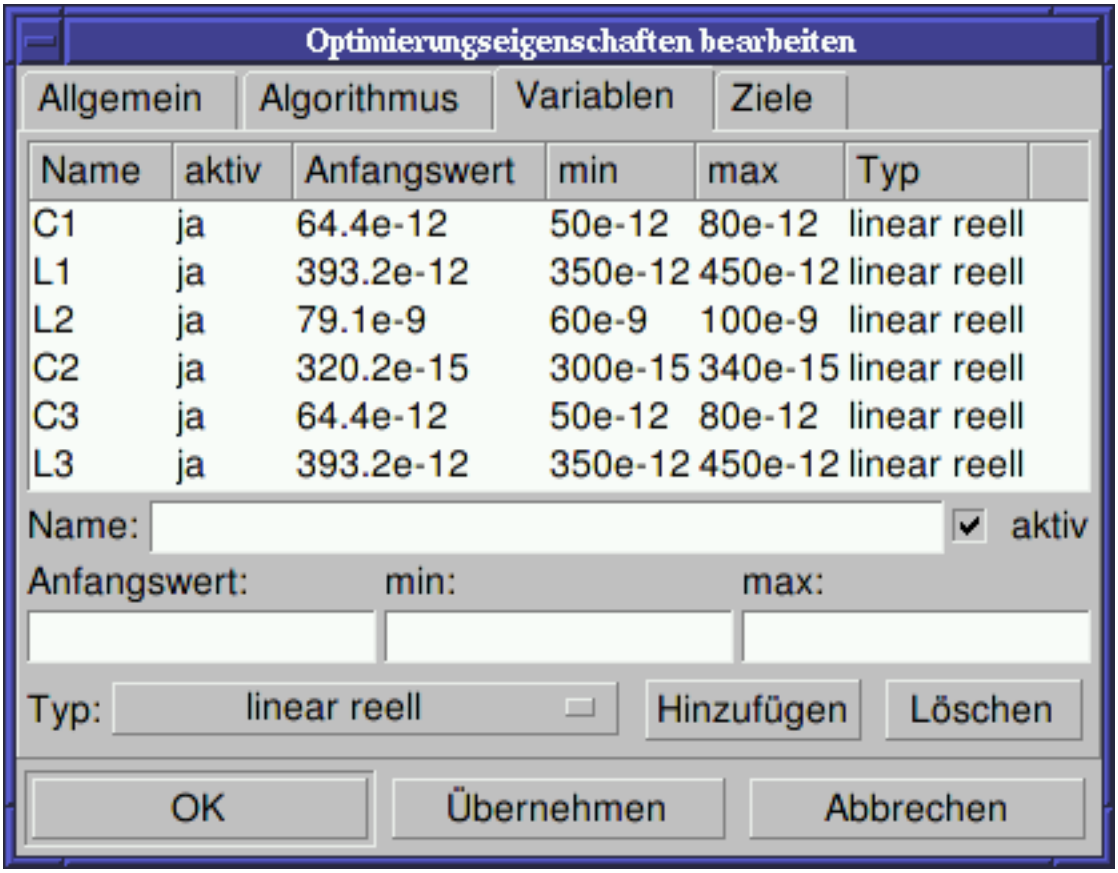


Abbildung 4 - Optimierungsdialog, Variablenoptionen.

Schließlich müssen noch die Ziele der Optimierung (maximieren, minimieren) und Optimierungsgrenzen (kleiner, größer, gleich) in der Zielschaltfläche eingegeben werden. ASCO kombiniert diese Ziele zu einer einzigen Kostenfunktion, die dann minimiert wird.

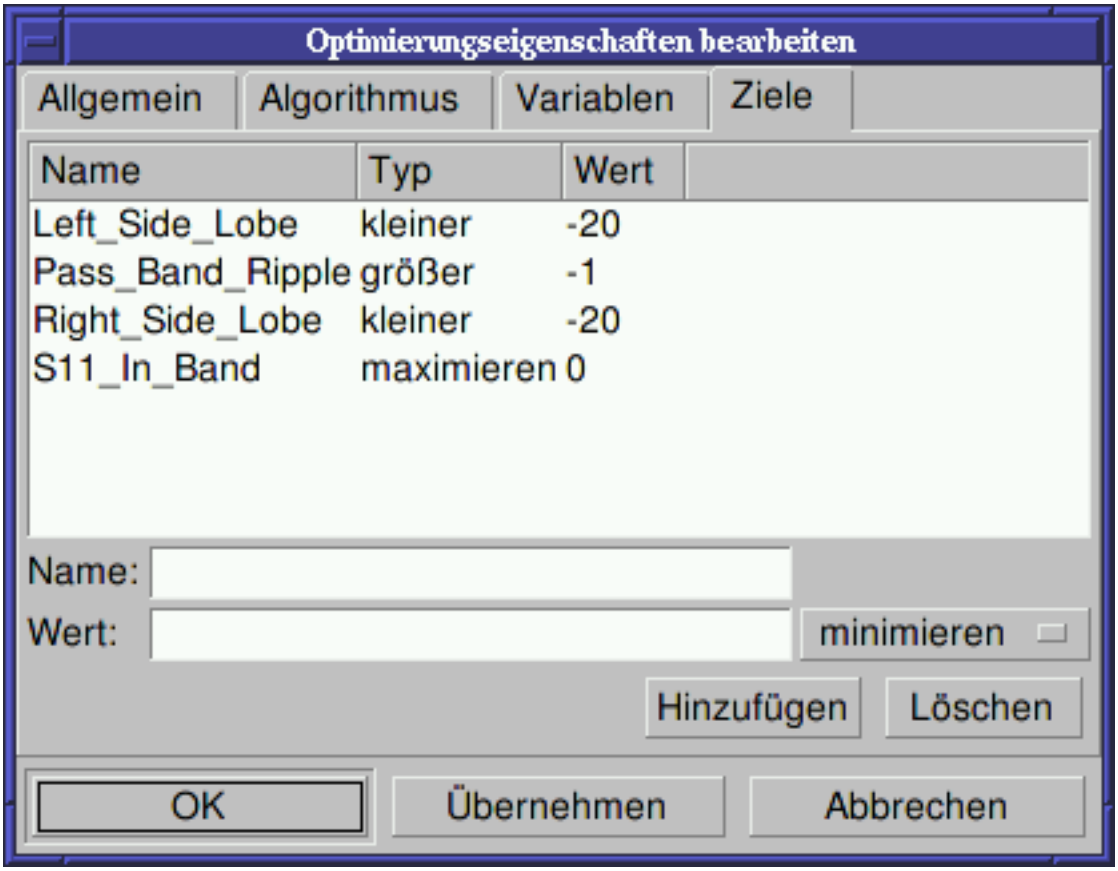


Abbildung 5 - Optimierungsdialog, Zieloptionen.

Der nächste Schritt ist die Veränderung des Schaltplans und die Definition der Schaltkreiselemente, die optimiert werden sollen. Der enstehende Schaltplan wird in Abbildung 6 dargestellt.

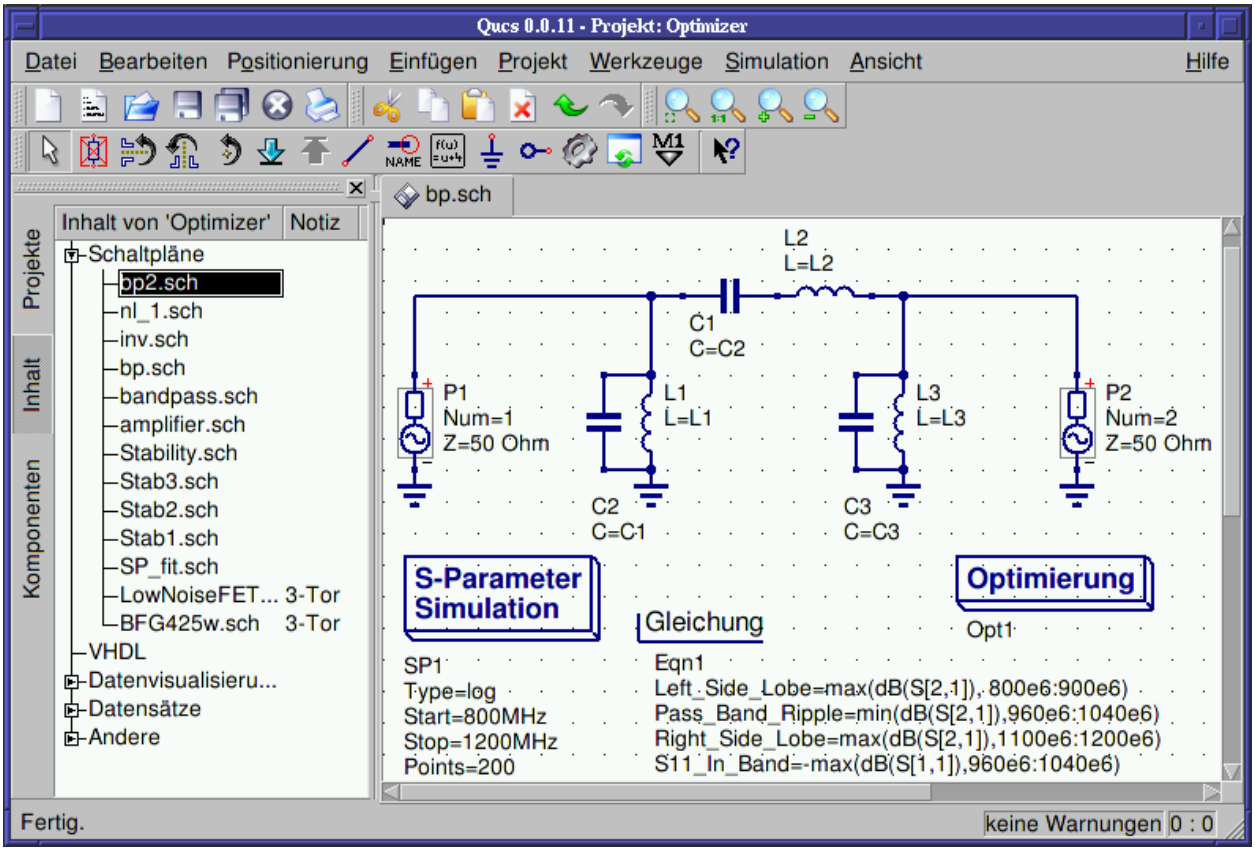


Abbildung 6 - Neues Qucs Hauptfenster.

Der letzte Schritt ist die Ausführung der Optimierung, d.h. das Starten der Simulation durch Drücken von F2. Wenn die Optimierung beendet ist, was auf einem modernen Computer ein paar Sekunden dauert, werden die besten Simulationsergebnisse angezeigt.

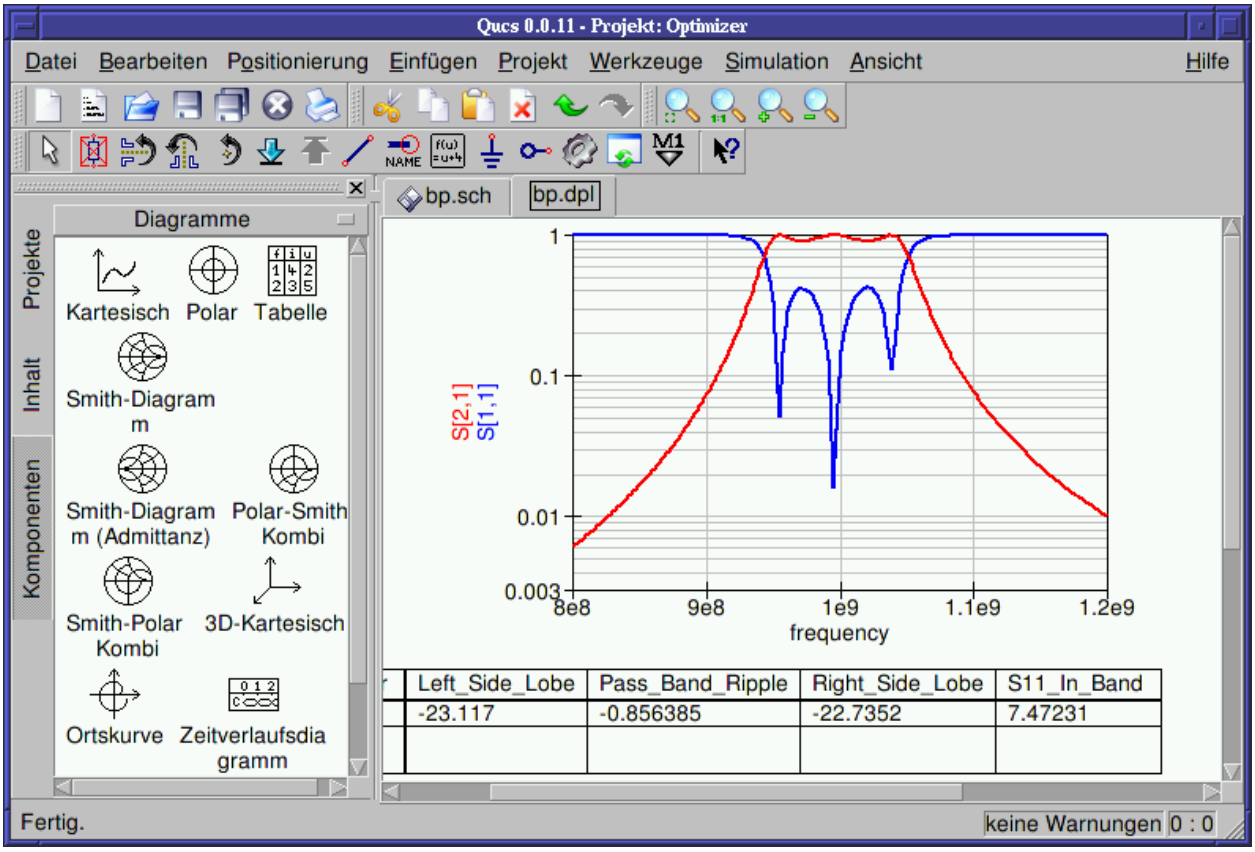


Abbildung 7 - Qucs Ergebnisfenster.

Die besten Schaltkreisgrößen findet man im Optimierungsdialog auf der Variablenschaltfläche. Es sind jetzt die Anfangswerte für jede der eingeführten Variablen (siehe Abbildung 8).

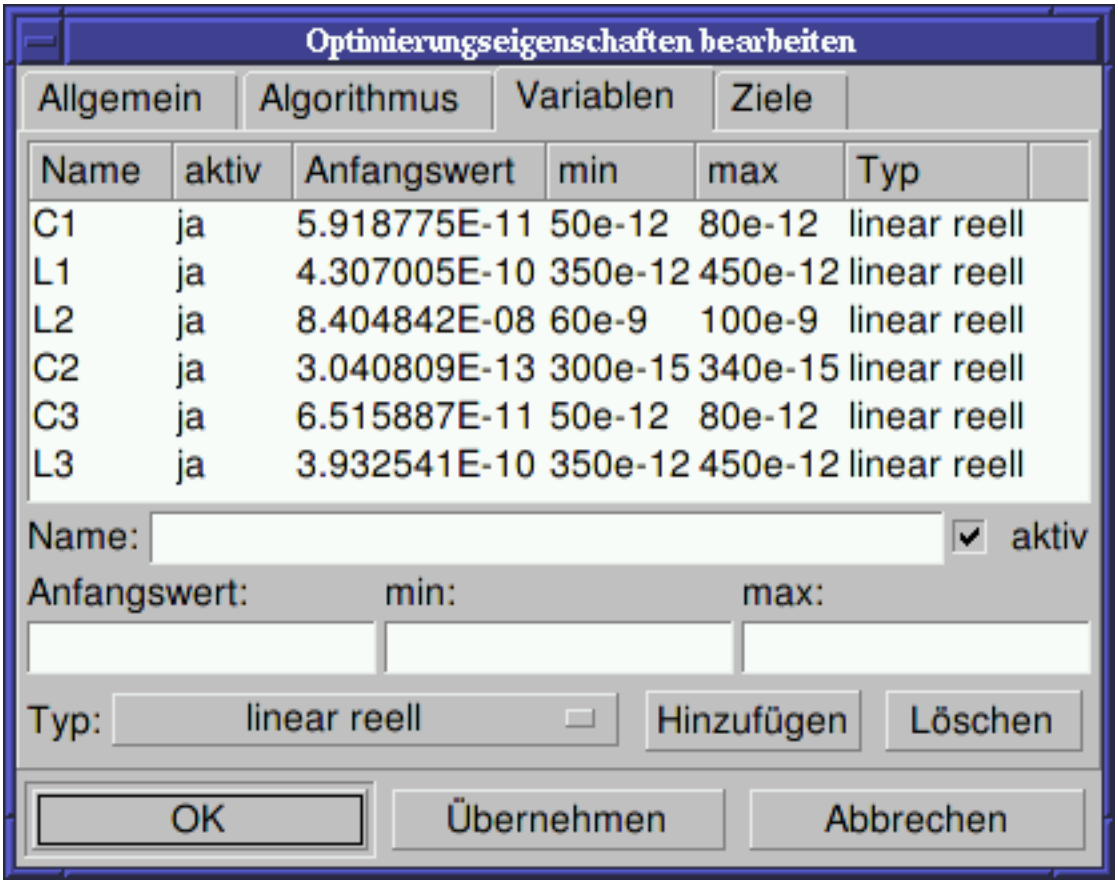


Abbildung 8 - Die optimierten Schaltkreisgrößen.

Getting Started with Octave Scripts

Qucs can also be used to develop Octave scripts (see http://www.octave.org). This document should give you a short description on how to do this.

If the user creates a new text document and saves it with the Octave extension, e.g. 'name.m' then the file will be listed at the Octave files of the active project. The script can be executed with F2 key or by pressing the simulate button in the toolbar. The output can bee seen in the Octave window that opens automatically (per default on the right-hand side). At the bottom of the Octave window there is a command line where the user can enter single commands. It has a history function that can be used with the cursor up/down keys.

There are two Octave functions that load Qucs simulation results from a dataset file: loadQucsVariable() and loadQucsDataset(). Please use the help function in the Octave command line to learn more about them (i.e. type help loadQucsVariable and help loadQucsDataset).

4.1 Postprocessing

Octave can also be used for automatic postprocessing of a Qucs simulation result. This is done by editing the data display file of a schematic (Document Settings... in File menu). If the filename of an Octave script (filename extension m) from the same project is entered, this script will be executed after the simulation is finished.

Kurze Beschreibung der Funktionen

5.1 Generelle Funktionen

(gültig in allen Betriebszuständen)

Mausrad	Bewegt den Mauszeiger vertikal durch den Zeichenbereich. Kann auch zum Bewegen außerhalb der gegenwärtigen Größe genutzt werden.
Mausrad + Umstelltaste	Bewegt den Mauszeiger horizontal durch den Zeichenbereich. Kann auch zum Bewegen außerhalb der gegenwärtigen Größe genutzt werden.
Mausrad + Ctrl-Taste	Vergrößert oder verkleinert den betrachteten Zeichen-Bereich.
drag'n'drop file into document area	(Menüeintrag: Bearbeiten->Auswählen (Esc))

5.2 linke Maustaste

(Menüeintrag: Bearbeiten->Auswählen (Esc))

rechte Maustaste	Wählt das Element aus, das sich unter dem Mauszeiger befindet. Wenn sich dort mehrere Komponenten befinden, so kann man einige Mal klicken, bis die gewünschte Komponente ausgewählt ist.Während man die Maustaste gedrückt hält, kann man die Maus bewegen und somit die selektierte Komponente bewegen. Sind mehrere Komponenten ausgewählt, so werden diese alle bewegt.Wird die Maustaste gedrückt, während sich unter ihr kein Element befindet, so öffnet sich ein Rechteck. Durch das Bewegen der Maus bei gedrückter Maustaste wird ein Rechteck aufgezogen. Nach dem Loslassen der Maustaste sind alle Elemente ausgewählt, die sich vollständig innerhalb das Rechtecks befanden.Ein ausgewähltes Diagramm oder eine Zeichnung kann vergrößert oder verkleinert werden, indem man auf eine der Ecken des Diagramms mit dem Mauszeiger klickt und während dessen die Maus bewegt.Nach Drücken auf Komponententext kann dieser direkt editiert werden. Mit der Eingabetaste gelangt man zur nächsten Zeile. Wenn der momentane Komponententext eine Auswahlliste ist, so kann er nur mit den Cursor hoch/runter Tasten verändert werden.
linke Maustaste + Ctrl-Taste	Erlaubt es, mehr als ein einziges Element auszuwählen, z.B. das Auswählen eines Elementes hebt die Auswahl für ein anderes zuvor selektiertes Element nicht auf. Diese Funktion kann ebenfall erreicht werden durch das Aufziehen eines Rechtecks mit der Maus. Dazu wird die linke Maustaste gedrückt und während dessen die Maus bewegt. Alle vollständig in dem Rechteck befindlichen Elemente werden ausgewählt (Eintrag “linke Maustaste” vorher beachten).
“Draht”-Modus	Klick auf einen Draht selektiert einen einzigen geraden Draht anstatt eines kompletten Leitungszuges.
“Einfügen Komponente”-Modus	Öffnet einen Dialog zum Verändern der Elementeigenschaften (Die Beschriftung von Drähten, die Parameter von Komponenten, etc.).

5.3 “Einfügen Komponente”-Modus

(Klick auf eine Komponente/Diagramm in der Hauptarbeitsfläche)

rechte Maustaste “Draht”-Modus	Plaziert eine neue Instanz einer Komponente in das Hauptarbeitsfenster. (Menüeintrag: Einfügen->Draht (STRG-E))

5.4 linke Maustaste

(Menüeintrag: Einfügen->Draht (STRG-E))

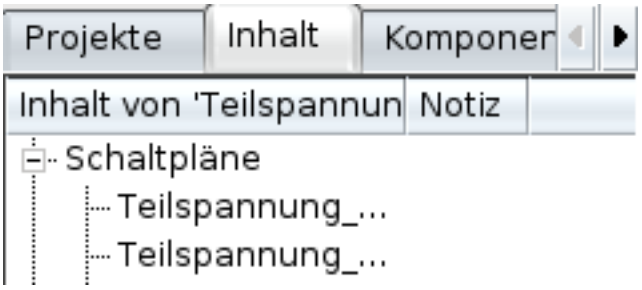
rechte Maustaste “Draht”-Modus “Einfügen Komponente”-Modus	Setzt den Start-/Endpunkt eines Drahtes. Beendet einen Draht ohne an einem Draht oder Port zu sein. (Menüeintrag: Bearbeiten->Einfügen (STRG-V))

5.5 linke Maustaste

(Menüeintrag: Bearbeiten->Einfügen (STRG-V))

rechte Maustaste	Plaziert eine Komponente in dem Hauptarbeitsfenster (aus der Zwischenablage).
“Draht”-Modus	Dreht das Element.

5.6 Wählt eine Datei aus.



doppel-Klick	Wählt eine Datei aus.	
rechts Klick	Öffnet eine Datei.	
Tastatur	Zeigt ein Menü mit:	
	“Öffnen”	• Öffnet die ausgewählte Datei
	“Umbenennen”	• Ändert den Namen der ausgewählten Datei
	“Löschen”	• Löscht die ausgewählte Datei
	“Gruppe löschen”	• Löscht die ausgewählte Datei und alle von ihr abgangigen Dateien (Schematic, Data-Diagramm, Simulationsdaten)

5.7 Tastatur

Viele Aktionen können über Tastatureingaben aktivert/ausgeführt werden. Die zu den Befehlen gehörigen Tastatureingaben kann man immer rechts neben den Texten im Menü finden. Einige weitere Tastaturkommandos sind in der folgenden Liste dargestellt:

Cursor links/rechts	Löscht das gegenwärtig ausgewählte Element oder schaltet den “Löschen”-Modus ein, wenn kein Element ausgewählt wurde.
Cursor hoch/runter	Ändert die Position des Markers innerhalb eines Graphen.Wenn kein Marker ausgewählt wurde, werden die selektierten Elemente bewegt.Wenn kein Element ausgewählt wurde, bewegt man sich durch das Dokument.
Tabulator-Taste	Ändert die Position des ausgewählten Markers in einem mehrdimensionalen Graphen.Wenn kein Marker ausgewählt wurde, werden die selektierten Elemente bewegt.Wenn kein Element ausgewählt wurde, bewegt man sich durch das Dokument.
Tabulator-Taste	Geht zum nächsten geöffneten Dokument (bezogen auf die oben vorhandenen Tabulator-Reiter).

KAPITEL 6

Arbeiten mit Schaltungshierarchien

Unterschaltungen werden benutzt, um mehr Klarheit in die Gesamtschaltung zu bringen. Die Funktion ist sehr vorteilhaft, wenn es sich um große Schaltungen handelt oder wenn Schaltungsteile mehrfach benutzt werden.

In Qucs wird jede Schaltung, die Ports enthält, als Unterschaltung betrachtet. Man erhält eine Unterschaltung, indem man den “Anschluß einfügen”-Knopf in der Knopfleiste betätigt, auf den Reiter “Komponenten” geht, dort den Blättertext “diskrete Komponenten” und darin die Komponente “Schaltkreis-Anschluß” auswählt. Alternativ kann auch aus dem Menü der Eintrag “Einfügen->Anschluß” benutzt werden. Nachdem an allen Ein- und Ausgängen der Schaltung Ports plaziert wurden, wird die Schaltung gespeichert, z.B. über “CRTL-S”. Durch einen Blick in den “Inhalts-Reiter” (Abbildung 1) erkennt man, das hinter dem Dateinamen die “Notiz” “2-Port” hinzugefügt wurde. Diese “Notiz” markiert alle Schaltungen bei denen es sich um Unterschaltungen handelt. Jetzt wird ein Schaltungsdesign geöffnet, in dem die Unterschaltung verwendet werden soll. Durch einen Klick auf den Namen der Unterschaltung im “Inhalt”-Reiter kann eine neue Komponente in die Schaltung eingefügt werden. Anschließend kann eine Simulation durchgeführt werden. Das Ergebnis ist wie erwartet das gleiche, als würde man die Komponenten direkt im Hauptarbeitsfenster plazieren.

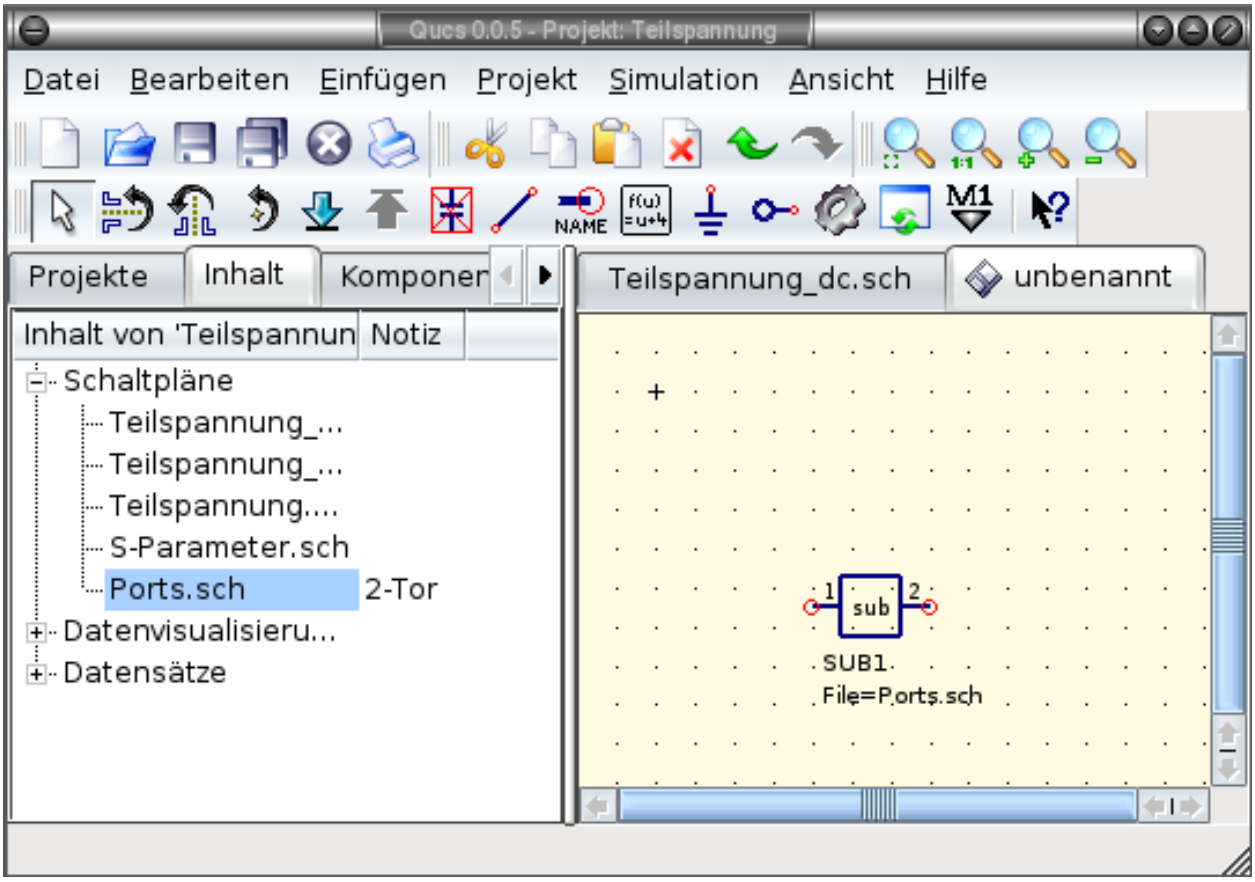


Abbildung 1 - Zugriff auf eine Unterschaltung

Wenn die Komponente markiert ist (durch das Anklicken der selbigen), die eine Unterschaltung darstellt, so kann deren Inhalt durch den Pfeil in der Knopfleiste mit der Beschriftung “Gehe in den Schaltkreis hinein (STRG-I), betrachtet werden. Alternativ erfolgt der Aufruf der Funktion über die Menüauswahl “Bearbeiten->Gehe in den Schaltkreis hinein (STRG-I)”. Das Verlassen der Unterschaltung erfolgt über die Betätigung des Knopfes “Verlasse den Schaltkreis (STRG-H)” bzw. über den Menüeintrag “Bearbeiten->Verlasse den Schaltkreis (STRG-H)”.

Wird nicht das vordefinierte Symbol für die Unterschaltung gewünscht, so kann ein eigenes Symbol gezeichnet werden und eigener Text hinzugefügt werden. Dazu geht man einfach in den Schaltungsentwurf mit der Unterschaltung hinein und wählt aus dem Menü den Eintrag “Datei->Schaltkreissymbol bearbeiten” aus. Wenn bis jetzt noch kein Symbol für die Schaltung gezeichnet wurde, so wird automatisch ein vordefiniertes Symbol erzeugt. Dieses Symbol kann durch das Zeichnen von Linien und Ecken modifiziert werden. Nachdem das Zeichnen beendet ist, wird das Symbol gespeichert. Anschließend wird es in einem anderen Schaltungsdesign plaziert, und voila, nun hat man ein neues Unterschaltungs-Symbol generiert.

Wenn die Komponente markiert ist (durch das Anklicken der selbigen), die eine Unterschaltung darstellt, so kann deren Inhalt durch den Pfeil in der Knopfleiste mit der Beschriftung “Gehe in den Schaltkreis hinein (STRG-I), betrachtet werden. Alternativ erfolgt der Aufruf der Funktion über die Menüauswahl “Bearbeiten->Gehe in den Schaltkreis hinein (STRG-I)”. Das Verlassen der Unterschaltung erfolgt über die Betätigung des Knopfes “Verlasse den Schaltkreis (STRG-H)” bzw. über den Menüeintrag “Bearbeiten->Verlasse den Schaltkreis (STRG-H)”.

6.1 Subcircuits with Parameters

A simple example using subcircuits with parameters and equations is provided here.

Create a subcircuit:

- Create a new project
- New schematic (for subcircuit)
- Add a resistor, inductor, and capacitor, wire them in series, add two ports
- Save the subcircuit as RLC.sch
- Give value of resistor as 'R1'
- Add equation 'ind = L1',
- Give value of inductor as 'ind'
- Give value of capacitor as 'C1'
- Save
- File > Edit Circuit Symbol
- Double click on the 'SUB File=name' tag under the rectangular box
  - Add name = R1, default value = 1
  - Add name = L1, default value = 1
  - Add name = C1, default value = 1
  - Ok

Insert subcircuit and define parameters:

- New schematic (for testbench)
- Save Test_RLC.sch
- Project Contents > pick and place the above RLC subcircuit
- Add AC voltage source (V1) and ground
- Add AC simulation, from 140Hz to 180Hz, 201 points
- Set on the subcircuit symbol
  - R1=1
  - L1=100e-3
  - C1=10e-6
- Simulate
- Add a Cartesian diagram, plot V1.i
- The result should be the resonance of the RLC circuit.
- The parameters of the RLC subcircuit can be changed on the top schematic.

Kurze Beschreibung der mathematischen Funktionen

Die folgenden Operationen und Funktionen können in Gleichungen von Qucs benutzt werden. Eine detaillierte Beschreibung entnehmen Sie bitte dem “Measurement Expressions Reference Manual”. Parameter in rechteckigen Klammern “[]” sind optional.

7.1 Operatoren

7.1.1 Arithmetische Operatoren

`+x`	Unär Plus
`-x`	Unär Minus
`x+y`	Addition
`x-y`	Subtraktion
`x*y`	Multiplikation
`x/y`	Division
`x%y`	Modulo-Operation (Nachkommateil einer Division)
`x^y`	Potenz

7.1.2 Logische Operatoren

`!x`	Negation
`x&&y`	Und
`x||y`	Oder
`x^^y`	Exklusiv-Oder
`x?y:z`	Abkürzung für die Bedingung `if x then y else z`
`x==y`	Gleich
`x!=y`	Ungleich
`x<y`	Kleiner als
`x<=y`	Kleiner als oder gleich
`x>y`	Größer als
`x>=y`	Größer als oder gleich

7.2 Mathematische Funktionen

7.2.1 Vektoren und Matrizen: Generierung

`eye(n)`	n x n Einheits-Matrix
`length(y)`	Liefert die Länge des gegebenen Vektors
`linspace(from,to,n)`	Erzeugt einen Vektor mit n linear gleichverteilten Werten zwischen von und bis, beide Werte mit eingeschlossen
`logspace(from,to,n)`	Erzeugt einen Vektor mit n logarithmisch gleichverteilten Werten zwischen von und bis, beide Werte mit eingeschlossen

7.2.2 Vektoren und Matrizen: Grundlegende Matrix-Funktionen

`adjoint(x)`	Transponierte und konjungiert komplexe Matrix zu x
`det(x)`	Determinante von x
`inverse(x)`	Inverse Matrix zu x
`transpose(x)`	Transponierte Matrix zu x (Zeilen und Spalten vertauscht)

7.2.3 Elementare mathematische Funktionen: Grundlegende reelle und komplexe Funktionen

`abs(x)`	Absoluter Wert, Betrag einer komplexen Zahl
`angle(x)`	Phase einer komplexen Zahl im Bogenmaß
`arg(x)`	Gleicher Ausdruck wie <code>angle(x)</code>
`conj(x)`	Konjungiert komplexe Werte der Zahl x
`deg2rad(x)`	Umrechnung von Grad nach Bogenmaß
`hypot(x,y)`	Euklidische Distanzfunktion
`imag(x)`	Imaginärteil einer komplexen Zahl
`mag(x)`	Gleicher Ausdruck wie abs(x)
`norm(x)`	Quadrat von <code>mag(x)</code>
`phase(x)`	Phase einer komplexen Zahl in Grad
`polar(m,p)`	Liefert komplexe Zahl mit gegebenem Betrag m und Phase p
`rad2deg(x)`	Umrechnung von Bogenmaß nach Grad
`real(x)`	Realteil einer komplexen Zahl
`sign(x)`	Berechnet die Signumfunktion
`sqr(x)`	Quadrat (<code>x</code> zur Potenz zwei)
`sqrt(x)`	Quadratwurzel
`unwrap(p[,tol[,step]])`	Gleicht Phasensprünge von p (im Bogenmaß – Standardsprungweite step ist 2*pi) aus und verwendet dabei die optionale Toleranzschwelle tol (Standardwert ist pi)

7.2.4 Elementare mathematische Funktionen: Exponential- und Logarithmus-Funktionen

`exp(x)`	Exponentialfunktion zur Basis e
`limexp(x)`	Begrenzte Exponentialfunktion
`log10(x)`	Dekadischer Logarithmus
`log2(x)`	Binärer Logarithmus
`ln(x)`	Natürlicher Logarithmus

7.2.5 Elementare mathematische Funktionen: Trigonometrie

`cos(x)`	Kosinus
`cosec(x)`	Kosekans
`cot(x)`	Kotangens
`sec(x)`	Sekans
`sin(x)`	Sinus
`tan(x)`	Tangens

7.2.6 Elementare mathematische Funktionen: Inverse trigonometrische Funktionen

`arccos(x)`	Arkuskosinus
`arccosec(x)`	Arkuskosekans
`arccot(x)`	Arkuskotangens
`arcsec(x)`	Arkussekans
`arcsin(x)`	Arkussinus
`arctan(x[,y])`	Arkustangens

7.2.7 Elementare mathematische Funktionen: Hyperbolische Funktionen

`cosh(x)`	Kosinus hyperbolicus
`cosech(x)`	Kosekans hyperbolicus
`coth(x)`	Kotangens hyperbolicus
`sech(x)`	Sekans hyperbolicus
`sinh(x)`	Sinus hyperbolicus
`tanh(x)`	Tangens hyperbolicus

7.2.8 Elementare mathematische Funktionen: Inverse hyperbolische Funktionen

`arcosh(x)`	Area Kosinus hyperbolicus
`arcosech(x)`	Area Kosekans hyperbolicus
`arcoth(x)`	Area Kotangens hyperbolicus
`arsech(x)`	Area Sekans hyperbolicus
`arsinh(x)`	Area Sinus hyperbolicus
`artanh(x)`	Area Tangens hyperbolicus

7.2.9 Elementare mathematische Funktionen: Runden

`ceil(x)`	Rundet zur nächstgrößeren Ganzzahl
`fix(x)`	Schneidet Nachkommastellen von reellen Zahlen ab
`floor(x)`	Rundet zur nächstkleineren Ganzzahl
`round(x)`	Rundet zur nächsten Ganzzahl

7.2.10 Elementare mathematische Funktionen: Spezielle Funktionen

`besseli0(x)`	Modifizierte Besselfunktion nullter Ordnung
`besselj(n,x)`	Besselfunktion erster Art und n-ter Ordnung
`bessely(n,x)`	Besselfunktion zweiter Art und n-ter Ordnung
`erf(x)`	Fehlerfunktion
`erfc(x)`	Komplementäre Fehlerfunktion
`erfinv(x)`	Inverse Fehlerfunktion
`erfcinv(x)`	Inverse komplementäre Fehlerfunktion
`sinc(x)`	Sinc-Funktion (sin(`x`)/`x` und 1 bei `x` =0)
`step(x)`	Sprungfunktion

7.2.11 Datenanalyse: Grundlegende Statistik-Funktionen

avg(x[,Bereich])	Arithmetischer Mittelwert aus den Werten in einem Vektor; wenn ein Bereich angegeben wird, dann muss x eine einfache Datenabhängigkeit aufweisen
`cumavg(x)`	Kumulativer Mittelwert der Werte eines Vektors
`max(x,y)`	Liefert den größeren der beiden Werte x und y
max(x[,Bereich])	Maximaler Wert in einem Vektor x; wenn ein Bereich angegeben wird, dann muss x eine einfache Datenabhängigkeit aufweisen
`min(x,y)`	Liefert den kleineren der beiden Werte x und y
min(x[,Bereich])	Minimaler Wert in einem Vektor x; wenn ein Bereich angegeben wird, dann muss x eine einfache Datenabhängigkeit aufweisen
`rms(x)`	Effektivwert aus den Werten eines Vektors
`runavg(x)`	Gleitender Mittelwert der Werte eines Vektors
`stddev(x)`	Standardabweichung der Werte eines Vektors
`variance(x)`	Varianz der Werte eines Vektors
`random()`	Zufallszahl zwischen 0.0 und 1.0
`srandom(x)`	Anfangswert für Zufallsgenerator

7.2.12 Datenanalyse: Grundlegende Operationen

`cumprod(x)`	Kumulatives Produkt der Werte in einem Vektor
`cumsum(x)`	Kumulative Summe der Werte in einem Vektor
`interpolate(f,x[,n])`	Berechnet eine Interpolation der reellen Funktion f(x)an n äquidistanten Punkten; letzterer Parameter kann weggelassen werden und erhält dann einen vernünftigen Standardwert
`prod(x)`	Produkt der Werte in einem Vektor
`sum(x)`	Summe der Werte in einem Vektor
`xvalue(f,yval)`	Liefert den X-Wert, der mit dem nächstliegenden Y-Wert zu yval aus dem Vektor f assoziiert ist; dafür muss der Vektor f eine einfache Datenabhängigkeit besitzen
`yvalue(f,xval)`	Liefert den Y-Wert des gegebenen Vektors f, der dem X-Wert xval am nächsten liegt; dafür muss der Vektor f eine einfache Datenabhängigkeit besitzen

7.2.13 Datenanalyse: Differentiation und Integration

`ddx(expr,var)`	Differenziert den mathematischen Ausdruck expr bezüglich der Variable var
`diff(y,x[,n])`	Differenziert n-mal den Vektor y in Bezug auf x. Wird n weggelassen, entspricht dies n=1.
`integrate(x,h)`	Integriert den Vektor x numerisch bei angenommener konstanter Schrittweite h

7.2.14 Datenanalyse: Signalverarbeitung

`dft(x)`	Berechnet die diskrete Fourier-Transformation (DFT) des Vektors x
`fft(x)`	Berechnet die schnelle Fourier-Transformation (FFT) des Vektors x
`fftshift(x)`	Schiebt die Werte des FFT-Vektors x so, dass die Frequenz 0 in die Mitte des Vektors verschoben wird
`Freq2Time(V,f)`	Berechnet die inverse diskrete Fourier-Transformation der Funktion V(f) und interpretiert die Werte physikalisch
`idft(x)`	Berechnet die inverse diskrete Fourier-Transformation (IDFT) des Vektors x
`ifft(x)`	Berechnet die inverse schnelle Fourier-Transformation (IFFT) des Vektors x
`kbd(x[,n])`	Kaiser-Bessel Fensterfunktion
`Time2Freq(v,t)`	Berechnet die diskrete Fourier-Transformation der Funktion v(t) und interpretiert die Werte physikalisch

7.3 Elektrotechnische Funktionen

7.3.1 Umrechnung von Maßeinheiten

`dB(x)`	Spannungsdezibel
`dbm(x)`	Wandelt Spannung in Leistung in dBm um
`dbm2w(x)`	Wandelt Leistung in dBm in Leistung in Watt um
`w2dbm(x)`	Wandelt Leistung in Watt in Leistung in dBm um
`vt(t)`	Temperaturspannung für eine gegebene Temperatur t in Kelvin

7.3.2 Reflexionskoeffizienten und Stehwellenverhältnisse

`rtoswr(x)`	Konvertiert einen Reflexionsfaktor in das (Spannungs-)Stehwellenverhältnis
`rtoy(x[,zref])`	Konvertiert einen Reflexionsfaktor (Referenzimpedanz ist standardmäßig 50 Ohm) in eine Admittanz
`rtoz(x[,zref])`	Konvertiert einen Reflexionsfaktor (Referenzimpedanz ist standardmäßig 50 Ohm) in eine Impedanz
`ytor(x[,zref])`	Konvertiert eine Admittanz in einen Reflexionsfaktor (Referenzimpedanz ist standardmäßig 50 Ohm)
`ztor(x[,zref])`	Konvertiert eine Impedanz in einen Reflexionsfaktor (Referenzimpedanz ist standardmäßig 50 Ohm)

7.3.3 Transformation von N-Tor-Matrizen

`stos(s,zref[,z0])`	Konvertiert die S-Parameter-Matrix in eine S-Parameter-Matrix mit unterschiedliche(r/n) Referenzimpedanz(en)
`stoy(s[,zref])`	Konvertiert die S-Parameter-Matrix in die Y-Parameter-Matrix
`stoz(s[,zref])`	Konvertiert die S-Parameter-Matrix in die Z-Parameter-Matrix
`twoport(m,from,to)`	Konvertiert eine gegebene 2-Port-Matrix von einer Darstellungsform in eine andere, mögliche Werte für von und nach sind “Y”, “Z”, “H”, “G”, “A”, “S” und “T”.
`ytos(y[,z0])`	Konvertiert die Y-Parameter-Matrix in die S-Parameter-Matrix
`ytoz(y)`	Konvertiert die Y-Parameter-Matrix in die Z-Parameter-Matrix
`ztos(z[,z0])`	Konvertiert die Z-Parameter-Matrix in die S-Parameter-Matrix
`ztoy(z)`	Konvertiert die Z-Parameter-Matrix in die Y-Parameter-Matrix

7.3.4 Verstärker

`GaCircle(s,Ga[,arcs])`	Kreis(e) mit konstanter verfügbarer Leistungsverstärkung Ga in der Quellebene
`GpCircle(s,Gp[,arcs])`	Kreis(e) mit konstanter Leistungsverstärkung Gp in der Lastebene
`Mu(s)`	Mu Stabilitätsfaktor der Zweitor-S-Parameter-Matrix <code>s</code>
`Mu2(s)`	Mu' Stabilitätsfaktor der Zweitor-S-Parameter-Matrix <code>s</code>
`NoiseCircle(Sopt,Fmin,Rn,F[,Winkel])`	Kreis mit konstanten Rauschzahlen F (kann eine Konstante oder ein Vektor sein). Winkel spezifiziert die Winkel in Grad, die z.B. mit linspace(0,360,100) erzeugt wurden. Wenn Winkel eine Zahl ist, dann steht diese für die Anzahl der gleichverteilten Kreissegmente. Wenn der Parameter weggelassen wurde, dann wird ein vernünftiger Standardwert eingesetzt
`PlotVs(data,dep)`	Liefert Daten zurück, die auf dem Vektor oder Matrizenvektor Daten basieren, in Abhängigkeit von dem gegebenen Vektor Abh. Beispiel: PlotVs(Gain,frequency/1e9)
`Rollet(s)`	Rollet Stabilitätsfaktor der Zweitor-S-Parameter-Matrix `s`
`StabCircleL(s[,arcs])`	Stabilitätskreise in der Lastebene
`StabCircleS(s[,arcs])`	Stabilitätskreise in der Quellebene
`StabFactor(s)`	Stabilitätsfaktor der Zweitor-S-Parameter-Matrix s. Synonym für Rollet()
`StabMeasure(s)`	Stabilitätsmaß B1 einer Zweitor-S-Parameter-Matrix

7.4 Schreibweisen

7.4.1 Intervalle

`LO:HI`	Intervall von LO bis HI
`:HI`	Bis zu HI
`LO:`	Von LO an
`:`	Keine Intervallgrenzen

7.4.2 Matrizen und ihre Elemente

`M`	Die gesamte Matrix M
`M[2,3]`	Element in der 2. Zeile und der 3. Spalte der Matrix M
`M[:,3]`	Vektor bestehend aus der 3. Spalte der Matrix M

7.4.3 Zahlen, Vektoren, Matrizen

`2.5`	Reelle Zahl
`1.4+j5.1`	Komplexe Zahl
`[1,3,5,7]`	Vektor
`[11,12;21,22]`	Matrix

7.4.4 Zahlenendungen

`E`	exa, 1e+18
`P`	peta, 1e+15
`T`	tera, 1e+12
`G`	giga, 1e+9
`M`	mega, 1e+6
`k`	kilo, 1e+3
`m`	milli, 1e-3
`u`	micro, 1e-6
`n`	nano, 1e-9
`p`	pico, 1e-12
`f`	femto, 1e-15
`a`	atto, 1e-18

7.4.5 Wertenamen

`S[1,1]`	S-Parameterwert
knotenname.V	DC-Spannung am Knoten *knotenname*
name.I	DC-Strom durch die Komponente *name*
knotenname.v	AC-Spannung am Knoten *knotenname*
name.i	AC-Strom durch die Komponente *name*
knotenname.vn	AC-Rauschspannung am Knoten *knotenname*
name.in	AC-Rauschstrom durch die Komponente *name*
knotenname.Vt	Transientenspannung am Knoten *knotenname*
name.It	Transientenstrom durch die Komponente *name*

Bitte beachten: Alle Spannungen und Ströme sind Spitzenwerte. Rauschspannungen sind Effektivwerte in 1 Hz Bandbreite.

7.5 Konstanten

`i, j`	Imaginäre Einheit (“Quadratwurzel von -1”)
`pi`	4*arctan(1) = 3.14159...
`e`	Eulerzahl = 2.71828...
`kB`	Boltzmann-Konstante = 1.38065e-23 J/K
`q`	Elementarladung = 1.6021765e-19 C

Spezielle Zeichen

In der Textkomponente und in den Achsenbeschriftungen der Diagramme können besondere Zeichen verwendet werden. Das wird mit LaTeX-Befehlen bewerkstelligt. Die folgende Tabelle beinhaltet die derzeit verfügbaren Zeichen.

Beachte: Welche dieser Zeichen auch tatsächlich richtig angezeigt werden können, hängt von dem Zeichensatz, der von Qucs verwendet wird, ab.

Kleine griechische Buchstaben

LaTeX-Befehl	Unicode	Beschreibung
\alpha	0x03B1	alpha
\beta	0x03B2	beta
\gamma	0x03B3	gamma
\delta	0x03B4	delta
\epsilon	0x03B5	epsilon
\zeta	0x03B6	zeta
\eta	0x03B7	eta
\theta	0x03B8	theta
\iota	0x03B9	iota
\kappa	0x03BA	kappa
\lambda	0x03BB	lambda
\mu	0x03BC	mu
\textmu	0x00B5	mu
\nu	0x03BD	nu
\xi	0x03BE	xi
\pi	0x03C0	pi
\varpi	0x03D6	pi
\rho	0x03C1	rho
\varrho	0x03F1	rho
\sigma	0x03C3	sigma
\tau	0x03C4	tau
\upsilon	0x03C5	upsilon
\phi	0x03C6	phi
\chi	0x03C7	chi
\psi	0x03C8	psi
\omega	0x03C9	omega

Große griechische Buchstaben

LaTeX-Befehl	Unicode	Beschreibung
\Gamma	0x0393	Gamma
\Delta	0x0394	Delta
\Theta	0x0398	Theta
\Lambda	0x039B	Lambda
\Xi	0x039E	Xi
\Pi	0x03A0	Pi
\Sigma	0x03A3	Sigma
\Upsilon	0x03A5	Upsilon
\Phi	0x03A6	Phi
\Psi	0x03A8	Psi
\Omega	0x03A9	Omega

Mathematische Symbole

LaTeX-Befehl	Unicode	Beschreibung
\cdot	0x00B7	Multiplikationspunkt (zentrierter Punkt)
\times	0x00D7	Multiplikationskreuz
\pm	0x00B1	Plus/Minus Zeichen
\mp	0x2213	Minus/Plus Zeichen
\partial	0x2202	Symbol für partielle Differentiation
\nabla	0x2207	Nablaoperator
\infty	0x221E	Symbol für die Unendlichkeit
\int	0x222B	Integralsymbol
\approx	0x2248	Näherungssymbol (geschwungenes Gleichheitszeichen)
\neq	0x2260	Ungleichzeichen
\in	0x220A	Symbol für “enthält”
\leq	0x2264	Kleiner-Gleich-Zeichen
\geq	0x2265	Größer-Gleich-Zeichen
\sim	0x223C	(mitteleuropäisches) Proportionalzeichen
\propto	0x221D	(amerikanisches) Proportionalzeichen
\diameter	0x00F8	Durchmesser-Zeichen (auch für Durchschnitt)
\onehalf	0x00BD	ein Halb
\onequarter	0x00BC	ein Viertel
\twosuperior	0x00B2	Quadrat (Zweierpotenz)
\threesuperior	0x00B3	Dreierpotenz
\ohm	0x03A9	Einheit für den elektrischen Widerstand (großes griechisches omega)

KAPITEL 9

Anpassungsnetzwerke

Das Erstellen von Anpassschaltungen ist eine häufig anzutreffende Aufgabe in der Mikrowellentechnik. Qucs kann das automatisch. Dies sind die dazu notwendigen Schritte:

Durchführen einer S-Parameter-Simulation zur Berechnung der Reflektionsfaktoren.

Einfügen eines Diagramms zur Darstellung eines Reflektionsfaktors (z.B. S[1,1] für Tor 1, S[2,2] für Tor 2 usw.)

Setzen eines Markers auf die Kurve des Reflektionsfaktors und schrittweise zur gewünschten Frequenz gehen.

Mit der rechten Maustaste auf den Marker klicken und “Leistungsanpassung” in dem erscheinenden Menü auswählen.

Es öffnet sich ein Dialogfenster, in dem die Werte auch von Hand angepasst werden können. Die Referenzimpedanz kann beispielsweise von 50 Ohm abweichend gewählt werden.

Nach dem Bestätigen mit dem “Erstellen”-Knopf wird auf den Schaltplan zurückgeschaltet und durch Bewegen der Maus kann die fertige Anpassschaltung eingefügt werden.

Die linke Seite der Anpassschaltung ist der Eingang und die rechte Seite muss mit dem Schaltkreis verbunden werden.

Falls der Marker auf eine Variable namens “Sopt” zeigt, beinhaltet das Menü die Option “Rauschanpassung”. Beachte, dass der einzige Unterschied zur “Leistungsanpassung” die Verwendung des konjugiert komplexen Reflektionsfaktors ist. Wenn die Variable also einen anderen Namen hat, kann Rauschanpassung durch eine kleine Änderung der Werte in dem Dialog erreicht werden.

Der Anpassungsdialog kann auch über das Menü aufgerufen werden (Werkzeuge->Anpassnetzwerk) oder durch das Tastenkürzel (<CTRL-5>). Dann müssen aber alle Werte von Hand eingetragen werden.

9.1 Zweitor-Anpassungsnetzwerke

Falls der Variablenname in dem Marker ein S-Parameter ist, dann existiert ein weiterer Menüpunkt für die gleichzeitige Anpassung am Ein- und Ausgang des Zweitors. Das funktioniert sehr ähnlich wie mit den oben beschriebenen Schritten. Das Ergebnis sind zwei Anpassnetzwerke: Der ganz linke Knoten muss an Tor 1 angeschlossen werden und der ganz rechte Knoten an Tor 2. Die zwei Knoten in der Mitte müssen mit dem Zweitor verbunden werden.

KAPITEL 10

Installierte Dateien

Das Qucs-System benötgt verschiedene Programme. Diese werden während des Installationsvorgangs mitinstalliert. Der Installationspfad von Qucs wird durch die Parameterübergabe beim Aufruf des Konfigurationsscripts bestimmt (z.B. configure –prefix=/pfad/zum/gewünschten/Verzeichnis). Die folgenden Ausführungen beziehen sich auf den angenommenen voreingestellten Installationpfad (/usr/local/).

- /usr/local/bin/qucs - die Benutzeroberfläche (GUI)
- /usr/local/bin/qucsator - Der Simulator (Konsolenprogramm)
- /usr/local/bin/qucsedit - Ein einfacher Texteditor
- /usr/local/bin/qucshelp - Ein kleines Programm zur Darstellung der Hilfe-Seiten
- `/usr/local/bin/qucstrans` - a program for calculation transmission line parameters
- `/usr/local/bin/qucsfilter` - a program synthesizing filter circuits
- /usr/local/bin/qucsconv - Ein Dateiformat-Konverter (Konsolenprogramm)

Alle Programme sind einzelnen Anwendungen, die unabhängig voneinander gestartet werden können. Die Benutzeroberfläche (GUI)

- ruft qucsator auf, wenn eine Simulation durchgeführt werden soll,
- ruft qucsedit auf, wenn eine Textdatei angezeigt werden soll,
- ruft qucshelp auf, wenn die Hilfe-Seiten dargestellt werden sollen,
- calls `qucstrans` when calling this program from menu “Tools”,
- calls `qucsfilter` when calling this program from menu “Tools”,
- ruft qucsconv auf, wenn eine SPICE-Komponente plaziert wird und wenn eine Simulation mit der SPICE-Komponente durchgeführt werden soll.

Desweiteren werden die folgenden Verzeichnisse während der Installation erzeugt:

- /usr/local/share/qucs/bitmaps - Enthält alle Bitmap-Bilder (Icons, etc.)
- /usr/local/share/qucs/docs - Enthält alle HTML-Dokumente für die Hilfe-Seiten
- /usr/local/share/qucs/lang - Enthält die Datien für die Sprachanpassung

10.1 Kommandozeilen Argumente

```
qucs [Datei1 [Datei2 ...]]
```

`qucsator [-b] -i Netzliste -o Datensatz` (b = Fortschrittsanzeige)

`qucsedit [-r] [Datei]` (r = nur-lesen)

`qucshelp (keine Argumente)`

`qucsconv -if spice -of qucs -i Netzliste.inp -o Netzliste.net`

Dateiformat der Schaltpläne

Dieses Dokument beschreibt kurz das Dateiformat der Schaltpläne von Qucs. Das Format wird für Schaltpläne (normalerweise mit der Dateiendung `.sch`) und für Datenvisualisierungen (normalerweise mit der Dateiendung `.dpl`) verwendet. Der folgende Text zeigt ein kurzes Beispiel für eine solche Datei.

```
<Qucs Schematic 0.0.6>
<Properties>
  <View=0,0,800,800,1,0,0>
</Properties>
<Symbol>
  <.ID -20 14 SUB>
</Symbol>
<Components>
  <R R1 1 180 150 15 -26 0 1 "50 Ohm" 1 "26.85" 0 "european" 0>
  <GND * 1 180 180 0 0 0 0>
</Components>
<Wires>
  <180 100 180 120 "" 0 0 0 "">
  <120 100 180 100 "Input" 170 70 21 "">
</Wires>
<Diagrams>
  <Polar 300 250 200 200 1 #c0c0c0 1 00 1 0 1 1 1 0 5 15 1 0 1 1 315 0 225 "" "" "">
  <"acnoise2:S[2,1]" #0000ff 0 3 0 0 0>
  <Mkr 6e+09 118 -195 3 0 0>
</Polar>
</Diagrams>
<Paintings>
  <Arrow 210 320 50 -100 20 8 #000000 0 1>
</Paintings>
```

Die Datei beinhaltet mehrere Abschnitte. Jeder dieser Abschnitte wird nachfolgend erklärt. Jede Zeile besteht aus einem einzigen Informationsblock, der mit dem Kleiner-Zeichen < beginnt und mit dem Größer-Zeichen > endet.

11.1 Eigenschaften

Der erste Abschnitt beginnt mit `<Properties>` und endet mit `</Properties>`. Er beinhaltet die Dokumenteneigenschaften der Datei. Jede dieser Zeilen ist optional. Die folgenden Eigenschaften werden unterstützt.

- `<View=x1,y1,x2,y2,scale,xpos,ypos>` beinhaltet die Pixelposition des Schaltplanfensters in den ersten vier Zahlen, die aktuelle Skalierung und die aktuelle Position der linken oberen Ecke (die letzten beiden Zahlen).

- <Grid=x,y,on> beinhaltet den Gitternetzabstand in Pixeln (die ersten beiden Zahlen) und ob das Gitternetz sichtbar ist (letzte Zahl 1) oder nicht (letzte Zahl 0).
- <DataSet=name.dat> beinhaltet den Dateinamen des Datensatzes, der mit diesem Schaltplan assoziiert wird.
- <DataDisplay=name.dpl> beinhaltet den Dateinamen der Datenvisualisierung, die mit diesem Schaltplan assoziiert wird (bzw. den Dateinamen des Schaltplan, falls das Dokument eine Dateinvisualisierung ist).
- <OpenDisplay=yes> beinhaltet eine 1, falls die Datenvisualisierung automatisch nach der Simulation angezeigt werden soll, anderenfalls eine 0.

11.2 Symbol

Dieser Abschnitt beginnt mit <Symbol> und endet mit </Symbol>. Er beinhaltet die Zeichnungselemente, die das Schaltplansymbol dieser Datei bilden. Das wird normalerweise nur bei Schaltplänen verwendet, die eine Unterschaltung darstellen.

11.3 Komponenten

Dieser Abschnitt beginnt mit <Components> und endet mit </Components>. Er beinhaltet die Schaltkreiskomponenten des Schaltplans. Das Zeilenformat ist wie folgt aufgebaut:

```
<type name active x y xtext ytext mirrorX rotate "Value1" visible "Value2" visible ...>
```

- Der type identifiziert die Komponente, z.B. steht R für einen Widerstand und C für einen Kondensator.
- Der name ist der Komponentenidentifizierer in dem Schaltplan, z.B. steht R1 für den ersten Widerstand.
- Eine 1 in dem active Feld zeigt an, dass die Komponenten aktiv ist, d.h. dass sie während der Simulation verwendet wird. Eine 0 zeigt an, dass die Komponente nicht aktiv ist.
- Die nächsten beiden Zahlen sind die x- und y-Koordinaten des Komponentenzentrums.
- Die folgenden beiden Zahlen sind die x- und y-Koordinaten der linken oberen Ecke des Komponententextes. Sie sind relativ zum Komponentenzentrum.
- Die nächsten beiden Zahlen zeigen an, ob die Komponente an der x-Achse gespiegelt ist (1 für gespiegelt, 0 für nicht gespiegelt) ist und ob die Komponente entgegen des Uhrzeigersinns gedreht ist (Vielfache von 90 Grad, d.h. 0...3).
- Die nächsten beiden Einträge sind die Werte der Komponenteneigenschaften (in Anführungszeichen) gefolgt von einer 1, falls die Eigenschaft in dem Schaltplan angezeigt wird (ansonsten eine 0).

11.4 Verbindungen

Der Abschnitt beginnt mit <Wires> und endet mit </Wires>. Er beinhaltet die Drähte (elektrische Verbindungen zwischen den Schlatkreiskomponenten) und ihre Bezeichnungen bzw. zusätzlichen Eigenschaften. Das Zeilenformat sieht wie folgt aus:

```
<x1 y1 x2 y2 "label" xlabel ylabel dlabel "node set">
```

- Die ersten vier Zahlen sind die Koordinaten des Drahtes in Pixel: x-Koordinate des Startpunktes, y-Koordinate des Startpunktes, x-Koordinate des Endpunktes und y-Koordinate des Endpunktes. Alle Drähte müssen entweder horizontal (beide x-Koordinaten gleich) oder vertikal (beide y-Koordinaten gleich) sein.

- Die erste Zeichenkette in Anführungszeichen ist der Name des Bezeichners. Er ist leer, wenn der Benutzer keine Drahtbezeichnung eingegeben hat.
- Die nächsten beiden Zahlen sind die x- und y-Koordinaten der Bezeichnung oder Null, falls es keine Bezeichnung gibt.
- Die folgenden beiden Zahlen sind der Abstand zwischen dem Startpunkt des Drahtes und dem Punkt, an dem der Bezeichner des Drahtes angezeigt werden soll.
- Die letzte Zeichenkette in Anführungszeichen ist der Anfangswert für die Knotenspannung an diesem Draht. Sie ist leer, falls der Benutzer keine Knotenspannung für diesen Draht angegeben hat..

11.5 Diagramme

Der Abschnitt beginnt mit *<Diagrams>*‘ und endet mit `</Diagrams>`. Er beinhaltet die Diagramme mit ihren Kurven und Markierungen.”

```
<x y width height grid gridcolor gridstyle log xAutoscale xmin xstep
 xmax yAutoscale ymin ystep ymax zAutoscale zmin zstep zmax xrotate
 yrotate zrotate "xlabel" "ylabel" "zlabel">
```

- Die ersten beiden Zahlen sind die x-und y-Koordinaten der linken unteren Ecke.
- Die nächsten beiden Zahlen sind die Breite und Höhe der Diagrammgrenzen.
- Die fünfte Zahl ist 1 falls das Gitternetz angezeigt werden soll und 0 falls nicht.
- Das nächste ist die Farbe des Gitternetzes als hexadezimaler 24-Bit RGB-Wert, z.B. ist #FF0000 rot.
- Die nächste Zahl legt den Stil des Gitternetzes fest.
- Die nächste Zahl legt fest, welche Achsen eine logarithmische Einteilung haben.

11.6 Zeichnungen

Der Abschnitt beginnt mit `<Paintings>` und endet mit `</Paintings>`. Er beinhaltet die Zeichnungselemente, die sich in dem Schaltplan befinden.”

Technische Beschreibungen den Simulator betreffend

sind erhältlich unter http://qucs.sourceforge.net/tech/technical.html

Beispielschaltungen

sind erhältlich unter http://qucs.sourceforge.net/download.html#example