

---

# Spice Documentation

*Release 1.7\_RC*

Oct 01, 2021

## Contents:

1	Spice IO-file	7
2	Spice Simulation Command	10
3	Spice DC Source	12
4	Spice Testbench	13
5	Spice Module	15
6	Indices and tables	15
	Python Module Index	16
	Index	17

---

Analog simulation interface package for TheSyDeKick.

Provides utilities to import spice-like modules to Python environment and generate testbenches for the various simulation cases.

Initially written by Okko Järvinen, 2019

Release 1.6, Jun 2020 supports Eldo and Spectre

### **class** `spice.spice`

Adding this class as a superclass enforces the definitions for Spice simulations in the subclasses.

#### **connect\_spice\_inputs()**

Automatically called function to connect iofiles (inputs) to top entity IOS Bundle items.

#### **connect\_spice\_outputs()**

Automatically called function to connect iofiles (outputs) to top entity IOS Bundle items.

#### **property** `dcsource_bundle`

Bundle

A thesdk.Bundle containing *spice\_dcsource* objects. The *spice\_dcsource* objects are automatically added to this Bundle, nothing should be manually added.

**property distributed\_run**

True | False (default)

If True, distributes applicable simulations (currently DC sweep supported) into the LSF cluster. The number of subprocesses launched is set by self.num\_processes.

**property dspf**

List of str

List containing filenames for DSPF-files to be included for post-layout simulations. The names given in this list are matched to dspf-files in './spice/' -directory. A postfix '.pex.dspf' is automatically appended to the given names (this will probably change later).

Example:

```
self.dspf = ['inv_v2', 'switch_v3']
```

would include files './spice/inv\_v2.pex.dspf' and './spice/switch\_v3.pex.dspf' as dspf-files in the testbench. If the dspf-file contains definition matching the original design name of the top-level netlist, it gets also renamed to match the module name (dspf-file for top-level instance is possible).

**property eldochisrc**

String

Path to the Eldo chi-file. ('./Simulations/spicesim/<runname>/tb\_entityname.chi'). Only applies to Eldo simulations. This shouldn't be set manually.

**property entitypath**

String

Path to the entity root.

**property errpreset**

String

Global accuracy parameter for Spectre simulations. Options include 'liberal', 'moderate' and 'conservative', in order of rising accuracy.

**execute\_spice\_sim()**

Automatically called function to execute spice simulation.

**extract\_powers()**

Automatically called function to extract transient power and current consumptions. The consumptions are extracted for spice\_dcsource objects with the attribute extract=True.

The extracted consumptions are accessible on the top-level after simulation as:

```
# Dictionary with averaged power of each supply + total
self.extracts.Members['powers']
# Dictionary with averaged current of each supply + total
self.extracts.Members['currents']
# Dictionary with transient current of each supply
self.extracts.Members['curr_tran']
```

The keys in the aforementioned dictionaries match the *name*-fields of the respective *spice\_dcsource* objects.

**property extracts**

Bundle

A thesdb.Bundle containing extracted quantities.

**get\_buswidth(*signame*)**

Extract buswidth from signal name.

Little-endian example:

```
start,stop,width,busrange = get_buswidth('BUS<10:0>')
# start = 10
# stop = 0
# width = 11
# busrange = range(10,-1,-1)
```

Big-endian example:

```
start,stop,width,busrange = get_buswidth('BUS<0:8>')
# start = 0
# stop = 8
# width = 9
# busrange = range(0,9)
```

**property has\_lsf**

True | False (default)

True if LSF submissions are properly defined.

**property interactive\_spice**

True | False (default)

Launch simulator in interactive mode. A waveform viewer (ezwave by default) is opened during the simulation for debugging. See *plotprogram* for selecting waveform viewer program.

**property iofile\_bundle**

Bundle

A thesdb.Bundle containing *spice\_iofile* objects. The *spice\_iofile* objects are automatically added to this Bundle, nothing should be manually added.

**property iofile\_eventdict**

Dictionary

Dictionary to store event type output from the simulations. This should speed up reading the results.

**property load\_state**

String (default '')

Feature for loading results of previous simulation. The Spice simulation is not re-executed, but the outputs will be read from existing files. The string value should be the *runname* of the desired simulation.

Loading the most recent result automatically:

```
self.load_state = 'last'
# or
self.load_state = 'latest'
```

Loading a specific past result using the *runname*:

```
self.load_state = '20201002103638_tmpdbw11nr4'
```

List available results by providing any non-existent *runname*:

```
self.load_state = 'this_does_not_exist'
```

**property name**

String

Name of the module.

**property nproc**

Integer

Requested maximum number of threads for multithreaded simulations. For Eldo, maps to command line parameter '-nproc'. For Spectre, maps to command line parameter '+mt'.

**property num\_processes**

Integer

Maximum number of spawned child processes for distributed runs.

**property plotlist**

List of str

List of net names to be saved in the waveform database.

---

**Note:** Obsolete! Moved to *spice\_simcmd* as a keyword argument.

---

**property plotprogcmd**

String

Sets the command to be run for interactive simulations.

**property plotprogram**

String

Sets the program to be used for visualizing waveform databases. Options are ezwave (default) or viva.

**property preserve\_iofiles**

True | False (default)

If True, do not delete file IO files after simulations. Useful for debugging the file IO.

---

**Note:** Replaced by *preserve\_result* in v1.7

---

**property preserve\_result**

True | False (default)

If True, do not delete result files after simulations.

**property preserve\_spicefiles**

True | False (default)

If True, do not delete generated Spice files (testbench, subcircuit, etc.) after simulations. Useful for debugging.

---

**Note:** Replaced by *preserve\_result* in v1.7

---

**read\_oppts()**

Internally called function to read the DC operating points of the circuit TODO: Implement for Eldo as well.

**read\_spice\_outputs()**

Automatically called function to call read() functions of each iofile with direction 'output'.

**run\_plotprogram()**

Starting a parallel process for waveform viewer program.

The plotting program command can be set with 'plotprogram'. Tested for spectre and eldo.

**run\_spice()**

Externally called function to execute spice simulation.

**property runname**

String

Automatically generated name for the simulation.

Formatted as timestamp\_randomtag, i.e. '20201002103638\_tmpdbw11nr4'. Can be overridden by assigning self.runname = 'myname'.

Example:

```
self.runname = 'test'
```

would generate the simulation files in *Simulations/spicesim/test/*.

**property si\_prefix\_mult**

Dictionary

Dictionary mapping SI-prefixes to multipliers.

**si\_string\_to\_float(strval)**

Convert SI-formatted string to float

E.g. self.si\_string\_to\_float('3 mV') returns 3e-3.

**property simcmd\_bundle**

Bundle

A thesdk.Bundle containing *spice\_simcmd* objects. The *spice\_simcmd* objects are automatically added to this Bundle, nothing should be manually added.

**property spice\_submission**

String

Defines spice submission prefix from thesdk.GLOBALS['LSFSUBMISSION'] and thesdk.GLOBALS['LSFINTERACTIVE'] for LSF submissions.

Usually something like 'bsub -K' or 'bsub -I'.

**property spicecmd**

String

Simulation command string to be executed on the command line. Automatically generated.

**property spicecorner**

Dictionary

Feature for specifying the 'section' of the model library file and simulation temperature. The path to model libraries should be set in TheSDK.config as either ELDOLIBFILE, SPECTRELIBFILE or NGSPICELIBFILE variable.

Example:

```
self.spicecorner = {
    'temp': 27,
    'corner': 'top_tt'
}
```

#### property spicedbpath

String

Path to output waveform database. (./Simulations/spicesim/<runname>/tb\_<entityname>.<resultfile\_ext>)  
For now only for spectre. This shouldn't be set manually.

#### property spicemisc

List of str

List of manual commands to be pasted to the testbench. The strings are pasted to their own lines (no linebreaks needed), and the syntax is unchanged.

For example, setting initial voltages from testbench (Eldo):

```
for i in range(nodes):
    self.spicemisc.append('.ic NODE<%d> 0' % i)
```

The same example can be done in Spectre with:

```
self.spicemisc.append('simulator lang=spice')
for i in range(nodes):
    self.spicemisc.append('.ic NODE<%d> 0' % i)
self.spicemisc.append('simulator lang=spectre')
```

#### property spiceoptions

Dictionary

Feature for specifying options for spice simulation. The key is the name of the option (as in simulator manual specifies), and the value is the value given to said option. Valid key-value pairs can be found from the manual of the simulator (Eldo, Spectre or Ngspice).

Example:

```
self.spiceoptions = {
    'save': 'lvlpub',
    'nestlvl': '1',
    'pwr': 'subckts',
    'digits': '12'
}
```

#### property spiceparameters

Dictionary

Feature for specifying simulation parameters for spice simulation. The key is the name of the parameter , and the value is the value given to said parameter.

Example:

```
self.spiceparameters = {
    'nf_pmos': 8,
    'nf_nmos': 4,
    'ibias': 100e-6
}
```

**property spicesimpath**

String

Simulation path. (./Simulations/spicesim/&lt;runname&gt;) This shouldn't be set manually.

**property spicesrc**

String

Path to the source netlist (i.e. 'spice/entityname.scs'). This shouldn't be set manually.

---

**Note:** Provided netlist name has to match entity name (entityname.scs or entityname.cir).

---

---

**Note:** Netlist has to contain the top-level design as a subcircuit definition.

---

**property spicesrcpath**

String

Path to the spice source of the entity ('./spice').

**property spicesubcktsrc**

String

Path to the parsed subcircuit file. ('./Simulations/spicesim/&lt;runname&gt;/subckt\_entityname.&lt;suffix&gt;'). This shouldn't be set manually.

**property spicetbsrc**

String

Path to the spice testbench ('./Simulations/spicesim/&lt;runname&gt;/tb\_entityname.&lt;suffix&gt;'). This shouldn't be set manually.

**property syntaxdict**

Dictionary

Internally used dictionary for common syntax conversions between Spectre, Eldo, and Ngspice.

**write\_spice\_inputs()**

Automatically called function to call write() functions of each iofile with direction 'input'.

## 1 Spice IO-file

Provides spice file IO related attributes and methods for TheSDK spice.

Initially written by Okko Järvinen, 2020

**class** spice.spice\_iofile.spice\_iofile(*parent=None, \*\*kwargs*)

Class to provide file IO for spice simulations. When created, adds a spice\_iofile object to the parents iofile\_bundle property. Accessible as iofile\_bundle.Members['name'].

**Parameters**

- **parent** (*object*) – The parent object initializing the spice\_iofile instance. Default None
- **name** (*str*) – Name of the IO.
- **dir** ('in' or 'out') – Direction of the IO.

- **iotype** ('event', 'sample' or 'time') – Type of the IO signal. Event type signals are time-value pairs (analog signal), whereas sample type signals are sampled by a clock signal (digital bus). Sample type signals can be used for discrete time & continuous amplitude outputs (sampled voltage for example), by setting iotype='sample' and ioformat='volt'. Time type signals return a vector of timestamps corresponding to threshold crossings.
- **ioformat** ('dec', 'bin' or 'volt') – Formatting of the sampled signals. Digital output buses are formatted to unsigned integers when ioformat = 'dec'. For 'bin', the digital output bus is returned as a string containing ones and zeros. When ioformat = 'volt', the output signal is sampled at the clock and the floating point value is returned. Voltage sampling is only supported for non-bus signals.
- **sourcetype** ('V', 'I' or 'ISUB') – Type of the source associated to a file. Default 'V'.
- **datatype** (str) – Inherited from the parent. If complex, the ioname is handled as a complex signal. Currently implemented only for writing the outputs in testbenches and reading them in.
- **trigger** (str or list<str>) – Name of the clock signal node in the Spice netlist. If a single string is given, the same clock signal is used for all bits/buses. If a list is given, and the length matches ionames list length, each ioname will be assigned its own clock. Applies only to sample type outputs.
- **vth** (float) – Threshold voltage of the trigger signal and the bit rounding. Applies only to sample type outputs.
- **edgetype** ('rising', 'falling' or 'both') – Type of triggering edge. When time type signal is used, the edgetype values can define the extraction type as 'risetime' or 'falltime' additionally. Default 'rising'.
- **after** (float) – Initial delay added to the input signal (sample) or time extraction (time). Useful for ignoring initial settling, for example. Applies only to sample and time outputs. Default 0.
- **big\_endian** (bool) – Flag to read the extracted bus as big-endian. Applies only to sample type outputs. Default False.
- **rs** (float) – Sample rate of the sample type input. Default None.
- **vhi** (float) – High bit value of sample type input. Default 1.0.
- **vlo** (float) – Low bit value of sample type input. Default 0.
- **tfall** (float) – Falltime of sample type input. Default 5e-12.
- **trise** (float) – Risetime of sample type input. Default 5e-12.

## Examples

Initiated in parent as:

```
_=spice_iofile(self,name='foobar')
```

Defining analog input voltage signals from python to spice:

```
_=spice_iofile(self,name='inp',dir='in',iotype='event',sourcetype='V',ionames='INP')
_=spice_iofile(self,name='inn',dir='in',iotype='event',sourcetype='V',ionames='INN')
```

Defining time-domain output signal containing rising edge threshold crossing timestamps of an analog clock signal:



```
_=spice_iofile(self,name='clk_rise',dir='out',iotype='time',sourcetype='V',
               ionames='CLK',edgetype='rising',vth=self.vdd/2)
```

Defining digital output signal triggered with a falling edge of the analog clock:

```
_=spice_iofile(self,name='dout',dir='out',iotype='sample',sourcetype='V',
               ioformat='bin',ionames='DOUT<7:0>',edgetype='falling',
               vth=self.vdd/2,trigger='CLK')
```

Defining a discrete time & continuous amplitude output signal triggered with a rising edge of the analog clock. The iofile returns a 2D-vector similar to ‘event’ type signals:

```
_=spice_iofile(self,name='sampled_input',dir='out',iotype='sample',sourcetype='V',
               ioformat='volt',ionames='INP',edgetype='rising',
               vth=self.vdd/2,trigger='CLK')
```

Defining digital input signal with decimal format. The input vector is a list of integers, which get converted to binary bus of 4-bits (inferred from ‘CTRL<3:0>’). The values are changed at 1 MHz interval in this example.:

```
_=spice_iofile(self,name='ctrl',dir='in',iotype='sample',ionames='CTRL<3:0>',rs=1e6,
               vhi=self.vdd,trise=5e-12,tfall=5e-12,ioformat='dec')
```

**append\_to\_data**(*arr=None, bits=False, buswidth=None*)

Helper method to append array to self.Data.

The array(s) are padded with np.nan when bits=False, and ‘UUUU’ when bits=True. This is called automatically for time and sample type IOs.

**property file**

List<str>

List containing filepaths to files associated with this spice\_iofile. For digital buses or arrays of signals, the list contains multiple files which are automatically handled together. These filepaths are set automatically.

**interp\_crossings**(*data, vth, nint, edgetype*)

Helper method called for ‘time’ and ‘sample’ type outputs.

Interpolates the requested threshold crossings (rising or falling) from the ‘event’ type input signal. Returns the time-stamps of the crossing instants in a 1D-vector.

**Parameters**

- **data** (*ndarray*) – Input data array. Expected an ‘event’ type 2D-vector where first column is time and second is voltage.
- **vth** (*float*) – Threshold voltage.
- **nint** (*int*) – Interpolation factor. The two closest points on each side of a threshold crossing are used for linear interpolation endpoints, where nint points are added to find as close x-value of the threshold crossing as possible.
- **edgetype** (*str*) – Direction of the crossing: ‘rising’, ‘falling’ or ‘both’.

**Returns** 1D-vector with time-stamps of interpolated threshold crossings.

**Return type** ndarray

**property ionames**

List<str>

Set by argument ‘ionames’. This property casts the given argument to a list if needed.

**parse\_io\_from\_file**(*filepath, start, stop, dtype, label, queue*)

Parse specific lines from a spectre print file.

This is wrapped to a function to allow parallelism.

**read**(\*\**kwargs*)

Function to read files associated with this spice\_iofile.

**sample\_signal**(*signal, trigger, nint=1*)

Helper method called for 'sample' type outputs.

Finds the signal y-values at time instants defined by the clock signal (trigger).

#### Parameters

- **data** (*ndarray*) – Input data array. Expected an 'event' type 2D-vector where first column is time and second is voltage.
- **vth** (*float*) – Threshold voltage.
- **nint** (*int*) – Interpolation factor. The two closest points on each side of a threshold crossing are used for linear interpolation endpoints, where nint points are added to find as close x-value of the threshold crossing as possible.
- **edgetype** (*str*) – Direction of the crossing: 'rising', 'falling' or 'both'.

**Returns** 1D-vector with time-stamps of interpolated threshold crossings.

**Return type** ndarray

**write**(\*\**kwargs*)

Function to write files associated with this spice\_iofile.

## 2 Spice Simulation Command

Class for spice simulation commands.

Initially written by Okko Järvinen, 9.1.2020

**class** spice.spice\_simcmd.**spice\_simcmd**(*parent, \*\*kwargs*)

Class to provide simulation command parameters to spice testbench. When instantiated in the parent class, this class automatically attaches spice\_simcmd objects to simcmd\_bundle -bundle in testbench.

#### Parameters

- **parent** (*object*) – The parent object initializing the spice\_simcmd instance. Default None.
- **sim** ('tran' or 'dc') – Simulation type.
- **plotlist** (*list(str)*) – List of node names or operating points to be plotted. Node names follow simulator syntax. For Eldo, the voltage/current specifier is expected:

```
self.plotlist = ['v(OUT)', 'v(CLK)']
```

For Spectre, the node name is enough:

```
self.plotlist = ['OUT', 'CLK']
```

---

**Note:** Below applies only for Spectre!

---

When capturing operating point information with Spectre, adding the instance name to plotlist saves all operating points for the device, e.g:

```
self.plotlist = [XTB_NAME.XSUBCKT/M0]
```

saves all operating points defined by the model for the device M0 in subckt XSUBCKT.

Wildcards are supported, but should be used with caution as the output file can quickly become excessively large. For example to capture the gm of all transistor use:

```
self.plotlist = [*:gm]
```

It is highly recommended to exclude the devices that are not needed from the output to reduce file size. Examples of such devices are RC-parasitics (include option 'savefilter=rc' in self.spiceoptions to exclude them) and dummy transistors. See exclude\_list below.

- **excludelist** (*list(str)*) – Applies for Spectre only! List of device names NOT to be included in the output report. Wildcards are supported. Exclude list is especially useful for DC simulations when specifying outputs with wildcards.

For example, when capturing gm for all transistors, use exclude list to exclude all dummy transistors with:

```
self.excludelist = [XTB_NAME*DUMMY_ID*],
```

where DUMMY\_ID is extraction tool / runset specific dummy identifier.

- **sweep** (*str*) – DC & Spectre models only. If given, sweeps the top-level parameter given as value. For example:

```
_spice_simcmd(sim='dc', sweep='temp', swpstart=27, swpstop=87)
```

sweeps the top-level parameter temp (temperature) from 27 to 87 at 10 degree increments.

- **subcktname** (*str*) – If given, sweeps the parameter defined by property sweep from the subcircuit given by this property. For example:

```
_spice_simcmd(sim='dc', sweep='Vb', subcktname='XSUBCKT',  
               swpstart=0.1, swpstop=1.5, step=0.05)
```

sweeps the Vb parameter from subcircuit XSUBCKT from 0.1 volts to 1.5 volts with 0.05 volt increments.

- **devname** (*str*) – If given, sweeps the parameter defined by property sweep from the device given by this property. For example:

```
_spice_simcmd(sim='dc', sweep='w', deviceswp='XSUBCKT.XNMOS',  
               swpstart=10u, swpstop=14u, step=0.1u)
```

sweeps the width of transistor XNMOS of subckt XSUBCKT from 10u to 14u in 0.1u increments.

- **swpstart** (*int, float or str*) – Starting point of DC sweep. Default 0.
- **swpstop** (*int, float or str*) – Stop point of DC sweep. Default 0.
- **swpstep** (*int, float or str*) – Step size of the sweep simulation. Default 10.
- **tprint** (*float or str*) – Print interval. Default '1p' or 1e-12.

- **tstop** (*float* or *str*) – Transient simulation duration. When not defined, the simulation time is the duration of the longest input signal.
- **uic** (*bool*) – Use initial conditions flag. Default False.
- **noise** (*bool*) – Noise transient flag. Default False.
- **fmin** (*float* or *str*) – Minimum noise frequency. Default 1 (Hz).
- **fmax** (*float* or *str*) – Maximum noise frequency. Default 5e9.
- **fscale** ('log' or 'lin') – Logarithmic or linear scale for frequency. Default 'log'.
- **fpoints** (*int*) – Number of points for frequency analysis. Default 0.
- **fstepsize** (*int*) – Step size for AC analysis, if scale if 'lin'. If fscale is 'log', this parameter gives number of points per decade. Default 0.
- **seed** (*int*) – Random generator seed for noise transient. Default None (random).
- **method** (*str*) – Transient integration method. Default None (Spectre takes method from errpreset).
- **cmin** (*float*) – Spectre cmin parameter: this much cap from each node to ground. Might speed up simulation. Default None (not used).
- **mc** (*bool*) – Enable Monte Carlo simulation. This flag will enable Monte Carlo modeling for a single simulation. It will NOT execute multiple runs or do any statistical analysis. Intended use case is to generate a group of entities in TheSyDeKick with each having mc=True, simulating them in parallel (see run\_parallel() of thesdk-class), post-processing results in TheSyDeKick.
- **mc\_seed** (*int*) – Random seed for the Monte Carlo instance. Default None (random seed).

## Examples

Initiated in parent as:

```
_=spice_simcmd(self,sim='tran',tprint=1e-12,tstop='10n',
               uic=True,noise=True,fmin=1,fmax=5e9)
```

For a simple transient with inferred simulation duration:

```
_=spice_simcmd(self,sim='tran')
```

## 3 Spice DC Source

Class for generating DC voltage/current sources in the Spectre or Eldo testbench.

Initially written by Okko Järvinen, 9.1.2020

**class** spice.spice\_dcsource.spice\_dcsource(*parent*, *\*\*kwargs*)

Class to provide DC source definitions to spice testbench. When instantiated in the parent class, this class automatically attaches spice\_dcsource objects to dsource\_bundle -bundle in testbench.

### Parameters

- **parent** (*object*) – The parent object initializing the spice\_dcsource instance. Default None.

- **name** (*str*) – Name of the source.
- **value** (*float*) – Value of the source.
- **sourcetype** ('V' or 'I') – Type of the DC source. Either 'V' for voltage or 'I' for current.
- **pos** (*str*) – Name of the positive net in the netlist.
- **neg** (*str*) – Name of the negative net in the netlist.
- **extract** (*bool*) – Flag the source for transient current and power consumption extraction. Extracted currents and powers are accessible through dictionaries `self.currents` and `self.powers` in the parent object. Default False.
- **ext\_start** (*float*) – Time to start extracting average transient power consumption. Default is 0.
- **ext\_stop** (*float*) – Time to stop extracting average transient power consumption. Default is simulation end time.
- **noise** (*bool*) – Enable the noise contribution of this source (only when transient noise is enabled). Default is True.
- **ramp** (*float*) – Ramp up the source from 0 to value in ramp seconds. Default is 0 (no ramping).

## Examples

A voltage source connected between circuit nodes 'VDD' and 'VSS', for which power and current consumptions are extracted in transient simulation. Initiated in parent as:

```
_=spice_dcsource(self,name='supply',value=1.0,extract=True,pos='VDD',neg='VSS')
_=spice_dcsource(self,name='ground',value=0,pos='VSS',neg='0') # Ground 'source'
```

A bias current flowing from 'VDD' to node 'IBIAS':

```
_=spice_dcsource(self,name='bias',sourcetype='I',value=25e-6,pos='VDD',neg='IBIAS')
```

### property ext\_file

String

Optional filepath for extracted transient current when `self.extract=True`.

## 4 Spice Testbench

Testbench generation class for spice simulations.

**class** `spice.testbench.testbench`(*parent=None, \*\*kwargs*)

This class generates all testbench contents. This class is utilized by the main spice class.

### property dcsourcestr

String

DC source definitions parsed from `spice_dcsource` objects instantiated in the parent entity.

### property dspfincludcmd

String

DSPF-file inclusion string pointing to files corresponding to `self.dspf` in the parent entity.

**esc\_bus**(*name, esc\_colon=True*)

Helper function to escape bus characters for Spectre simulations bus<3:0> -> bus<3:0>.

**export**(*\*\*kwargs*)

Internally called function to write the testbench to a file.

**export\_subckt**(*\*\*kwargs*)

Internally called function to write the parsed subcircuit definitions to a file.

**property file**

String

Filepath to the testbench file (i.e. './spice/tb\_entityname.scs').

**generate\_contents**()

Internally called function to generate testbench contents.

**property includecmd**

String

Subcircuit inclusion string pointing to generated subckt\_\* -file.

**property inputsignals**

String

Input signal definitions parsed from spice\_iofile objects instantiated in the parent entity.

**property libcmd**

String

Library inclusion string. Parsed from self.spicecorner -dictionary in the parent entity, as well as 'EL-DOLIBFILE' or 'SPECTRELIBFILE' global variables in TheSDK.config.

**property misccmd**

String

Miscellaneous command string corresponding to self.spicemisc -list in the parent entity.

**property options**

String

Spice options string parsed from self.spiceoptions -dictionary in the parent entity.

**property parameters**

String

Spice parameters string parsed from self.spiceparameters -dictionary in the parent entity.

**property plotcmd**

String

All output IOs are mapped to plot or print statements in the testbench. Also manual plot commands through spice\_simcmd.plotlist are handled here.

**property simcmdstr**

String

Simulation command definition parsed from spice\_simcmd object instantiated in the parent entity.

## 5 Spice Module

Class for Spice netlist parsing.

**class** spice.spice\_module.spice\_module(\*\*kwargs)

This class parses source netlist for subcircuits and handles the generation of a separate pruned subckt\_\* -file.

This class is internally utilized by the spice\_testbench module.

**property name**

String

Entity name.

**property postlayout**

Boolean

Flag for detected post-layout netlists. This will enable post-layout optimizations in the simulator command options. Automatically detected from given netlist for Calibre extracted netlists, or when 'dspf' attribute is defined.

**property subckt**

String

String containing the contents of the subckt\_\* -file. This attribute parses the source netlist when called, and generates the contents to be written to the subckt-file.

**property subinst**

String

String containing the subcircuit instance to be placed in the testbench. The instance is parsed from the previously generated subckt\_\* -file.

**subinst\_constructor(\*\*kwargs)**

Method that parses the subcircuit definition and constructs a subcircuit instance out of it

subckt : string

## 6 Indices and tables

- genindex
- modindex
- search

## Python Module Index

### S

- `spice`, [1](#)
- `spice.spice_dcsource`, [12](#)
- `spice.spice_iofile`, [7](#)
- `spice.spice_module`, [14](#)
- `spice.spice_simcmd`, [10](#)
- `spice.testbench`, [13](#)



## Index

### A

`append_to_data()` (*spice.spice\_iofile.spice\_iofile* method), 9

### C

`connect_spice_inputs()` (*spice.spice* method), 1  
`connect_spice_outputs()` (*spice.spice* method), 1

### D

`dcsource_bundle` (*spice.spice* property), 1  
`dcsourcestr` (*spice.testbench.testbench* property), 13  
`distributed_run` (*spice.spice* property), 1  
`dspf` (*spice.spice* property), 2  
`dspfincludecmd` (*spice.testbench.testbench* property), 13

### E

`eldochoice` (*spice.spice* property), 2  
`entitypath` (*spice.spice* property), 2  
`errpreset` (*spice.spice* property), 2  
`esc_bus()` (*spice.testbench.testbench* method), 13  
`execute_spice_sim()` (*spice.spice* method), 2  
`export()` (*spice.testbench.testbench* method), 14  
`export_subckt()` (*spice.testbench.testbench* method), 14  
`ext_file` (*spice.spice\_dcsource.spice\_dcsource* property), 13  
`extract_powers()` (*spice.spice* method), 2  
`extracts` (*spice.spice* property), 2

### F

`file` (*spice.spice\_iofile.spice\_iofile* property), 9  
`file` (*spice.testbench.testbench* property), 14

### G

`generate_contents()` (*spice.testbench.testbench* method), 14  
`get_buswidth()` (*spice.spice* method), 2

### H

`has_1sf` (*spice.spice* property), 3

### I

`includecmd` (*spice.testbench.testbench* property), 14  
`inputsignals` (*spice.testbench.testbench* property), 14  
`interactive_spice` (*spice.spice* property), 3  
`interp_crossings()` (*spice.spice\_iofile.spice\_iofile* method), 9  
`iofile_bundle` (*spice.spice* property), 3

`iofile_eventdict` (*spice.spice* property), 3  
`ionames` (*spice.spice\_iofile.spice\_iofile* property), 9

### L

`libcmd` (*spice.testbench.testbench* property), 14  
`load_state` (*spice.spice* property), 3

### M

`misccmd` (*spice.testbench.testbench* property), 14  
`module`  
    *spice*, 1  
    *spice.spice\_dcsource*, 12  
    *spice.spice\_iofile*, 7  
    *spice.spice\_module*, 14  
    *spice.spice\_simcmd*, 10  
    *spice.testbench*, 13

### N

`name` (*spice.spice* property), 4  
`name` (*spice.spice\_module.spice\_module* property), 15  
`nproc` (*spice.spice* property), 4  
`num_processes` (*spice.spice* property), 4

### O

`options` (*spice.testbench.testbench* property), 14

### P

`parameters` (*spice.testbench.testbench* property), 14  
`parse_io_from_file()` (*spice.spice\_iofile.spice\_iofile* method), 9  
`plotcmd` (*spice.testbench.testbench* property), 14  
`plotlist` (*spice.spice* property), 4  
`plotprogcmd` (*spice.spice* property), 4  
`plotprogram` (*spice.spice* property), 4  
`postlayout` (*spice.spice\_module.spice\_module* property), 15  
`preserve_iofiles` (*spice.spice* property), 4  
`preserve_result` (*spice.spice* property), 4  
`preserve_spicefiles` (*spice.spice* property), 4

### R

`read()` (*spice.spice\_iofile.spice\_iofile* method), 10  
`read_oppts()` (*spice.spice* method), 4  
`read_spice_outputs()` (*spice.spice* method), 4  
`run_plotprogram()` (*spice.spice* method), 5  
`run_spice()` (*spice.spice* method), 5  
`runname` (*spice.spice* property), 5

## S

`sample_signal()` (*spice.spice\_iofile.spice\_iofile*  
    *method*), 10

`si_prefix_mult` (*spice.spice* property), 5

`si_string_to_float()` (*spice.spice* method), 5

`simcmd_bundle` (*spice.spice* property), 5

`simcmdstr` (*spice.testbench.testbench* property), 14

`spice`  
    module, 1

`spice` (*class in spice*), 1

`spice.spice_dcsource`  
    module, 12

`spice.spice_iofile`  
    module, 7

`spice.spice_module`  
    module, 14

`spice.spice_simcmd`  
    module, 10

`spice.testbench`  
    module, 13

`spice_dcsource` (*class in spice.spice\_dcsource*), 12

`spice_iofile` (*class in spice.spice\_iofile*), 7

`spice_module` (*class in spice.spice\_module*), 15

`spice_simcmd` (*class in spice.spice\_simcmd*), 10

`spice_submission` (*spice.spice* property), 5

`spicecmd` (*spice.spice* property), 5

`spicecorner` (*spice.spice* property), 5

`spicedbpath` (*spice.spice* property), 6

`spicemisc` (*spice.spice* property), 6

`spiceoptions` (*spice.spice* property), 6

`spiceparameters` (*spice.spice* property), 6

`spicesimpath` (*spice.spice* property), 6

`spicesrc` (*spice.spice* property), 7

`spicesrcpath` (*spice.spice* property), 7

`spicesubcktsrc` (*spice.spice* property), 7

`spicetbsrc` (*spice.spice* property), 7

`subckt` (*spice.spice\_module.spice\_module* property), 15

`subinst` (*spice.spice\_module.spice\_module* property),  
    15

`subinst_constructor()`  
    (*spice.spice\_module.spice\_module* method), 15

`syntaxdict` (*spice.spice* property), 7

## T

`testbench` (*class in spice.testbench*), 13

## W

`write()` (*spice.spice\_iofile.spice\_iofile* method), 10

`write_spice_inputs()` (*spice.spice* method), 7