Penthode 1.20 User Manual

Simone Pernice

1st September 2018

Chapter 1

Introduction

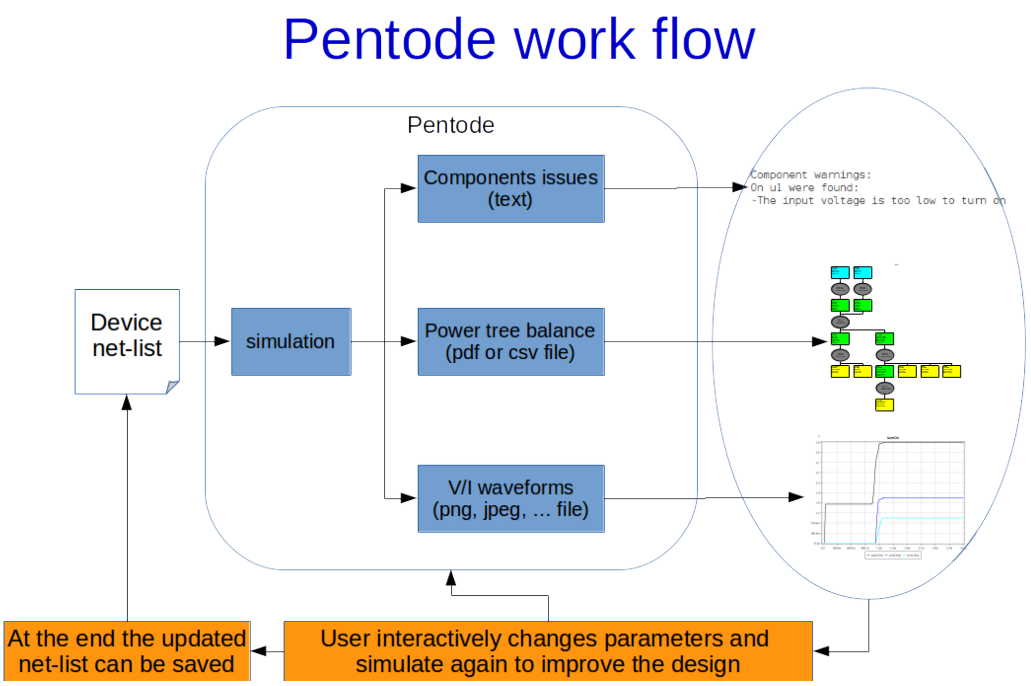
Pentode design begun on 3rd December 2016 by Simone Pernice. When Pentode was born there was not any tool to design a power distribution. Pentode names comes from PowErTreeDEsigner. Pentode was a device used as transistor in high quality audio pre-amplifier of the old age before the solid state transistor coming.
Pentode is open source however its development takes a lot of time. If you make a donation you can grant its manual, get help, request features, etc.
Pentode is a tool to design, simulate, draw the high level power distribution of a device. The power distribution, sometime called power tree of a device, shows how the power is delivered from the sources (battery, DC supplier), converted to the required voltages, provided to the loads (display, CPU, ...).
Pentode session usually begins loading the high level net list of the device. The user can prepare that net list describing the device power architecture. That is just a starting point, it is also possible begin from an empty net list and interactively: add, modify and delete components of the power distribution. It is possible to simulate the power distribution looking for components running out of specification, plotting their current and voltage waveforms. It is also possible compute the power tree BOM (bill of material) cost. If the result is not satisfying the power tree can be modified and simulated again. When the iterations are completed it is possible to save all the history commands given to continue from there later.
After starting Pentode, the user runs can interactively:

- Commands to simulate the voltage and current up to the steady state: the simulation reporting components working out of specification

- Simulate same circuit several times stepping some parameter to find the best performance

- Draw a nice power tree diagram showing the currents/powers balance on pdf/svg file
- Plot node transient voltage and gate current waveforms on png/jpeg file
- Change component parameters interactively to improve the design, delete or add components
- Iterate through loops and lists, check conditionals, manipulate variables and perform basic calculus to automate repetitive tasks
- In the net-list commands or interactively, it is possible to modify the power tree to test new configurations: modified components properties and adding or deleting components

Pentode workflow is showed in the next picture:


Pentode work flow
Pentode
Device net-list
simulation
Components issues (text)
Power tree balance (pdf or csv file)
V/I waveforms (png, jpeg, ... file)
At the end the updated net-list can be saved
User interactively changes parameters and simulate again to improve the design

```
Component warnings:
On u1 were found:
-The input voltage is too low to turn on
```

The main concerns designing the power distribution of a new device are:

- Efficiency: it is good to minimize the input power especially on portable devices to provide long battery life. High efficiency can be achieved with:

  - Scalability: the capacity to turn off the not used regulators in order to reduce the energy waste
  - Standby current: is the special case of scalability where almost all components are turned off and the device drains a negligible current to preserve the battery charge.

- Cost: the device has to be the cheapest possible but that goes against efficiency because usually more efficient converters are more expensive. To achieve the optimum it is required to compare several solutions.
- Ratings: every source, converter and drain has to work within its specification
- Correct suppliers turn on and off sequence: that is very important when the components linked together use several rails

Usually the power distribution design requires to:

- Define all the user cases: Standby, WiFi, Video + WiFi, Video + Ethernet, ...
- Define the most suitable power trees: That is a trial and error procedure which requires to draw on paper the most promising architectures
- Compute the cost of every architecture
- For each couple: use case architecture the designer has to evaluate the efficiency computing the power balance
- Choose the best trade off between cost and efficiency

Pentode can automate and simplify all those steps to get a faster and more accurate result.

Chapter 2

Basics

This chapter describes Pentode basics. Pentode simulate a *device*. A device is made by a net of *electric components* linked to the *nodes* of that device. There are two main types of electric components *node* and *black box*.
Pentode session begins reading an input file (.ptd extension). Usually that file describes the device components and their links, however it is possible to begin with an empty file and adding interactively all the components required by the architecture. Each line is a command to be executed: if the command is the description of a component it is added to the device. A component is defined by its unique label, type (supplier, converter, drainer, etc.), by its parameters (output voltage, etc.) and interconnections to the nodes of the devices. After the device is defined a set of commands to simulate and show its results are used. Once the commands in the input file are executed, it is possible to continue interactively to write commands or modify the device at Pentode's prompt. When the target result is achieved a *save* command will add to the input file the new lines added interactively after its loading. Pentode session ends with the *quit* command.

2.1 Basic Components

Pentode simulates a *device* made by a net of *electric components* linked to the *nodes* of that device. There are two main types of electric components *node* and *black box*. Every time an input line describing a component is found, it is added to the device. Pentode always try to add a new component although something is not understood due to syntax errors which are reported.

2.1.1 Node

The nodes are defined by their label and capacitance value. It is optionally possible to set their simulation begin voltage to use to start the simulation. By default the simulation begins with all nodes at 0V. Nodes are usually defined automatically when the black box is declared because it is required to write the nodes at which it is linked to. Nodes label are defined by user. Nodes save their voltage history to be plotted after the simulation if required. Its electrical capacitance stores or provides charge when there is a mismatch among the current magnitudes entering and exiting on the node.

2.1.2 Black box

The black boxes is defined by its label and the link between its *gate*(s) and the nodes. Every black box has one or more *gate*s through which the current flows to the device's nodes. The gates at the same voltage are linked to the same node. Gates store the black box current histories which can be showed after the simulation.
Power suppliers and drainers have just one gate (to output or input their power), while converter at least two (power flows from input to output). Pentode names gates by a sequential number: 0 for single gate, 0 is input and 1 output for dual gate. The currents flowing through those gates are defined by not-linear time-variant behavioral-equations depending on the voltage of the nodes at which they are linked to. By definition the current entering a component is positive and negative if it exits.
The behaviors of the most commons electric components are already stored in Pentode. The user has to customize their characteristic setting *parameter*s like the output voltage of a voltage regulator. Almost all parameters have a default value to be used if it is not set by the user.
The black boxes providing power are called *source*. They have just one gate (number 0) to output their power. For instance a source can be a DC adapter or a battery. Their behavioral model takes into account of parameters like: internal resistance, maximum current, voltage waveform (for time-variant source), etc. The purpose of those parameters is to simulate the real component behavior.
The *converters* have at least an input (gate 0) and an output (gate 1) usually. There are also controlled converter with a third gate (2) working as enable. Their purpose is to convert the power from the input gate to the output changing its voltage for instance. Examples of converters are diodes, switches, LDOs, buck or boost converters. Their behavioral model takes into

account of parameters like: load and line regulation, maximum current, minimum input voltage, quiescent and disable current; for switching converters: switching and resistant current loss, working mode (PWM or PFM), etc.
The *drains* use the power provided by sources and converters like LED or resistor. They have just the input gate (0). Their behavioral model takes into account of parameters like: minimum operating voltage, equivalent resistance, current or power, etc.

2.2 Commands

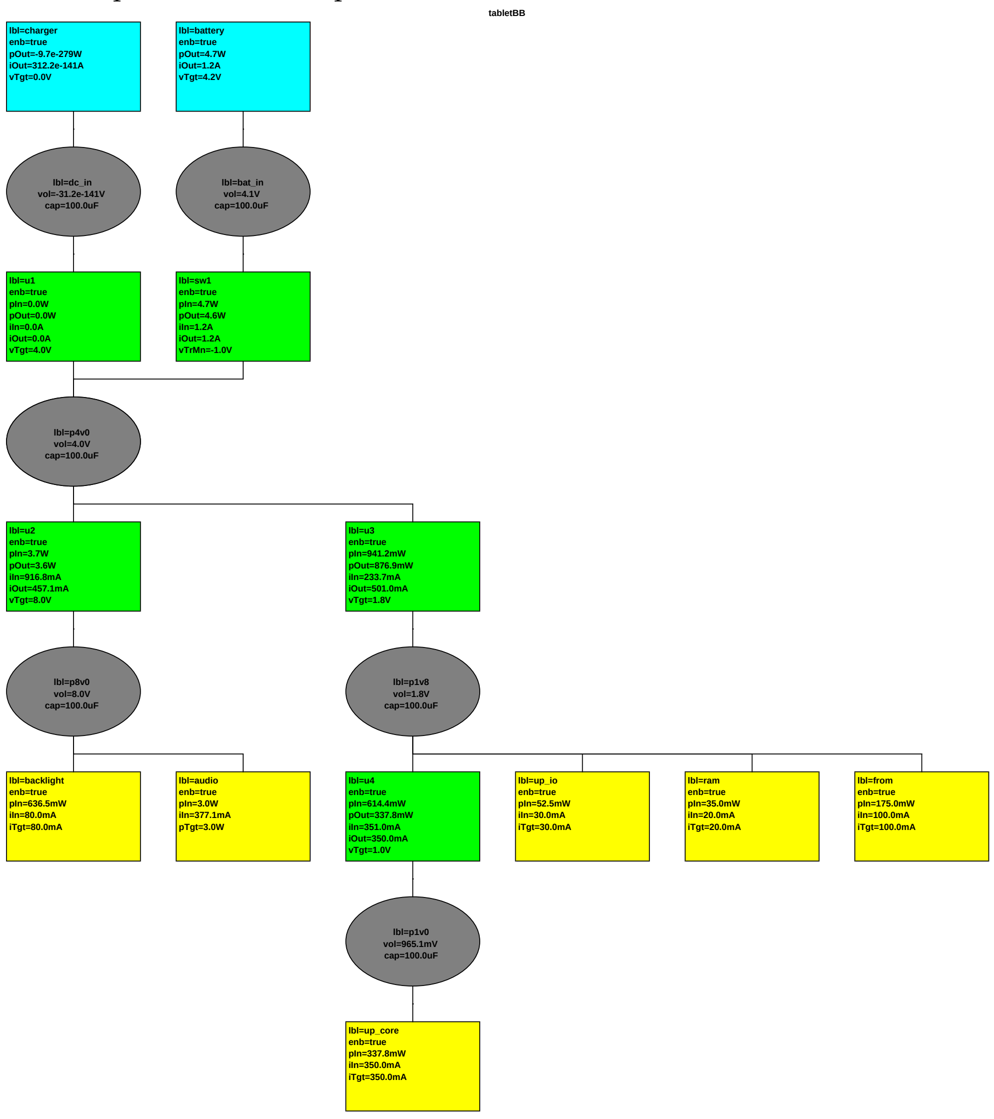
Once the power architecture is defined some commands are used to play with it.
The work begins simulating the power architecture. The second step is to show the simulation results to be checked for problems. If some problem is found there it is possible to modify the power architecture to restart a new cycle.

2.2.1 Simulate Commands

Pentode performs a time domain simulation based on Runge Kutta 3 as default. It is possible to use several integration algorithms setting a parameter: increasing their complexity improves the result at expense of calculation length. The simulation begins from the initial state of the node voltages (0V by default).
With the *simulateTransient* several node voltages and gate currents during the simulation are stored and the simulation ends at the user defined time. That simulation is useful to verify power on and off sequences and other dynamic behavior looking at their voltage and current plots.
It is also available a *simulateSteady* command which is a transient simulation automatically interrupted when the node currents mismatch is below a given threshold or if the maximum user defined simulation time is reached. In this simulation only the last voltage and current state is saved. This simulation is useful to check for power balance, efficiency and power consumption in several cases.
At the end of simulations all the components are checked against strange behavior: like a reverse biased diode or regulator exceeding its maximum current.

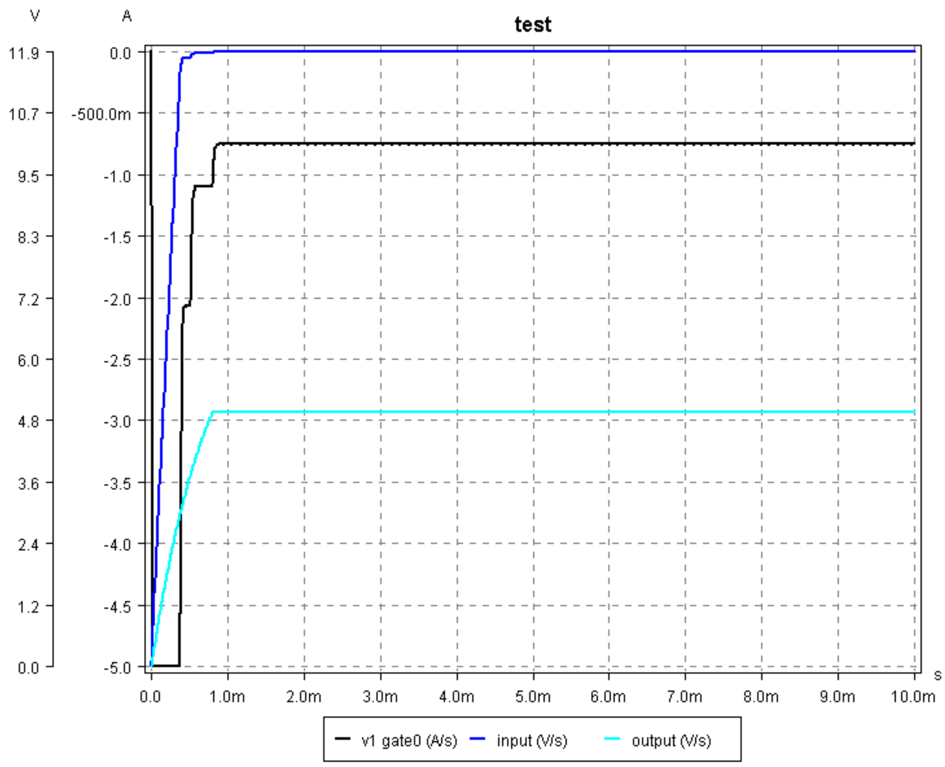
2.2.2 Show Commands

Once a simulation is run it is possible to *draw* a power tree. The power tree is a graphical picture of the last state computed on the (usually steady) simulation. The black boxes are drawn as squares with color and placement depending on their nature. Sources are drawn on top in light blue boxes, converters are green in the middle and the drainer are in yellow at the bottom. The black box gates are linked together by nodes drown as gray oval. On the next picture an example is visible:


tabletBB
lbl=charger enb=true pOut=-9.7e-279W iOut=312.2e-141A vTgt=0.0V
lbl=battery enb=true pOut=4.7W iOut=1.2A vTgt=4.2V
lbl=dc_in vol=-31.2e-141V cap=100.0uF
lbl=bat_in vol=4.1V cap=100.0uF
lbl=u1 enb=true pIn=0.0W pOut=0.0W iIn=0.0A iOut=0.0A vTgt=4.0V
lbl=sw1 enb=true pIn=4.7W pOut=4.6W iIn=1.2A iOut=1.2A vTrMn=-1.0V
lbl=p4v0 vol=4.0V cap=100.0uF
lbl=u2 enb=true pIn=3.7W pOut=3.6W iIn=916.8mA iOut=457.1mA vTgt=8.0V
lbl=u3 enb=true pIn=941.2mW pOut=876.9mW iIn=233.7mA iOut=501.0mA vTgt=1.8V
lbl=p8v0 vol=8.0V cap=100.0uF
lbl=p1v8 vol=1.8V cap=100.0uF
lbl=backlight enb=true pIn=636.5mW iIn=80.0mA iTgt=80.0mA
lbl=audio enb=true pIn=3.0W iIn=377.1mA pTgt=3.0W
lbl=u4 enb=true pIn=614.4mW pOut=337.8mW iIn=351.0mA iOut=350.0mA vTgt=1.0V
lbl=up_io enb=true pIn=52.5mW iIn=30.0mA iTgt=30.0mA
lbl=ram enb=true pIn=35.0mW iIn=20.0mA iTgt=20.0mA
lbl=from enb=true pIn=175.0mW iIn=100.0mA iTgt=100.0mA
lbl=p1v0 vol=965.1mV cap=100.0uF
lbl=up_core enb=true pIn=337.8mW iIn=350.0mA iTgt=350.0mA

Those graphical elements have text showing the value of a subset of the

component parameters: label, power, voltage, etc. The label is always printed and also the input and output power for black box. The parameters to show are different depending on the component type. It is possible to customized the parameters to show and the color to use. The diagram is saved on svg or pdf files.
Once a transient simulation is run all node voltages and gate currents are saved. A new variable is created (or updated) called TIME. It contains the time points at which the voltage and current values are taken. It is possible to set the number of points to store for the simulation, the default is 1000. Pentode is capable to take a variable for x-axis (usually TIME) and one or more variables for y-axis to plot them on a graph:



The plots are saved on several graphic formats and also in a text file as csv to be manipulated by other programs.
It is possible to *print* a component. The print shows its parameter status.
It is possible to *list* the components current available on the net list, the variables, etc.

2.2.3 Modify

The main way to check a power architecture is to try to *modify* some parameter of the components used and simulate again to check the result. It is possible to *delete* components and add new ones just with their definition.
One important property is the component *cost*. It is defined by its value and currency. All the components costs can be extracted to compute the BOM (bill of material) amount. In this context all costs are converted to the base currency value.
Pentode can define variables. They are single value or arrays with an optional measurement unit. Few variables are automatic made by Pentode, those are in upper case as the variable TIME in a transient simulations: that vector keeps trace of the time at which voltages and currents values are taken. A calculator allows to set a variable value based on the result of computation of other variables. There are iteration commands used to *loop* in a condition or iterate *for* a list. Also the basic *if* conditional statements available.
The new commands given can be saved updating the net list file or load and execute another file.

2.3 Syntax

Pentode read user input in two ways: from a file (ptd extension) and/or interactively by the command line. If the input comes from the command line it is executed just after enter key is pressed. However in few conditions like loops it is not possible to execute the command until all the commands within the loop are written. In that condition it is required to end the lines with + symbol. In that condition Pentode store the lines in its internal buffer until a line without the ending + is found. That is not required for file input.
Comment begin with the symbol # and goes ahead up to the end of line. A comment can fill a whole line or can begin at the end of a command.
Every Pentode command has the following structure:
name label node1 ... nodeN unlabeled-parameter parameter1 = value1 ... parameterN = valueN
Where:

- name of the command to execute (it may be the definition of component that will be added)
- label is the component label or the label required by the command: not all the commands need a label

- node1 to nodeN if the command is a black box with N gates it is required to define the node1 to nodeN labels the black box gates are linked to. The nodes not already defined are made just before the black box. It is possible to explicitly define the nodes before the black boxes but it is not required. It is done only if the node have not-default parameters.

- unlabeled-property if available it must be defined just after the node labels

- property it is possible to set every property to customize the component behavior. The properties can be enter in which ever order. When a property is not set its default value is used. If a property label is not recognized a warning is provided and it is skipped. As a rule Pentode try to go ahead with the input file. There are very few properties without default value which must be explicitly set. Usually the property label are made removing the vocals and using 3 chars: therefore the reverse resistance become rRev. Since there are a lot of properties for every command the help followed by a command name shows the meaning of each property and print followed by a label shows the given component properties settings.

Chapter 3

Components

This chapter describes the components available on Pentode and their basic properties.

3.1 Parameter

A parameter specifies the behavior of a command. It may be the output voltage of a battery or the capacitance of a capacitor.
Every property has an unique label made by few chars: usually beginning 3 chars of its name skipping vocals. To set a property it is required to write its label, the equal symbol '=', the new value. The property may be of different types: double number, integer number, string, waveform, boolean, list, vector of double, color, money. There are few read-only property which rises an exception if the value is changed. Almost all the properties have a default value that is kept if not set. All properties have a simple help visible looking for the help of the whole command.

3.1.1 Parameter Types

There are several parameter types available: double number (double precision real number like 1.1), vector of double, integer, boolean, string, electric component, list, money, option, read only.
Double number are used to set parameter for the black boxes internal behavior. It may be the resistance of a resistor or the output voltage of a voltage source. Real number are entered in engineer notation: the value magnitude is followed by one of the following prefix: ”f”, ”p”, ”n”, ”u”, ”m”, ””, ”k”,

”M”, ”G”, ”T” that stands for 10 at power of -15, -12, -9, -6, -6, -3, 3, 3, 6, 9, 12. After the prefix it is also possible to insert the measurement unit (it is used the mksA system, for Ohm it is required to write Ohm because the symbol is not available on the keyboard). The measurement unit is optional however if it is used it must match the correct one otherwise a parse error is generated and the value is discarded. Is is suggested to use the measurement unit when setting parameter to avoid any mistake. As example a 1000 Ohm resistor can be entered as 1kOhm or 1k or 0.001MOhm or 0.001M.
A vector of double can be entered with the following syntax: [v1 v2 .... vn —] to understand the meaning see the section about the calculator.
An integer value is just an integer value for instance the point of a simulation to save is an integer.
A boolean value is true or false, for instance the enable property for the converter is a true of false.
A string is a sequence of characters. If the string contains characters it must be contained between double apices (”). String are used as label on the components.
An electric component parameter is given by the label of the electric component optionally followed by a dot (.) and one of it parameter. That syntax is used to print and plot components.
A list is a list of some other type (of the ones explained before). For instance Plot can print several function on Y axes and they are provided after the plot command separated by commas.
A option is a value to be selected between a set of options. For instance the output file for a power tree can be selected between the options pdf and svg.
A color parameter represent a color with its red, green and blue components. For instance it is used to set the filling color for power tree drawing.
A waveform is used for transient analysis: it describes the output voltage of a source. It can be a simple DC voltage or a PWL (piece wise linear), sinusoidal, trapezoidal or triangular wavers. It can be periodic or not and it can have a set of periods. Please have a look at the manual for further details because it requires to set a lot of parameters.
A money used to represent the cost of a component. It is expected a number with dot (.) for separation and ending by a currency. If the currency is not provided the default one is used. To compute the BOM cost all the costs are converted to the default currency. It is possible to set the conversion factor with the *currency* command.
A read only parameter is a parameter that cannot be overwritten. It is used to show important internal values depending on other parameter that cannot be overwritten. For instance pOut is the output power computed as output voltage by output current and therefore it cannot be modified.

3.1.2 Calculator

3.1.3 Electrical Components Basic Properties

There are some properties common to all the electrical components which are explained below:

- name is read-only, it is the name of the component, which is the component command.
- lbl is read-only, it is the label of the component. It is the unique label of the given component.
- drwPrm is the list of parameters to plot on the power tree diagram.
- pltPr is the priority on placement from left to right in the power tree: the lower the sooner is drawn.
- drwLnk is a option set to choose how children are linked to father in the power tree diagram
- cst is the cost of the component, it is used to compute the BOM cost.

The black boxes have some common parameters:

- enb if the component is enabled: usually there is a little leakage current also when disabled.
- pLos is read only, it returns the power loss between output and input.

Source Properties

The source have further common parameters:

- eff is the efficiency and it is used to compute the power loss.
- pOut is read only, it computes the output power.
- iOut is read only, it returns the output current.
- drwClr is the fill color to be used for the drawing.

Converter Properties

The source have further common parameters:

- pIn is read only, it computes the input power.
- iIn is read only, it returns the input current.
- pOut is read only, it computes the output power.
- iOut is read only, it returns the output current.
- drwClr is the fill color to be used for the drawing.
- iOff is the input current when the device is disabled.

Drain Properties

The source have further common parameters:

- eff is the efficiency and it is used to compute the power loss.
- pIn is read only, it computes the input power.
- iIn is read only, it returns the input current.
- iOff is the input current when the device is disabled.
- drwClr is the fill color to be used for the drawing.

3.2 Source

Usually the *sources* are used to provide power to the device however they may drain as well: for instance a current source with a negative current of a voltage source with a voltage lower then the node to which it is linked. Usually sources are not used to drain because their definition is more complex because they can provide arbitrary waveforms.

3.3 Converter

Converter are used to change the power characteristic, usually its voltage, to supply other sections of the device.

3.4 Drain

Drain is the user of the power delivered by source or converter like the display back light or a Bluetooth module.

Chapter 4

Simulate Commands

This chapter explains simulation commands.

Chapter 5

Show commands

This chapter describe the commands used to show the simulation results.

Chapter 6

Modify Commands

This chapter describes the commands to modify the structure.

Chapter 7

Functions Detailed Description

7.1 command

7.1.1 help

-Name = help
-Type = command
-Label = NOT REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (default) the first chars of the command requiring help if the string contains spaces use ”
–fleNme = (default help) the output file name if was selected to output to a file if the string contains spaces use ”
–frmt = (default read) the output format has to be read or in latex format (read—latex)
–out = (default scr) the output of the help goes to the screen or to a file (scr—fle)
-Type = Command
-Function = returns the help of all the component beginning with the given text

7.2 command conditional if

7.2.1 iEnd

-Name = iEnd
-Type = command conditional if

-Label = NOT REQUIRED
-Parameters = (none)
-Type = Command
-Function = defines the end of the block if

7.2.2 if

-Name = if
-Type = command conditional if
-Label = NOT REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (must be defined) the condition checked not-equal to 0. is true
-Type = Command
-Function = defines the begin of the block if condition / iEnd. The block is executed only if the condition after if is not 0. To enter a multi line statement in interactive mode ends the lines with symbol +

7.3 command currency

7.3.1 currency

-Name = currency
-Type = command currency
-Label = REQUIRED
-Parameters =
–base = (default false) to set the base currency used to compute cost (true—false)
–cnvrsnrto = (default 1.0) the conversion ratio used to multiply the original currency to compute the BOM amount on the base currency value
–dcmldgts = (default 2) the number of decimal digits to print after the decimal point for this currency, internally all the values are stored
-Type = Command
-Function = create or modify a new currency convert factor setting its conversion factor, decimal digits and the base currency. To convert any currency to the base currency it is first multipied by its 'conversion factor' then divided by the base currency 'conversion factor' which usually is 1 and the result is rounded up. Every time a currency is set the COST variable is updated

with all the blackboxes and nodes money converted to the base currency. That variable can be used to compute the total BOM cost with the internal calculator.

7.4 command iteration

7.4.1 continue

-Name = continue
-Type = command iteration
-Label = NOT REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (default 1.0) the condition checked not-equal to 0. will restart the current loop
-Type = Command
-Function = restarts the current loop from its beginning

7.4.2 break

-Name = break
-Type = command iteration
-Label = NOT REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (default 1.0) the condition checked not-equal to 0. will break the current loop going to its end
-Type = Command
-Function = ends the current iteration blok (do-loop, if-iEnd, while-wEnd, for-next)

7.5 command iteration do

7.5.1 do

-Name = do
-Type = command iteration do
-Label = NOT REQUIRED
-Parameters = (none)
-Type = Command

-Function = defines the begin of the iteration do / loop block. To insert a multi line statement in interactive mode ends the required lines by symbol +

7.5.2 loop

-Name = loop
-Type = command iteration do
-Label = NOT REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (must be defined) the condition checked not-equal to 0. is true
-Type = Command
-Function = defines the end of do / loop condition. The loop restarts from do while the condition is != 0.

7.6 command iteration for

7.6.1 next

-Name = next
-Type = command iteration for
-Label = NOT REQUIRED
-Parameters = (none)
-Type = Command
-Function = if a new element of the vector is available restart for cycle otherwise end it. The for variable should not be modified inside the block otherwise for will restart from the first element of the vector

7.6.2 for

-Name = for
-Type = command iteration for
-Label = REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (must be defined) the array of elements to go through

-Type = Command
-Function = This command defines the begin of the iteration for vector / next which go through all the vector elements. The variable should not be changed within the cycle and should not begin with a value of the vector. To enter a multi line statement in interactive mode complete the lines with symbol +

7.7 command iteration while

7.7.1 while

-Name = while
-Type = command iteration while
-Label = NOT REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (must be defined) the condition checked not-equal to 0. is true
-Type = Command
-Function = defines the begin of the iteration while condition / wEnd which loops while the condition is not 0. To enter a multi line statement in interactive mode ends the lines with symbol +

7.7.2 wEnd

-Name = wEnd
-Type = command iteration while
-Label = NOT REQUIRED
-Parameters = (none)
-Type = Command
-Function = defines the end of the iterative while-wEnd block

7.8 command modifier

7.8.1 delete

-Name = delete
-Type = command modifier

-Label = REQUIRED
-Parameters = (none)
-Type = Command
-Function = a command to delete a blackbox, a node (without any gate linked), variable or currency. It it useful when a power tree is build interatively.

7.8.2 define

-Name = define
-Type = command modifier
-Label = REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (must be defined) A definition to be used as shortcut for write more times the same device requiring a lot of parameter, write the parameters between apex if the string contains spaces use ”
-Type = Command
-Function = defines a string to be used as replacement in the following lines. It can be used to define a long set of parameters to be used several times later. Write the string between ”

7.8.3 modify

-Name = modify
-Type = command modifier
-Label = REQUIRED
-Parameters =
–LOOKAT = (default label) the parameter that should be looked for the match (label—name)
–MATCH = (default exact) the type of match to use in the search (exact—begin—end)
-Type = Command
-Function = modifies a parameter of the component already entered in the net list whose [label—name] [matches—begin—end] with given text. Modify expects a single string value for each modified parameter. Therefore if the parameter to modify is longer it needs to be written between ”. For instance to modify drwClr it is required to write drwClr=”1,2,3” or to modify drwPrm list it requires drwPrm=”lbl,vol”

7.9 command simulation

7.9.1 simulateTransient

-Name = simulateTransient
-Type = command simulation
-Label = NOT REQUIRED
-Parameters =
–diSteadyMax = (default 1.0nA) the maximum current variation on the current at worst node to be in steady state
–dtBegin = (default 1.0ns) the simulation initial time step
–dtMax = (default 1.0us) the simulation max time step
–dtMin = (default 1.0ps) the simulation min time step
–dvMax = (default 1.0mV) the maximum node voltage variation require to decrease time step
–dvMin = (default 10.0uV) the minimum node voltage variation require to increase time step
–savePoints = (default 1000) how many calculated point has to be saved for a simulation. First and last are saved by default
–saveTransient = (default true) choose if save transient analisys. If the transient is not saved the source are set constant low value. (true—false)
–stopAtSteadyState = (default false) choose if stop when steady state is reached (true—false)
–tEnd = (default 1.0s) the simulation end time
–tSave = (default 0.0s) the time to begin data saving
-Type = Command
-Function = runs the transient simulation up to endTime saving the history. It is useful to verify turn on and off sequence and over shoots.

7.9.2 simulateSteady

-Name = simulateSteady
-Type = command simulation
-Label = NOT REQUIRED
-Parameters =
–diSteadyMax = (default 1.0nA) the maximum current variation on the current at worst node to be in steady state
–dtBegin = (default 1.0ns) the simulation initial time step
–dtMax = (default 1.0us) the simulation max time step
–dtMin = (default 1.0ps) the simulation min time step

–dvMax = (default 1.0mV) the maximum node voltage variation require to decrease time step
–dvMin = (default 10.0uV) the minimum node voltage variation require to increase time step
–measurementUnit = (default) the measurement unit to use for each parameter stepped
–parameter = (default) the parameter to step
–savePoints = (default 1000) how many calculated point has to be saved for a simulation. First and last are saved by default
–saveTransient = (default false) choose if save transient analisys. If the transient is not saved the source are set constant low value. (true—false)
–stopAtSteadyState = (default true) choose if stop when steady state is reached (true—false)
–tEnd = (default 1.0s) the simulation end time
–tSave = (default 0.0s) the time to begin data saving
–values = (default) the vector with values to simulate
-Type = Command
-Function = runs the transient simulation stopping as soon as a steady state is reached only the last point is saved on history

7.10 command system

7.10.1 save

-Name = save
-Type = command system
-Label = NOT REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (default) the file name where the current parsed lines are saved if the string contains spaces use ”
-Type = Command
-Function = saves the current parsed lines (which contains the interactive session input). It is possible to provide an optional fileName otherwise the device fileName is used.

7.10.2 load

-Name = load
-Type = command system
-Label = REQUIRED
-Parameters = (none)
-Type = Command
-Function = a command to load and parse the lines found in the given file-Name

7.10.3 setup

-Name = setup
-Type = command system
-Label = NOT REQUIRED
-Parameters =
–decimalDigits = (default 1) the number of decima digit to show
–drawBackGroundColor = (default color RGB components: 255, 255, 255) the background color used to color the page of the drawing
–drawBoxHeight = (default 100) the height (in pixels) of the boxes used to draw the power tree
–drawBoxSpaceX = (default 40) the x spacing (in pixels) between boxes
–drawBoxSpaceY = (default 40) the y spacing (in pixels) between boxes
–drawBoxWidth = (default 150) the width (in pixels) of the boxes used to draw the power tree
–drawForeGroundColor = (default color RGB components: 0, 0, 0) the foreground color used to color the labels and boxes of the drawing
–integratorAlgorithm = (default rungekutta3) sets the integration algorithm to use. Euler needs 1 step to compute the next node voltages. Midpoint, Heun and Ralston 2 steps. RungeKutta3 3 steps, while the other RungeKutta needs 4 steps for 1 point. RungeKutta3 is the best compromise speed and transient accuracy. Euler is the fastest and can be used safely if a steady state analysis is performed. RungeKutta4 methods are the most accurate but also require to reduce the dvMax in order to get an accurate transient simulation. (rungekutta3—euler—midpoint—rungekutta4bis3/8—rungeKutta4—ralston—heun)
-Type = Command
-Function = sets up the basic behaviour of the power tree designer: decimal digits, ...

7.10.4 quit

-Name = quit
-Type = command system
-Label = NOT REQUIRED
-Parameters = (none)
-Type = Command
-Function = exits Power Tree Designer. Any command given in the interactive session is not saved

7.11 command variable

7.11.1 set

-Name = set
-Type = command variable
-Label = REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (default) the value, the variable to copy or the calculation to set in the new variable
–measurementUnit = (default) the measurement unit assigned to the variable if the string contains spaces use ”
-Type = Command
-Function = defines a new variable or modify an existing one. The first field is the variable name. It requires a value and a vector. Whenever a vector is required it is possible to use the calculator. The calculator is a simple program working in reverse polish notation between square brackets [...]. The following calculator commands are available:
-Command -Command * takes 2 arguments from the stack to times the last two values of the stack
-Command + takes 2 arguments from the stack to sum two values
-Command ++ takes 1 arguments from the stack to increment the last vector of the stack
-Command - takes 2 arguments from the stack to subtract two values
-Command – takes 1 arguments from the stack to decrement the last vector of the stack
-Command / takes 2 arguments from the stack to divides the last two values of the stack
-Command 0- takes 1 arguments from the stack to negate the last vector of the stack

-Command 1/ takes 1 arguments from the stack to invert the last vector of the stack
-Command :=: takes 2 arguments from the stack to check if two number are the same value
-Command ¡ takes 2 arguments from the stack to check if the first number is less than second
-Command ¡= takes 2 arguments from the stack to check if the first number is less than or equal second
-Command ¿ takes 2 arguments from the stack to check if the first number is greater than second
-Command ¿= takes 2 arguments from the stack to check if the first number is greater than or equal second
-Command abs(takes 1 arguments from the stack to return the absolute value
-Command and(takes 2 arguments from the stack to return 1 of the last two valueas are not 0
-Command ceil(takes 1 arguments from the stack to return the ceil given value
-Command drop(takes 1 arguments from the stack to remove the last value from the stack
-Command duplicate(takes 1 arguments from the stack to duplicate the last value from the stack
-Command floor(takes 1 arguments from the stack to return the floor given value
-Command get(get the given element from a vector
-Command join(join in a vector the last n elements of the stack
-Command ne(takes 2 arguments from the stack to check if two number are different value
-Command not(takes 1 arguments from the stack to return 1 if the last two valueas is 0
-Command or(takes 2 arguments from the stack to return 1 if at least one of the last two valueas are not 0
-Command pow(takes 2 arguments from the stack to rise to the given power
-Command sign(takes 1 arguments from the stack to return the sign
-Command size(takes 1 arguments from the stack to return the size of the last vector of the stack
-Command sqr(takes 1 arguments from the stack to return the square of the given value
-Command sqrt(takes 1 arguments from the stack to return the square root of the given value
-Command sumup(takes 1 arguments from the stack to return the sum of

the elements of the last vector of the stack
-Command swap(takes 2 arguments from the stack to swaps the last two vectors from the stack
-Command vectorLinear(takes 3 arguments from the stack to create a vector from start value, number of points, end value
-Command vectorStep(takes 3 arguments from the stack to create a vector from start value, with step value up to end value
-Command xor(takes 2 arguments from the stack to return 1 if one but not both of the last two valueas are not 0
-Command — join in a vector all the elements of the stack

7.12 command work output

7.12.1 plot

-Name = plot
-Type = command work output
-Label = REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (default) the list of parameters to show
–decimalPoint = (default .) the chars used as decimal point if the string contains spaces use ”
–fieldsSeparator = (default ;) the chars used as field separator if the string contains spaces use ”
–outputFormat = (default png) the output file format (png—jpeg—bmp—wbmp—gif—csv—
-Type = Command
-Function = saves on a file csv, a graphic plot or on the screen the history of the given list of parameters. On graph the first parameter is used for X axis, the outhers are plotted as Ys. If measurement units are defined they are visible on the axis.

7.12.2 draw

-Name = draw
-Type = command work output
-Label = REQUIRED
-Parameters =

–outputFormat = (default pdf) the output file format (pdf—svg)
-Type = Command
-Function = draws the power tree of the given netlist on a file in pdf or svg file

7.12.3 list

-Name = list
-Type = command work output
-Label = NOT REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (default blackboxes) the kind of element to list (blackboxes—variables—parsedLines—nodes—currencies—definitions)
–show = (default name, lbl, lnks) the list of parameter to show for black boxes
-Type = Command
-Function = lists the variable or blackBox (which is the netlist) or node or parsedLine

7.12.4 print

-Name = print
-Type = command work output
-Label = NOT REQUIRED
-Parameters =
–UNLABELLEDPARAMETER = (must be defined) the list electric component of parameters to print
-Type = Command
-Function = prints on screen a parameter of the given node or blackBox with syntax label.parameter. If it is used an integer: label.0—1 the history of last simulation on that port will be printed

7.13 component converter

7.13.1 cnvSwitch

-Name = cnvSwitch
-Type = component converter

-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, pOut, iIn, iOut, res) the list of parameter to show in the power tree drawing
–enb = (default true) to enable or disable the device (true—false)
–iIn = (read only) the input current
–iMax = (default 10.0A) switch maximum forward current used only for warnings (not for simulation)
–iOff = (default 1.0uA) the input current when the device is disabled
–iOut = (read only) the output current
–iQst = (default 1.0mA) the current required to supply the switch circuitry
–lbl = (read only) the component label
–lnks = (read only) the component links
–name = (read only) the component name
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pOut = (read only) the output power
–pltPr = (default 2000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–rRev = (default 1.0Ohm) the parassitic diode forward resistance
–res = (default 50.0mOhm) the on resistance
–vMin = (default 2.0V) the minimum operative input voltage
–vRev = (default 1.0V) the parassitic diode forward voltage
-Type = Electric Component
-Number of gates = 2
-Class = converter
-Function = is a switch

7.13.2 cnvBoost

-Name = cnvBoost
-Type = component converter

-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, pOut, iIn, iOut, vTgt) the list of parameter to show in the power tree drawing
–enb = (default true) to enable or disable the device (true—false)
–iIMax = (default 10.0A) the maximum input current
–iIn = (read only) the input current
–iOff = (default 1.0uA) the input current when the device is disabled
–iOut = (read only) the output current
–iPfm = (default 10.0mA) the minimum current below which the frequency is reduced and the switching current decreases proportionally
–iQst = (default 1.0mA) the baseline current to supply the converter circuitry
–iRev = (default 1.0uA) the diode reverse current
–iSwt = (default 2.5mA) the current required for switching
–lbl = (read only) the component label
–lnRgl = (default 10.0m–lnks = (read only) the component links
–name = (read only) the component name
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pOut = (read only) the output power
–pltPr = (default 2000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–rLdRg = (default 100.0mOhm) the load regulation expressed as resistance
–rSwt = (default 200.0mOhm) the switching circuit resistance
–vFrw = (default 500.0mV) the diode forward voltage
–vMin = (default 1.5V) the minimum startup voltage
–vTgt = (must be defined) the output target voltage
-Type = Electric Component
-Number of gates = 2
-Class = converter
-Function = is a switching boost converter

7.13.3 cnvBuck

-Name = cnvBuck
-Type = component converter
-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, pOut, iIn, iOut, vTgt) the list of parameter to show in the power tree drawing
–enb = (default true) to enable or disable the device (true—false)
–iIn = (read only) the input current
–iOMax = (default 4.0A) the maximum output current
–iOff = (default 1.0uA) the input current when the device is disabled
–iOut = (read only) the output current
–iPfm = (default 10.0mA) the minimum current below which the frequency is reduced and the switching current decreases proportionally
–iQst = (default 1.0mA) the baseline current to supply the converter circuitry
–iSwt = (default 2.5mA) the current required for switching
–lbl = (read only) the component label
–lnRgl = (default 10.0m–lnks = (read only) the component links
–name = (read only) the component name
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pOut = (read only) the output power
–pltPr = (default 2000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–rLdRg = (default 100.0mOhm) the load regulation expressed as resistance
–rRev = (default 1.0Ohm) the high side parassitic diode forward resistance
–rSwt = (default 200.0mOhm) the switching circuit resistance
–vMin = (default 1.5V) the minimum startup voltage
–vRev = (default 1.0V) the high side parassitic diode forward voltage
–vTgt = (must be defined) the output target voltage
-Type = Electric Component

-Number of gates = 2
-Class = converter
-Function = is a switching buck converter

7.13.4 cnvResistor

-Name = cnvResistor
-Type = component converter
-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, pOut, iIn, iOut, ioRes, oRes) the list of parameter to show in the power tree drawing
–enb = (default true) to enable or disable the device (true—false)
–iIn = (read only) the input current
–iOff = (default 1.0uA) the input current when the device is disabled
–iOut = (read only) the output current
–iRes = (default 1.0MOhm) the input resistance: from input to GND
–ioRes = (default 1.0Ohm) the resistance from input to output
–lbl = (read only) the component label
–lnks = (read only) the component links
–name = (read only) the component name
–oRes = (default 1.0MOhm) the output resistance: from output to GND
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pOut = (read only) the output power
–pltPr = (default 2000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
-Type = Electric Component
-Number of gates = 2
-Class = converter
-Function = is a Greek-Pi resistor partition network. It is made by three resistors: input resistor, output resistor and input-output resistor.

7.13.5 cnvInductor

-Name = cnvInductor
-Type = component converter
-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, pOut, iIn, iOut, ind) the list of parameter to show in the power tree drawing
–enb = (default true) to enable or disable the device (true—false)
–iBgn = (default 0.0A) simulation starting current
–iIn = (read only) the input current
–iMax = (default 10.0A) inductor maximum forward current used only for warnings (not for simulation)
–iOff = (default 1.0uA) the input current when the device is disabled
–iOut = (read only) the output current
–ind = (must be defined) inductance value
–lbl = (read only) the component label
–lnks = (read only) the component links
–name = (read only) the component name
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pOut = (read only) the output power
–pltPr = (default 2000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–res = (default 0.0Ohm) inductance series resistance
-Type = Electric Component
-Number of gates = 2
-Class = converter
-Function = is an inductor modeled by an ideal inductor in series to a resistor.

7.13.6 cnvDiode

-Name = cnvDiode
-Type = component converter
-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, pOut, iIn, iOut, vFrw) the list of parameter to show in the power tree drawing
–enb = (default true) to enable or disable the device (true—false)
–iIn = (read only) the input current
–iMax = (default 10.0A) diode maximum forward current used only for warnings (not for simulation)
–iOff = (default 1.0uA) the input current when the device is disabled
–iOut = (read only) the output current
–iRev = (default 1.0uA) diode reverse leakage current
–lbl = (read only) the component label
–lnks = (read only) the component links
–name = (read only) the component name
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pOut = (read only) the output power
–pltPr = (default 2000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–rFrw = (default 300.0mOhm) diode forward resistance
–vFrw = (default 600.0mV) diode forward voltage
-Type = Electric Component
-Number of gates = 2
-Class = converter
-Function = is a diode

7.13.7 cnvLDO

-Name = cnvLDO
-Type = component converter
-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, pOut, iIn, iOut, vTgt) the list of parameter to show in the power tree drawing
–enb = (default true) to enable or disable the device (true—false)
–iIn = (read only) the input current
–iMax = (default 1.0A) the maximum output current
–iOff = (default 1.0uA) the input current when the device is disabled
–iOut = (read only) the output current
–iQst = (default 1.0mA) the device baseline supply current
–lbl = (read only) the component label
–lnRgl = (default 10.0m–lnks = (read only) the component links
–name = (read only) the component name
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pMax = (default 10.0W) the maximum power dissipable
–pOut = (read only) the output power
–pltPr = (default 2000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–rFrw = (default 100.0mOhm) the parassitic diode forward resistance
–rLdRg = (default 100.0mOhm) the load regulation expressed as resistance
–rRev = (default 1.0Ohm) the parassitic diode reverse resistance
–vDrp = (default 200.0mV) the minimum required voltage drop
–vRev = (default 1.0V) the parassitic diode forward voltage
–vTgt = (must be defined) the target output voltage
-Type = Electric Component
-Number of gates = 2
-Class = converter
-Function = is a linear (low drop out) regulator

7.13.8 cnvCntSwitch

-Name = cnvCntSwitch
-Type = component converter
-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, pOut, iIn, iOut, vTrMn, vTrMx) the list of parameter to show in the power tree drawing
–enb = (default true) to enable or disable the device (true—false)
–iIn = (read only) the input current
–iMax = (default 10.0A) switch maximum forward current used only for warnings (not for simulation)
–iOff = (default 1.0uA) the input current when the device is disabled
–iOut = (read only) the output current
–iQst = (default 1.0mA) the current required to supply the switch circuitry
–lbl = (read only) the component label
–lnks = (read only) the component links
–name = (read only) the component name
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pOut = (read only) the output power
–pltPr = (default 2000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–rRev = (default 1.0Ohm) the parassitic diode forward resistance
–res = (default 50.0mOhm) the on resistance
–vMin = (default 2.0V) the minimum operative input voltage
–vRev = (default 1.0V) the parassitic diode forward voltage
–vTrMn = (default 1.0V) the low threshold voltage, it is active above it
–vTrMx = (default 10.0V) the high threshold voltage, it is active below it
-Type = Electric Component
-Number of gates = 3

-Class = converter
-Class = voltage controlled converter
-Function = is voltage controlled switch. The third node is the gate. The switch is controlled by the gate voltage. The switch is closed if the voltage at the third node it between vTrMn and vTrMx

7.14 component drain

7.14.1 drnPower

-Name = drnPower
-Type = component drain
-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 255, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, iIn, pTgt) the list of parameter to show in the power tree drawing
–eff = (default 0.0–enb = (default true) to enable or disable the device (true—false)
–iIn = (read only) the input current
–iMax = (default 1.0A) the maximum drained current
–iOff = (default 1.0uA) the current drained when the device is disabled
–lbl = (read only) the component label
–lnks = (read only) the component links
–name = (read only) the component name
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pTgt = (must be defined) the target drained power
–pltPr = (default 3000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–vMin = (default 2.0V) the minimum operative voltage
-Type = Electric Component

-Number of gates = 1
-Class = drainer
-Function = is a constant power drain

7.14.2 drnResistor

-Name = drnResistor
-Type = component drain
-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 255, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, iIn, res) the list of parameter to show in the power tree drawing
–eff = (default 0.0–enb = (default true) to enable or disable the device (true—false)
–iIn = (read only) the input current
–iMax = (default 10.0A) maximum forward current used only for warnings (not for simulation)
–iOff = (default 1.0uA) the current drained when the device is disabled
–lbl = (read only) the component label
–lnks = (read only) the component links
–name = (read only) the component name
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pltPr = (default 3000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–res = (must be defined) the component resistance
–vMin = (default 0.0V) the minimum operative voltage
-Type = Electric Component
-Number of gates = 1
-Class = drainer
-Function = is a resistor

7.14.3 drnCurrent

-Name = drnCurrent
-Type = component drain
-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 255, 255, 0) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pIn, iIn, iTgt) the list of parameter to show in the power tree drawing
–eff = (default 0.0–enb = (default true) to enable or disable the device (true—false)
–iIn = (read only) the input current
–iOff = (default 1.0uA) the current drained when the device is disabled
–iTgt = (must be defined) the target constant current drained
–lbl = (read only) the component label
–lnks = (read only) the component links
–name = (read only) the component name
–pIn = (read only) the input power
–pLos = (read only) the power loss computed with efficiency
–pltPr = (default 3000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–vMax = (default 50.0V) maximum applicable voltage used only for warnings (not for simulation)
–vMin = (default 1.0V) the minimum operative voltage, below this level the curred linearly reduce to zero
-Type = Electric Component
-Number of gates = 1
-Class = drainer
-Function = is constant current drain

7.15 component node

7.15.1 node

-Name = node
-Type = component node
-Label = REQUIRED
-Parameters =
–cap = (default 100.0uF) the capacity linked to the node
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 128, 128, 128) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, vol, cap) the list of parameter to show in the power tree drawing
–lbl = (read only) the component label
–lnks = (read only) the component links
–name = (read only) the component name
–pltPr = (default 0) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–vBgn = (default 0.0V) the initial voltage used to begin simulation
–vol = (read only) the node voltage
-Type = Electric Component
-Class = voltage node
-Function = is a node of the power tree. It is intrinsecally defined when a component is liked to one ore more nodes. So usually it is not required to define it. It can be defined to set its capacity and initial voltage. If defined it has to be declared before its usage.

7.16 component source

7.16.1 srcBattery

-Name = srcBattery
-Type = component source
-Label = REQUIRED

-Parameters =
–aCpty = (read only) the available capacity after simulation
–cpty = (default 3.6kC) The battery capacity expressed in Culomb, it is usually provided in Ah: 1Ah=3600C
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 255) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pOut, iOut, vBtt, cpty) the list of parameter to show in the power tree drawing
–eRes = (default 20.0mOhm) The battery external resistance, together to the external resistance it models the short DC pulse current capability than the DC current
–eff = (default 0.0–enb = (default true) to enable or disable the device (true—false)
–iCap = (default 100.0mF) The battery internal capacitor, together to the external resistance it models the short DC pulse current capability than the DC current
–iOut = (read only) the output current
–iRes = (default 80.0mOhm) The battery internal resistance used to charge its internal capacitor.It models the long DC current capability of the battery
–lbl = (read only) the component label
–lnks = (read only) the component links
–name = (read only) the component name
–pLos = (read only) the power loss computed with efficiency
–pOut = (read only) the output power
–pltPr = (default 1000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–rchrgbl = (default true) An input current increases the internal capacity, if the battery is not rechargeable that rises a warning (true—false)
–v0 = (default 3.2V) The battery voltage at 0–v10 = (default 3.5V) The battery voltage at 10–v100 = (default 4.2V) The battery voltage at 100–v90 = (default 3.9V) The battery voltage at 90–vBtt = (read only) the internal battery voltage value during simulation
-Type = Electric Component
-Number of gates = 1
-Class = sourcer

-Function = is a battery

7.16.2 srcCurrent

-Name = srcCurrent
-Type = component source
-Label = REQUIRED
-Parameters =
–aHgh = (default 10.0A) the high peak for the waveform
–aLow = (default 0.0A) the voltage value for dc, the low peak for other waveforms, the value used for steady state simulation
–aVct = (default) the list of A points to use for a pwl waveform
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 255) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pOut, iOut, iTgt) the list of parameter to show in the power tree drawing
–eff = (default 0.0–enb = (default true) to enable or disable the device (true—false)
–iOut = (read only) the output current
–iTgt = (read only) the last current value during simulation
–lbl = (read only) the component label
–lnks = (read only) the component links
–nPrds = (default 0) is used to set the number of periods if the wavefor is periodic, ¡= 0 means infinity
–name = (read only) the component name
–pLos = (read only) the power loss computed with efficiency
–pOut = (read only) the output power
–pltPr = (default 1000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–res = (default 1.0kOhm) the output resistance
–tDly = (default 0.0s) the delay of the trapezoidal or phase for sin waveform
–tFll = (default 10.0ms) the fall time for trapezoidal wave
–tPls = (default 500.0ms) the pulse duration for trapezoidal wave
–tPrd = (default 1.0s) the period of the waveform, use wPrdc if the wave-

form has to repeat aftera 1 period
–tRse = (default 10.0ms) the rise time for trapezoidal wave
–tVct = (default) the list of time points to use for a pwl waveform
–vMax = (default 5.0V) the maximum output voltage
–wPrdc = (default false) is used to set if the wavefor is periodic, in case tPrd is used for period (true—false)
–wTyp = (default dc) is used to set the waveform type:
—dc constant waveform requires (v/i)low to be defined
—pwl piece wise linear waveform requires to define tVct=[t0 t1 t2 .. tn —] (v/i)Vct=[y0 y1 y2 .. yn —] were ti are absolute time instant if positive, if negative it is considered time increment respect to the previous step
—saw saw wave requires to define: (v/i)Low the low voltage, (v/i)Hgh the high voltage, tRse the rise time, tFll the fall time, and tPrd the period
—sin constant waveform requires (v/i)low to be defined
—tri triangulare wave requires to define: (v/i)Low the low voltage, (v/i)Hgh the high voltage, and tPrd the period
—trp trapezoidal wave requires to define: (v/i)Low the low voltage, (v/i)Hgh the high voltage, tDly the high pulse delay, tRse the rise time, tPls the high pulse time, tFll the fall time, tPrd the period (dc—pwl—trp—saw—tri—sin)
-Type = Electric Component
-Number of gates = 1
-Class = sourcer
-Function = is a constant current source

7.16.3 srcVoltage

-Name = srcVoltage
-Type = component source
-Label = REQUIRED
-Parameters =
–cst = (default 0.00) the component cost Every money amout should be followed by its currency symbol. If the symbol was not used before it is automatically added to the currency database. If no currency is specified the base currency is used.
–drwClr = (default color RGB components: 0, 255, 255) the fill color used for drawing
–drwLnk = (default full) how should be draw the link from child to his father (full—dash—cpChild—void)
–drwPrm = (default lbl, enb, pOut, iOut, vTgt) the list of parameter to show in the power tree drawing

–eff = (default 0.0–enb = (default true) to enable or disable the device (true—false)
–iMax = (default 5.0A) the maximum output current
–iOut = (read only) the output current
–lbl = (read only) the component label
–lnks = (read only) the component links
–nPrds = (default 0) is used to set the number of periods if the wavefor is periodic, ¡= 0 means infinity
–name = (read only) the component name
–pLos = (read only) the power loss computed with efficiency
–pOut = (read only) the output power
–pltPr = (default 1000) the draw priority, the lower more on the left is drawed. By default source have 1000, converter 2000 and drain 3000.
–res = (default 100.0mOhm) the output resistance
–tDly = (default 0.0s) the delay of the trapezoidal or phase for sin waveform
–tFll = (default 10.0ms) the fall time for trapezoidal wave
–tPls = (default 500.0ms) the pulse duration for trapezoidal wave
–tPrd = (default 1.0s) the period of the waveform, use wPrdc if the waveform has to repeat aftera 1 period
–tRse = (default 10.0ms) the rise time for trapezoidal wave
–tVct = (default) the list of time points to use for a pwl waveform
–vHgh = (default 10.0V) the high peak for the waveform
–vLow = (default 0.0V) the voltage value for dc, the low peak for other waveforms, the value used for steady state simulation
–vTgt = (read only) the last voltage value during simulation
–vVct = (default) the list of V points to use for a pwl waveform
–wPrdc = (default false) is used to set if the wavefor is periodic, in case tPrd is used for period (true—false)
–wTyp = (default dc) is used to set the waveform type:
—dc constant waveform requires (v/i)low to be defined
—pwl piece wise linear waveform requires to define tVct=[t0 t1 t2 .. tn —] (v/i)Vct=[y0 y1 y2 .. yn —] were ti are absolute time instant if positive, if negative it is considered time increment respect to the previous step
—saw saw wave requires to define: (v/i)Low the low voltage, (v/i)Hgh the high voltage, tRse the rise time, tFll the fall time, and tPrd the period
—sin constant waveform requires (v/i)low to be defined
—tri triangulare wave requires to define: (v/i)Low the low voltage, (v/i)Hgh the high voltage, and tPrd the period
—trp trapezoidal wave requires to define: (v/i)Low the low voltage, (v/i)Hgh the high voltage, tDly the high pulse delay, tRse the rise time, tPls the high pulse time, tFll the fall time, tPrd the period (dc—pwl—trp—saw—tri—sin)

-Type = Electric Component
-Number of gates = 1
-Class = sourcer
-Function = is a constant voltage source