CircuiTi*k*Z

version git:0ffd80a (2019/02/27)

Massimo A. Redaelli (m.redaelli@gmail.com)
Stefan Lindner (stefan.lindner@fau.de)
Stefan Erhardt (stefan.erhardt@fau.de)
Romano Giannetti (romano@rgtti.com)

February 27, 2019

Contents

1 Introduction

1.1 About

CircuiTi*k*Z was initiated by Massimo Redaelli in 2007, who was working as a research assistant at the Polytechnic University of Milan, Italy, and needed a tool for creating exercises and exams. After he left University in 2010 the development of CircuiTi*k*Z slowed down, since LaTeX is mainly established in the academic world. In 2015 Stefan Lindner and Stefan Erhardt, both working as research assistants at the University of Erlangen-Nürnberg, Germany, joined the team and now maintain the project together with the initial author.

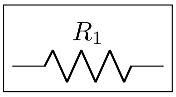
The use of CircuiTi*k*Z is, of course, not limited to academic teaching. The package gets widely used by engineers for typesetting electronic circuits for articles and publications all over the world.

This documentation is somewhat scant. Hopefully the authors will find the leisure to improve it some day.

1.2 Loading the package

LaTeX	ConTeXt[1]
`\usepackage{circuitikz}`	`\usemodule[circuitikz]`

Ti*k*Z will be automatically loaded.
CircuiTi*k*Z commands are just Ti*k*Z commands, so a minimum usage example would be:




```
\tikz \draw (0,0) to[R=$R_1$] (2,0);
```

1.3 Requirements

- `tikz`, version ≥ 3;
- `xstring`, not older than 2009/03/13;
- `siunitx`, if using `siunitx` option.

1.4 Incompatible packages

Ti*k*Z's own `circuit` library, which is based on CircuiTi*k*Z, (re?)defines several styles used by this library. In order to have them work together you can use the `compatibility` package option, which basically prefixes the names of all CircuiTi*k*Z `to[]` styles with an asterisk.

So, if loaded with said option, one must write `(0,0) to[*R] (2,0)` and, for transistors on a path, `(0,0) to[*Tnmos] (2,0)`, and so on (but `(0,0) node[nmos] {}`). See example at page 73.

1.5 License

Copyright © 2007–2017 Massimo Redaelli. This package is author-maintained. Permission is granted to copy, distribute and/or modify this software under the terms of the LaTeX Project Public License, version 1.3.1, or the GNU Public License. This software is provided 'as is', without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

[1] ConTeXt support was added mostly thanks to Mojca Miklavec and Aditya Mahajan.

1.6 Feedback

The easiest way to contact the authors is via the official Github repository: `https://github.com/circuitikz/circuitikz/issues`

2 Incompabilities between version

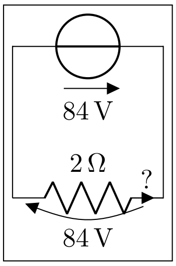
Here, we will provide a list of incompabilitys between different version of circuitikz. We will try to hold this list short, but sometimes it is easier to break with old syntax than including a lot of switches and compatibility layers. You can check the used version at your local installation using the macro `\pgfcircversion{}`.

- After v0.8.4: the parameters `tripoles/american or port/aaa`, `...bbb`, `...ccc` and `...ddd` are no longer used and are silently ignored; the same stands for `nor`, `xor`, and `xnor` ports.
- After v0.8.4: voltage and current directions/sign (plus and minus signs in case of `american voltages` and arrows in case of `european voltages` have been rationalized with a couple of new options (see details in section 7.2. The default case is still the same as v0.8.4.
- Since v0.8.2: voltage and current label directions(v<= / i<=) do NOT change the orientation of the drawn source shape anymore. Use the "invert" option to rotate the shape of the source. Furthermore, from this version on, the current label(i=) at current sources can be used independent of the regular label(l=).
- Since v0.7?: The label behaviour at mirrored bipoles has changes, this fixes the voltage drawing, but perhaps you have to adjust your label positions.
- Since v0.5.1: The parts pfet,pigfete,pigfetebulk and pigfetd are now mirrored by default. Please adjust your yscale-option to correct this.
- Since v0.5: New voltage counting direction, here exists an option to use the old behaviour

For older projects, you can use an older version locally using the git-version and picking the correct commit from the repository (branch gh-pages).

3 Package options

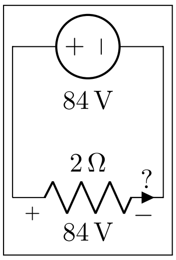
Circuit people are very opinionated about their symbols. In order to meet the individual gusto you can set a bunch of package options. The standard options are what the authors like, for example you get this:




```
\begin{circuitikz}
  \draw (0,0) to[R=2<\ohm>, i=?, v=84<\volt>] (2,0) --
     (2,2) to[V<=84<\volt>] (0,2)
     -- (0,0);
\end{circuitikz}
```

Feel free to load the package with your own cultural options:

LaTeX	ConTeXt
`\usepackage[american]{circuitikz}`	`\usemodule[circuitikz][american]`




```
\begin{circuitikz}
  \draw (0,0) to[R=2<\ohm>, i=?, v=84<\volt>] (2,0) --
     (2,2) to[V<=84<\volt>] (0,2)
     -- (0,0);
\end{circuitikz}
```

Here is the list of all the options:

- `europeanvoltages`: uses arrows to define voltages, and uses european-style voltage sources;
- `straightvoltages`: uses arrows to define voltages, and and uses straight voltage arrows;
- `americanvoltages`: uses − and + to define voltages, and uses american-style voltage sources;
- `europeancurrents`: uses european-style current sources;
- `americancurrents`: uses american-style current sources;
- `europeanresistors`: uses rectangular empty shape for resistors, as per european standards;
- `americanresistors`: uses zig-zag shape for resistors, as per american standards;
- `europeaninductors`: uses rectangular filled shape for inductors, as per european standards;
- `americaninductors`: uses ”4-bumps” shape for inductors, as per american standards;
- `cuteinductors`: uses my personal favorite, ”pig-tailed” shape for inductors;
- `americanports`: uses triangular logic ports, as per american standards;
- `europeanports`: uses rectangular logic ports, as per european standards;
- `americangfsurgearrester`: uses round gas filled surge arresters, as per american standards;
- `europeangfsurgearrester`: uses rectangular gas filled surge arresters, as per european standards;
- `european`: equivalent to `europeancurrents`, `europeanvoltages`, `europeanresistors`, `europeaninductors`, `europeanports`, `europeangfsurgearrester`;
- `american`: equivalent to `americancurrents`, `americanvoltages`, `americanresistors`, `americaninductors`, `americanports`, `americangfsurgearrester`;
- `siunitx`: integrates with `SIunitx` package. If labels, currents or voltages are of the form `#1<#2>` then what is shown is actually `\SI{#1}{#2}`;
- `nosiunitx`: labels are not interpreted as above;
- `fulldiode`: the various diodes are drawn *and* filled by default, i.e. when using styles such as `diode`, `D`, `sD`, …Other diode styles can always be forced with e.g. `Do`, `D-`, …
- `strokediode`: the various diodes are drawn *and* stroke by default, i.e. when using styles such as `diode`, `D`, `sD`, …Other diode styles can always be forced with e.g. `Do`, `D*`, …
- `emptydiode`: the various diodes are drawn *but not* filled by default, i.e. when using styles such as `D`, `sD`, …Other diode styles can always be forced with e.g. `Do`, `D-`, …
- `arrowmos`: pmos and nmos have arrows analogous to those of pnp and npn transistors;

- `noarrowmos`: pmos and nmos do not have arrows analogous to those of pnp and npn transistors;
- `fetbodydiode`: draw the body diode of a FET;
- `nofetbodydiode`: do not draw the body diode of a FET;
- `fetsolderdot`: draw solderdot at bulk-source junction of some transistors;
- `nofetsolderdot`: do not draw solderdot at bulk-source junction of some transistors;
- `emptypmoscircle`: the circle at the gate of a pmos transistor gets not filled;
- `lazymos`: draws lazy nmos and pmos transistors. Chip designers with huge circuits prefer this notation;
- `straightlabels`: labels on bipoles are always printed straight up, i.e. with horizontal baseline;
- `rotatelabels`: labels on bipoles are always printed aligned along the bipole;
- `smartlabels`: labels on bipoles are rotated along the bipoles, unless the rotation is very close to multiples of 90°;
- `compatibility`: makes it possibile to load CircuiTi*k*Z and Ti*k*Z circuit library together.
- Voltage directions: until v0.8.3, there was an error in the coherence between american and european voltages styles (see section 7.2 for the batteries. This has been fixed, but to guarantee backward compatibility and nasty surprises, the fix is available with new options:
  - `oldvoltagedirection`: Use old way of voltage direction having a difference between european and american direction, with wrong default labelling for batteries;
  - `nooldvoltagedirection`: The standard from 0.5 onward, utilize the (German?) standard of voltage arrows in the direction of electric fields (without fixing batteries);
  - `RPvoltages` (meaning Rising Potential voltages): the arrow is in direction of rising potential, like in `oldvoltagedirections`, but batteries and current sources are fixed to follow the passive/active standard;
  - `EFvoltages` (meaning Electric Field voltages): the arrow is in direction of the electric field, like in `nooldvoltagedirections`, but batteries are fixed;

  If none of these option are given, the package will default to `nooldvoltagedirections`, but will give a warning. The behavior is also selectable circuit by circuit with the `voltage dir` style.
- `betterproportions`[2]: nicer proportions of transistors in comparision to resistors;

The old options in the singular (like `american voltage`) are still available for compatibility, but are discouraged.

Loading the package with no options is equivalent to my own personal liking, that is to the following options:
`[nofetsolderdot,nooldvoltagedirection,europeancurrents,europeanvoltages,americanports,americanres`

In ConTEXt the options are similarly specified: `current=european|american`, `voltage=european|american`, `resistor=american|european`, `inductor=cute|american|european`, `logic=american|european`, `siunitx=true|false`, `arrowmos=false|true`.

[2]May change in the future!

4 The components

Here follows the list of all the shapes defined by CircuiTi*k*Z. These are all `pgf` nodes, so they are usable in both `pgf` and Ti*k*Z.

Drawing normal components

Normal components (monopoles, multipoles) can be drawn at a specified point with this syntax, where `#1` is the name of the component:

```
\begin{center}\begin{circuitikz} \draw
   (0,0) node[#1,#2] (#3) {#4}
; \end{circuitikz} \end{center}
```

Explanation of the parameters:
`#1`: component name[3] (mandatory)
`#2`: list of comma separated options (optional)
`#3`: name of an anchor (optional)
`#4`: text written to the text anchor of the component (optional)

> **Note for Ti*k*Z newbies:** Nodes must have curly brackets at the end, even when empty. An optional anchor (`#3`) can be defined within round brackets to be addressed again later on. And please don't forget the semicolon to terminate the `\draw` command.

Drawing bipoles/two-ports

Bipoles/Two-ports (plus some special components) can be drawn between two points using the following command:

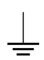
```
\begin{center}\begin{circuitikz} \draw
   (0,0) to[#1,#2] (2,0)
; \end{circuitikz} \end{center}
```

Explanation of the parameters:
`#1`: component name (mandatory)
`#2`: list of comma separated options (optional)
Transistors and some other components can also be placed using the syntax for bipoles. See section 8.6.

> If using the `\tikzexternalize` feature, as of Ti*k*z 2.1 all pictures must end with `\end{tikzpicture}`. Thus you *cannot* use the `circuitikz` environment.
> Which is ok: just use the environment `tikzpicture`: everything will work there just fine.

4.1 Monopoles

- Ground (node[`ground`]{})



- Reference ground (node[`rground`]{})

[3]For using bipoles as nodes, the name of the node is `#1shape`.

- Signal ground (node[sground]{})

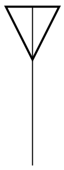
- Thicker ground (node[tground]{})

- Noiseless ground (node[nground]{})

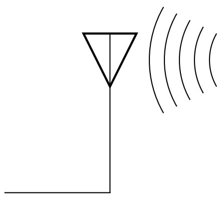
- Protective ground (node[pground]{})

- Chassis ground[4] (node[cground]{})

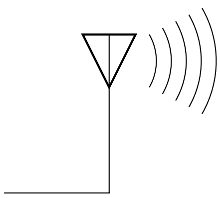
- Antenna (node[antenna]{})



- Receiving antenna (node[rxantenna]{})

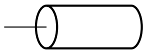


- Transmitting antenna (node[txantenna]{})



[4]These last three were contributed by Luigi «Liverpool»)

- Transmission line stub (node[`tlinestub`]{})



- VCC/VDD (node[`vcc`]{})

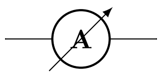
- VEE/VSS (node[`vee`]{})

- match (node[`match`]{})

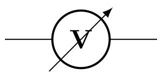
4.2 Bipoles

4.2.1 Instruments

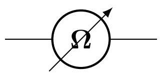
- Ammeter (`ammeter`)



- Voltmeter (`voltmeter`)



- Ohmmeter (`ohmmeter`)

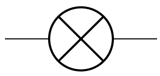


4.2.2 Basic resistive bipoles

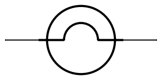
- Short circuit (`short`)

- Open circuit (`open`)

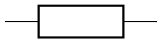
- Lamp (`lamp`)



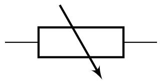
- Bulb (`bulb`)



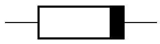
- Generic (symmetric) bipole (`generic`)



- Tunable generic bipole (`tgeneric`)

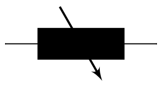


- Generic asymmetric bipole (`ageneric`)

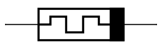


- Generic asymmetric bipole (full) (`fullgeneric`)

- Tunable generic bipole (full) (`tfullgeneric`)



- Memristor (`memristor`, or `Mr`)



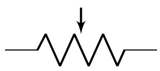
4.2.3 Resistors and the like

If (default behaviour) `americanresistors` option is active (or the style `[american resistors]` is used), the resistor is displayed as follows:

- Resistor (`R`, or `american resistor`)

- Variable resistor (`vR`, or `variable american resistor`)

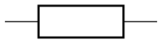
- Potentiometer (`pR`, or `american potentiometer`)



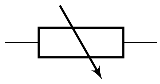
- Resisitive sensor (`sR`, or `american resisitive sensor`)

If instead `europeanresistors` option is active (or the style `[european resistors]` is used), the resistors, variable resistors and potentiometers are displayed as follows:

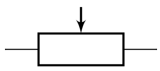
- Resistor (`R`, or `european resistor`)



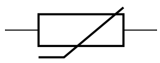
- Variable resistor (`vR`, or `european variable resistor`)



- Potentiometer (`pR`, or `european potentiometer`)

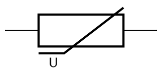


- Resisitive sensor (`sR`, or `european resisitive sensor`)



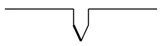
Other miscellaneous resistor-like devices:

- Varistor (`varistor`)

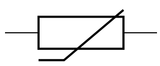


- Photoresistor (`phR`, or `photoresistor`)

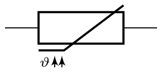
- Thermocouple (`thermocouple`)



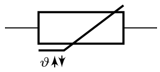
- Thermistor (`thR`, or `thermistor`)



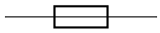
- PTC thermistor (`thRp`, or `thermistor ptc`)



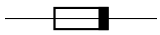
- NTC thermistor (`thRn`, or `thermistor ntc`)



- Fuse (`fuse`)

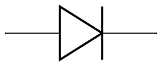


- Asymmetric fuse (`afuse`, or `asymmetric fuse`)

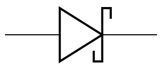


4.2.4 Diodes and such

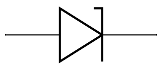
- Empty diode (`empty diode`, or `Do`)



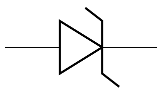
- Empty Schottky diode (`empty Schottky diode`, or `sDo`)



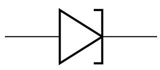
- Empty Zener diode (`empty Zener diode`, or `zDo`)



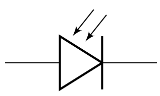
- Empty ZZener diode (`empty ZZener diode`, or `zzDo`)



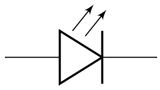
- Empty tunnel diode (`empty tunnel diode`, or `tDo`)



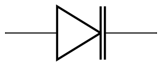
- Empty photodiode (`empty photodiode`, or `pDo`)



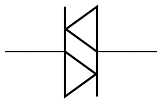
- Empty led (`empty led`, or `leDo`)



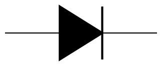
- Empty varcap (`empty varcap`, or `VCo`)



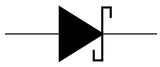
- Empty bidirectionaldiode (`empty bidirectionaldiode`, or `biDo`)



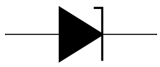
- Full diode (`full diode`, or `D*`)



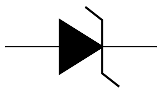
- Full Schottky diode (`full Schottky diode`, or `sD*`)



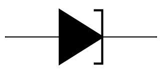
- Full Zener diode (`full Zener diode`, or `zD*`)



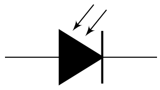
- Full ZZener diode (`full ZZener diode`, or `zzD*`)



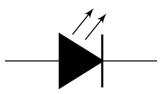
- Full tunnel diode (`full tunnel diode`, or `tD*`)



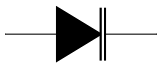
- Full photodiode (`full photodiode`, or `pD*`)



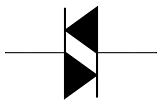
- Full led (`full led`, or `leD*`)



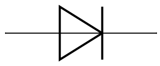
- Full varcap (`full varcap`, or `VC*`)



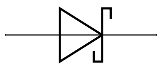
- Full bidirectionaldiode (`full bidirectionaldiode`, or `biD*`)



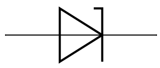
- Stroke diode (`stroke diode`, or `D-`)



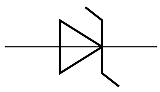
- Stroke Schottky diode (`stroke Schottky diode`, or `sD-`)



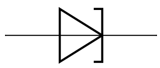
- Stroke Zener diode (`stroke Zener diode`, or `zD-`)



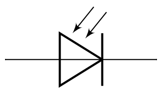
- Stroke ZZener diode (`stroke ZZener diode`, or `zzD-`)



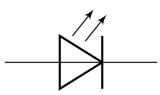
- Stroke tunnel diode (`stroke tunnel diode`, or `tD-`)



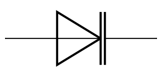
- Stroke photodiode (`stroke photodiode`, or `pD-`)



- Stroke led (`stroke led`, or `leD-`)



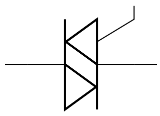
- Stroke varcap (`stroke varcap`, or `VC-`)



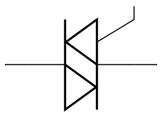
4.2.5 Other tripole-like diodes

The following tripoles are entered with the usual command of the form

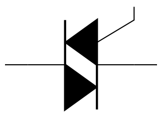
- Standard triac (shape depends on package option) (`triac`, or `Tr`)



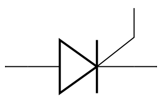
- Empty triac (`empty triac`, or `Tro`)



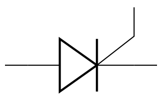
- Full triac (`full triac`, or `Tr*`)



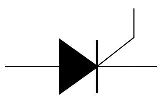
- Standard thyristor (shape depends on package option) (`thyristor`, or `Ty`)



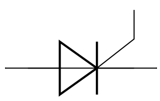
- Empty thyristor (`empty thyristor`, or `Tyo`)



- Full thyristor (`full thyristor`, or `Ty*`)



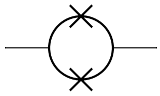
- Stroke thyristor (`stroke thyristor`, or `Ty-`)



See chapter 8.1.3 for information how access the third connector

> The package options `fulldiode`, `strokediode`, and `emptydiode` (and the styles `[full diodes]`, `[stroke diodes]`, and `[empty diodes]`) define which shape will be used by abbreviated commands such that `D`, `sD`, `zD`, `zzD`, `tD`, `pD`, `leD`, `VC`, `Ty`,`Tr`(no stroke symbol available!).

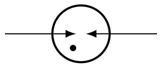
- Squid (`squid`)



- Barrier (`barrier`)

- European gas filled surge arrester (`european gas filled surge arrester`)

- American gas filled surge arrester (`american gas filled surge arrester`)

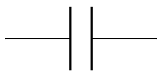


If (default behaviour) `europeangfsurgearrester` option is active (or the style `[european gas filled surge arrester]` is used), the shorthands `gas filled surge arrester` and `gf surge arrester` are equivalent to the european version of the component.

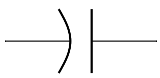
If otherwise `americangfsurgearrester` option is active (or the style `[american gas filled surge arrester]` is used), the shorthands the shorthands `gas filled surge arrester` and `gf surge arrester` are equivalent to the american version of the component.

4.2.6 Basic dynamical bipoles

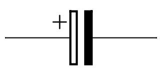
- Capacitor (`capacitor`, or `C`)



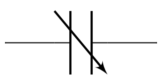
- Polar capacitor (`polar capacitor`, or `pC`)



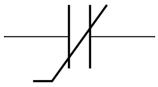
- Electrolytic capacitor (`ecapacitor`, or `eC,elko`)



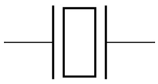
- Variable capacitor (`variable capacitor`, or `vC`)



- Capacitive sensor (`capacitive sensor`, or `sC`)



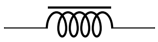
- Piezoelectric Element (`piezoelectric`, or `PZ`)



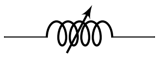
If (default behaviour) `cuteinductors` option is active (or the style `[cute inductors]` is used), the inductors are displayed as follows:

- Inductor (`L`, or `cute inductor`)

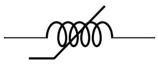
- Choke (`cute choke`)



- Variable inductor (`vL`, or `variable cute inductor`)



- Inductive sensor (`sL`, or `cute inductive sensor`)



If `americaninductors` option is active (or the style `[american inductors]` is used), the inductors are displayed as follows:

- Inductor (`L`, or `american inductor`)

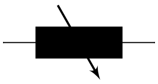
- Variable inductor (`vL`, or `variable american inductor`)

- Inductive sensor (`sL`, or `american inductive sensor`)

Finally, if `europeaninductors` option is active (or the style `[european inductors]` is used), the inductors are displayed as follows:

- Inductor (`L`, or `european inductor`)

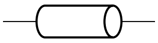
- Variable inductor (vL, or `variable european inductor`)



- Inductive sensor (sL, or `european inductive sensor`)

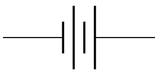
There is also a transmission line:

- Transmission line (TL, or `transmission line, tline`)

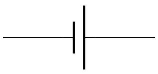


4.2.7 Stationary sources

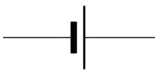
- Battery (`battery`)



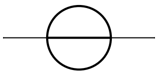
- Single battery cell (`battery1`)



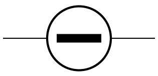
- Single battery cell (`battery2`)



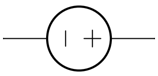
- Voltage source (european style) (`european voltage source`)



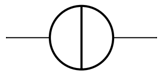
- Voltage source (cute european style) (`cute european voltage source`, or `vsourceC, ceV`)



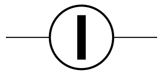
- Voltage source (american style) (`american voltage source`)



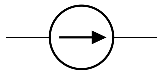
- Current source (european style) (`european current source`)



- Current source (cute european style) (`cute european current source`, or `isourceC, ceI`)



- Current source (american style) (`american current source`)



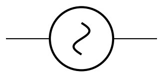
If (default behaviour) `europeancurrents` option is active (or the style `[european currents]` is used), the shorthands `current source`, `isource`, and `I` are equivalent to `european current source`. Otherwise, if `americancurrents` option is active (or the style `[american currents]` is used) they are equivalent to `american current source`.

Similarly, if (default behaviour) `europeanvoltages` option is active (or the style `[european voltages]` is used), the shorthands `voltage source`, `vsource`, and `V` are equivalent to `european voltage source`. Otherwise, if `americanvoltages` option is active (or the style `[american voltages]` is used) they are equivalent to `american voltage source`.

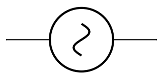
4.2.8 Sinusoidal sources

Here because I was asked for them. But how do you distinguish one from the other?!

- Sinusoidal voltage source (`sinusoidal voltage source`, or `vsourcesin, sV`)



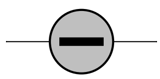
- Sinusoidal current source (`sinusoidal current source`, or `isourcesin, sI`)



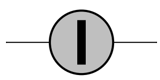
4.2.9 Noise sources

In this case, the "direction" of the source has no sense.

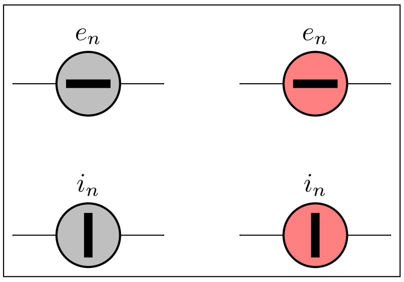
- Sinusoidal voltage source (`noise voltage source`, or `vsourceN, nV`)



- Sinusoidal current source (`noise current source`, or `isourceN, nI`)

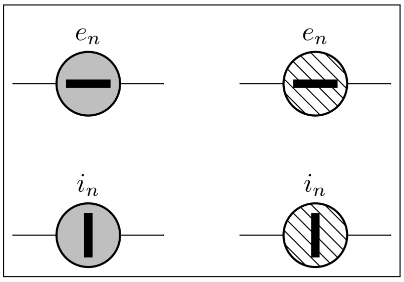


You can change the fill color with the key `circuitikz/bipoles/noise sources/fillcolor`:




```
\begin{circuitikz}
  \draw(0,0) to [nV, l=$e_n$] ++(2,0);
  \draw(0,-2) to [nI, l=$i_n$] ++(2,0);
  \begin{scope}[circuitikz/bipoles/noise sources/fillcolor=red!50]
    \draw(3,0) to [nV, l=$e_n$] ++(2,0);
    \draw(3,-2) to [nI, l=$i_n$] ++(2,0);
  \end{scope}
\end{circuitikz}
```

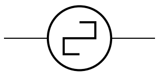
If you prefer a patterned noise generator (similar to the one you draw by hand) you can use the fake color `dashed`:



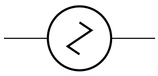

```
\begin{circuitikz}
  \draw(0,0) to [nV, l=$e_n$] ++(2,0);
  \draw(0,-2) to [nI, l=$i_n$] ++(2,0);
  \begin{scope}[circuitikz/bipoles/noise sources/fillcolor=dashed]
    \draw(3,0) to [nV, l=$e_n$] ++(2,0);
    \draw(3,-2) to [nI, l=$i_n$] ++(2,0);
  \end{scope}
\end{circuitikz}
```

4.2.10 Special sources

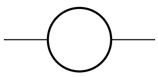
- Square voltage source (`square voltage source`, or `vsourcesquare, sqV`)



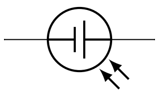
- Triangle voltage source (`vsourcetri`, or `tV`)



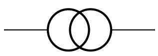
- Empty voltage source (`esource`)



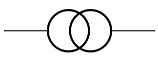
- Photovoltaic-voltage source (`pvsource`)



- Double Zero style current source (`ioosource`)

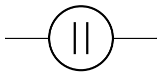


- Double Zero style voltage source (`voosource`)

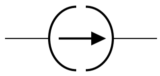


4.2.11 DC sources

- DC voltage source (`dcvsource`)

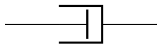


- DC current source (`dcisource`)



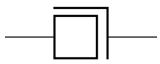
4.2.12 Mechanical Analogy

- Mechanical Damping (`damper`)



- Mechanical Stiffness (`spring`)

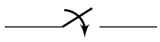
- Mechanical Mass (`mass`)



4.2.13 Switch

- Switch (`switch`, or `spst`)

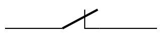
- Closing switch (`closing switch`, or `cspst`)



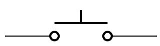
- Opening switch (`opening switch`, or `ospst`)

- Normally open switch (`normal open switch`, or `nos`)

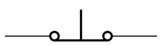
- Normally closed switch (`normal closed switch`, or `ncs`)



- Normally open push button (`push button`, or `normally open push button, nopb`)



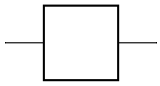
- Normally closed push button (`normally closed push button`, or `ncpb`)



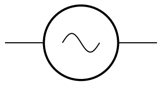
4.2.14 Block diagram components

Contributed by Stefan Erhardt.

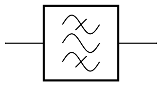
- generic two port[5] (`twoport`)



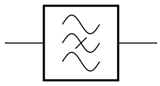
- vco (`vco`)



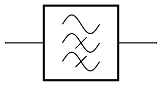
- bandpass (`bandpass`)



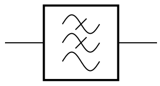
- bandstop (`bandstop`)



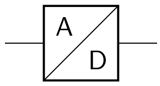
- highpass (`highpass`)



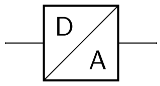
- lowpass (`lowpass`)



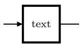
- A/D converter (`adc`)

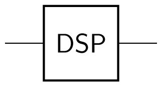


- D/A converter (`dac`)

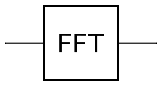


- DSP (`dsp`)

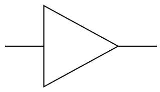
[5] To specify text to be put in the component: `twoport[t=text]`): 



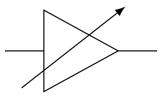
- FFT (`fft`)



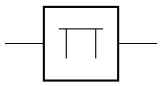
- amplifier (`amp`)



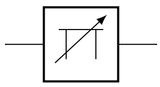
- VGA (`vamp`)



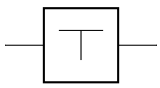
- π attenuator (`piattenuator`)



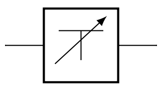
- var. π attenuator (`vpiattenuator`)



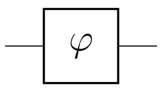
- T attenuator (`tattenuator`)



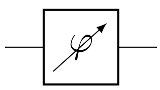
- var. T attenuator (`vtattenuator`)



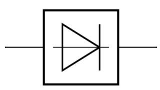
- phase shifter (`phaseshifter`)



- var. phase shifter (`vphaseshifter`)



- detector (`detector`)

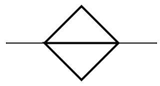


4.3 Tripoles

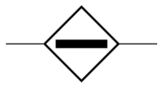
4.3.1 Controlled sources

Admittedly, graphically they are bipoles. But I couldn't…

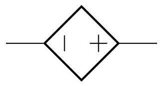
- Controlled voltage source (european style) (`european controlled voltage source`)



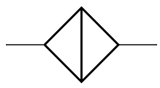
- Voltage source (cute european style) (`cute european controlled voltage source`, or `cvsourceC, cceV`)



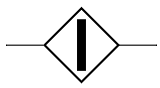
- Controlled voltage source (american style) (`american controlled voltage source`)



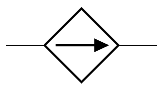
- Controlled current source (european style) (`european controlled current source`)



- Current source (cute european style) (`cute european controlled current source`, or `cisourceC, cceI`)



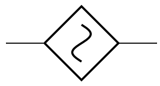
- Controlled current source (american style) (`american controlled current source`)



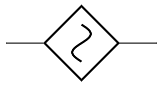
If (default behaviour) `europeancurrents` option is active (or the style `[european currents]` is used), the shorthands `controlled current source`, `cisource`, and `cI` are equivalent to `european controlled current source`. Otherwise, if `americancurrents` option is active (or the style `[american currents]` is used) they are equivalent to `american controlled current source`.

Similarly, if (default behaviour) `europeanvoltages` option is active (or the style `[european voltages]` is used), the shorthands `controlled voltage source`, `cvsource`, and `cV` are equivalent to `european controlled voltage source`. Otherwise, if `americanvoltages` option is active (or the style `[american voltages]` is used) they are equivalent to `american controlled voltage source`.

- Controlled sinusoidal voltage source (`controlled sinusoidal voltage source`, or `controlled vsourcesin, cvsourcesin, csV`)

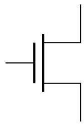


- Controlled sinusoidal current source (`controlled sinusoidal current source`, or `controlled isourcesin, cisourcesin, csI`)

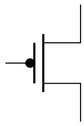


4.3.2 Transistors

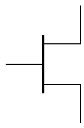
- NMOS (node[`nmos`]{})



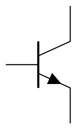
- PMOS (node[`pmos`]{})



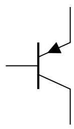
- HEMT (node[`hemt`]{})



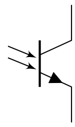
- NPN (node[`npn`]{})



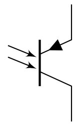
- PNP (node[`pnp`]{})



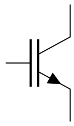
- NPN (node[`npn,photo`]{})



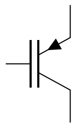
- PNP (node[**pnp,photo**]{})



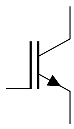
- NIGBT (node[**nigbt**]{})



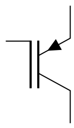
- PIGBT (node[**pigbt**]{})



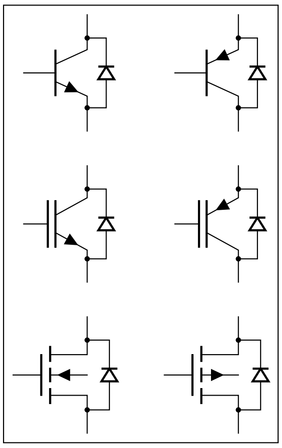
- LNIGBT (node[**Lnigbt**]{})



- LPIGBT (node[**Lpigbt**]{})

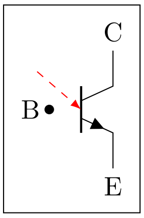


For all transistors a bodydiode(or freewheeling diode) can automatically be drawn. Just use the global option bodydiode, or for single transistors, the tikz-option bodydiode:



```
\begin{circuitikz}
  \draw (0,0) node[npn,bodydiode](npn){}++(2,0)node[pnp,
      bodydiode](npn){};
  \draw (0,-2) node[nigbt,bodydiode](npn){}++(2,0)node[
      pigbt,bodydiode](npn){};
  \draw (0,-4) node[nfet,bodydiode](npn){}++(2,0)node[
      pfet,bodydiode](npn){};
\end{circuitikz}
```

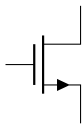
The Base/Gate connection of all transistors can be disable by using the options *nogate* or *nobase*, respectively. The Base/Gate anchors are floating, but there an additional anchor "nogate"/"nobase", which can be used to point to the unconnected base:



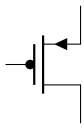

```
\begin{circuitikz}
  \draw (2,0) node[npn,nobase](npn){};
  \draw (npn.E) node[below]{E};
  \draw (npn.C) node[above]{C};
  \draw (npn.B) node[circ]{} node[left]{B};
  \draw[dashed,red,-latex] (1,0.5)--(npn.nobase);
\end{circuitikz}
```

If the option `arrowmos` is used (or after the command `\ctikzset{tripoles/mos style/arrows}` is given), this is the output:

- NMOS (node[`nmos`]{})

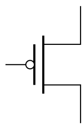


- PMOS (node[`pmos`]{})

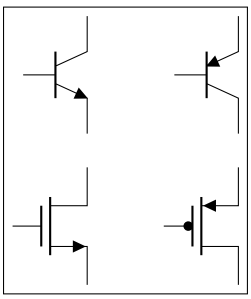


To draw the PMOS circle non-solid, use the option `emptycircle` or the command `\ctikzset{tripoles/pmos styl`

- PMOS (node[`pmos,emptycircle`]{})



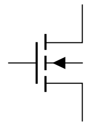
If you prefer different position of the arrows in transistors and FETs, you can adjust them like this (it works for the other BJT-based transistors, too):



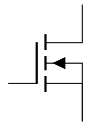
```
\begin{circuitikz}
   \ctikzset{tripoles/mos style/arrows,
   tripoles/npn/arrow pos=0.8,
   tripoles/pnp/arrow pos=0.8,
   tripoles/nmos/arrow pos=0.8,
   tripoles/pmos/arrow pos=0.6, }
  \draw (0,0) node[npn, ](npn){};
  \draw (2,0) node[pnp, ](npn){};
  \draw (0,-2) node[nmos, ](npn){};
  \draw (2,-2) node[pmos, ](npn){};
\end{circuitikz}
```

NFETs and PFETs have been incorporated based on code provided by Clemens Helfmeier and Theodor Borsche. Use the package options `fetsolderdot/nofetsolderdot` to enable/disable solderdot at some fet-transistors. Additionally, the solderdot option can be enabled/disabled for single transistors with the option "solderdot" and "nosolderdot", respectively.

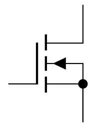
- NFET (node[`nfet`]{})



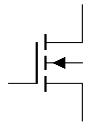
- NIGFETE (node[`nigfete`]{})



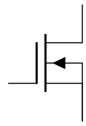
- NIGFETE (node[`nigfete,solderdot`]{})



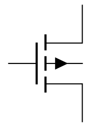
- NIGFETEBULK (node[`nigfetebulk`]{})



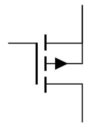
- NIGFETD (node[`nigfetd`]{})



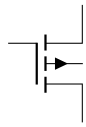
- PFET (node[`pfet`]{})



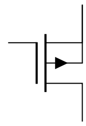
- PIGFETE (node[`pigfete`]{})



- PIGFETEBULK (node[`pigfetebulk`]{})

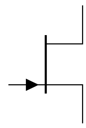


- PIGFETD (node[`pigfetd`]{})

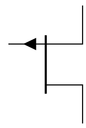


NJFET and PJFET have been incorporated based on code provided by Danilo Piazzalunga:

- NJFET (node[`njfet`]{})

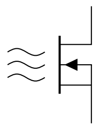


- PJFET (node[`pjfet`]{})



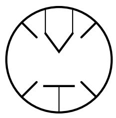
ISFET

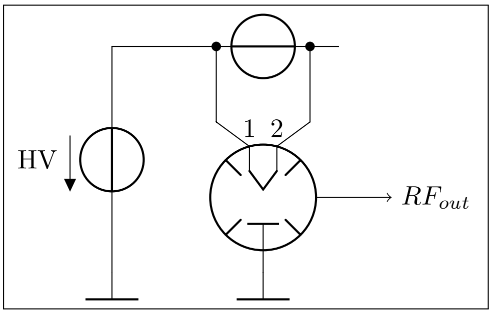
- ISFET (node[`isfet`]{})



4.3.3 Electronic Tubes

- Magnetron (node[`magnetron`]{})



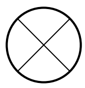



```
\begin{circuitikz}
\draw (0,-2)node[rground](gnd){} to[voltage source,v<={HV}]++(0,3)--++(1,0)to[V,n=DC]++(2,0);
\draw (2,-1) node[magnetron,scale=1](magn){};
\draw (DC.left)++(-0.2,0)to [short,*-] ++(0,-1) to [short] (magn.cathode1);
\draw (DC.right)++(0.2,0)to [short,*-] ++(0,-1) to [short] (magn.cathode2);
\draw (magn.anode) to [short] (magn.anode|-gnd) node[rground]{};
\draw (magn.cathode1)node[above]{$1$};
\draw (magn.cathode2)node[above]{$2$};
\draw[->](magn.east) --++(1,0)node[right]{$RF_{out}$};
\end{circuitikz}
```

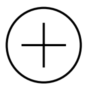
4.3.4 Block diagram

These come from Stefan Erhardt's contribution of block diagram components. Add a box around them with the option `box`.

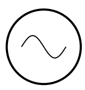
- MIXER (node[`mixer`]{})



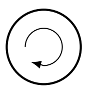
- ADDER (node[`adder`]{})



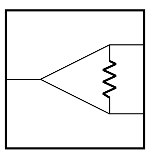
- OSCILLATOR (node[`oscillator`]{})



- CIRCULATOR (node[`circulator`]{})

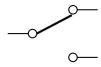


- WILKINSON DIVIDER (node[`wilkinson`]{})

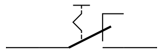


4.3.5 Switch

- SPDT (node[**spdt**]{})



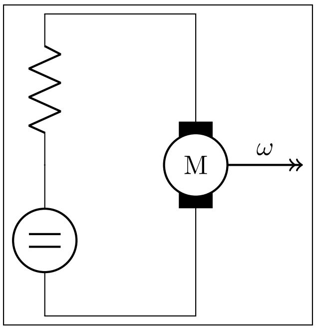
- Toggle switch (`toggle switch`)



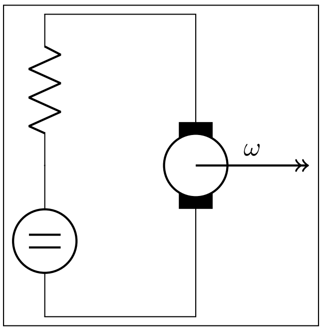
4.3.6 Electro-Mechanical Devices

- MOTOR (node[**elmech**]{M})

- GENERATOR (node[**elmech**]{G})

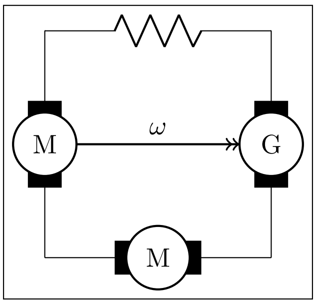



```
\begin{circuitikz}
\draw (2,0) node[elmech](motor){M};
\draw (motor.north) |-(0,2) to [R] ++(0,-2) to[
    dcvsource]++(0,-2) -| (motor.bottom);
\draw[thick,->>](motor.right)--++(1,0)node[midway,
    above]{$\omega$};
\end{circuitikz}
```




```
\begin{circuitikz}
\draw (2,0) node[elmech](motor){};
\draw (motor.north) |-(0,2) to [R] ++(0,-2) to[
    dcvsource]++(0,-2) -| (motor.bottom);
\draw[thick,->>](motor.center)--++(1.5,0)node[midway,
    above]{$\omega$};
\end{circuitikz}
```

The symbols can also be used along a path, using the transistor-path-syntax(T in front of the shape name, see section 8.6). Don´t forget to use parameter n to name the node and get acces to the anchors:

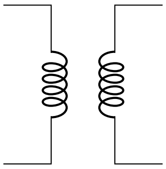



```
\begin{circuitikz}
\draw (0,0) to [Telmech=M,n=motor] ++(0,-3) to [Telmech=M] ++(3,0) to [Telmech=G,n=generator] ++(0,3) to [R] (0,0);
\draw[thick,->>](motor.left)--(generator.left)node[midway,above]{$\omega$};
\end{circuitikz}
```

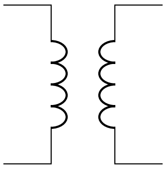
4.4 Double bipoles

Transformers automatically use the inductor shape currently selected. These are the three possibilities:

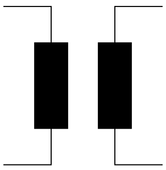
- Transformer (cute inductor) (node[`transformer`]{})



- Transformer (american inductor) (node[`transformer`]{})

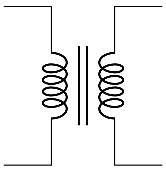


- Transformer (european inductor) (node[`transformer`]{})

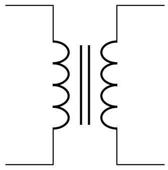


Transformers with core are also available:

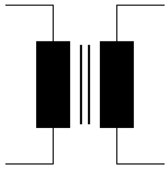
- Transformer core (cute inductor) (node[`transformer core`]{})



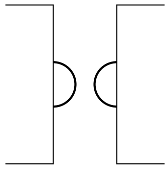
- Transformer core (american inductor) (node[`transformer core`]{})



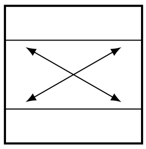
- Transformer core (european inductor) (node[`transformer core`]{})



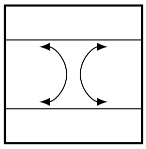
- Gyrator (node[`gyrator`]{})



- Coupler (node[`coupler`]{})

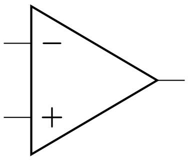


- Coupler, 2 (node[`coupler2`]{})

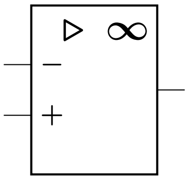


4.5 Amplifiers

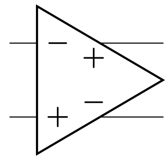
- Operational amplifier (node[`op amp`]{})



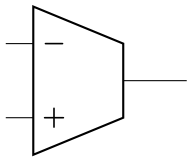
- Operational amplifier compliant to DIN/EN 60617 standard (node[`en amp`]{})



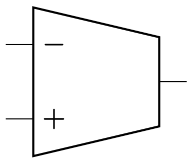
- Fully differential operational amplifier[6] (node[`fd op amp`]{})



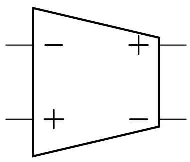
- transconductance amplifier (node[`gm amp`]{})



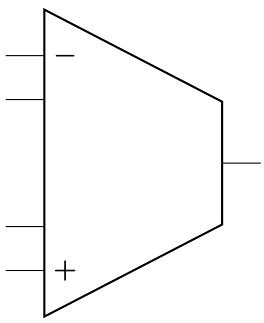
- plain instrumentation amplifier (node[`inst amp`]{})



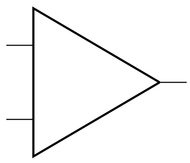
- Fully differential instrumentation amplifier (node[`fd inst amp`]{})



- instrumentation amplifier with amplification resistance terminals (node[`inst amp ra`]{})

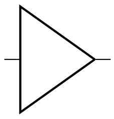


- Plain amplifier (node[`plain amp`]{})



[6]Contributed by Kristofer M. Monisit.

- Buffer (node[`buffer`]{})



4.6 Support shapes

- Arrows (current and voltage) (node[`currarrow`]{})

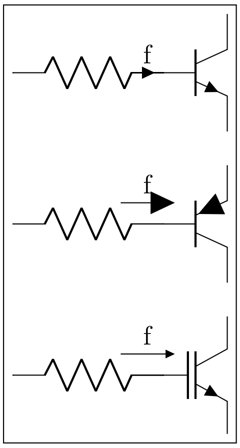
- Arrow to draw at its tip, useful for block diagrams. (node[`inputarrow`]{})

- Connected terminal (node[`circ`]{})

- Unconnected terminal (node[`ocirc`]{})

- Diamond-style terminal (node[`diamondpole`]{})

You can use the parameter `current arrow scale` to change the size of the arrows in various components and indicators; the normal value is 16, higher numbers give smaller arrows and so on. You need to use `circuitikz/current arrow scale` if you use it into a node.



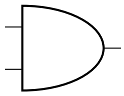

```
\begin{circuitikz}
   \draw (0,0) to[R, i=f] ++(2,0) node[npn, anchor=B]{};
   \draw (0,-2) to[R, f=f, current arrow scale=8] ++(2,0)
          node[pnp, anchor=B, circuitikz/current arrow scale
              =8]{};
   \draw (0,-4) to[R, f=f, current arrow scale=24] ++(2,0)
          node[nigbt, anchor=B]{};
\end{circuitikz}
```

5 Logic gates

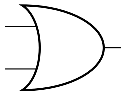
Logic gates, with two or more input, are supported. Albeit in principle these components are multipoles, the are considered tripoles here, for historical reasons (when they just had two inputs).

5.0.1 American Logic gates

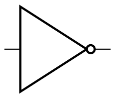
- American AND port (node[`american and port`]{})



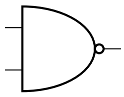
- American OR port (node[`american or port`]{})



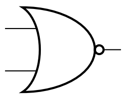
- American NOT port (node[`american not port`]{})



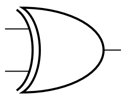
- American NAND port (node[`american nand port`]{})



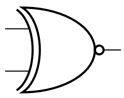
- American NOR port (node[`american nor port`]{})



- American XOR port (node[`american xor port`]{})

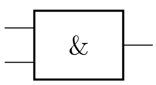


- American XNOR port (node[`american xnor port`]{})

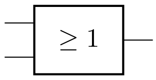


5.0.2 European Logic gates

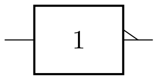
- European AND port (node[`european and port`]{})



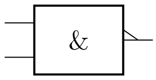
- European OR port (node[`european or port`]{})



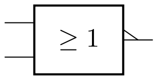
- European NOT port (node[`european not port`]{})



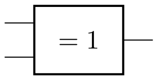
- European NAND port (node[`european nand port`]{})



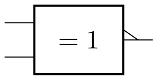
- European NOR port (node[`european nor port`]{})



- European XOR port (node[`european xor port`]{})



- European XNOR port (node[`european xnor port`]{})



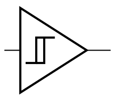
If (default behaviour) `americanports` option is active (or the style `[american ports]` is used), the shorthands `and port`, `or port`, `not port`, `nand port`, `not port`, `xor port`, and `xnor port` are equivalent to the american version of the respective logic port.

If otherwise `europeanports` option is active (or the style `[european ports]` is used), the shorthands `and port`, `or port`, `not port`, `nand port`, `not port`, `xor port`, and `xnor port` are equivalent to the european version of the respective logic port.

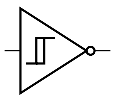
5.0.3 special components

There is no european version of these symbols.

- Non-Inverting SCHMITTTRIGGER (node[`schmitt`]{})

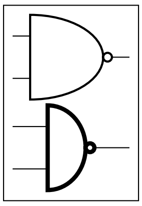


- Inverting SCHMITTTRIGGER (node[`invschmitt`]{})



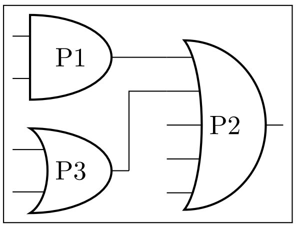
5.1 Logic port customization

As for most components, you can change the width and height of the ports; the thickness is given by the parameter `tripoles/thickness` (default 2):



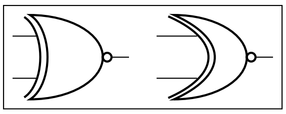
```
\tikz \draw (0,0) node[nand port] {}; \par
\ctikzset{tripoles/american nand port/input height=.2}
\ctikzset{tripoles/american nand port/port width=.4}
\ctikzset{tripoles/thickness=4}
\tikz \draw (0,0) node[nand port] {};
```

This is especially useful if you have ports with more than two inputs, which are instantiated with the parameter `number inputs` :




```
\begin{circuitikz}
\draw (0,3) node[american and port] (A) {P1};
\begin{scope}
    \ctikzset{tripoles/american or port/height=1.6}
    \draw (A.out) -- ++(0.5,0)
    node[american or port,
    number inputs=5,
    anchor=in 1] (B) {P2};
\end{scope}
\draw (0,1.5) node[american or port] (C) {P3};
\draw (C.out) |- (B.in 2);
\end{circuitikz}
```

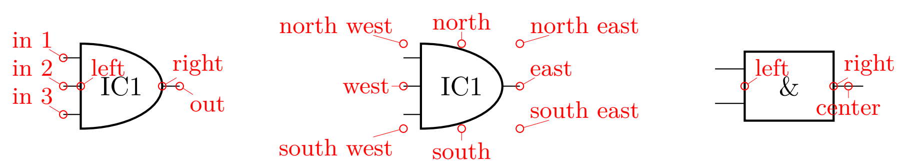
You can tweak the appearance of american “or” family (`or`, `nor`, `xor` and `xnor`) ports, too, with the parameters `inner` (how much the base circle go “into” the shape, default 0.3) and `angle` (the angle at which the base starts, default 70).



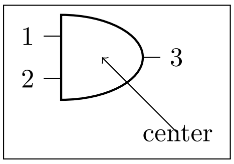
```
\tikz \draw (0,0) node[xnor port] {};
\ctikzset{tripoles/american xnor port/inner=.7}
\ctikzset{tripoles/american xnor port/angle=40}
\tikz \draw (0,0) node[xnor port] {};
```

5.2 Logic port anchors

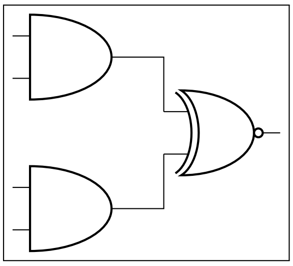
These are the anchors for logic ports:



As you can see, the `center` anchor is (for historic reasons) not in the center at all. You can fix this with the command `\ctikzset{logic ports origin=center}`:

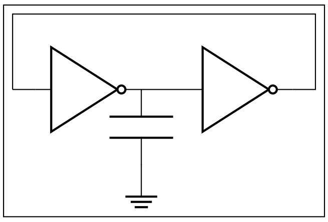



```
\begin{circuitikz}
\ctikzset{logic ports origin=center}
\draw (0,0) node[and port] (myand) {}
  (myand.in 1) node[anchor=east] {1}
  (myand.in 2) node[anchor=east] {2}
  (myand.out) node[anchor=west] {3};
\draw[<-] (myand.center) -- ++(1,-1)
   node{center};
\end{circuitikz}
```



```
\begin{circuitikz} \draw
  (0,2) node[and port] (myand1) {}
  (0,0) node[and port] (myand2) {}
  (2,1) node[xnor port] (myxnor) {}
  (myand1.out) -| (myxnor.in 1)
  (myand2.out) -| (myxnor.in 2)
;\end{circuitikz}
```

In the case of NOT, there are only `in` and `out` (although for compatibility reasons `in 1` is still defined and equal to `in`):

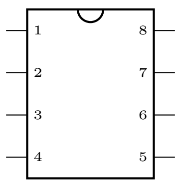


```
\begin{circuitikz} \draw
  (1,0) node[not port] (not1) {}
  (3,0) node[not port] (not2) {}
  (0,0) -- (not1.in)
  (not2.in) -- (not1.out)
  ++(0,-1) node[ground] {} to[C] (not1.out)
  (not2.out) -| (4,1) -| (0,0)
;\end{circuitikz}
```

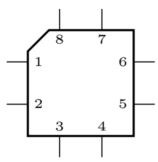
6 Chips

`CircuiTikZ` supports two types of variable-pin chips: DIP (Dual-in-Line Package) and QFP (Quad-Flat Package).

- Dual-in-Line Package chip (node[`dipchip`]{})



- Quad-Flat Package chip (node[`qfpchip`]{})



6.1 DIP and QFP chips customization

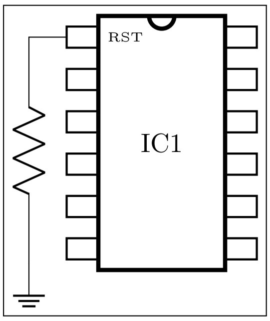
You can customize the DIP chip with the keys `multipoles/dipchip/width` (default 1.2) and `multipoles/dipchip/pin spacing` (default 0.4) that are expressed in fraction of basic lengths (see section 9.2). The height of the chip will be equal to half the numbers of pins multiplied by the spacing, plus one spacing for the borders. For the QFP chips, you can only chose the pin spacing with `multipoles/qfpchip/pin spacing` key.

The pins of the chip can be "hidden" (that is, just a spot in the border, optionally marked with a number) or "stick out" with a thin lead by setting `multipoles/external pins width` greater than 0 (default value is 0.2, so you'll have leads as shown above). Moreover, you can transform the thin lead into a pad by setting the key `multipoles/external pad fraction` to something different form 0 (default is 0); the value expresses the fraction of the pin spacing space that the pad will use on both sides of the pin.

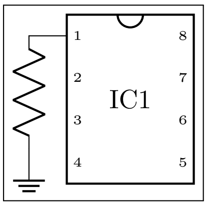
The number of pins is settable with the key `num pins`. **Please notice** that the number of pins **must** be *even* for `dipchips` and *multiple of 4* for `qfpchips`, otherwise havoc will ensue.

You can, if you want, avoid printing the numbers of the pin with `hide numbers` (default `show numbers`) if you prefer positioning them yourself (see the next section for the anchors you can use). The font used for the pins is adjustable with the key `multipoles/font` (default `\tiny`) For special use you can suppress the orientation mark with the key `no topmark` (default `topmark`).

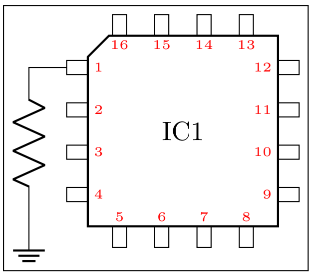
The line thickness of the main shape is controlled by `multipoles/thickness` (default 2) and the one of the external pins/pads with `multipoles/external pins thickness` (default 1).




```
\begin{circuitikz}
    \ctikzset{multipoles/thickness=4}
    \ctikzset{multipoles/external pins thickness=2}
    \draw (0,0) node[dipchip,
      num pins=12,
      hide numbers,
      external pins width=0.3,
      external pad fraction=4 ](C){IC1};
    \draw (C.pin 1) -- ++(-0.5,0) to[R]
      ++(0,-3) node[ground]{};
      \node [right, font=\tiny]
      at (C.bpin 1) {RST};
\end{circuitikz}
```




```
\begin{circuitikz}
    \draw (0,0) node[dipchip,
      num pins=8,
      external pins width=0.0](C){IC1};
    \draw (C.pin 1) -- ++(-0.5,0) to[R]
      ++(0,-1.5) node[ground]{};
\end{circuitikz}
```

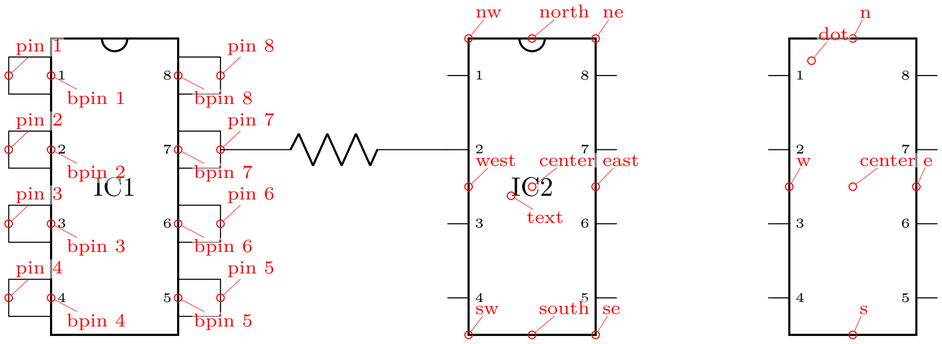



```
\begin{circuitikz}
    \ctikzset{multipoles/font={\color{red}\tiny}}
    \draw (0,0) node[qfpchip,
      num pins=16,
      external pad fraction=6](C){IC1};
    \draw (C.pin 1) -- ++(-0.5,0) to[R]
      ++(0,-2) node[ground]{};
\end{circuitikz}
```

6.2 Chip's anchors

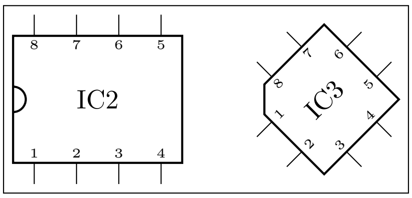
Chips have anchors on pins and global anchors for the main shape. The pin anchors to be used to connect wires to the chip are called `pin 1`, `pin 2` , …, with just one space between `pin` and the number. Border pin anchors (`bpin 1`…) are always on the box border, and can be used to add numbers or whatever markings are needed. Obviously, in case of `multipoles/external pins width` equal to cero, border and normal pin anchors will coincide.

Additionally, you have geometrical anchors on the chip "box", see the following figure. The nodes are available with the full name (like `north`) and with the short abbreviations `n`, `nw`, `w`…. The `dot` anchor is useful to add a personalized marker if you use the `no topmark` key.



6.3 Chips rotation

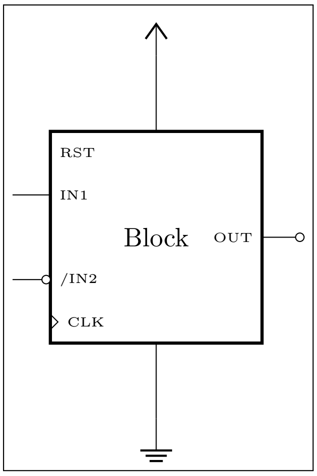
You can rotate chips, and normally the pin numbers are kept straight (option `straight numbers`, which is the default), but you can rotate them if you like with `rotated numbers`. Notice that the main label has to be (counter-)rotated manually in this case.




```
\begin{circuitikz}
    \draw (0,0) node[dipchip,
        rotate=90]{%
        \rotatebox{-90}{IC2}};
    \draw (3,0) node[qfpchip,
        rotated numbers,
        rotate=45]{IC3};
\end{circuitikz}
```

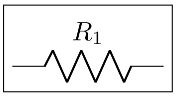
6.4 Chip special usage

You can use the chips to have special, personalized blocks. Look at the following example, which is easily put into a macro.

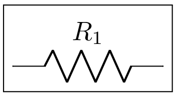



```
\begin{circuitikz}
  \ctikzset{multipoles/thickness=3}
  \ctikzset{multipoles/dipchip/width=2}
  \draw (0,0) node[dipchip,
     num pins=10, hide numbers, no topmark,
     external pins width=0](C){Block};
  \node [right, font=\tiny] at (C.bpin 1) {RST};
  \node [right, font=\tiny] at (C.bpin 2) {IN1};
  \node [right, font=\tiny] at (C.bpin 4) {/IN2};
  \node [left, font=\tiny] at (C.bpin 8) {OUT};
  \draw (C.bpin 2) -- ++(-0.5,0) coordinate(extpin);
  \node [ocirc, anchor=0](notin2) at (C.bpin 4) {};
  \draw (notin2.180) -- (C.bpin 4 -| extpin);
  \draw (C.bpin 8) to[short,-o] ++(0.5,0);
  \draw (C.bpin 5) ++(0,0.1) -- ++(0.1,-0.1)
     node[right, font=\tiny]{CLK} -- ++(-0.1,-0.1);
  \draw (C.n) -- ++(0,1) node[vcc]{};
  \draw (C.s) -- ++(0,-1) node[ground]{};
\end{circuitikz}
```

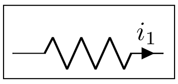
7 Usage



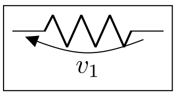

```
\begin{circuitikz}
  \draw (0,0) to[R, l=$R_1$] (2,0);
\end{circuitikz}
```



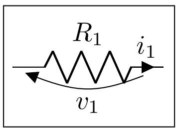

```
\begin{circuitikz}
  \draw (0,0) to[R=$R_1$] (2,0);
\end{circuitikz}
```



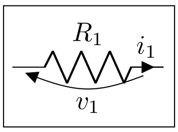

```
\begin{circuitikz}
  \draw (0,0) to[R, i=$i_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[R, v=$v_1$] (2,0);
\end{circuitikz}
```

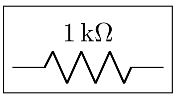



```
\begin{circuitikz}
  \draw (0,0) to[R=$R_1$, i=$i_1$, v=$v_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[R=$R_1$, i=$i_1$, v=$v_1$] (2,0);
\end{circuitikz}
```

Long names/styles for the bipoles can be used:

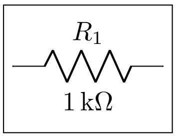



```
\begin{circuitikz}\draw
 (0,0) to[resistor=1<\kilo\ohm>] (2,0)
;\end{circuitikz}
```

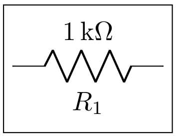
7.1 Labels and Annotations

Since Version 0.7, beside the original label (l) option, there is a new option to place a second label, called annotation (a) at each bipole. Up to now this is a beta-test and there can be problems. For example, up to now this option is not compatible with the concurrent use of voltage labels.

The position of (a) and (l) labels can be adjusted with _ and ^, respectively.

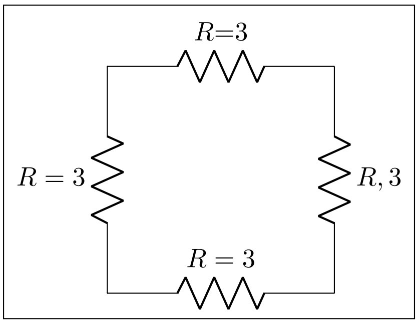



```
\begin{circuitikz}
  \draw (0,0) to[R, l=$R_1$,a=1<\kilo\ohm>] (2,0);
\end{circuitikz}
```



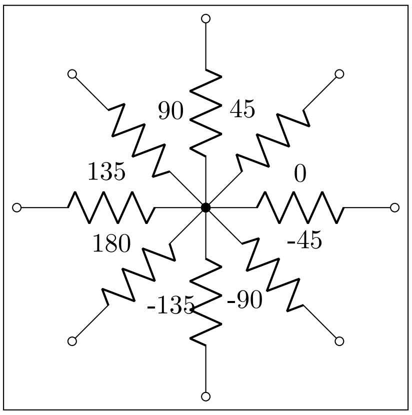

```
\begin{circuitikz}
  \draw (0,0) to[R, l_=$R_1$,a^=1<\kilo\ohm>] (2,0);
\end{circuitikz}
```

Caveat: notice that the way in which `circuitikz` processes the options, there will be problems if the label (or annotation, or voltage, or current) contains one of the characters = (equal) or , (comma), giving unexpected errors and wrong output. These two characters must be protected to the option parser using an `\mbox` command, or redefining the characters with a TEX `\def`:

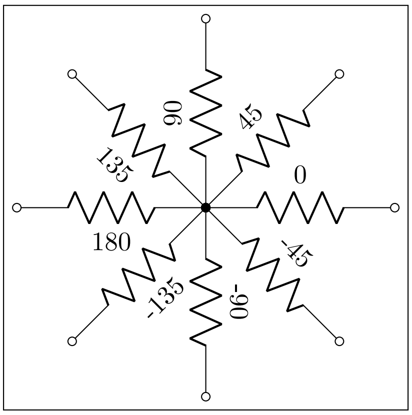



```
\def\eq{=}
\begin{circuitikz}
    % the following will fail:
    % \draw (0,0) to[R, l={$R=3}] (3,0);
    \draw (0,0) to[R, l=\mbox{$R=3$}] (3,0);
    \draw (0,0) to[R, l=$R\eq3$] (0,3);
    \draw (3,3) to[R, l=\mbox{$R,3$}] (3,0);
    % this works, but it has wrong spacing
    \draw (0,3) to[R, l=$R{=}3$] (3,3);
\end{circuitikz}
```

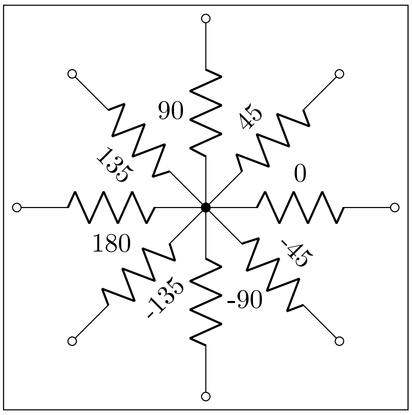
The default orientation of labels is controlled by the options `smartlabels`, `rotatelabels` and `straightlabels` (or the corresponding `label/align` keys). Here are examples to see the differences:




```
\begin{circuitikz}
\ctikzset{label/align = straight}
\def\DIR{0,45,90,135,180,-90,-45,-135}
\foreach \i in \DIR {
  \draw (0,0) to[R=\i, *-o] (\i:2.5);
}
\end{circuitikz}
```

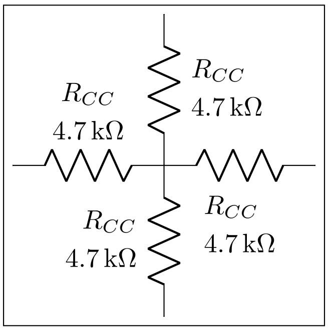



```
\begin{circuitikz}
\ctikzset{label/align = rotate}
\def\DIR{0,45,90,135,180,-90,-45,-135}
\foreach \i in \DIR {
  \draw (0,0) to[R=\i, *-o] (\i:2.5);
}
\end{circuitikz}
```

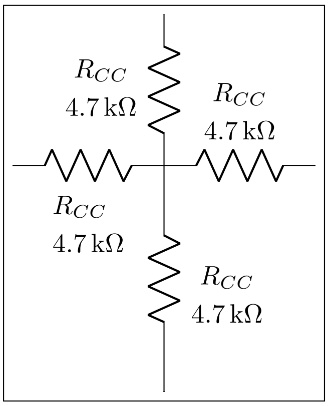



```
\begin{circuitikz}
\ctikzset{label/align = smart}
\def\DIR{0,45,90,135,180,-90,-45,-135}
\foreach \i in \DIR {
  \draw (0,0) to[R=\i, *-o] (\i:2.5);
}
\end{circuitikz}
```

You also can use stacked (two lines) labels. The example should be self-explanatory: the two lines are specified as `l2=`*line1* `and` *line2*. You can use the keys `l2 halign` to control horizontal position (`l`eft, `c`enter, `r`ight) and `l2 valign` to control the vertical one (`b`ottom, `c`center, `t`op).




```
\begin{circuitikz}[ american, ]
    %
    % default is l2 halign=l, l2 valign=c
    %
    \draw (0,0) to[R, l2_=$R_{CC}$ and \SI{4.7}{k\ohm},   , l2 valign=t] (2,0);
    \draw (0,0) to[R, l2_=$R_{CC}$ and \SI{4.7}{k\ohm},   ,          ] (0,2);
    \draw (0,0) to[R, l2_=$R_{CC}$ and \SI{4.7}{k\ohm}, l2 halign=c, l2 valign=b] (-2,0);
    \draw (0,0) to[R, l2_=$R_{CC}$ and \SI{4.7}{k\ohm}, l2 halign=r, l2 valign=c] (0, -2);
\end{circuitikz}
```




```
\begin{circuitikz}[ american, ]
    \draw (0,0) to[R, l2^=$R_{CC}$ and \SI{4.7}{k\ohm}, l2 halign=c, l2 valign=b] (2,0);
    \draw (0,0) to[R, l2^=$R_{CC}$ and \SI{4.7}{k\ohm}, l2 halign=c, ] (0,2);
    \draw (0,0) to[R, l2^=$R_{CC}$ and \SI{4.7}{k\ohm},   , l2 valign=t] (-2,0);
    \draw (0,0) to[R, l2^=$R_{CC}$ and \SI{4.7}{k\ohm}, l2 halign=c, l2 valign=t](0, -3);
\end{circuitikz}
```

7.2 Currents and voltages

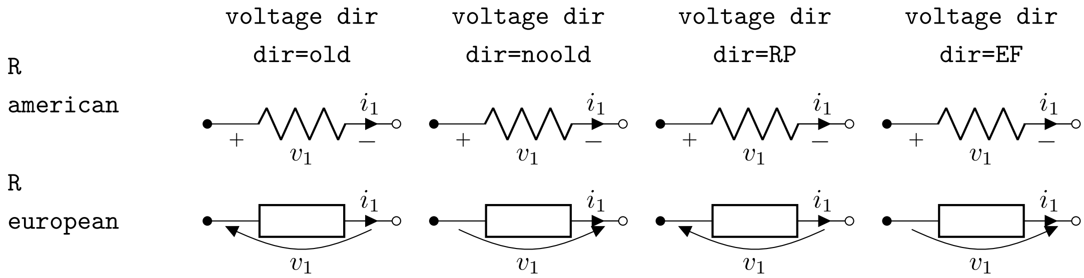
The default direction/sign for currents and voltages in the components is, unfortunately, not standard, and can change across country and sometime across different authors. This unfortunate situation created a bit of confusion in `circuitikz` across the versions, with several incompatible changes starting from version 0.5. From version 0.8.4 onward, the maintainers agreed a new policy for the directions of bipoles's voltages and currents, depending on 4 different possible options:

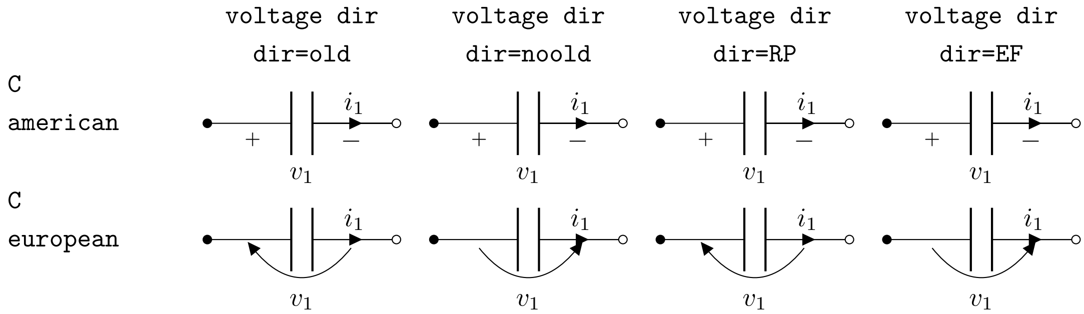
- `oldvoltagedirection`, or the key style `voltage dir=old`: Use old way of voltage direction having a difference between european and american direction, with wrong default labelling for batteries (it was the default before version 0.5);
- `nooldvoltagedirection`, or the key style `voltage dir=noold`: The standard from version 0.5 onward, utilize the (German?) standard of voltage arrows in the direction of electric fields (without fixing batteries);
- `RPvoltages` (meaning Rising Potential voltages), or the key style `voltage dir=RP`: the arrow is in direction of rising potential, like in `oldvoltagedirections`, but batteries and current sources are fixed so that they follow the passive/active standard: the default direction of `v` and `i` are chosen so that, when both values are positive:
  - in passive component, the element is *dissipating power*;
  - in active components (generators), the element is *generating power*.
- `EFvoltages` (meaning Electric Field voltages), or the key style `voltage dir=EF`: the arrow is in direction of the electric field, like in `nooldvoltagedirections`, but batteries are fixed;

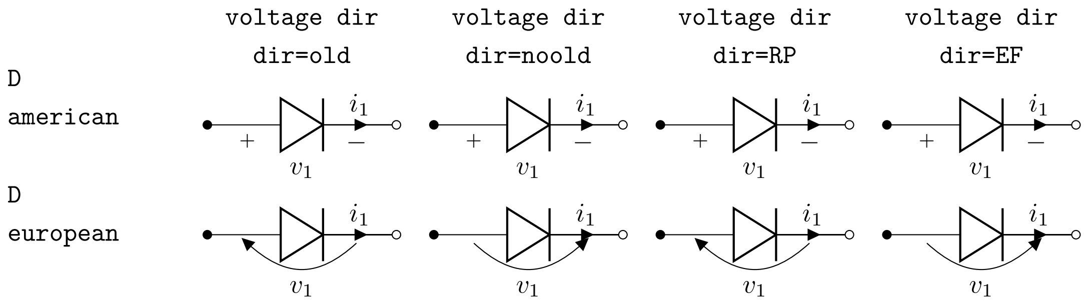
The standard direction of currents, flows and voltages are changed by these options; notice that the default drops in case of passive and active elements is normally different. Take care that in the case of `noold` and `EFvoltages` also the currents can switch directions. It is much easier to understand the several behaviors by looking at the following examples, that have been generated by the code:

```
\foreach\element in {R, C, D, battery2, V, I, sV, cV, cI}{%
    \noindent\ttfamily
    \begin{tabular}{p{2cm}}
        \element \\ american \\[15pt]
        \element \\ european \\
    \end{tabular}
    \foreach\mode in {old, noold, RP, EF} {
        \begin{tabular}{@{}l@{}}
            \multicolumn{1}{c}{voltage dir} \\
            \multicolumn{1}{c}{dir=\mode} \\[4pt]
```

```
            \begin{tikzpicture}[
                american, voltage dir=\mode,
                ]
                \draw (0,0) to[\element, *-o, v=$v_1$, i=$i_1$, ] (2.5,0);
            \end{tikzpicture}\\
            \begin{tikzpicture}[
                european, voltage dir=\mode,
                ]
                \draw (0,0) to[\element, *-o, v=$v_1$, i=$i_1$, ] (2.5,0);
            \end{tikzpicture}
        \end{tabular}
        \medskip
    }
    \par
}
```



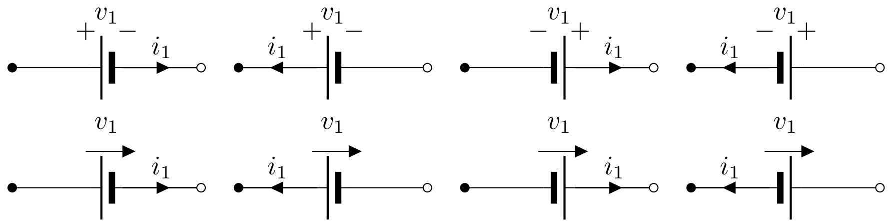




	voltage dir dir=old	voltage dir dir=noold	voltage dir dir=RP	voltage dir dir=EF

battery2
american

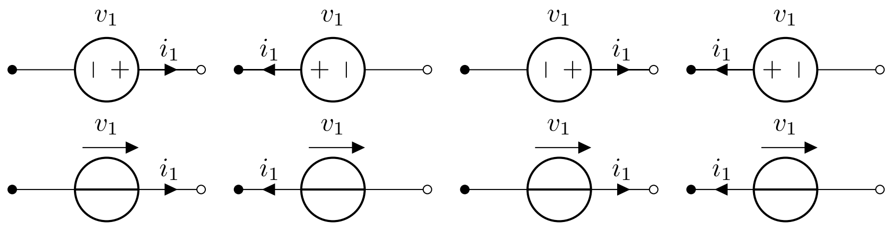
battery2
european



	voltage dir dir=old	voltage dir dir=noold	voltage dir dir=RP	voltage dir dir=EF

V
american

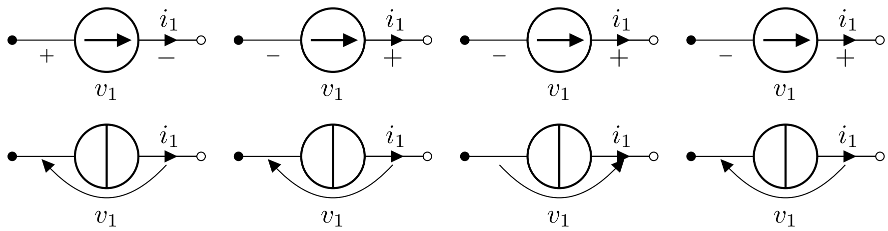
V
european



	voltage dir dir=old	voltage dir dir=noold	voltage dir dir=RP	voltage dir dir=EF

I
american

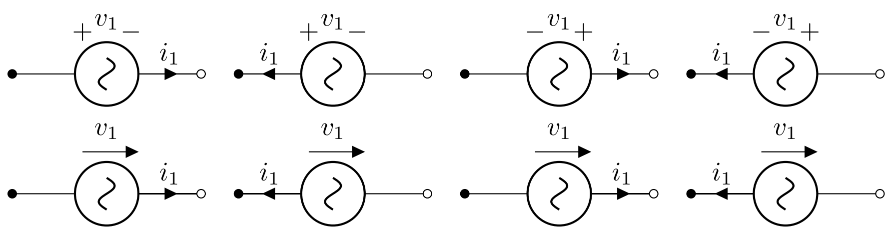
I
european

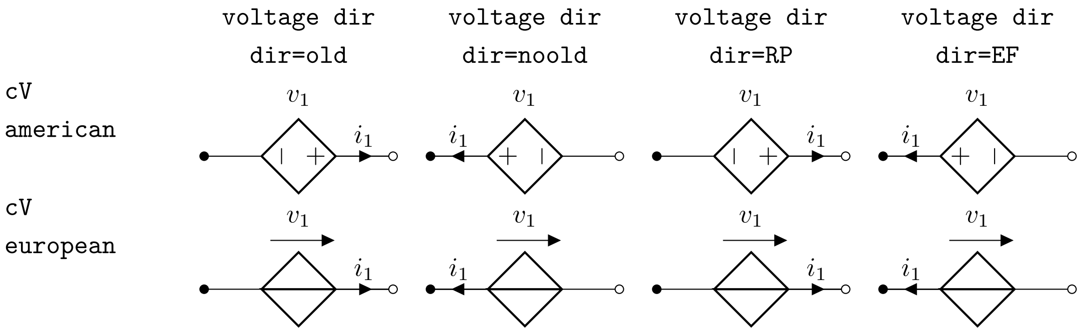


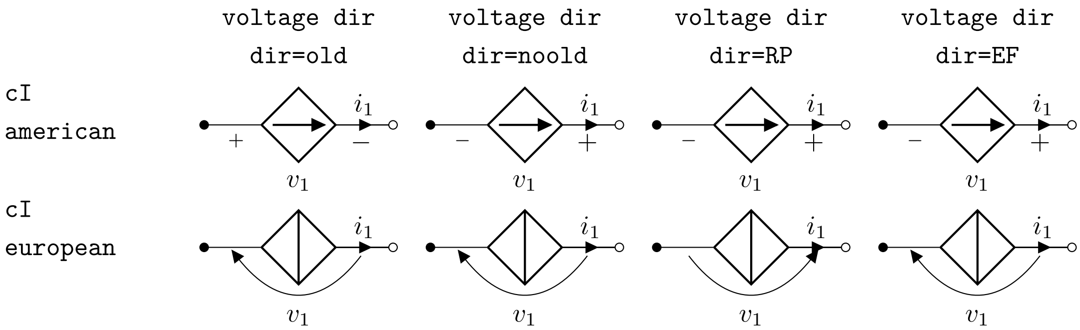
	voltage dir dir=old	voltage dir dir=noold	voltage dir dir=RP	voltage dir dir=EF

sV
american

sV
european





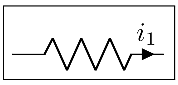


Obviously, you normally use just one between current and flows, but anyway you can change direction of the voltages, currents and flows using the complete keys `i_>`, `i^<`, `i>_`, `i>^`, as shown in the following examples.

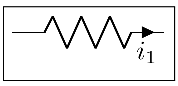
This manual has been typeset with the option `RPvoltages`.

7.3 Currents

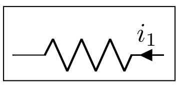
Inline (along the wire) currents are selected with `i_>`, `i^<`, `i>_`, `i>^`, and various simplification; the default position and direction is obtained with the key `i=...`.



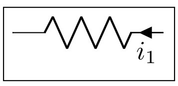

```
\begin{circuitikz}
  \draw (0,0) to[R, i^>=$i_1$] (2,0);
\end{circuitikz}
```



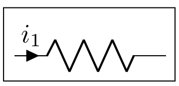

```
\begin{circuitikz}
  \draw (0,0) to[R, i_>=$i_1$] (2,0);
\end{circuitikz}
```



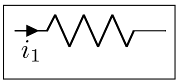

```
\begin{circuitikz}
  \draw (0,0) to[R, i^<=$i_1$] (2,0);
\end{circuitikz}
```



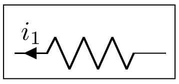

```
\begin{circuitikz}
  \draw (0,0) to[R, i_<=$i_1$] (2,0);
\end{circuitikz}
```



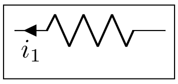

```
\begin{circuitikz}
  \draw (0,0) to[R, i>^=$i_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[R, i>_=$i_1$] (2,0);
\end{circuitikz}
```

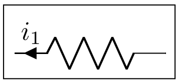



```
\begin{circuitikz}
  \draw (0,0) to[R, i<^=$i_1$] (2,0);
\end{circuitikz}
```

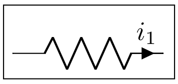



```
\begin{circuitikz}
  \draw (0,0) to[R, i<_=$i_1$] (2,0);
\end{circuitikz}
```

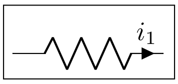
Also



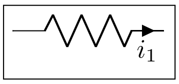

```
\begin{circuitikz}
  \draw (0,0) to[R, i<=$i_1$] (2,0);
\end{circuitikz}
```



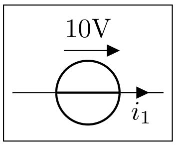

```
\begin{circuitikz}
  \draw (0,0) to[R, i>=$i_1$] (2,0);
\end{circuitikz}
```



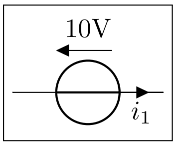

```
\begin{circuitikz}
  \draw (0,0) to[R, i^=$i_1$] (2,0);
\end{circuitikz}
```



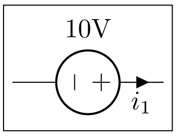

```
\begin{circuitikz}
  \draw (0,0) to[R, i_=$i_1$] (2,0);
\end{circuitikz}
```



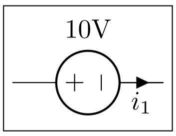

```
\begin{circuitikz}
  \draw (0,0) to[V=10V, i_=$i_1$] (2,0);
\end{circuitikz}
```



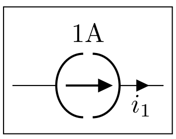

```
\begin{circuitikz}
  \draw (0,0) to[V<=10V, i_=$i_1$] (2,0);
\end{circuitikz}
```



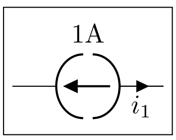

```
\begin{circuitikz}[american]
  \draw (0,0) to[V=10V, i_=$i_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}[american]
  \draw (0,0) to[V=10V,invert, i_=$i_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}[american]
  \draw (0,0) to[dcisource=1A, i_=$i_1$] (2,0);
\end{circuitikz}
```

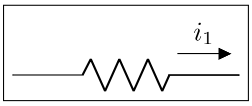



```
\begin{circuitikz}[american]
  \draw (0,0) to[dcisource=1A,invert, i_=$i_1$] (2,0);
\end{circuitikz}
```

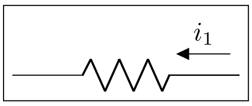
7.4 Flows

As an alternative for the current arrows, you can also use the following flows. They can also be used to indicate thermal or power flows. The syntax is pretty the same as for currents.

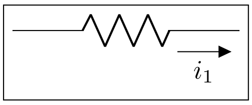
This is a new beta feature since version 0.8.3, therefore, please provide bugreports or hints to optimize this feature regarding placement and appearance! This means, that the appearance may change in the future!



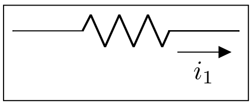

```
\begin{circuitikz}
  \draw (0,0) to[R, f=$i_1$] (3,0);
\end{circuitikz}
```



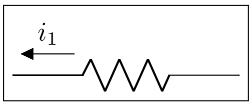

```
\begin{circuitikz}
  \draw (0,0) to[R, f<=$i_1$] (3,0);
\end{circuitikz}
```



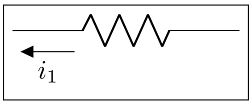

```
\begin{circuitikz}
  \draw (0,0) to[R, f_=$i_1$] (3,0);
\end{circuitikz}
```



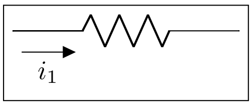

```
\begin{circuitikz}
  \draw (0,0) to[R, f_>=$i_1$] (3,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[R, f<^=$i_1$] (3,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[R, f<_=$i_1$] (3,0);
\end{circuitikz}
```



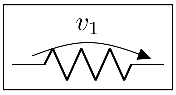

```
\begin{circuitikz}
  \draw (0,0) to[R, f>_=$i_1$] (3,0);
\end{circuitikz}
```

7.5 Voltages

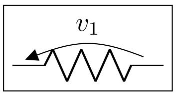
See introduction note at Currents (chapter 7.2, page 44)!

7.5.1 European style

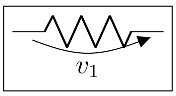
The default, with arrows. Use option `europeanvoltage` or style `[european voltages]`.



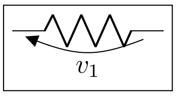

```
\begin{circuitikz}[european voltages]
  \draw (0,0) to[R, v^>=$v_1$] (2,0);
\end{circuitikz}
```



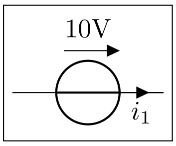

```
\begin{circuitikz}[european voltages]
  \draw (0,0) to[R, v^<=$v_1$] (2,0);
\end{circuitikz}
```



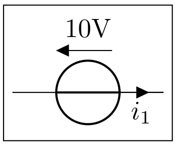

```
\begin{circuitikz}[european voltages]
  \draw (0,0) to[R, v_>=$v_1$] (2,0);
\end{circuitikz}
```



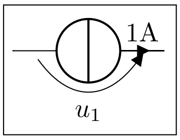

```
\begin{circuitikz}[european voltages]
  \draw (0,0) to[R, v_<=$v_1$] (2,0);
\end{circuitikz}
```



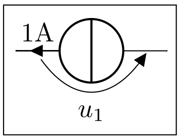

```
\begin{circuitikz}
  \draw (0,0) to[V=10V, i_=$i_1$] (2,0);
\end{circuitikz}
```



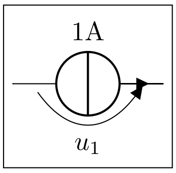

```
\begin{circuitikz}
  \draw (0,0) to[V<=10V, i_=$i_1$] (2,0);
\end{circuitikz}
```



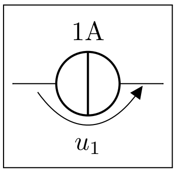

```
\begin{circuitikz}
  \draw (0,0) to[I=1A, v_=$u_1$] (2,0);
\end{circuitikz}
```



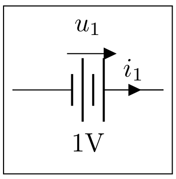

```
\begin{circuitikz}
  \draw (0,0) to[I<=1A, v_=$u_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[I=$~$,l=1A, v_=$u_1$] (2,0);
\end{circuitikz}
```



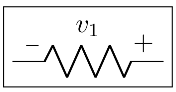

```
\begin{circuitikz}
  \draw (0,0) to[I,l=1A, v_=$u_1$] (2,0);
\end{circuitikz}
```



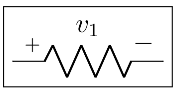

```
\begin{circuitikz}
  \draw (0,0) to[battery,l_=1V, v=$u_1$, i=$i_1$] (2,0);
\end{circuitikz}
```

7.5.2 American style

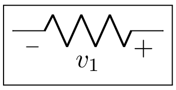
For those who like it (not me). Use option `americanvoltage` or set `[american voltages]`.



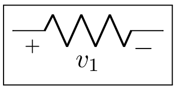

```
\begin{circuitikz}[american voltages]
  \draw (0,0) to[R, v^>=$v_1$] (2,0);
\end{circuitikz}
```



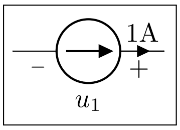

```
\begin{circuitikz}[american voltages]
  \draw (0,0) to[R, v^<=$v_1$] (2,0);
\end{circuitikz}
```



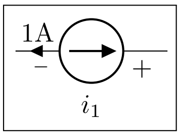

```
\begin{circuitikz}[american voltages]
  \draw (0,0) to[R, v_>=$v_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}[american voltages]
  \draw (0,0) to[R, v_<=$v_1$] (2,0);
\end{circuitikz}
```



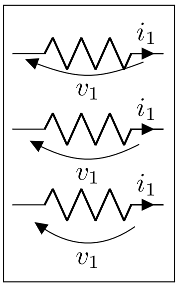

```
\begin{circuitikz}[american]
  \draw (0,0) to[I=1A, v_=$u_1$] (2,0);
\end{circuitikz}
```



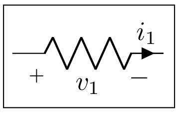

```
\begin{circuitikz}[american]
  \draw (0,0) to[I<=1A, v_=$i_1$] (2,0);
\end{circuitikz}
```

7.5.3 Voltage position

It is possible to move away the arrows and the plus or minus signs with the key `voltages shift` (default value is `0`, which gives the standard position):

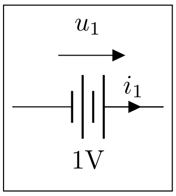



```
\begin{circuitikz}[]
  \draw (0,0) to[R, v=$v_1$, i=$i_1$] (2,0);
  \draw (0,-1) to[R, v=$v_1$, i=$i_1$,
     voltage shift=0.5] (2,-1);
  \draw (0,-2) to[R, v=$v_1$, i=$i_1$,
     voltage shift=1.0, ] (2,-2);
\end{circuitikz}
```

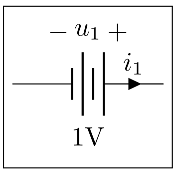



```
\begin{circuitikz}[american voltages, voltage shift=0.5]
  \draw (0,0) to[R, v=$v_1$, i=$i_1$] (2,0);
\end{circuitikz}
```

Notes that `american voltage` will not affect batteries (and that the default direction of `european voltages` is a bit strange).

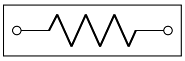



```
   \begin{circuitikz}[voltage shift=0.5]
  \draw (0,0) to[battery,l_=1V, v=$u_1$, i=$i_1$] (2,0);
\end{circuitikz}
```

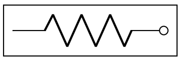



```
\begin{circuitikz}[american voltages, voltage shift=0.5]
  \draw (0,0) to[battery,l_=1V, v=$u_1$, i=$i_1$] (2,0);
\end{circuitikz}
```

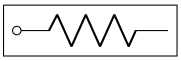
7.6 Nodes



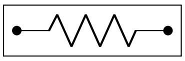
```
\begin{circuitikz}
  \draw (0,0) to[R, o-o] (2,0);
\end{circuitikz}
```



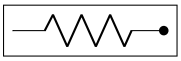
```
\begin{circuitikz}
  \draw (0,0) to[R, -o] (2,0);
\end{circuitikz}
```



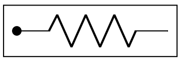
```
\begin{circuitikz}
  \draw (0,0) to[R, o-] (2,0);
\end{circuitikz}
```



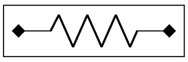
```
\begin{circuitikz}
  \draw (0,0) to[R, *-*] (2,0);
\end{circuitikz}
```



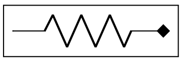
```
\begin{circuitikz}
  \draw (0,0) to[R, -*] (2,0);
\end{circuitikz}
```



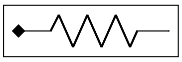
```
\begin{circuitikz}
  \draw (0,0) to[R, *-] (2,0);
\end{circuitikz}
```



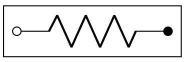
```
\begin{circuitikz}
  \draw (0,0) to[R, d-d] (2,0);
\end{circuitikz}
```



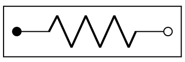
```
\begin{circuitikz}
  \draw (0,0) to[R, -d] (2,0);
\end{circuitikz}
```



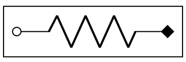
```
\begin{circuitikz}
  \draw (0,0) to[R, d-] (2,0);
\end{circuitikz}
```



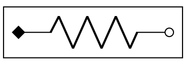
```
\begin{circuitikz}
  \draw (0,0) to[R, o-*] (2,0);
\end{circuitikz}
```



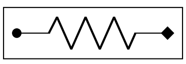
```
\begin{circuitikz}
  \draw (0,0) to[R, *-o] (2,0);
\end{circuitikz}
```



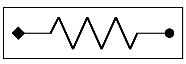
```
\begin{circuitikz}
  \draw (0,0) to[R, o-d] (2,0);
\end{circuitikz}
```



```
\begin{circuitikz}
  \draw (0,0) to[R, d-o] (2,0);
\end{circuitikz}
```



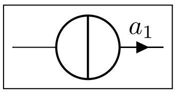
```
\begin{circuitikz}
  \draw (0,0) to[R, *-d] (2,0);
\end{circuitikz}
```



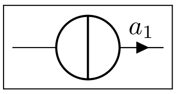
```
\begin{circuitikz}
  \draw (0,0) to[R, d-*] (2,0);
\end{circuitikz}
```

7.7 Special components

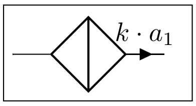
For some components label, current and voltage behave as one would expect:



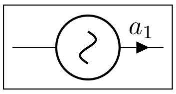

```
\begin{circuitikz}
  \draw (0,0) to[I=$a_1$] (2,0);
\end{circuitikz}
```



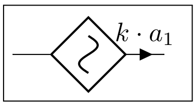

```
\begin{circuitikz}
  \draw (0,0) to[I, i=$a_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[cI=$k\cdot a_1$] (2,0);
\end{circuitikz}
```

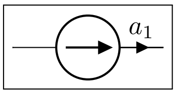



```
\begin{circuitikz}
  \draw (0,0) to[sI=$a_1$] (2,0);
\end{circuitikz}
```

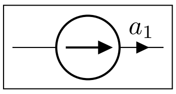



```
\begin{circuitikz}
  \draw (0,0) to[csI=$k\cdot a_1$] (2,0);
\end{circuitikz}
```

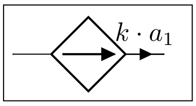
The following results from using the option `americancurrent` or using the style `[american currents]`.



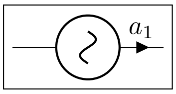

```
\begin{circuitikz}[american currents]
  \draw (0,0) to[I=$a_1$] (2,0);
\end{circuitikz}
```



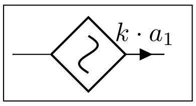

```
\begin{circuitikz}[american currents]
  \draw (0,0) to[I, i=$a_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}[american currents]
  \draw (0,0) to[cI=$k\cdot a_1$] (2,0);
\end{circuitikz}
```

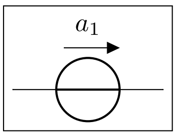



```
\begin{circuitikz}[american currents]
  \draw (0,0) to[sI=$a_1$] (2,0);
\end{circuitikz}
```

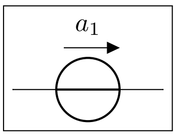



```
\begin{circuitikz}[american currents]
  \draw (0,0) to[csI=$k\cdot a_1$] (2,0);
\end{circuitikz}
```

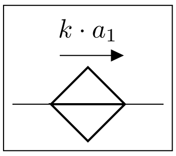
The same holds for voltage sources:



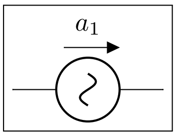

```
\begin{circuitikz}
  \draw (0,0) to[V=$a_1$] (2,0);
\end{circuitikz}
```



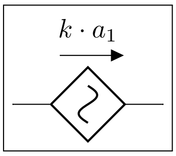

```
\begin{circuitikz}
  \draw (0,0) to[V, v=$a_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[cV=$k\cdot a_1$] (2,0);
\end{circuitikz}
```

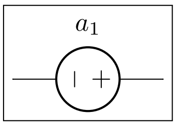



```
\begin{circuitikz}
  \draw (0,0) to[sV=$a_1$] (2,0);
\end{circuitikz}
```

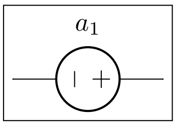



```
\begin{circuitikz}
  \draw (0,0) to[csV=$k\cdot a_1$] (2,0);
\end{circuitikz}
```

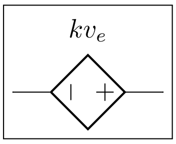
The following results from using the option `americanvoltage` or the style `[american voltages]`.



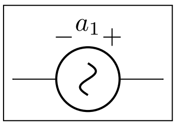

```
\begin{circuitikz}[american voltages]
  \draw (0,0) to[V=$a_1$] (2,0);
\end{circuitikz}
```



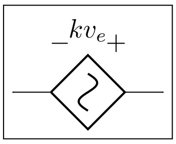

```
\begin{circuitikz}[american voltages]
  \draw (0,0) to[V, v=$a_1$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}[american voltages]
  \draw (0,0) to[cV=$k v_e$] (2,0);
\end{circuitikz}
```



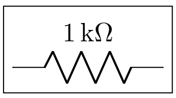

```
\begin{circuitikz}[american voltages]
  \draw (0,0) to[sV=$a_1$] (2,0);
\end{circuitikz}
```



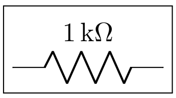

```
\begin{circuitikz}[american voltages]
  \draw (0,0) to[csV=$k v_e$] (2,0);
\end{circuitikz}
```

7.8 Integration with `siunitx`

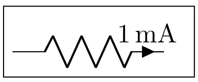
If the option `siunitx` is active (and *not* in ConTEXt), then the following are equivalent:



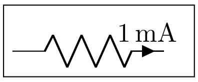

```
\begin{circuitikz}
  \draw (0,0) to[R, l=1<\kilo\ohm>] (2,0);
\end{circuitikz}
```



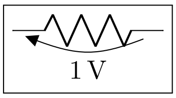

```
\begin{circuitikz}
  \draw (0,0) to[R, l=$\SI{1}{\kilo\ohm}$] (2,0);
\end{circuitikz}
```



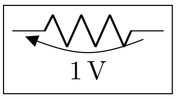

```
\begin{circuitikz}
  \draw (0,0) to[R, i=1<\milli\ampere>] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[R, i=$\SI{1}{\milli\ampere}$] (2,0);
\end{circuitikz}
```



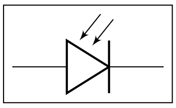

```
\begin{circuitikz}
  \draw (0,0) to[R, v=1<\volt>] (2,0);
\end{circuitikz}
```



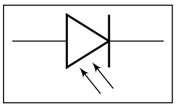

```
\begin{circuitikz}
  \draw (0,0) to[R, v=$\SI{1}{\volt}$] (2,0);
\end{circuitikz}
```

7.9 Mirroring and Inverting

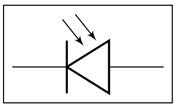
Bipole paths can also mirrored and inverted (or reverted) to change the drawing direction.



```
\begin{circuitikz}
  \draw (0,0) to[pD] (2,0);
\end{circuitikz}
```

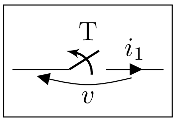


```
\begin{circuitikz}
  \draw (0,0) to[pD, mirror] (2,0);
\end{circuitikz}
```

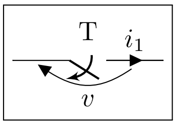


```
\begin{circuitikz}
  \draw (0,0) to[pD, invert] (2,0);
\end{circuitikz}
```

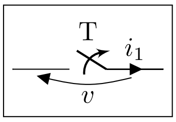
Placing labels, currents and voltages works also, please note, that mirroring and inverting does not incfluence the positioning of labels and voltages. Labels are by default above/right of the bipole and voltages below/left, respectively.



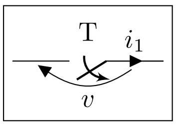

```
\begin{circuitikz}
  \draw (0,0) to[ospst=T, i=$i_1$, v=$v$] (2,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[ospst=T, mirror, i=$i_1$, v=$v$] (2,0);
\end{circuitikz}
```

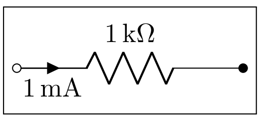



```
\begin{circuitikz}
  \draw (0,0) to[ospst=T, invert, i=$i_1$, v=$v$] (2,0);
\end{circuitikz}
```

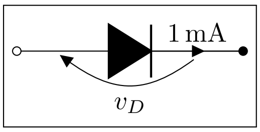



```
\begin{circuitikz}
  \draw (0,0) to[ospst=T,mirror,invert, i=$i_1$, v=$v$] (2,0);
\end{circuitikz}
```

7.10 Putting them together



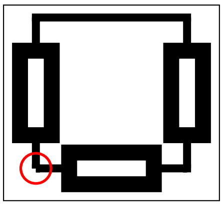

```
\begin{circuitikz}
  \draw (0,0) to[R=1<\kilo\ohm>,
     i>_=1<\milli\ampere>, o-*] (3,0);
\end{circuitikz}
```




```
\begin{circuitikz}
  \draw (0,0) to[D*, v=$v_D$,
    i=1<\milli\ampere>, o-*] (3,0);
\end{circuitikz}
```

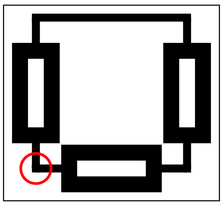
7.11 Line joins between Path Components

Line joins should be calculated correctly, if the were on the same path and if the path is not closed. For example, the following path is not closed correctly(–*cycle* does not work here!):



```
\begin{tikzpicture}[line width=3pt,european]
\draw (0,0) to[R]++(2,0)to[R]++(0,2)
  --++(-2,0)to[R]++(0,-2);
\draw[red,line width=1pt] circle(2mm);
\end{tikzpicture}
```

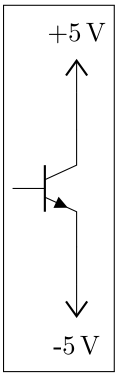
To correct the line ending, there are support shapes to fill the missing rectangle. They can be used like the support shapes(*,o,d) using a dot (.) on one or both ends of a component(have a look at the last resistor in this example:



```
\begin{tikzpicture}[line width=3pt,european]
\draw (0,0) to[R]++(2,0)to[R]++(0,2)
  --++(-2,0)to[R,-.]++(0,-2);
\draw[red,line width=1pt] circle(2mm);
\end{tikzpicture}
```

8 Not only bipoles

Since only bipoles (but see section 8.6) can be placed ”along a line”, components with more than two terminals are placed as nodes:



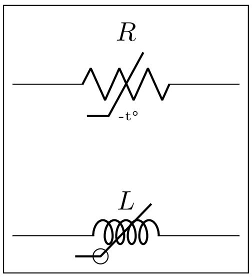

```
\begin{circuitikz}
\draw (0,0) node[npn](npn) at (0,0) {};
\draw (npn.C) --++(0,0.5) node[vcc]{+5\,\textnormal{V}};
\draw (npn.E) --++(0,-0.5) node[vee]{-5\,\textnormal{V}};
\end{circuitikz}
```

8.1 Anchors

In order to allow connections with other components, all components define anchors.

8.1.1 Sensors

generic sensors have an extra label to help positioning the type of dependence, if needed:

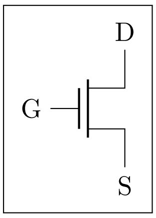



```
\begin{circuitikz}
  \draw (0,2) to[sR, l=$R$, name=mySR] ++(3,0);
  \node [font=\tiny, right] at(mySR.label) {-t\si{\degree}};
  \draw (0,0) to[sL, l=$L$, name=mySL] ++(3,0);
  \node [draw, circle, inner sep=2pt] at(mySL.label) {};
\end{circuitikz}
```

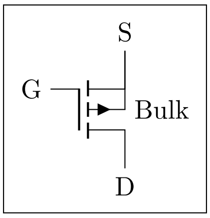
The anchor is positioned just on the corner of the segmented line crossing the component.

8.1.2 Transistors

For NMOS, PMOS, NFET, NIGFETE, NIGFETD, PFET, PIGFETE, and PIGFETD transistors one has `base`, `gate`, `source` and `drain` anchors (which can be abbreviated with `B`, `G`, `S` and `D`):

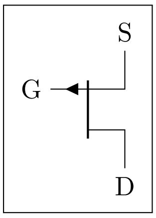



```
\begin{circuitikz} \draw
 (0,0) node[nmos] (mos) {}
 (mos.gate) node[anchor=east] {G}
 (mos.drain) node[anchor=south] {D}
 (mos.source) node[anchor=north] {S}
;\end{circuitikz}
```



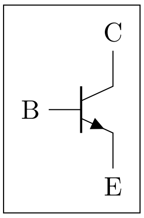

```
\begin{circuitikz} \draw
 (0,0) node[pigfete] (pigfete) {}
 (pigfete.G) node[anchor=east] {G}
 (pigfete.D) node[anchor=north] {D}
 (pigfete.S) node[anchor=south] {S}
 (pigfete.bulk) node[anchor=west] {Bulk}
;\end{circuitikz}
```

Similarly NJFET and PJFET have `gate`, `source` and `drain` anchors (which can be abbreviated with `G`, `S` and `D`):

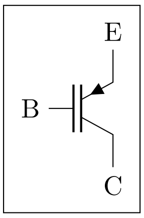



```
\begin{circuitikz} \draw
 (0,0) node[pjfet] (pjfet) {}
 (pjfet.G) node[anchor=east] {G}
 (pjfet.D) node[anchor=north] {D}
 (pjfet.S) node[anchor=south] {S}
;\end{circuitikz}
```

For NPN, PNP, NIGBT, and PIGBT transistors the anchors are `base`, `emitter` and `collector` anchors (which can be abbreviated with `B`, `E` and `C`):

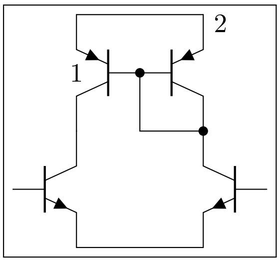



```
\begin{circuitikz} \draw
 (0,0) node[npn] (npn) {}
 (npn.base) node[anchor=east] {B}
 (npn.collector) node[anchor=south] {C}
 (npn.emitter) node[anchor=north] {E}
;\end{circuitikz}
```



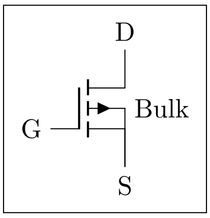

```
\begin{circuitikz} \draw
 (0,0) node[pigbt] (pigbt) {}
 (pigbt.B) node[anchor=east] {B}
 (pigbt.C) node[anchor=north] {C}
 (pigbt.E) node[anchor=south] {E}
;\end{circuitikz}
```

Here is one composite example (please notice that the `xscale=-1` style would also reflect the label of the transistors, so here a new node is added and its text is used, instead of that of `pnp1`):

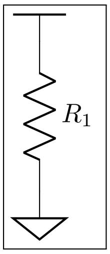



```
\begin{circuitikz} \draw
 (0,0) node[pnp] (pnp2) {2}
 (pnp2.B) node[pnp, xscale=-1, anchor=B] (pnp1) {}
   (pnp1) node {1}
 (pnp1.C) node[npn, anchor=C] (npn1) {}
 (pnp2.C) node[npn, xscale=-1, anchor=C] (npn2) {}
 (pnp1.E) -- (pnp2.E) (npn1.E) -- (npn2.E)
 (pnp1.B) node[circ] {} |- (pnp2.C) node[circ] {}
;\end{circuitikz}
```

Similarly, transistors and other components can be reflected vertically:



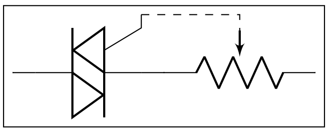

```
\begin{circuitikz} \draw
 (0,0) node[pigfete, yscale=-1] (pigfete) {}
 (pigfete.bulk) node[anchor=west] {Bulk}
 (pigfete.G) node[anchor=east] {G}
 (pigfete.D) node[anchor=south] {D}
 (pigfete.S) node[anchor=north] {S}
;\end{circuitikz}
```




```
 \begin{circuitikz}
     \draw (0,2)
         node[rground, yscale=-1] {}
     to[R=$R_1$] (0,0)
         node[sground] {};
  \end{circuitikz}
```

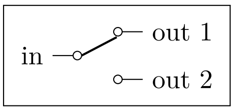
8.1.3 Other tripoles

When inserting a thrystor, a triac or a potentiometer, one needs to refer to the third node–gate (`gate` or `G`) for the former two; wiper (`wiper` or `W`) for the latter one. This is done by giving a name to the bipole:

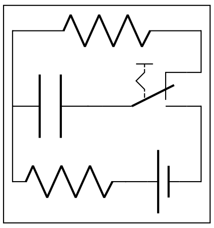


```
\begin{circuitikz} \draw
 (0,0) to[Tr, n=TRI] (2,0)
       to[pR, n=POT] (4,0);
 \draw[dashed] (TRI.G) -| (POT.wiper)
;\end{circuitikz}
```

As for the switches:

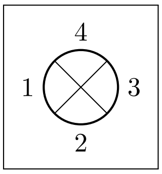



```
\begin{circuitikz} \draw
 (0,0) node[spdt] (Sw) {}
 (Sw.in) node[left] {in}
 (Sw.out 1) node[right] {out 1}
 (Sw.out 2) node[right] {out 2}
;\end{circuitikz}
```



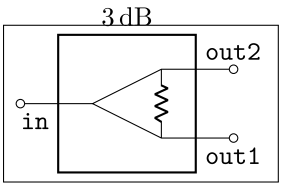
```
\begin{circuitikz} \draw
(0,0) to[C] (1,0) to[toggle switch , n=Sw] (2.5,0)
  -- (2.5,-1) to[battery1] (1.5,-1) to[R] (0,-1) -| (0,0)
 (Sw.out 2) -| (2.5, 1) to[R] (0,1) -- (0,0)
;\end{circuitikz}
```

The ports of the mixer and adder can be addressed with numbers or `west`/`south`/`east`/`north`:




```
\begin{circuitikz} \draw
 (0,0) node[mixer] (mix) {}
 (mix.1) node[left] {1}
 (mix.2) node[below] {2}
 (mix.3) node[right] {3}
 (mix.4) node[above] {4}
;\end{circuitikz}
```

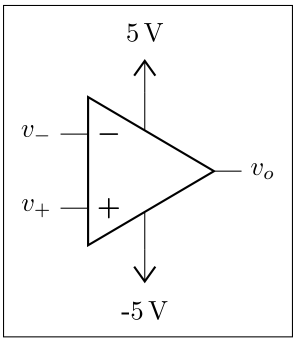
The Wilkinson divider has:




```
\begin{circuitikz} \draw
 (0,0) node[wilkinson] (w) {\SI{3}{dB}}
 (w.in) to[short,-o] ++(-0.5,0)
 (w.out1) to[short,-o] ++(0.5,0)
 (w.out2) to[short,-o] ++(0.5,0)
 (w.in) node[below left] {\texttt{in}}
 (w.out1) node[below right] {\texttt{out1}}
 (w.out2) node[above right] {\texttt{out2}}
 ;
\end{circuitikz}
```

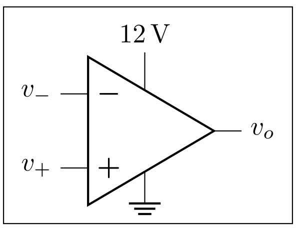
8.1.4 Operational amplifier

The op amp defines the inverting input (`-`), the non-inverting input (`+`) and the output (`out`) anchors:



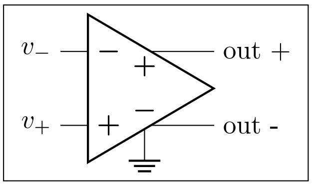

```
\begin{circuitikz} \draw
 (0,0) node[op amp] (opamp) {}
 (opamp.+) node[left] {$v_+$}
 (opamp.-) node[left] {$v_-$}
 (opamp.out) node[right] {$v_o$}
 (opamp.up) --++(0,0.5) node[vcc]{5\,\textnormal{V}}
 (opamp.down) --++(0,-0.5) node[vee]{-5\,\textnormal{V
    }}
;\end{circuitikz}
```

There are also two more anchors defined, `up` and `down`, for the power supplies:



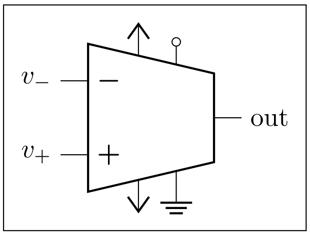

```
\begin{circuitikz} \draw
 (0,0) node[op amp] (opamp) {}
 (opamp.+) node[left] {$v_+$}
 (opamp.-) node[left] {$v_-$}
 (opamp.out) node[right] {$v_o$}
 (opamp.down) node[ground] {}
 (opamp.up) ++ (0,.5) node[above] {\SI{12}{\volt}}
   -- (opamp.up)
;\end{circuitikz}
```

The fully differential op amp defines two outputs:



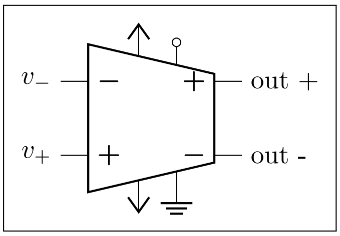

```
\begin{circuitikz} \draw
  (0,0) node[fd op amp] (opamp) {}
  (opamp.+) node[left] {$v_+$}
  (opamp.-) node[left] {$v_-$}
  (opamp.out +) node[right] {out +}
  (opamp.out -) node[right] {out -}
  (opamp.down) node[ground] {}
;\end{circuitikz}
```

The instrumentation amplifier inst amp defines also references (normally you use the ”down”, unless you are flipping the component):



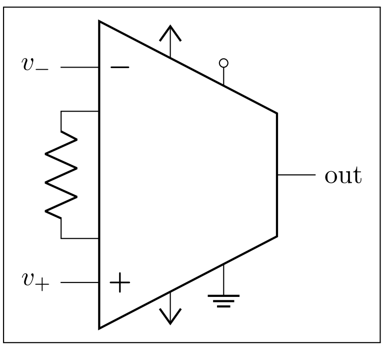

```
\begin{circuitikz} \draw
  (0,0) node[inst amp] (opamp) {}
  (opamp.+) node[left] {$v_+$}
  (opamp.-) node[left] {$v_-$}
  (opamp.out) node[right] {out}
  (opamp.up) node[vcc]{}
  (opamp.down) node[vee] {}
  (opamp.refv down) node[ground]{}
  (opamp.refv up) to[short, -o] ++(0,0.3)
;\end{circuitikz}
```

The fully diffential instrumentation amplifier inst amp defines two outputs:



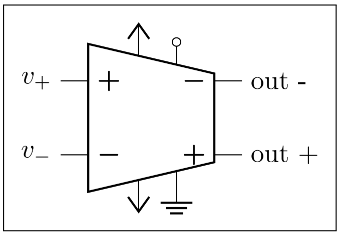

```
\begin{circuitikz} \draw
  (0,0) node[fd inst amp] (opamp) {}
  (opamp.+) node[left] {$v_+$}
  (opamp.-) node[left] {$v_-$}
  (opamp.out +) node[right] {out +}
  (opamp.out -) node[right] {out -}
  (opamp.up) node[vcc]{}
  (opamp.down) node[vee] {}
  (opamp.refv down) node[ground]{}
  (opamp.refv up) to[short, -o] ++(0,0.3)
;\end{circuitikz}
```

The instrumentation amplifier with resistance terminals (`inst amp ra`) defines also terminals to add an amplification resistor:




```
\begin{circuitikz} \draw
  (0,0) node[inst amp ra] (opamp) {}
  (opamp.+) node[left] {$v_+$}
  (opamp.-) node[left] {$v_-$}
  (opamp.out) node[right] {out}
  (opamp.up) node[vcc]{}
  (opamp.down) node[vee] {}
  (opamp.refv down) node[ground]{}
  (opamp.refv up) to[short, -o] ++(0,0.3)
  (opamp.ra-) to[R] (opamp.ra+)
;\end{circuitikz}
```

All these amplifier have the possibility to flip input and output (if needed) polarity. You can change polarity of the input with the `noinv input down` (default) or `noinv input up` key; and the output with `noinv output up` (default) or `noinv output down` key:



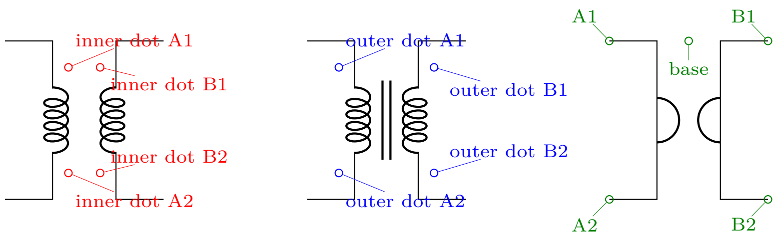

```
\begin{circuitikz} \draw
  (0,0) node[fd inst amp,
     noinv input up,
     noinv output down] (opamp) {}
  (opamp.+) node[left] {$v_+$}
  (opamp.-) node[left] {$v_-$}
  (opamp.out +) node[right] {out +}
  (opamp.out -) node[right] {out -}
  (opamp.up) node[vcc]{}
  (opamp.down) node[vee] {}
  (opamp.refv down) node[ground]{}
  (opamp.refv up) to[short, -o] ++(0,0.3)
;\end{circuitikz}
```

When you use the `noinv input/output ...` keys the anchors (`+`, `-`, `out +`, `out -`) will change with the effective position of the terminals. You have also the anchors `in up`, `in down`, `out up`, `out down` that will not change with the positive or negatie sign.

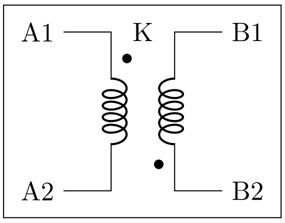
8.1.5 Double bipoles: transformers and gyrator

All the (few, actually) double bipoles/quadrupoles have the four anchors, two for each port. The first port, to the left, is port `A`, having the anchors `A1` (up) and `A2` (down); same for port `B`.

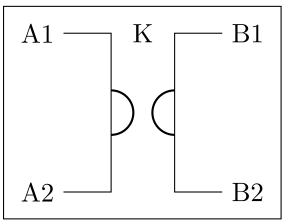
They also expose the `base` anchor, for labelling, and anchors for setting dots or signs to specify polarity. The set of anchors, to which the standard "geographical" `north`, `north east`, etc. is here:



Alse the standard "geographical" `north`, `north east`, etc. are defined. A couple of examples follow:

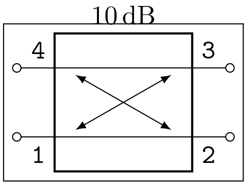



```
\begin{circuitikz} \draw
  (0,0) node[transformer] (T) {}
  (T.A1) node[anchor=east] {A1}
  (T.A2) node[anchor=east] {A2}
  (T.B1) node[anchor=west] {B1}
  (T.B2) node[anchor=west] {B2}
  (T.base) node{K}
  (T.inner dot A1) node[circ]{}
  (T.inner dot B2) node[circ]{}
;\end{circuitikz}
```

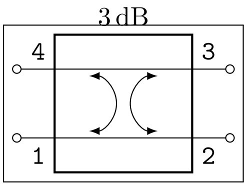



```
\begin{circuitikz} \draw
  (0,0) node[gyrator] (G) {}
  (G.A1) node[anchor=east] {A1}
  (G.A2) node[anchor=east] {A2}
  (G.B1) node[anchor=west] {B1}
  (G.B2) node[anchor=west] {B2}
  (G.base) node{K}
;\end{circuitikz}
```

8.1.6 Couplers




```
\begin{circuitikz} \draw
  (0,0) node[coupler] (c) {\SI{10}{dB}}
  (c.1) to[short,-o] ++(-0.5,0)
  (c.2) to[short,-o] ++(0.5,0)
  (c.3) to[short,-o] ++(0.5,0)
  (c.4) to[short,-o] ++(-0.5,0)
  (c.1) node[below left] {\texttt{1}}
  (c.2) node[below right] {\texttt{2}}
  (c.3) node[above right] {\texttt{3}}
  (c.4) node[above left] {\texttt{4}}
  ;
\end{circuitikz}
```

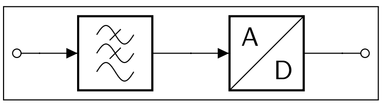



```
\begin{circuitikz} \draw
  (0,0) node[coupler2] (c) {\SI{3}{dB}}
  (c.1) to[short,-o] ++(-0.5,0)
  (c.2) to[short,-o] ++(0.5,0)
  (c.3) to[short,-o] ++(0.5,0)
  (c.4) to[short,-o] ++(-0.5,0)
  (c.1) node[below left] {\texttt{1}}
  (c.2) node[below right] {\texttt{2}}
  (c.3) node[above right] {\texttt{3}}
  (c.4) node[above left] {\texttt{4}}
  ;
\end{circuitikz}
```

8.2 Input arrows

Two ports

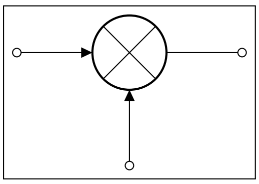
With the option > you can draw an arrow to the input of the block diagram symbols.




```
\begin{circuitikz} \draw
  (0,0) to[short,o-] ++(0.3,0)
  to[lowpass,>] ++(2,0)
  to[adc,>] ++(2,0)
  to[short,-o] ++(0.3,0);
\end{circuitikz}
```

Multi ports

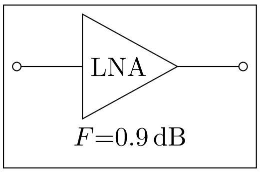
Since inputs and outputs can vary, input arrows can be placed as nodes. Note that you have to rotate the arrow on your own:



```
\begin{circuitikz} \draw
  (0,0) node[mixer] (m) {}
  (m.1) to[short,-o] ++(-1,0)
  (m.2) to[short,-o] ++(0,-1)
  (m.3) to[short,-o] ++(1,0)
  (m.1) node[inputarrow] {}
  (m.2) node[inputarrow,rotate=90] {};
\end{circuitikz}
```

8.3 Labels and custom twoport boxes

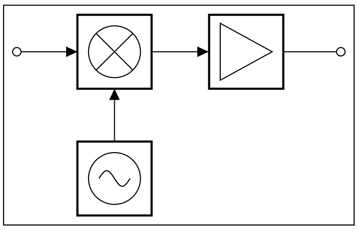
Some twoports have the option to place a normal label (`l=`) and a inner label (`t=`).




```
\begin{circuitikz}
  \ctikzset{bipoles/amp/width=0.9}
  \draw (0,0) to[amp,t=LNA,l_=$F{=}0.9\,$dB,o-o] ++(3,0);
\end{circuitikz}
```

8.4 Box option

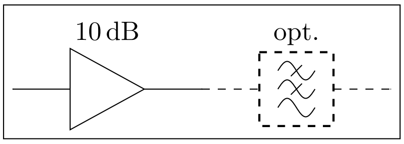
Some devices have the possibility to add a box around them. The inner symbol scales down to fit inside the box.



```
\begin{circuitikz} \draw
  (0,0) node[mixer,box,anchor=east] (m) {}
    to[amp,box,>,-o] ++(2.5,0)
  (m.west) node[inputarrow] {} to[short,-o]
      ++(-0.8,0)
  (m.south) node[inputarrow,rotate=90] {} --
    ++(0,-0.7) node[oscillator,box,anchor=north] {};
\end{circuitikz}
```

8.5 Dash optional parts

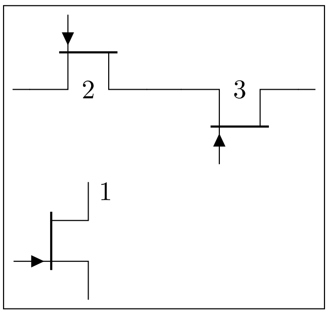
To show that a device is optional, you can dash it. The inner symbol will be kept with solid lines.




```
\begin{circuitikz}
  \draw (0,0) to[amp,l=\SI{10}{dB}] ++(2.5,0);
  \draw[dashed] (2.5,0) to[lowpass,l=opt.]
      ++(2.5,0);
\end{circuitikz}
```

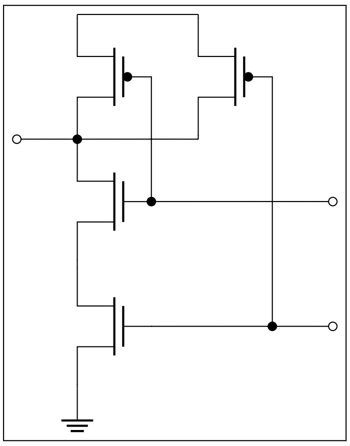
8.6 Transistor paths

For syntactical convenience transistors can be placed using the normal path notation used for bipoles. The transitor type can be specified by simply adding a "T" (for transistor) in front of the node name of the transistor. It will be placed with the base/gate orthogonal to the direction of the path:




```
\begin{circuitikz} \draw
 (0,0) node[njfet] {1}
 (-1,2) to[Tnjfet=2] (1,2)
   to[Tnjfet=3, mirror] (3,2);
;\end{circuitikz}
```

Access to the gate and/or base nodes can be gained by naming the transistors with the `n` or `name` path style:



```
\begin{circuitikz} \draw[yscale=1.1, xscale=.8]
  (2,4.5) -- (0,4.5) to[Tpmos, n=p1] (0,3)
     to[Tnmos, n=n1] (0,1.5)
     to[Tnmos, n=n2] (0,0) node[ground] {}
  (2,4.5) to[Tpmos,n=p2] (2,3) to[short, -*] (0,3)
  (p1.G) -- (n1.G) to[short, *-o] ($(n1.G)+(3,0)$)
  (n2.G) ++(2,0) node[circ] {} -| (p2.G)
  (n2.G) to[short, -o] ($(n2.G)+(3,0)$)
  (0,3) to[short, -o] (-1,3)
;\end{circuitikz}
```

The `name` property is available also for bipoles, although this is useful mostly for triac, potentiometer and thyristor (see 4.2.5).

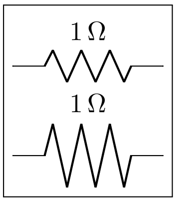
9 Customization

9.1 Parameters

Pretty much all CircuiTi*k*Z relies heavily on `pgfkeys` for value handling and configuration. Indeed, at the beginning of `circuitikz.sty` a series of key definitions can be found that modify all the graphical characteristics of the package.

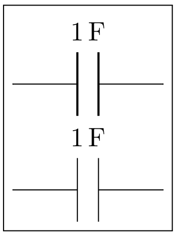
All can be varied using the `\ctikzset` command, anywhere in the code.

Shape of the components (on a per-component-class basis)



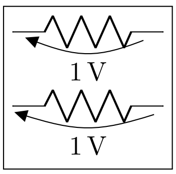

```
\tikz \draw (0,0) to[R=1<\ohm>] (2,0); \par
\ctikzset{bipoles/resistor/height=.6}
\tikz \draw (0,0) to[R=1<\ohm>] (2,0);
```

Thickness of the lines (globally)

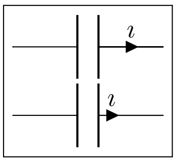



```
\tikz \draw (0,0) to[C=1<\farad>] (2,0); \par
\ctikzset{bipoles/thickness=1}
\tikz \draw (0,0) to[C=1<\farad>] (2,0);
```

Global properties Of voltage and current

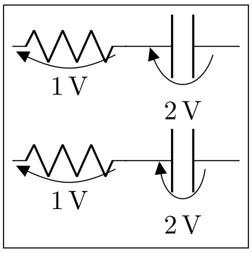



```
\tikz \draw (0,0) to[R, v=1<\volt>] (2,0); \par
\ctikzset{voltage/distance from node=.1}
\tikz \draw (0,0) to[R, v=1<\volt>] (2,0);
```



```
\tikz \draw (0,0) to[C, i=$\imath$] (2,0); \par
\ctikzset{current/distance = .2}
\tikz \draw (0,0) to[C, i=$\imath$] (2,0);
```

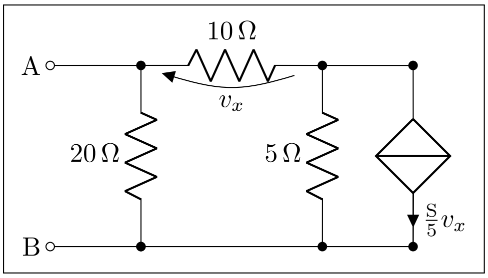
However, you can override the properties `voltage/distance from node`[7], `voltage/bump b`[8] and `voltage/european label distance`[9] on a per-component basis, in order to fine-tune the voltages:




```
\tikz \draw (0,0) to[R, v=1<\volt>] (1.5,0)
      to[C, v=2<\volt>] (3,0); \par
\ctikzset{bipoles/capacitor/voltage/%
    distance from node/.initial=.7}
\tikz \draw (0,0) to[R, v=1<\volt>] (1.5,0)
      to[C, v=2<\volt>] (3,0); \par
```

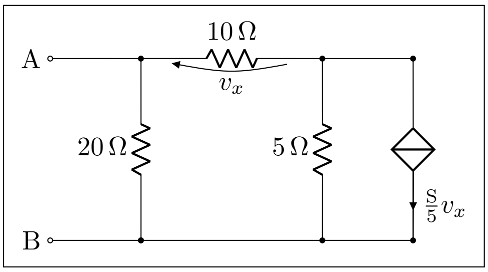
9.2 Components size

Perhaps the most important parameter is `\circuitikzbasekey/bipoles/length`, which can be interpreted as the length of a resistor (including reasonable connections): all other lenghts are relative to this value. For instance:




```
\ctikzset{bipoles/length=1.4cm}
\begin{circuitikz}[scale=1.2]\draw
  (0,0) node[anchor=east] {B}
        to[short, o-*] (1,0)
        to[R=20<\ohm>, *-*] (1,2)
        to[R=10<\ohm>, v=$v_x$] (3,2) -- (4,2)
        to[cI=$\frac{\si{\siemens}}{5} v_x$, *-*] (4,0) -- (3,0)
        to[R=5<\ohm>, *-*] (3,2)
  (3,0) -- (1,0)
  (1,2) to[short, -o] (0,2) node[anchor=east]{A}
;\end{circuitikz}
```

[7] That is, how distant from the initial and final points of the path the arrow starts and ends.
[8] Controlling how high the bump of the arrow is — how curved it is.
[9] Controlling how distant from the bipole the voltage label will be.

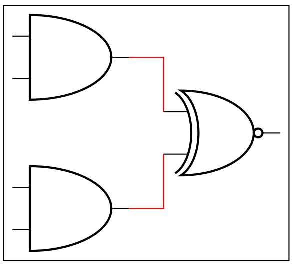



```
\ctikzset{bipoles/length=.8cm}
\begin{circuitikz}[scale=1.2]\draw
 (0,0) node[anchor=east] {B}
      to[short, o-*] (1,0)
      to[R=20<\ohm>, *-*] (1,2)
      to[R=10<\ohm>, v=$v_x$] (3,2) -- (4,2)
      to[cI=$\frac{\siemens}{5} v_x$, *-*] (4,0) -- (3,0)
      to[R=5<\ohm>, *-*] (3,2)
 (3,0) -- (1,0)
 (1,2) to[short, -o] (0,2) node[anchor=east]{A}
;\end{circuitikz}
```

9.3 Colors

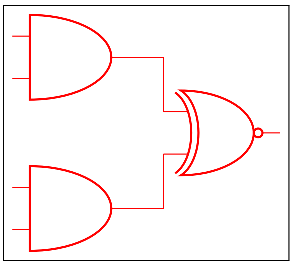
The color of the components is stored in the key `\circuitikzbasekey/color`. CircuiTi*k*Z tries to follow the color set in Ti*k*Z, although sometimes it fails. If you change color in the picture, please do not use just the color name as a style, like `[red]`, but rather assign the style `[color=red]`.

Compare for instance



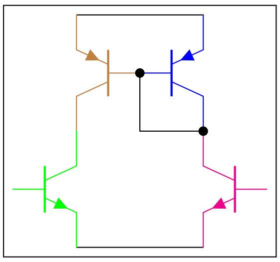
```
\begin{circuitikz} \draw[red]
 (0,2) node[and port] (myand1) {}
 (0,0) node[and port] (myand2) {}
 (2,1) node[xnor port] (myxnor) {}
 (myand1.out) -| (myxnor.in 1)
 (myand2.out) -| (myxnor.in 2)
;\end{circuitikz}
```

and



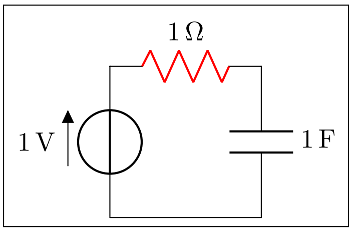
```
\begin{circuitikz} \draw[color=red]
 (0,2) node[and port] (myand1) {}
 (0,0) node[and port] (myand2) {}
 (2,1) node[xnor port] (myxnor) {}
 (myand1.out) -| (myxnor.in 1)
 (myand2.out) -| (myxnor.in 2)
;\end{circuitikz}
```

One can of course change the color *in medias res*:



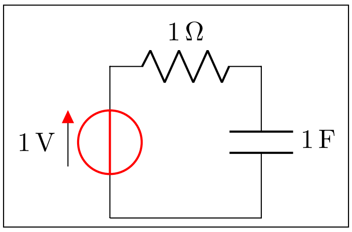
```
\begin{circuitikz} \draw
  (0,0) node[pnp, color=blue] (pnp2) {}
  (pnp2.B) node[pnp, xscale=-1, anchor=B, color=brown] (pnp1) {}
  (pnp1.C) node[npn, anchor=C, color=green] (npn1) {}
  (pnp2.C) node[npn, xscale=-1, anchor=C, color=magenta] (npn2) {}
  (pnp1.E) -- (pnp2.E) (npn1.E) -- (npn2.E)
  (pnp1.B) node[circ] {} |- (pnp2.C) node[circ] {}
;\end{circuitikz}
```

The all-in-one stream of bipoles poses some challanges, as only the actual body of the bipole, and not the connecting lines, will be rendered in the specified color. Also, please notice the curly braces around the `to`:



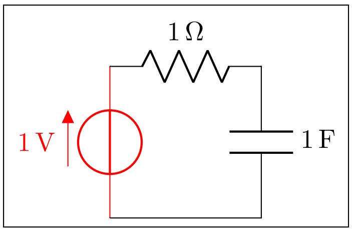

```
\begin{circuitikz} \draw
  (0,0) to[V=1<\volt>] (0,2)
     { to[R=1<\ohm>, color=red] (2,2) }
       to[C=1<\farad>] (2,0) -- (0,0)
;\end{circuitikz}
```

Which, for some bipoles, can be frustrating:




```
\begin{circuitikz} \draw
  (0,0){to[V=1<\volt>, color=red] (0,2) }
       to[R=1<\ohm>] (2,2)
       to[C=1<\farad>] (2,0) -- (0,0)
;\end{circuitikz}
```

The only way out is to specify different paths:




```
\begin{circuitikz} \draw[color=red]
  (0,0) to[V=1<\volt>, color=red] (0,2);
  \draw (0,2) to[R=1<\ohm>] (2,2)
       to[C=1<\farad>] (2,0) -- (0,0)
;\end{circuitikz}
```

And yes: this is a bug and *not* a feature...

10 FAQ

Q: When using `\tikzexternalize` I get the following error:

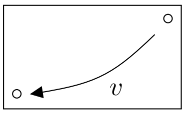
```
! Emergency stop.
```

A: The Ti*k*Z manual states:

> Furthermore, the library assumes that all LaTeX pictures are ended with `\end{tikzpicture}`.

Just substitute every occurrence of the environment `circuitikz` with `tikzpicture`. They are actually pretty much the same.

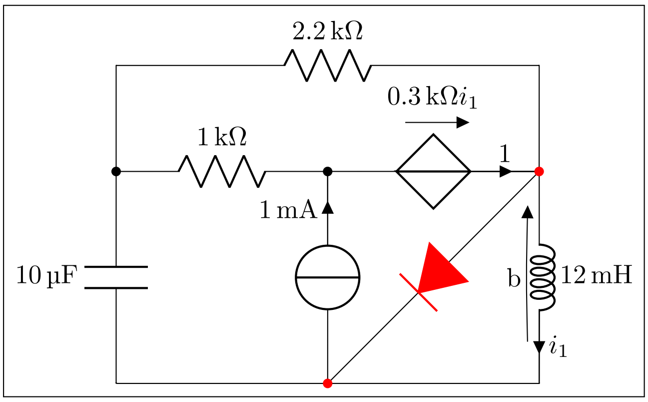
Q: How do I draw the voltage between two nodes?
A: Between any two nodes there is an open circuit!



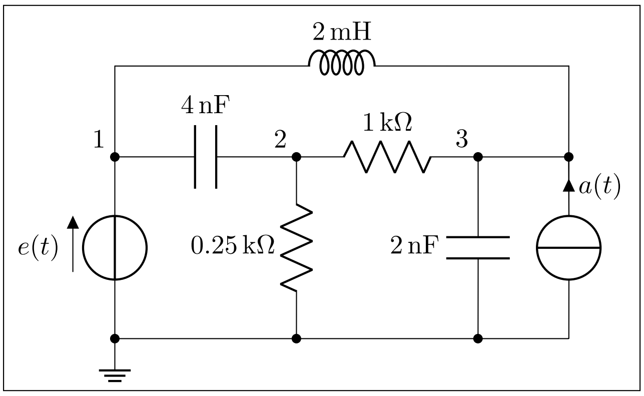

```
\begin{circuitikz} \draw
  node[ocirc] (A) at (0,0) {}
  node[ocirc] (B) at (2,1) {}
  (A) to[open, v=$v$] (B)
;\end{circuitikz}
```

Q: I cannot write `to[R = $R_1=12V$]` nor `to[ospst = open, 3s]`: I get errors.
A: It is a limitation of the parser.
Use `\def{\eq}{=}` `to[R = $R_1\eq 12V$]` and `to[ospst = open{,} 3s]` instead; see caveat in section 7.1.

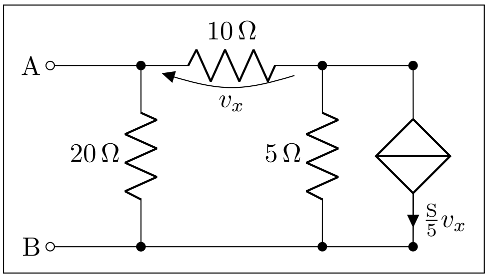
11 Examples



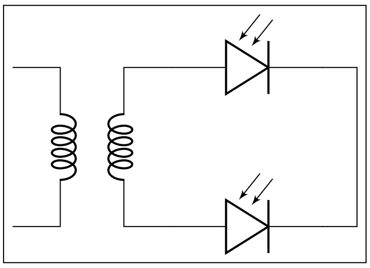

```
\begin{circuitikz}[scale=1.4]\draw
  (0,0) to[C, l=10<\micro\farad>] (0,2) -- (0,3)
        to[R, l=2.2<\kilo\ohm>] (4,3) -- (4,2)
        to[L, l=12<\milli\henry>, i=$i_1$,v=b] (4,0) -- (0,0)
  (4,2) { to[D*, *-*, color=red] (2,0) }
  (0,2) to[R, l=1<\kilo\ohm>, *-] (2,2)
        to[cV, i=1,v=$\SI{.3}{\kilo\ohm} i_1$] (4,2)
  (2,0) to[I, i=1<\milli\ampere>, -*] (2,2)
;\end{circuitikz}
```



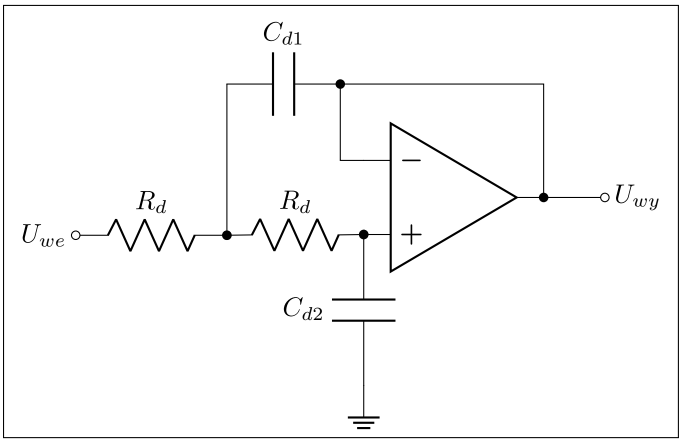

```
\begin{circuitikz}[scale=1.2]\draw
 (0,0) node[ground] {}
      to[V=$e(t)$, *-*] (0,2) to[C=4<\nano\farad>] (2,2)
      to[R, l_=.25<\kilo\ohm>, *-*] (2,0)
 (2,2) to[R=1<\kilo\ohm>] (4,2)
      to[C, l_=2<\nano\farad>, *-*] (4,0)
 (5,0) to[I, i_=$a(t)$, -*] (5,2) -- (4,2)
 (0,0) -- (5,0)
 (0,2) -- (0,3) to[L, l=2<\milli\henry>] (5,3) -- (5,2)

{[anchor=south east] (0,2) node {1} (2,2) node {2} (4,2) node {3}}
;\end{circuitikz}
```



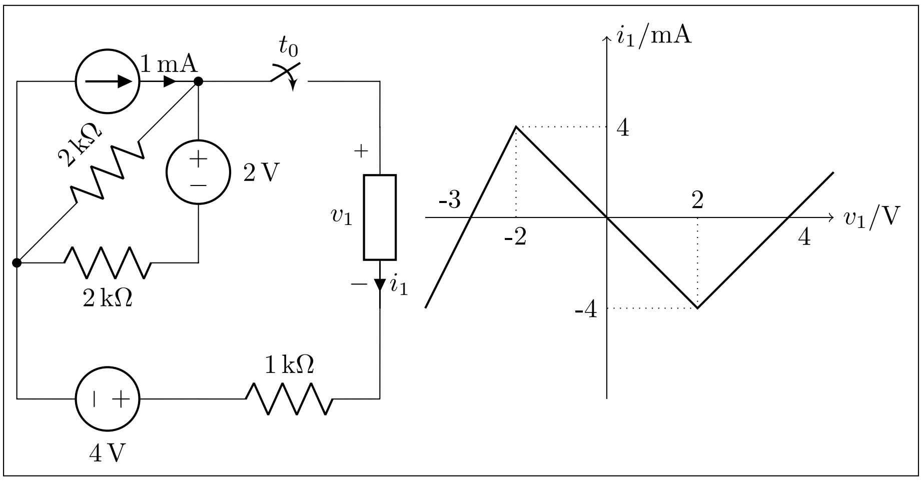

```
\begin{circuitikz}[scale=1.2]\draw
 (0,0) node[anchor=east] {B}
      to[short, o-*] (1,0)
      to[R=20<\ohm>, *-*] (1,2)
      to[R=10<\ohm>, v=$v_x$] (3,2) -- (4,2)
      to[cI=$\frac{\siemens}{5} v_x$, *-*] (4,0) -- (3,0)
      to[R=5<\ohm>, *-*] (3,2)
 (3,0) -- (1,0)
 (1,2) to[short, -o] (0,2) node[anchor=east]{A}
;\end{circuitikz}
```



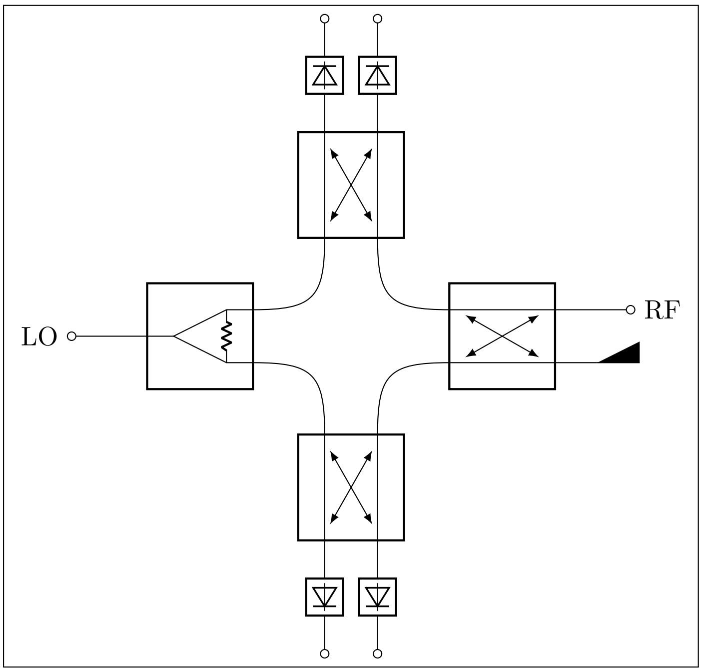
```
\begin{circuitikz}[scale=1]\draw
  (0,0) node[transformer] (T) {}
  (T.B2) to[pD] ($(T.B2)+(2,0)$) -| (3.5, -1)
  (T.B1) to[pD] ($(T.B1)+(2,0)$) -| (3.5, -1)
;\end{circuitikz}
```



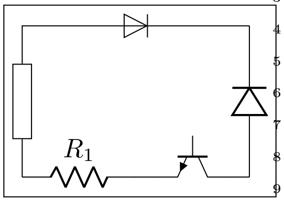

```
\begin{circuitikz}[scale=1]\draw
  (5,.5) node [op amp] (opamp) {}
  (0,0) node [left] {$U_{we}$} to [R, l=$R_d$, o-*] (2,0)
  to [R, l=$R_d$, *-*] (opamp.+)
  to [C, l_=$C_{d2}$, *-] ($(opamp.+)+(0,-2)$) node [ground] {}
  (opamp.out) |- (3.5,2) to [C, l_=$C_{d1}$, *-] (2,2) to [short] (2,0)
  (opamp.-) -| (3.5,2)
  (opamp.out) to [short, *-o] (7,.5) node [right] {$U_{wy}$}
;\end{circuitikz}
```




```
\begin{circuitikz}[scale=1.2, american]\draw
 (0,2) to[I=1<\milli\ampere>] (2,2)
      to[R, l_=2<\kilo\ohm>, *-*] (0,0)
      to[R, l_=2<\kilo\ohm>] (2,0)
      to[V, v_=2<\volt>] (2,2)
      to[cspst, l=$t_0$] (4,2) -- (4,1.5)
      to [generic, i=$i_1$, v=$v_1$] (4,-.5) -- (4,-1.5)
 (0,2) -- (0,-1.5) to[V, v_=4<\volt>] (2,-1.5)
      to [R, l=1<\kilo\ohm>] (4,-1.5);

  \begin{scope}[xshift=6.5cm, yshift=.5cm]
   \draw [->] (-2,0) -- (2.5,0) node[anchor=west] {$v_1/\volt$};
   \draw [->] (0,-2) -- (0,2) node[anchor=west] {$i_1/\SI{}{\milli\ampere}$} ;
   \draw (-1,0) node[anchor=north] {-2} (1,0) node[anchor=south] {2}
        (0,1) node[anchor=west] {4} (0,-1) node[anchor=east] {-4}
        (2,0) node[anchor=north west] {4}
        (-1.5,0) node[anchor=south east] {-3};
   \draw [thick] (-2,-1) -- (-1,1) -- (1,-1) -- (2,0) -- (2.5,.5);
   \draw [dotted] (-1,1) -- (-1,0) (1,-1) -- (1,0)
        (-1,1) -- (0,1) (1,-1) -- (0,-1);
  \end{scope}
\end{circuitikz}
```




```
\begin{circuitikz}[scale=1]
  \ctikzset{bipoles/detector/width=.35}
  \ctikzset{quadpoles/coupler/width=1}
  \ctikzset{quadpoles/coupler/height=1}
  \ctikzset{tripoles/wilkinson/width=1}
  \ctikzset{tripoles/wilkinson/height=1}
  %\draw[help lines,red,thin,dotted] (0,-5) grid (5,5);
  \draw
  (-2,0) node[wilkinson](w1){}
  (2,0) node[coupler] (c1) {}
  (0,2) node[coupler,rotate=90] (c2) {}
  (0,-2) node[coupler,rotate=90] (c3) {}
  (w1.out1) .. controls ++(0.8,0) and ++(0,0.8) .. (c3.3)
  (w1.out2) .. controls ++(0.8,0) and ++(0,-0.8) .. (c2.4)
  (c1.1) .. controls ++(-0.8,0) and ++(0,0.8) .. (c3.2)
  (c1.4) .. controls ++(-0.8,0) and ++(0,-0.8) .. (c2.1)
  (w1.in) to[short,-o] ++(-1,0)
  (w1.in) node[left=30] {LO}
  (c1.2) node[match,yscale=1] {}
  (c1.3) to[short,-o] ++(1,0)
  (c1.3) node[right=30] {RF}
  (c2.3) to[detector,-o] ++(0,1.5)
  (c2.2) to[detector,-o] ++(0,1.5)
  (c3.1) to[detector,-o] ++(0,-1.5)
  (c3.4) to[detector,-o] ++(0,-1.5)
  ;
\end{circuitikz}
```




```
\documentclass{standalone}

\usepackage{tikz}
\usetikzlibrary{circuits.ee.IEC}
\usetikzlibrary{positioning}

\usepackage[compatibility]{circuitikz}
\ctikzset{bipoles/length=.9cm}

\begin{document}
 \begin{tikzpicture}[circuit ee IEC]
  \draw (0,0) to [resistor={name=R}] (0,2)
   to[diode={name=D}] (3,2);
  \draw (0,0) to[*R=$R_1$] (1.5,0) to[*Tnpn] (3,0)
    to[*D](3,2);
 \end{tikzpicture}
\end{document}
```

12 Changelog

The major changes among the different circuitikz versions are listed here. See `https://github.com/circuitikz/circuitikz/commits` for a full list of changes.

- Version git (unreleased)
  - Fixed placement of straightlabels within 4th quadrant
  - Fixed straightvoltages at Diodes, varcap and some other components
  - Adjusted ground symbols to better match ISO standard
  - Fixed a bug about straightlabels (thanks to @fotesan)
  - Added Romano as contributor
  - Added a CONTRIBUTING file
  - Added new sources (cute european versions, noise sources)
  - Added new types of amplifiers, and option to flip inputs and outputs
  - Added bidirectional diodes (diac) thanks to Andre Lucas Chinazzo
  - Added L,R,C sensors (with european, american and cute variants)
  - Added stacked labels (thanks to the original work by Claudio Fiandrino)
  - Make the position of voltage symbols adjustable
  - Make the position of arrows in FETs and BJTs adjustable
  - Added the bulb symbol
  - Added options for solving the voltage direction problems.
  - Added chips (DIP, QFP) with a generic number of pins.
  - Added special anchors for transformers (and fixed the wrong center anchor)
  - Changed the logical port implementation to multiple inputs (thanks to John Kormylo)
  - Changed labels spacing so that they are independent on scale factor
- Version 0.8.3 (2017-05-28)
  - Removed unwanted lines at to-paths if the starting point is a node without a explicit anchor.

- Fixed scaling option, now all parts are scaled by bipoles/length
- Surge arrester appears no more if a to path is used without []-options
- Fixed current placement now possible with paths at an angle of around 280°
- Fixed voltage placement now possible with paths at an angle of around 280°
- Fixed label and annotation placement (at some angles position not changable)
- Adjustable default distance for straight-voltages: 'bipoles/voltage/straight label distance'
- Added Symbol for bandstop filter
- New annotation type to show flows using f=... like currents, can be used for thermal, power or current flows

- Version 0.8.2 (2017-05-01)
  - Fixes pgfkeys error using alternatively specified mixed colors(see pgfplots manual section "4.7.5 Colors")
  - Added new switches "ncs" and "nos"
  - Reworked arrows at spst-switches
  - Fixed direction of controlled american voltage source
  - "v<=" and "i<=" do not rotate the sources anymore(see them as "counting direction indication", this can be different then the shape orientation); Use the option "invert" to change the direction of the source/apperance of the shape.
  - current label "i=" can now be used independent of the regular label "l=" at current sources
  - rewrite of current arrow placement. Current arrows can now also be rotated on zero-length paths
  - New DIN/EN compliant operational amplifier symbol "en amp"
- Version 0.8.1 (2017-03-25)
  - Fixed unwanted line through components if target coordinate is a name of a node
  - Fixed position of labels with subscript letters.
  - Absolute distance calculation in terms of ex at rotated labels
  - Fixed label for transistor paths (no label drawn)
- Version 0.8 (2017-03-08)
  - Allow use of voltage label at a [short]
  - Correct line joins between path components (to[...])
  - New Pole-shape .-. to fill perpendicular joins
  - Fixed direction of controlled american current source
  - Fixed incorrect scaling of magnetron
  - Fixed: Number of american inductor coils not adjustable
  - Fixed Battery Symbols and added new battery2 symbol
  - Added non-inverting Schmitttrigger
- Version 0.7 (2016-09-08)
  - Added second annotation label, showing, e.g., the value of an component

  - Added new symbol: magnetron
  - Fixed name conflict of diamond shape with tikz.shapes package
  - Fixed varcap symbol at small scalings
  - New packet-option ”straightvoltages, to draw straight(no curved) voltage arrows
  - New option “invert” to revert the node direction at paths
  - Fixed american voltage label at special sources and battery
  - Fixed/rotated battery symbol(longer lines by default positive voltage)
  - New symbol Schmitttrigger

- Version 0.6 (2016-06-06)

  - Added Mechanical Symbols (damper,mass,spring)
  - Added new connection style diamond, use (d-d)
  - Added new sources voosource and ioosource (double zero-style)
  - All diode can now drawn in a stroked way, just use globel option “strokediode” or stroke instead of full/empty, or D-. Use this option for compliance with DIN standard EN-60617
  - Improved Shape of Diodes:tunnel diode, Zener diode, schottky diode (bit longer lines at cathode)
  - Reworked igbt: New anchors G,gate and new L-shaped form Lnigbt, Lpigbt
  - Improved shape of all fet-transistors and mirrored p-chan fets as default, as pnp, pmos, pfet are already. This means a backward-incompatibility, but smaller code, because p-channels mosfet are by default in the correct direction(source at top). Just remove the ‘yscale=-1’ from your p-chan fets at old pictures.

- Version 0.5 (2016-04-24)

  - new option boxed and dashed for hf-symbols
  - new option solderdot to enable/disable solderdot at source port of some fets
  - new parts: photovoltaic source, piezo crystal, electrolytic capacitor, electromechanical device(motor, generator)
  - corrected voltage and current direction(option to use old behaviour)
  - option to show body diode at fet transistors

- Version 0.4

  - minor improvements to documentation
  - comply with TDS
  - merge high frequency symbols by Stefan Erhardt
  - added switch (not opening nor closing)
  - added solder dot in some transistors
  - improved ConTeXt compatibility

- Version 0.3.1

  - different management of color…
  - fixed typo in documentation
  - fixed an error in the angle computation in voltage and current routines

- fixed problem with label size when scaling a tikz picture
- added gas filled surge arrester
- added compatibility option to work with Tikz's own circuit library
- fixed infinite in arctan computation

- Version 0.3.0
  - fixed gate node for a few transistors
  - added mixer
  - added fully differential op amp (by Kristofer M. Monisit)
  - now general settings for the drawing of voltage can be overridden for specific components
  - made arrows more homogeneous (either the current one, or latex' bt pgf)
  - added the single battery cell
  - added fuse and asymmetric fuse
  - added toggle switch
  - added varistor, photoresistor, thermocouple, push button
  - added thermistor, thermistor ptc, thermistor ptc
  - fixed misalignment of voltage label in vertical bipoles with names
  - added isfet
  - added noiseless, protective, chassis, signal and reference grounds (Luigi «Liverpool»)

- Version 0.2.4
  - added square voltage source (contributed by Alistair Kwan)
  - added buffer and plain amplifier (contributed by Danilo Piazzalunga)
  - added squid and barrier (contributed by Cor Molenaar)
  - added antenna and transmission line symbols contributed by Leonardo Azzinnari
  - added the changeover switch spdt (suggestion of Fabio Maria Antoniali)
  - rename of context.tex and context.pdf (thanks to Karl Berry)
  - updated the email address
  - in documentation, fixed wrong (non-standard) labelling of the axis in an example (thanks to prof. Claudio Beccaria)
  - fixed scaling inconsistencies in quadrupoles
  - fixed division by zero error on certain vertical paths
  - introduced options straighlabels, rotatelabels, smartlabels

- Version 0.2.3
  - fixed compatibility problem with label option from tikz
  - Fixed resizing problem for shape ground
  - Variable capacitor
  - polarized capacitor
  - ConTeXt support (read the manual!)
  - nfet, nigfete, nigfetd, pfet, pigfete, pigfetd (contribution of Clemens Helfmeier and Theodor Borsche)

- njfet, pjfet (contribution of Danilo Piazzalunga)
- pigbt, nigbt
- *backward incompatibility* potentiometer is now the standard resistor-with-arrow-in-the-middle; the old potentiometer is now known as variable resistor (or vR), similarly to variable inductor and variable capacitor
- triac, thyristor, memristor
- new property “name” for bipoles
- fixed voltage problem for batteries in american voltage mode
- european logic gates
- *backward incompatibility* new american standard inductor. Old american inductor now called “cute inductor”
- *backward incompatibility* transformer now linked with the chosen type of inductor, and version with core, too. Similarly for variable inductor
- *backward incompatibility* styles for selecting shape variants now end are in the plural to avoid conflict with paths
- new placing option for some tripoles (mostly transistors)
- mirror path style

* Version 0.2.2 - 20090520
  - Added the shape for lamps.
  - Added options `europeanresistor`, `europeaninductor`, `americanresistor` and `americaninductor`, with corresponding styles.
  - FIXED: error in transistor arrow positioning and direction under negative `xscale` and `yscale`.

* Version 0.2.1 - 20090503
  - Op-amps added
  - added options arrowmos and noarrowmos, to add arrows to pmos and nmos

* Version 0.2 - 20090417 First public release on CTAN
  - *Backward incompatibility*: labels ending with `:`*angle* are not parsed for positioning anymore.
  - Full use of Ti*k*Z keyval features.
  - White background is not filled anymore: now the network can be drawn on a background picture as well.
  - Several new components added (logical ports, transistors, double bipoles, …).
  - Color support.
  - Integration with {`siunitx}.`
  - `Voltage, american style.`
  - `Better code, perhaps. General cleanup at the very least.`

* Version 0.1 - 2007-10-29 First public release

Index of the components