SiliconCompiler

Release 0.21.11

SiliconCompiler Authors

May 01, 2024

CONTENTS

CHAPTER

ONE

USER GUIDE

This guide provides an overview for users. You will also want to look at API References for details.

1.1 What is SiliconCompiler?

SiliconCompiler is an open source, modular, build system that automates translation from hardware design source code to silicon ("make for silicon").

1.1.1 Motivation

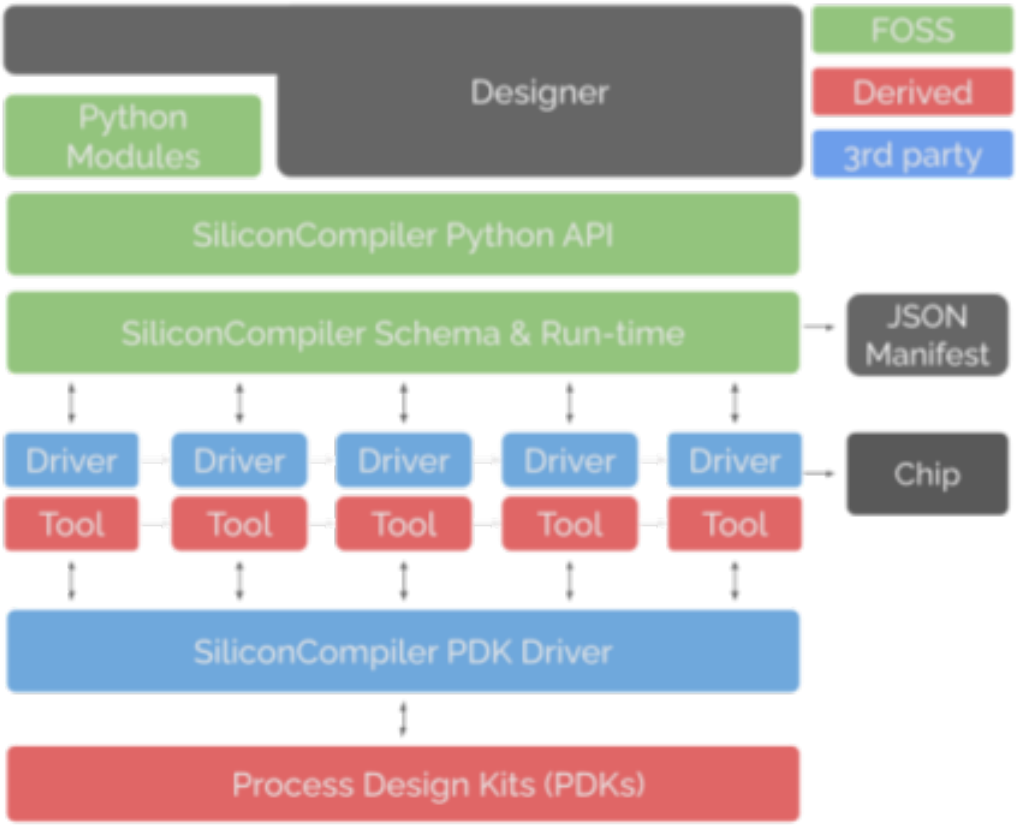
Silicon had an enormous positive impact on the world over the last 50 years and it is a social imperative that we surf the exponential Moore's Law as long as possible. Extreme hardware specialization is the only viable path for extending the current exponential electronics improvement trajectory indefinitely, but the path is currently blocked by the high engineering effort of chip design.

Hardware specialization for a long tail of future applications will require the creation of completely automated end-to-end compilers that are orders of magnitude faster than today's tools. The enormity of these challenges means single machine execution and monolithic single company efforts are unlikely to be sufficient. In this work, we take a distributed systems approach to compilers, with the goal of creating infrastructure that scales to thousands of developers and millions of servers.

1.1.2 Our Approach

The SiliconCompiler project is based on a standardized *Schema* that supports orthogonal combinations of design, tools, and Process Design Kits (PDKs). The schema design philosophy is to "make the complex possible while keeping the simple simple".

To simplify flow development, the project incorporates a simple object oriented *Python API*. The API includes abstracted set/get access to the Schema, a flowgraph based parallel programming model, and a suite of utility functions for compilation setup and metric tracking.



The expansive data schema, standardized plug-in interfaces, and built-in dynamic module search functionality enables SiliconCompiler to scale effectively to a large number of tools and PDKs. The open source *Building Blocks* sections in the reference manual serves as a good starting point for folks who want to add their own PDKs and tools.

To further reduce design access barriers, the project also supports a *client-server* execution model that leverages the cloud to: 1) reduce tool installation barriers, 2) reduce the barrier to massively parallel elastic compute, and 3) address the NDA barrier for PDK and EDA tools.

1.1.3 Supported Technologies

Type	Supported
Languages	C, Verilog, SV, VHDL, Chisel, Migen/Amaranth, Bluespec
Simulation	Verilator, Icarus, GHDL
Synthesis	Yosys, Vivado, Synopsys, Cadence
ASIC APR	OpenRoad, Synopsys, Cadence
FPGA APR	VPR, nextpnr, Vivado
Layout Viewer	Klayout, OpenRoad, Cadence, Synopsys
DRC/LVS	Magic, Synopsys, Siemens
PDKs	sky130, asap7, freepdk45, gf12lp, intel16

1.1.4 Authors

SiliconCompiler project authors in chronological order: Andreas Olofsson, William Ransohoff, Noah Moroze, Zachary Yedidia, Massimiliano Giacometti, Kimia Talaei, Peter Gadfort, Aulihan Teng, Peter Grossmann, Gabriel Aguirre, Martin Troiber.

1.2 Installation

1.2.1 Installing Python

Before installing the SiliconCompiler package you will need to set up a Python environment. Currently Python 3 is supported. The following sections will walk you through how to install the appropriate python dependencies and start a Python virtual environment. Note that at any time, if you need to exit the Python virtual environment, type 'deactivate' and hit enter.

Ubuntu (>=18.04)

Open up a terminal and enter the following command sequence.

```
python3 --version                                    # check for Python 3
sudo apt update                                      # update package information
sudo apt install python3-dev python3-pip python3-venv  # install dependencies
python3 -m venv  ./venv                              # create a virtual env
source ./venv/bin/activate                           # active virtual env (bash/zsh)
```

Note: If you plan to generate any docs or create any flowgraphs, you'll also need to install Graphviz. You can make sure you have this dependency by running `sudo apt install graphviz xdot`

Skip ahead to *SC Install*.

RHEL (>=RHEL 7)

Open up a terminal and enter the following command sequence.

```
sudo subscription-manager repos --enable rhel-server-rhscl-7-rpms  # enable Red Hat␣
↪Software Collections repository
sudo yum -y install rh-python38                                    # install Python 3.8
scl enable rh-python38 bash                                        # enable Python in␣
↪current environment
python3 --version                                                  # check for Python 3
python3 -m venv ./venv                                             # create a virtual env
source ./venv/bin/activate                                         # active virtual env␣
↪(bash/zsh)
pip install --upgrade pip                                          # upgrade Pip
```

Note: If you plan to generate any docs or create any flowgraphs, you'll also need to install Graphviz. You can make sure you have this dependency by running `sudo yum -y install graphviz xdot`

Skip ahead to *SC Install*.

macOS (>=10.15)

Open up a terminal and enter the following command sequence.

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/
↪install.sh)"
export PATH="/usr/local/opt/python/libexec/bin:$PATH"
brew update
brew install python
python3 --version                                  # check for Python 3
python3 -m venv  ./venv                            # create a virtual env
source ./venv/bin/activate                         # active virtual env
```

Note: If you plan to generate any docs or create any flowgraphs, you'll also need to install Graphviz. You can make sure you have this dependency by running `brew install graphviz xdot`

Skip ahead to *SC Install*.

Windows (>= Windows 10)

Install the latest Python package from Python.org using the Windows installer. Open up a Windows shell by:

1. Pressing the 'Windows' key
2. Typing 'cmd', and pressing enter.

From the command shell, enter the following sequence to create and activate a virtual environment.

```
python -m venv  .\venv
.\venv\Scripts\activate
```

Note: If you plan to generate any docs or create any flowgraphs, you'll also need to install Graphviz.

1.2.2 Installing SiliconCompiler

After you've got the python dependencies installed, you will need to install SiliconCompiler. There are a few different ways to do this:

1. The *recommended method* is to install the last stable version published to pypi.org, or
2. You can do an *offline install* with a tarball (for Linux only), or
3. You can install *directly from the git repository* (best for developers).

Install from pypi.org

SiliconCompiler can be installed directly from pypi.org using pip. Activate your Python Virtual Environment and follow the instructions below.

```
(venv) pip install --upgrade pip                # upgrade pip in virtual env
(venv) pip list                                 # show installed packages in venv
(venv) pip install --upgrade siliconcompiler    # install SiliconCompiler in venv
(venv) python -m pip show siliconcompiler       # will display SiliconCompiler package␣
↪information
```

To confirm your installation:

```
(venv) python -c "import siliconcompiler;print(siliconcompiler.__version__)"
```

The expected version should be printed to the display:

```
0.21.11
```

Skip to *asic demo*.

Offline Install (Linux only)

We also provide packages that bundle SC with all of its Python dependencies to enable installation on machines without an external internet connection.

To access them:

1. Go our builds page.
2. Click on the most recent, passing Wheels package. This should be the first green-colored build in the list.
3. On the bottom of that page, you will see an "Artifacts" section. Click on the "sc_plus_dependencies" to download it.
4. The packages are named `scdeps-<pyversion>.tar.gz`, depending on which Python version they are associated with.

Then untar the package and install SiliconCompiler:

```
(venv) tar -xzvf scdeps-<pyversion>.tar.gz
(venv) pip install --upgrade pip --no-index --find-links scdeps
(venv) pip install siliconcompiler --no-index --find-links scdeps
```

To confirm your installation:

```
(venv) python -c "import siliconcompiler;print(siliconcompiler.__version__)"
```

The expected version should be printed to the display:

```
0.21.11
```

Skip to *asic demo*.

Install from GitHub Repo (Linux/MacOS)

You can also install SiliconCompiler from the latest SiliconCompiler GitHub Repository.

Install Dependencies, Bison and Flex

For Linux, you can use:

```
sudo apt-get install flex bison
```

On MacOS, note that you must first install Bison and Flex from Homebrew.

```
brew install bison
brew install flex
```

Ensure that the path to the Homebrew packages takes priority over system packages in your `$PATH`. Run `brew --prefix` to determine where Homebrew installs packages on your machine.

Install SiliconCompiler

Finally, to clone and install SiliconCompiler, run the following:

```
(venv) git clone -b v0.21.11 https://github.com/siliconcompiler/siliconcompiler
(venv) cd siliconcompiler
(venv) python -m pip install -e .
```

To confirm your installation:

```
(venv) python -c "import siliconcompiler;print(siliconcompiler.__version__)"
```

The expected version should be printed to the display:

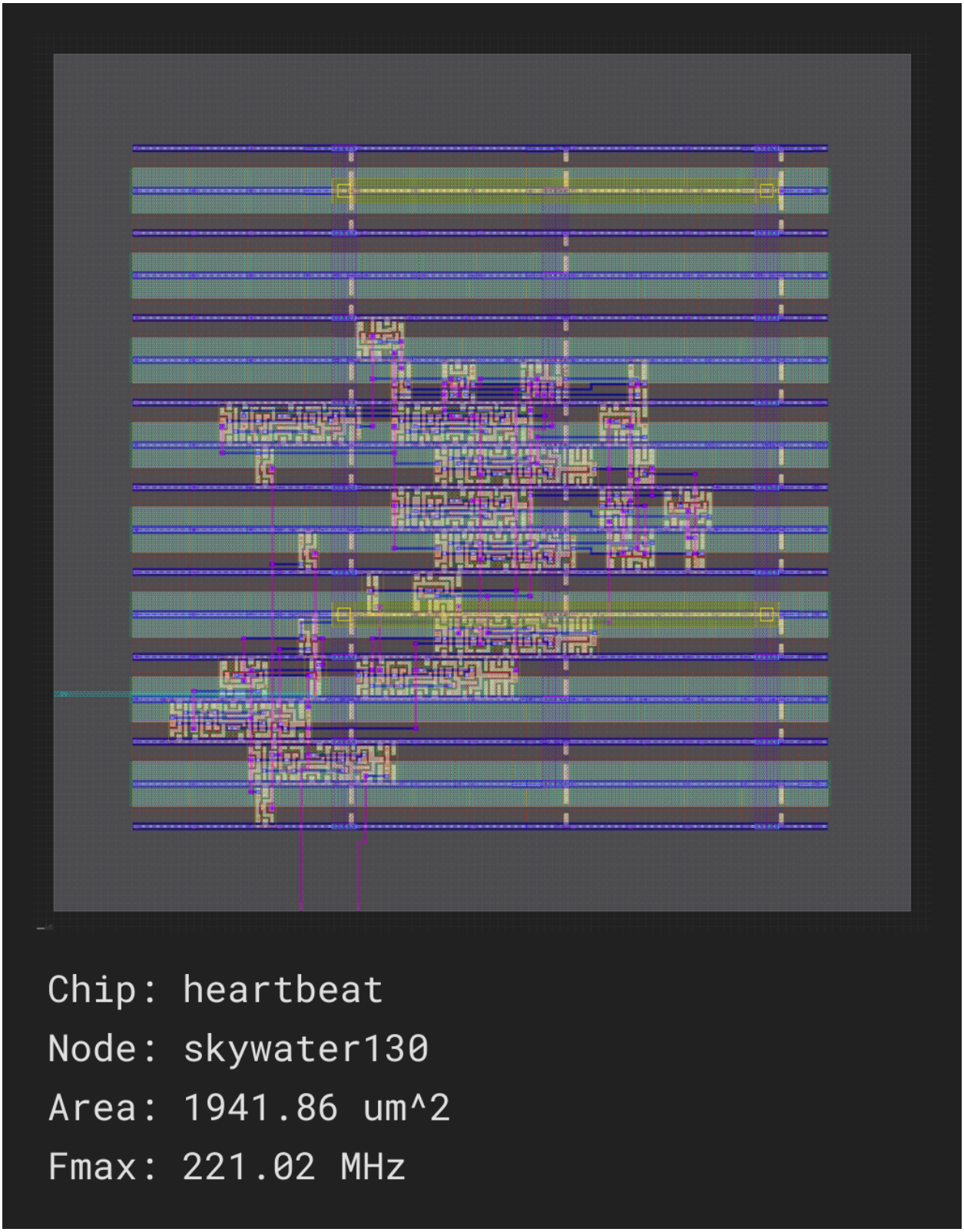
```
0.21.11
```

1.2.3 ASIC Demo

Now that you have installed SiliconCompiler, you can test your installation by running a quick demo through the ASIC design flow in the cloud.:

```
sc -target asic_demo -remote
```

Your remote job should only take a few minutes to run if the servers aren't too busy. It should end with a results directory where you can find `png` file which displays your results. It should look something like this:



See *Quickstart guide* next to go through the design and run details of the quick demo above.

1.2.4 External Tools

If you wish to run on your machine instead of remotely in the cloud as in the quick *asic demo* target above, there will be some tools you need to install first.

Note: The minimum set of tools required for an ASIC flow are: *Surelog*, *Yosys*, *OpenROAD*, and *KLayout*. Links to individual tool installation instructions and platform limitations can be found in the *pre-defined tool drivers*.

We have provided the following helper install scripts for this minimum toolset for the ASIC flow as well as other external tools, but keep in mind that they are for reference only. If you should run into issues, please consult the official download instructions for the tool itself. All official tool documentation links can be found in the *pre-defined tool drivers* section.

- *bambu*: ubuntu20, ubuntu22
- *bluespec*: ubuntu20, ubuntu22
- *chisel*: ubuntu20, ubuntu22
- *ghdl*: ubuntu20, ubuntu22
- icarus: ubuntu20, ubuntu22
- icepack: ubuntu20, ubuntu22
- *klayout*: ubuntu20, ubuntu22
- *magic*: ubuntu20, ubuntu22
- *montage*: ubuntu20, ubuntu22
- netgen: ubuntu20, ubuntu22
- nextpnr: ubuntu20, ubuntu22
- openfpga: ubuntu20, ubuntu22
- *openroad*: ubuntu20, ubuntu22
- slurm: ubuntu20, ubuntu22
- *surelog*: ubuntu20, ubuntu22
- *sv2v*: ubuntu20, ubuntu22
- verible: ubuntu20, ubuntu22
- *verilator*: ubuntu20, ubuntu22
- *vpr*: ubuntu20, ubuntu22
- *xyce*: ubuntu20, ubuntu22
- *yosys*: ubuntu20, ubuntu22

See *Quickstart guide* next to see how to run locally on your machine with these tools.

1.3 Quickstart guide

If you've completed the *Installation* section and were able to run the *ASIC Demo*, you will have completed a simple remote run through an ASIC design flow!

In the following sections, you will find more details about *the design*, *the flow* and *the results* of the run.

1.3.1 Design Details

The simple design that was used in the *demo target* is a single clock cycle pulse ("heartbeat") generated by a counter. You can see the design here: heartbeat.v.

1.3.2 Run Setup

SiliconCompiler includes a Python API to simplify the hardware compilation flow process. The following code snippet below shows how the *demo design* was loaded in and run through the Python API.

Listing 1: heartbeat.py (remote run)

```
#!/usr/bin/env python3

import siliconcompiler                          # import python package

if __name__ == "__main__":
    chip = siliconcompiler.Chip('heartbeat')  # create chip object
    chip.input('heartbeat.v')                 # define list of source files
    chip.clock('clk', period=10)              # define clock speed of design
    chip.load_target('skywater130_demo')      # load predefined technology and flow␣
↪target
    chip.set('option', 'remote', True)        # run remote in the cloud
    chip.run()                                # run compilation of design and target
    chip.summary()                            # print results summary
```

The following sub-sections will describe each line in more detail.

Object Creation

The hardware build flow centers around the chip data object. This chip object is instantiated by calling the *Chip()* class constructor defined in the *Core API*

```
import siliconcompiler                  # import python package

chip = siliconcompiler.Chip('heartbeat')  # create chip object
```

Define Design

Once the chip object is created, design parameters can be set up with the chip object's pre-defined functions, or methods. In this case, the helper function *.input()* allows you to specify the hardware description input file(s) and the `.clock()` helper function allows you to specify the design frequency.

```
chip.input('heartbeat.v')                  # define list of source files
chip.clock('clk', period=10)               # define clock speed of design
```

Define PDK and Flow

In addition to design parameters, you can also set up your PDK and libraries. The compilations of this design is using the `Chip.load_target()` function to load the pre-defined flow target *skywater130_demo* which is set up to use the *skywater130 pdk*. This *pre-built target* is also set up to run a full RTL to GDS run flow, from design synthesis to design placement and routing. You can take a look at the other *Pre-Defined Targets* to see other options for other PDKs and libraries.

```
chip.load_target('skywater130_demo')       # load predefined technology and flow target
```

Specify Run Location

Next, the *['option', 'remote']* parameter of the chip object is directly being accessed by the `Chip.set()` method to `True`. This means it's run in the cloud. If you were to remove this, it would run on your *local machine*.:

```
chip.set('option', 'remote', True)         # run remote in the cloud
```

Design Compilation

Now that the design compilation is set up, it's time to *run()* the compilation and print the results with *summary()*.

```
chip.run()                                 # run compilation of design and target
chip.summary()                             # print results summary
```

1.3.3 Run Flow

Running this python script directly produces the same results as the *ASIC Demo* target.

```
python3 heartbeat.py
```

Alternatively, since this is a simple design with just one design input file, you can also run from the command line:

```
sc heartbeat.v heartbeat.sdc -target "skywater130_demo" -remote
```

Note: You can use heartbeat.sdc for the constraints file; this replaces the clock definition in the python script.

1.3.4 Remote Run Controls

When your job starts on a remote server, it will log a job ID which you can use to query your job if you close the terminal window or otherwise interrupt the run before it completes:

```
| INFO    | job0  | remote     | 0  | Your job's reference ID is:␣
↪0123456789abcdeffedcba9876543210
```

You can use this job ID to interact with a running job using the *sc-remote* CLI app:

```
# Check on a job's progress.
sc-remote -jobid 0123456789abcdeffedcba9876543210

# Cancel a running job.
sc-remote -jobid 0123456789abcdeffedcba9876543210 -cancel

# Ask the server to delete a job from its active records.
sc-remote -jobid 0123456789abcdeffedcba9876543210 -delete

# Reconnect to an active job.
sc-remote -jobid 0123456789abcdeffedcba9876543210 -reconnect -cfg [build/design/jobname/
↪import/0/outputs/design.pkg.json]
```

The *sc-remote* app also accepts a *-credentials* input parameter which works the same way as the *['option', 'credentials'] Schema* parameter.

1.3.5 Run Results

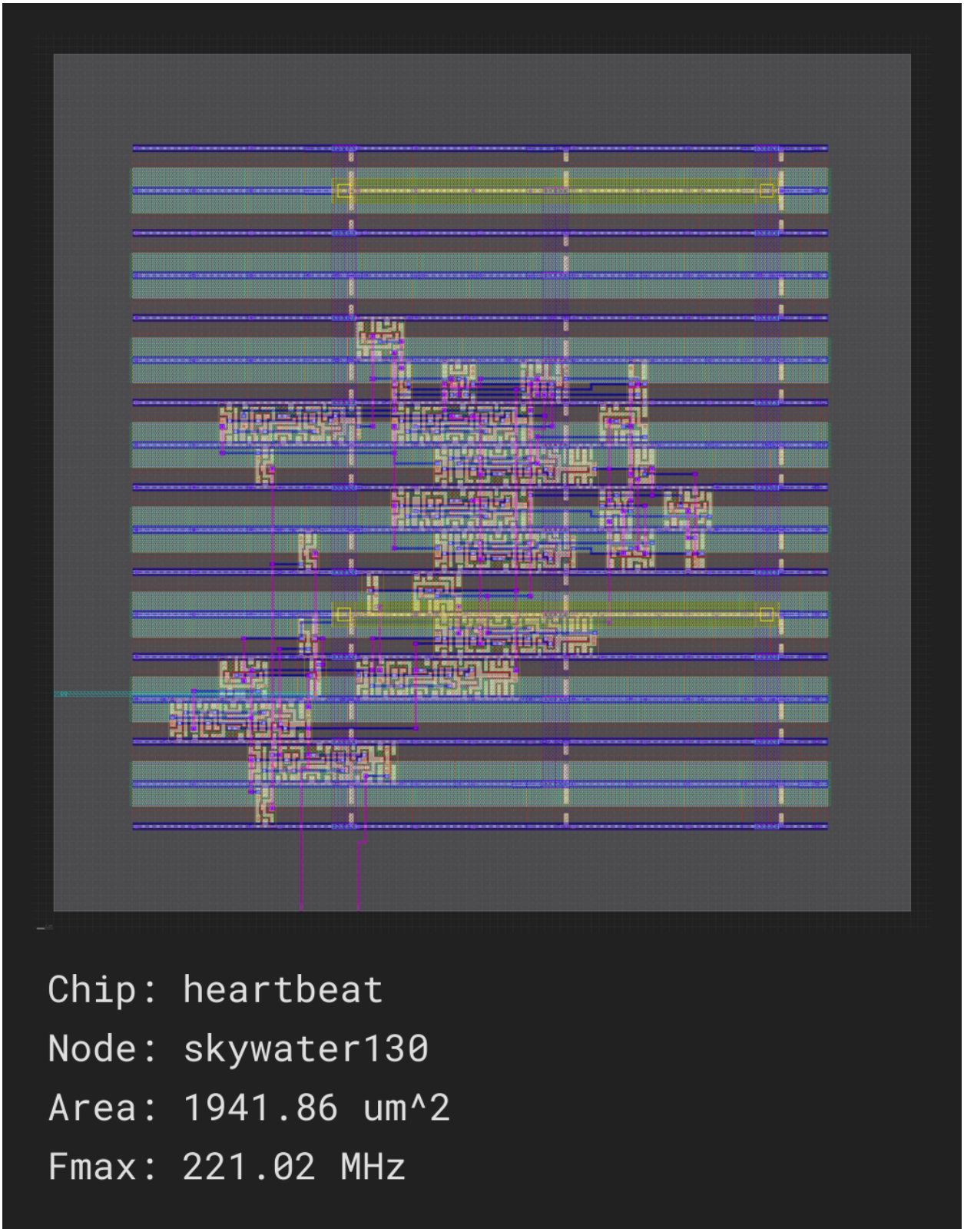
Your run will first show the the SiliconCompiler banner/info, followed by design INFO messages.

As the run goes through each step of the flow, a message will be printed to the screen every 30 seconds.

Then, at the end of the run, a summary table will be printed similar to the one show below. This table is generated by calling the *summary()* function call in your python script *above*.

```
-------------------------------------------------------------------------------------------------------------------------------------------------
SUMMARY:

design : heartbeat
params : None
jobdir : ./build/heartbeat/job0
foundry : skywater
process : skywater130
targetlibs : sky130hd

                  import0       syn0     floorplan0     physyn0       place0        cts0        route0        dfm0       export0       export1
errors              0           0           0           0           0           0           0           0           0           0
warnings            1         237          17           0           0           2           3          51           1           0
drvs              ---         ---           0           0           0           0           0           0         ---           0
unconstrained     ---         ---           1           1           1           1           1           1         ---           1
cellarea          ---       346.0       375.36      375.36      372.858      384.118     384.118     384.118       ---       384.118
totalarea         ---         ---       1941.86     1941.86     1941.86     1941.86     1941.86     1941.86       ---       1941.86
utilization       ---         ---      0.193299    0.193299     0.19201    0.197809    0.197809    0.197809       ---      0.197809
peakpower         ---         ---     4.3291e-05  4.3291e-05  4.36818e-05 5.62034e-05 6.43086e-05 5.51926e-05       ---     5.67837e-05
leakagepower      ---         ---     7.97075e-08 7.97075e-08 8.35005e-08  8.3504e-08  8.3504e-08  8.3504e-08       ---      8.3504e-08
holdpaths         ---         ---           0           0           0           0           0           0         ---           0
setuppaths        ---         ---           0           0           0           0           0           0         ---           0
holdslack         ---         ---       0.294133    0.294133    0.298687    0.313026    0.354662    0.305927       ---      0.314895
holdwns           ---         ---           0.0         0.0         0.0         0.0         0.0         0.0         ---           0.0
holdtns           ---         ---           0.0         0.0         0.0         0.0         0.0         0.0         ---           0.0
setupslack        ---         ---       5.96039     5.96039     5.57303     5.51415     4.97672     5.66156       ---       5.47544
setupwns          ---         ---           0.0         0.0         0.0         0.0         0.0         0.0         ---           0.0
setuptns          ---         ---           0.0         0.0         0.0         0.0         0.0         0.0         ---           0.0
fmax              ---         ---     247549000.0 247549000.0 225888000.0 222923000.0 199073000.0 230498000.0       ---     221016000.0
macros            ---         ---           0           0           0           0           0           0         ---           0
cells             ---          26          52          52          52          55          55          55         ---          55
registers         ---           9           9           9           9           9           9           9         ---           9
buffers           ---         ---           2           2           2           5           5           5         ---           5
pins              ---         ---           3           3           3           3           3           3         ---           3
nets              ---         ---          29          29          29          32          32          32         ---          32
vias              ---         ---         ---         ---         ---         ---         179         ---         ---         ---
wirelength        ---         ---         ---         ---         ---         ---       496.0         ---         ---         ---
memory       27348992.0  40378368.0  138579968.0 131424256.0 211271680.0 179249152.0 547594240.0 144228352.0 621031424.0 142176256.0
exetime           0.32        1.25        1.47        1.06        1.98        2.88        3.17        0.96        3.02        1.15
tasktime          0.58        3.42        2.03        1.51        2.44        3.31        3.62        1.37        3.95        1.56
-------------------------------------------------------------------------------------------------------------------------------------------------
```

All design outputs are located in `build/<design>/<jobname>`. When running remote, you will not get all the tool-specific output that you would with a *local run*, but you will be able to find a screenshot of the demo design `heartbeat.png` and a summary report in `report.html`:


Chip: heartbeat
Node: skywater130
Area: 1941.86 um^2
Fmax: 221.02 MHz

1.3.6 Other Ways to Run

The *ASIC Demo* was run in public beta server in the cloud. SiliconCompiler also supports running on *private* servers or also local runs on your own machine.

See *Remote processing* to see details on how to run on a private server, and see *External Tools* to see the additional tool installation requirements for running on your machine locally.

1.3.7 Local Run

If you have the *prerequisite tools* installed for a local run, you can also give that a try. Local runs are useful if you're interested in more detailed logs and reports for each step run and the ability to browse a gds at the end.

In order to run on your local machine, the only thing you need to do differently than *the remote run* is to set the *['option', 'remote']* to `False` in your python script and *re-run*.

Or, if you want to run from the command line, just remove the `-remote` option.

```
sc heartbeat.v heartbeat.sdc -target "skywater130_demo"
```

Local Run Results

By default, only the summary of each step is printed, in order to not clutter up the screen with tool-specific output. If you wish to see the output from each tool, you can find the log files associated with each tool in: `build/<design>/<jobname>/<step>/<index>/<step>.log`

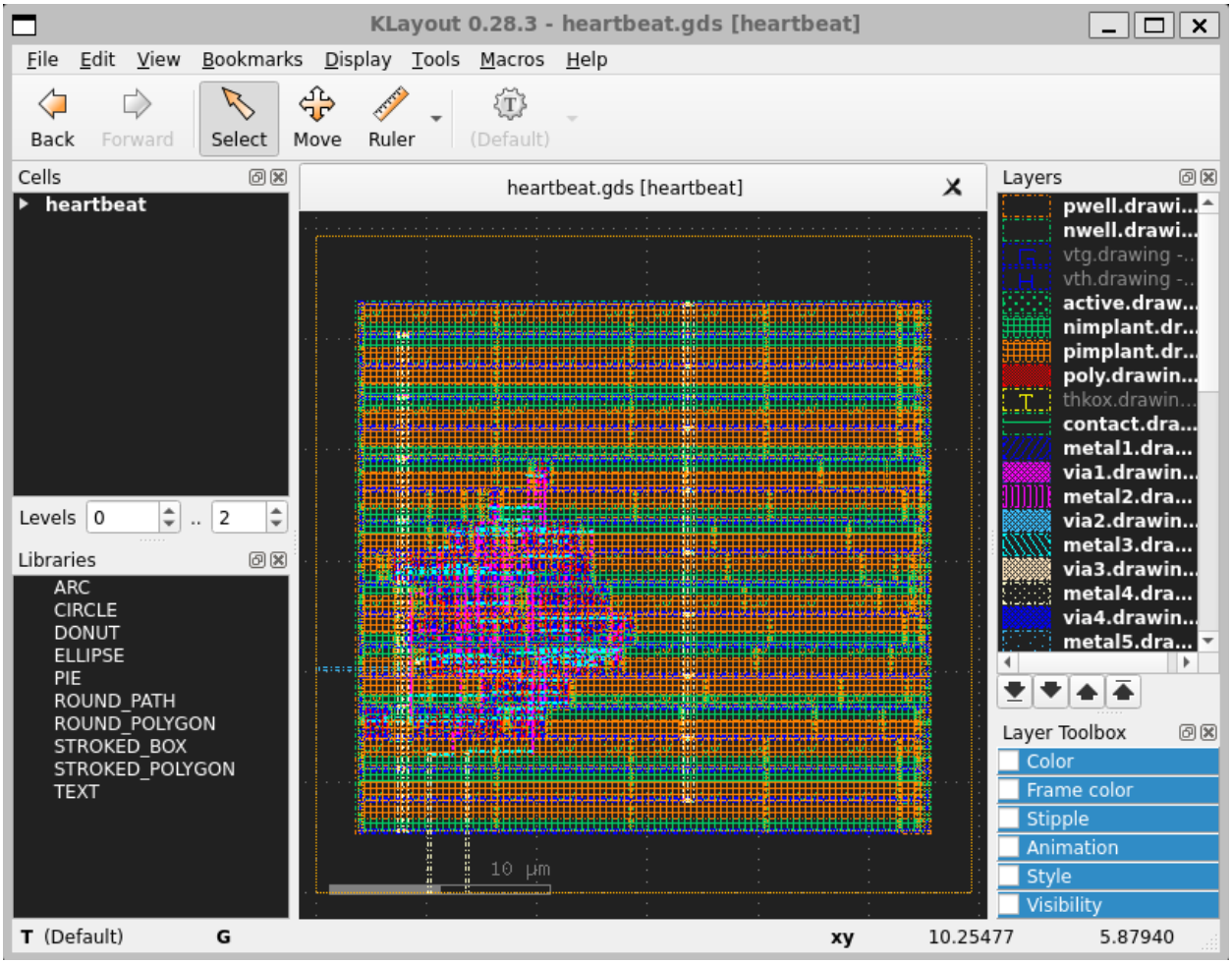
If you wish to see all the tool-specific information printed onto the screen, you can turn the *quiet* option off.

View Design

For viewing IC layout files (DEF, GDSII) we recommend installing the open source multi-platform *Klayout viewer* (available for Windows, Linux, and macOS). Installation instructions for Klayout can be found in the *tools directory*.

If you have Klayout installed, you can browse your completed design by calling *sc-show* directly from the command line as shown below:

```
(venv) sc-show -design heartbeat
```



If you want to have this window pop up automatically at the end of your script, you can add `show()` to the end of your *python script*.

```
chip.show()       # pops open a window with the layout
```

1.3.8 What Next?

Now that you've quickly run a simple example, you can proceed to a larger example like *Building Your Own SoC*, or you can dive deeper into the SiliconCompiler build flow you ran from this quickstart (asic_demo) by looking through how the flow is constructed with the *Design and Compilation Data* and *Compilation Process* in the Fundamentals section.

1.4 Design and Compilation Data

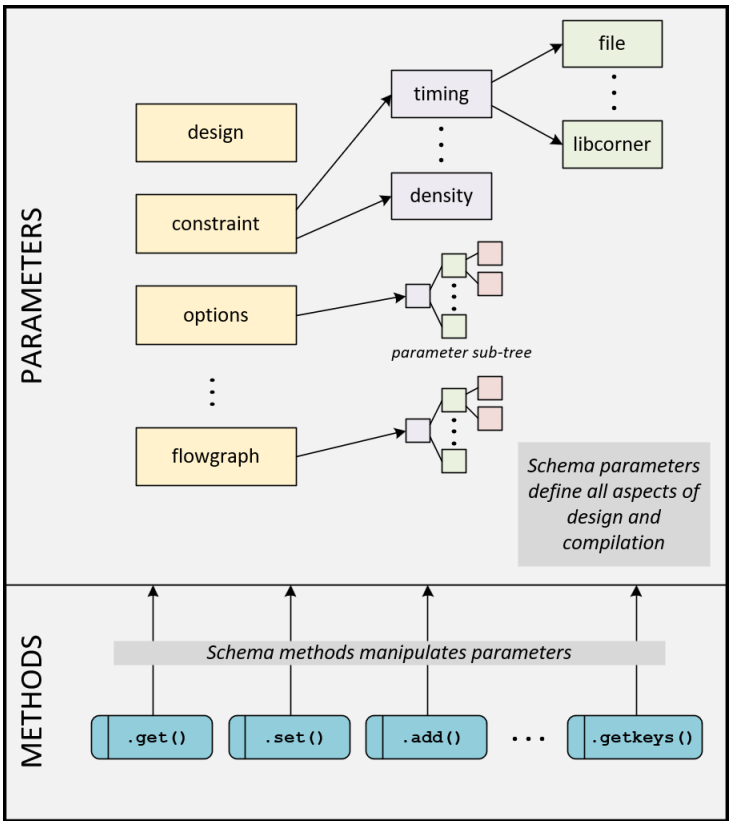
SiliconCompiler uses a data structure object, called `Schema`, also referred to as "the schema" in subsequent docs, to store all information associated with the compilation process and the design that's being compiled.

The types of information stored by the schema include, but is not limited to:

- How the design is defined (i.e. HW architectural definitions)
- How the design is compiled (i.e. Build tools and technology specifics)

- How the design is optimized (i.e. Different tool options for build experiments)

This data is stored in Schema parameters, and accessed through Schema methods.



The diagram above shows a few examples of Schema parameters and methods for an overview of how data is stored and accessed.

The following sections provide more detail on how information in the schema is initialized and manipulated.

1.4.1 Schema Configuration

The schema is "configured," or defined, based on its parameters.

Major Parameter Categories

The SiliconCompiler Schema is divided into the following major sub-groups of parameters:

Group	Parameters	Description
`['constraint', ...]`	31	Design constraint settings
`['option', ...]`	63	Compilation options
`['unit', ...]`	11	Global units
`['fpga', ...]`	11	FPGA related settings
`['asic', ...]`	20	ASIC related settings
`['pdk', ...]`	40	PDK related settings
`['tool', ...]`	30	Individual tool settings
`['flowgraph', ...]`	10	Execution flow definition

continues on next page

Table 1 – continued from previous page

['checklist', ...]	9	Checklist related settings
['metric', ...]	43	Metric tracking
['record', ...]	18	Compilation history tracking
['datasheet', ...]	174	Design interface specifications
['package', ...]	23	Packaging manifest
Total	483	

Parameter Sub-tree Example

Some parameters have their own subtrees in order to be fully defined. The table below shows an example of a parameter, called *constraint*, which specifies the design constraints, from timing-specific parameters to physical design parameters.

parameter	description
['constraint', 'timing', ...]	Contains sub-tree of parameters. See Schema.
['constraint', 'component', ...]	Contains sub-tree of parameters. See Schema.
['constraint', 'pin', ...]	Contains sub-tree of parameters. See Schema.
['constraint', 'net', ...]	Contains sub-tree of parameters. See Schema.
['constraint', 'outline']	Constraint: Layout outline
['constraint', 'corearea']	Constraint: Layout core area
['constraint', 'coremargin']	Constraint: Layout core margin
['constraint', 'density']	Constraint: Layout density
['constraint', 'aspectratio']	Constraint: Layout aspect ratio

Accessing Schema Parameters

While all the design and compilation information are stored in the Schema object, this information is manipulated through a separate data structured called *Chip*.

The Chip Object

This separate data structure is different from the *Schema* since it instantiates the Schema object and is used to define methods that manipulate the compilation process.

class siliconcompiler.**Chip**(*design*, *loglevel=None*)

Object for configuring and executing hardware design flows.

This is the main object used for configuration, data, and execution within the SiliconCompiler platform.

Parameters

design (*string*) – Name of the top level chip design module.

Examples

```
>>> siliconcompiler.Chip(design="top")
Creates a chip object with name "top".
```

Chip Creation and Schema Parameter Access

The following example shows how to create a chip object and manipulate the *input* schema parameter in Python by setting the parameter with the *Chip.set()* method, accessing it with the *Chip.get()* method, and appending to the parameter field with the *Chip.add()* method.

```
>>> import siliconcompiler
>>> chip = siliconcompiler.Chip('fulladder')

>>> chip.set('input', 'rtl', 'verilog', 'fulladder.v')
>>> print(chip.get('input', 'rtl', 'verilog'))
['fulladder.v']

>>> chip.add('input', 'rtl', 'verilog', 'halfadder.v')

>>> print(chip.get('input', 'rtl', 'verilog'))
['fulladder.v', 'halfadder.v']
```

The *Chip* object provides many useful *helper functions*. For example, in the *quickstart guide* , the *Chip.input()* helper function was used to set the chip timing constraints file, a simpler call than using *Chip.set()*.

```
>>> chip.input('fulladder.sdc')
| INFO    | fulladder.sdc inferred as constraint/sdc

>>> print(chip.get('input', 'constraint', 'sdc'))
['fulladder.sdc']
```

Chip.getkeys() is another example of a useful function, provided by *the chip object*, for checking your parameters.

```
>>> chip.getkeys('input')
['rtl', 'constraint']

>>> chip.getkeys('input', 'rtl')
['verilog']

>>> chip.getkeys('input', 'constraint')
['sdc']
```

You can see from the example above that using the *Chip.getkeys()* function, you're able to query the subtree of the parameter called `input`, where the parameter tree can be visually represented as:

```
└── input
    ├── constraint
    │   └── sdc
    └── rtl
        └── verilog
```

If you further go one step further down, you'll see that `verilog` is a leaf parameter, so the *Chip.getkeys()* function returns its parameter fields.

```
>>> chip.getkeys('input', 'rtl', 'verilog')
['type', 'scope', 'require', 'lock', 'switch', 'shorthelp', 'example', 'help', 'notes',
↪'pernode', 'node', 'hashalgo', 'copy']
```

Parameter fields are standardized variables which help to define the parameter. In the case below, you can see that *Chip.get()* can also be used to query parameter fields to provide more information about the parameters:

```
>>> chip.get('input', 'rtl', 'verilog', field='type')
'[file]'

>>> chip.get('input', 'rtl', 'verilog', field='example')
["cli: -input 'rtl verilog hello_world.v'", "api: chip.set(input, 'rtl','verilog','hello_
↪world.v')"]
```

`getkeys()` is just one useful helpfer function; see *Core API* for more information on methods which can be used to manipulate Schema parameters.

1.4.2 Manifest

The Schema is recorded to a *manifest*. This file serves not only as a reference of all the design and compilation parameters, it also provides a mechanism to reload a design.

If you ran the *ASIC Demo*, you should have a manifest written out to

```
build/<design>/job0/<design>.pkg.json
```

The *Chip.read_manifest()* and *Chip.write_manifest()* Python API methods handle reading and writing the Schema to/from disk. Besides JSON, other supported export file formats include TCL, and YAML. By default, only non-empty values are written to disk.

```
import siliconcompiler
chip = siliconcompiler.Chip('hello_world')
chip.write_manifest('hello_world.json')
```

The *Chip.write_manifest()* method above writes out the JSON file below, showing the standardized key/value pairs ("fields") associated with the *design* parameter.

```
"design": {
    "lock": false,
    "node": {
        "default": {
           "default": {
                "signature": null,
                "value": null
           }
        },
        "global": {
            "global": {
                "signature": null,
                "value": "hello_world"
            }
```

(continues on next page)

(continued from previous page)

```
        }
    },
    "notes": null,
    "pernode": "never",
    "require": "all",
    "scope": "global",
    "shorthelp": "Design top module name",
    "switch": [
        "-design <str>"
    ],
    "type": "str"
},
```

1.4.3 Additional Schema Information

Refer to the *Schema* and *Python API* sections of the reference manual for more information. Another good resource is the schema configuration file Schema source code.

1.5 Compilation Process

The complete SiliconCompiler compilation is handled by a single call to the *run()* function. Within that function call, a static data *flowgraph*, consisting of *nodes* and *edges* is traversed and "executed."

The static flowgraph approach was chosen for a number reasons:

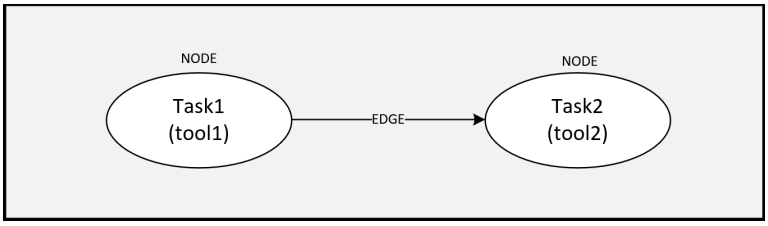
- Performance scalability ("cloud-scale")
- High abstraction level (not locked into one language and/or shared memory model)
- Deterministic execution
- Ease of implementation (synchronization is hard)

1.5.1 The Flowgraph

Nodes and Edges

A SiliconCompiler flowgraph consists of a set of connected nodes and edges, where:

- A *node* is an executable *tool* performing some ("*task*"), and
- An *edge* is the connection between those tasks, specifying execution order.

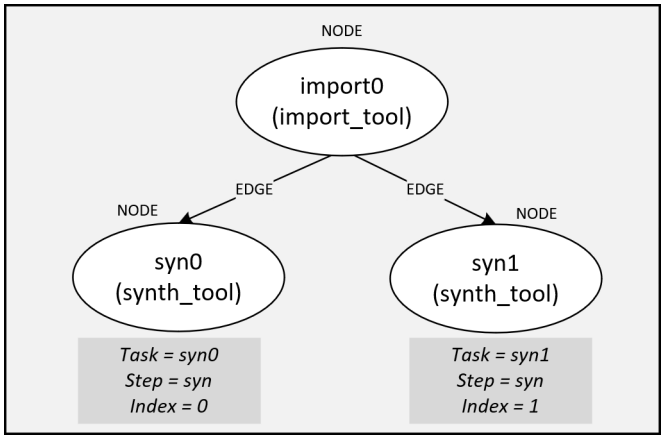


Tasks

SiliconCompiler breaks down a "task" into an atomic combination of a step and an index, where:

1. A *step* is defined as discrete function performed within compilation flow such as synthesis, linting, placement, routing, etc, and
2. An *index* is defined as variant of a step operating on identical data.

An example of this might be two parallel synthesis runs with different settings after an import task. The two synthesis "tasks" might be called `syn0` and `syn1`, where:



See *Using Index for Optimization* for more information on why using indices to build your flowgraph are helpful.

Execution

Flowgraph execution is done through the *`run()`* function which checks the flowgraph for correctness and then executes all tasks in the flowgraph from start to finish.

1.5.2 Flowgraph Examples

The flowgraph, used in the *ASIC Demo*, is a built-in compilation flow, called *asicflow*. This compilation flow is a pre-defined flowgraph customized for an ASIC build flow, and is called through the *`load_target()`* function, which calls a *pre-defined PDK module* that uses the asicflow flowgraph.

You can design your own chip compilation build flows by easily creating custom flowgraphs through:

- *`node()`*/*`edge()`* methods

The user is free to construct a flowgraph by defining any reasonable combination of steps and indices based on available tools and PDKs.

A Two-Node Flowgraph

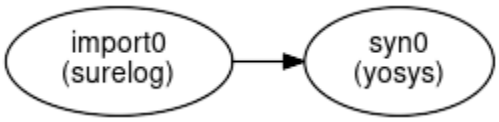
The example below shows a snippet which creates a simple two-step (import + synthesis) compilation pipeline.

Listing 2: Snippet from examples/heartbeat_flowgraph.py

```
flow = 'synflow'
chip.node(flow, 'import', parse)              # use surelog for import
chip.node(flow, 'syn', syn_asic)              # use yosys for synthesis
chip.edge(flow, 'import', 'syn')              # perform syn after import
chip.set('option', 'flow', flow)
```

At this point, you can visually examine your flowgraph by using *write_flowgraph()*. This function is very useful in debugging graph definitions.

```
chip.write_flowgraph("flowgraph.svg", landscape=True)
```



Note: **[In Progress]** Insert link to tutorial which has step-by-step instruction on how to set up this flow with libs and pdk through run and execution.

Using Index for Optimization

The previous example did not include any mention of *index*, so the index defaults to `0`.

While not essential to basic execution, the 'index' is fundamental to searching and optimizing tool and design options.

One example use case for the index feature would be to run a design through synthesis with a range of settings and then selecting the optimal settings based on power, performance, and area. The snippet below shows how a massively parallel optimization flow can be programmed using the SiliconCompiler Python API.

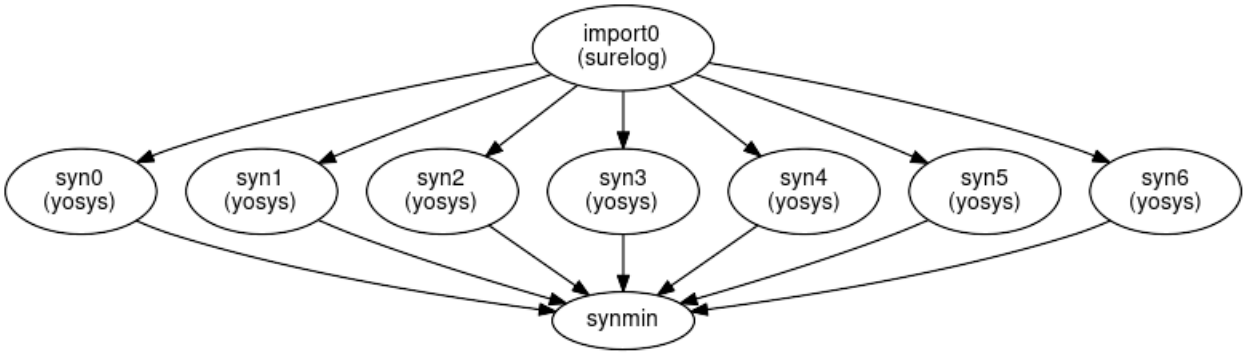
Listing 3: Snippet from examples/flowgraph_doe.py that sets up parallel synthesis runs for optimization

```
syn_strategies = ['DELAY0', 'DELAY1', 'DELAY2', 'DELAY3', 'AREA0', 'AREA1', 'AREA2']

# define flowgraph name
flow = 'synparallel'

# create import node
chip.node(flow, 'import', parse)

# create node for each syn strategy (first node called import and last node called␣
↪synmin)
# and connect all synth nodes to both the first node and last node
for index in range(len(syn_strategies)):
    chip.node(flow, 'syn', syn_asic, index=str(index))
    chip.edge(flow, 'import', 'syn', head_index=str(index))
    chip.edge(flow, 'syn', 'synmin', tail_index=str(index))
```

(continues on next page)

(continued from previous page)

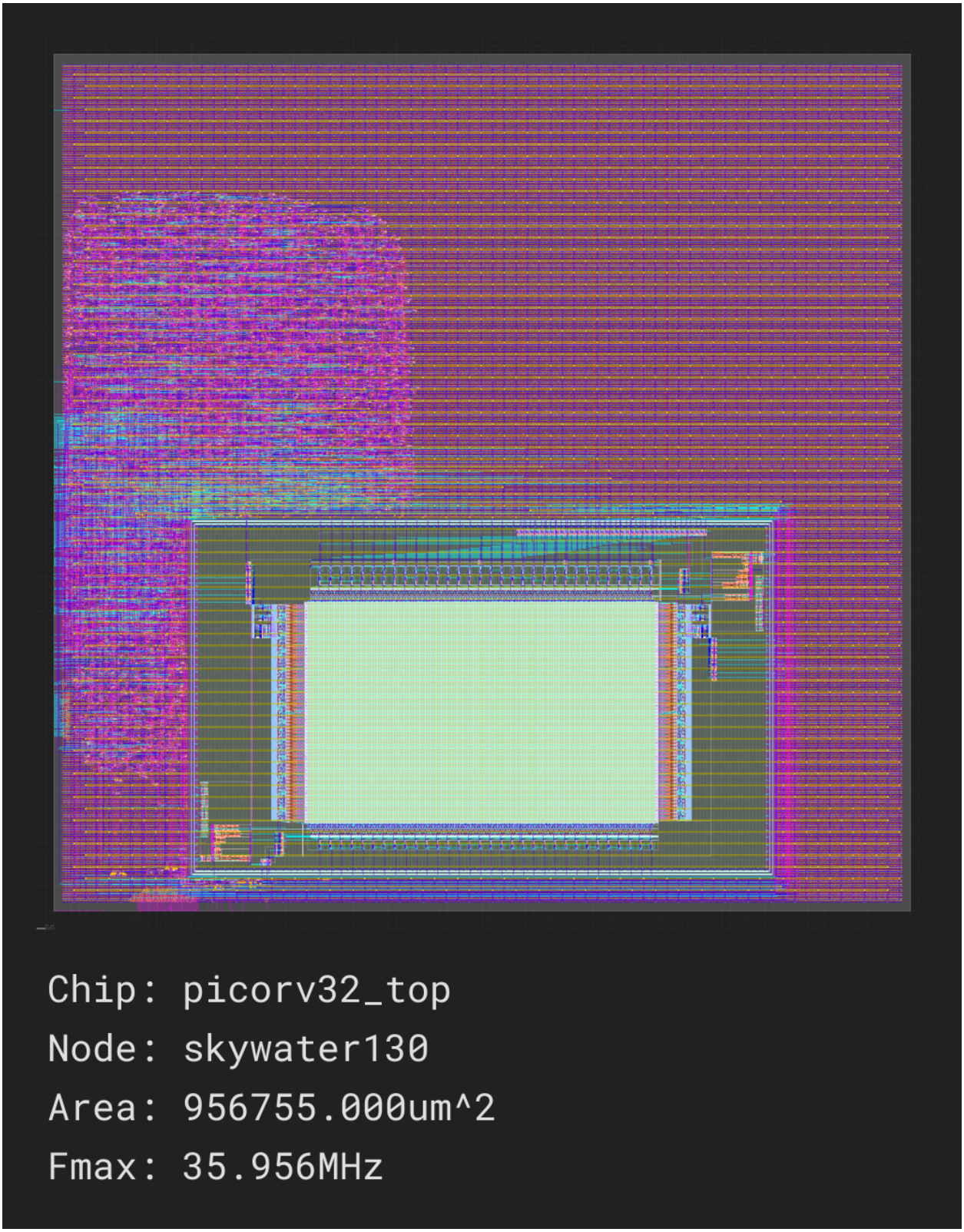
```
    chip.set('tool', 'yosys', 'task', 'syn_asic', 'var', 'strategy', syn_
↪strategies[index],
             step='syn', index=index)

    # set synthesis metrics that you want to optimize for
    for metric in ('cellarea', 'peakpower', 'standbypower'):
        chip.set('flowgraph', flow, 'syn', str(index), 'weight', metric, 1.0)

# create node for optimized (or minimum in this case) metric
chip.node(flow, 'synmin', minimum)

chip.set('option', 'flow', flow)
```



Note: **[In Progress]** Provide pointer to a tutorial on optimizing a metric

1.6 Building Your Own SoC

This tutorial will walk you through the process of building an ASIC containing one PicoRV32 RISC-V CPU core and 2 kilobytes of SRAM, on an open-source 130nm Skywater process node, with SiliconCompiler's remote workflow:



We will walk through the process of downloading the design files and writing a build script, but for your reference, you can find complete example designs which reflect the contents of this tutorial in the public SiliconCompiler repository. The first part of the tutorial will cover building the CPU core without RAM, and the second part will describe how to

add an SRAM block.

See the *Installation* section for information on how to install SiliconCompiler, and the *Remote Processing* section for instructions on setting up the remote workflow.

1.6.1 Download PicoRV32 Verilog Code

The heart of any digital design is its HDL code, typically written in a language such as Verilog or VHDL. High-level synthesis languages are gaining in popularity, but most of them still output their final design sources in a traditional HDL such as Verilog.

PicoRV32 is an open-source implementation of a small RISC-V CPU core, the sort you might find in a low-power microcontroller. Its source code, license, and various tooling can be found in its GitHub repository.

1.6.2 Build the PicoRV32 Core using SiliconCompiler

Before we add the complexity of a RAM macro block, let's build the core design using the open-source *Skywater 130* PDK. Copy the following build script into the same directory which you copied `picorv32.v` into:

Listing 4: <project_dir>/picorv32.py

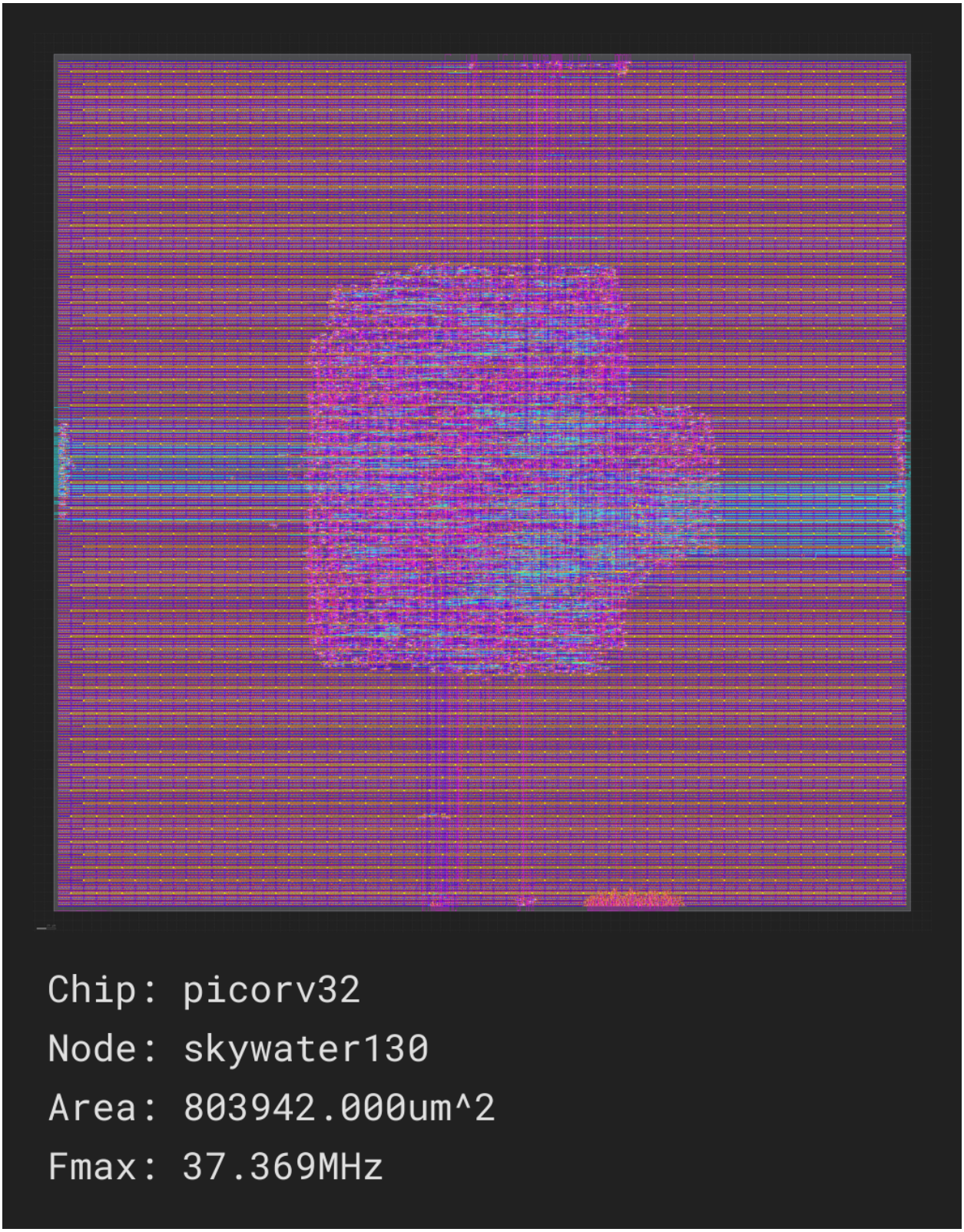
```
#!/usr/bin/env python3
import siliconcompiler


def rtl2gds(target="skywater130_demo"):
    '''RTL2GDS flow'''

    # CREATE OBJECT
    chip = siliconcompiler.Chip('picorv32')

    # SETUP
    chip.load_target(target)

    chip.register_package_source(name='picorv32',
                                 path='git+https://github.com/YosysHQ/picorv32.git',
                                 ref='c0acaebf0d50afc6e4d15ea9973b60f5f4d03c42')

    chip.input('picorv32.v', package='picorv32')

    chip.set('option', 'relax', True)
    chip.set('option', 'quiet', True)
    chip.set('option', 'remote', False)

    chip.clock('clk', period=25)

    # RUN
    chip.run()

    # ANALYZE
    chip.summary()

    return chip
```

(continues on next page)

(continued from previous page)

```
if __name__ == '__main__':
    rtl2gds()
```

Note in the code snippet above that *remote* is set to `False`. If this is set to `True`, this means it is set up for *Remote processing*, and if you run this example as a Python script, it should take approximately 20 minutes to run if the servers are not too busy. We have not added a RAM macro yet, but this script will build the CPU core with I/O signals placed pseudo-randomly around the edges of the die area. Once the job finishes, you should receive a screenshot of your final design, and a report containing metrics related to the build in `build/picorv32/job0/report.html`. SiliconCompiler will try to open the file after the job completes, but it may not be able to do so if you are running in a headless environment.



For the full GDS-II results and intermediate build artifacts, you can run the build locally. See the *Local Run* section for more information.

1.6.3 Adding an SRAM block

A CPU core is not very useful without any memory. Indeed, a real system-on-chip would need quite a few supporting IP blocks to be useful in the real world. At the very least, you would want a SPI interface for communicating with external non-volatile memory, a UART to get data in and out of the core, a debugging interface, and a small on-die cache.

In this tutorial, we'll take the first step by adding a small (2 kilobyte) SRAM block and wiring it to the CPU's memory interface. This will teach you how to import and place a hard IP block in your design.

The open-source Skywater130 PDK does not currently include foundry-published memory macros. Instead, they have a set of OpenRAM configurations which are blessed by the maintainers. You can use those configurations to generate RAM macros from scratch if you are willing to install the OpenRAM utility, or you can download pre-built files.

We will use the sky130_sram_2kbyte_1rw1r_32x512_8 block in this example.

Create a Python script called `sky130_sram_2k.py` to describe the RAM macro in a format which can be imported by SiliconCompiler:

Listing 5: <project_dir>/sky130_sram_2k.py

```
import os
import siliconcompiler


def setup(chip):
    # Core values.
    design = 'sky130_sram_2k'
    stackup = chip.get('option', 'stackup')

    # Create library Chip object.
    lib = siliconcompiler.Library(chip, design)
    lib.register_package_source('vlsida',
                                'git+https://github.com/VLSIDA/sky130_sram_macros',
                                'c2333394e0b0b9d9d71185678a8d8087715d5e3b')
    lib.set('output', stackup, 'gds',
            'sky130_sram_2kbyte_1rw1r_32x512_8/sky130_sram_2kbyte_1rw1r_32x512_8.gds',
            package='vlsida')
    lib.set('output', stackup, 'lef',
            'sky130_sram_2kbyte_1rw1r_32x512_8/sky130_sram_2kbyte_1rw1r_32x512_8.lef',
            package='vlsida')

    rootdir = os.path.dirname(__file__)
    lib.set('output', 'blackbox', 'verilog', os.path.join(rootdir, "sky130_sram_2k.bb.v
↪"))
    # Ensure this file gets uploaded to remote
    lib.set('output', 'blackbox', 'verilog', True, field='copy')

    return lib
```

You will also need a "blackbox" Verilog file to assure the synthesis tools that the RAM module exists: you can call this file `sky130_sram_2k.bb.v`. You don't need a full hardware description of the RAM block to generate an ASIC design, but the open-source workflow needs some basic information about the module:

Listing 6: <project_dir>/sky130_sram_2k.bb.v

```
(* blackbox *)
module sky130_sram_2kbyte_1rw1r_32x512_8 (
`ifdef USE_POWER_PINS
    vccd1,
    vssd1,
`endif
    // Port 0: RW
    input clk0,
    input csb0,
    input web0,
    input [3:0] wmask0,
    input [8:0] addr0,
    input [31:0] din0,
    output reg [31:0] dout0,
    // Port 1: R
    input clk1,
    input csb1,
    input [8:0] addr1,
    output reg [31:0] dout1
);
endmodule
```

Next, you need to create a top-level Verilog module containing one `picorv32` CPU core, one `sky130_sram_2k` memory, and signal wiring to connect their I/O ports together. Note that for the sake of brevity, this module does not include some optional parameters and signals:

Listing 7: <project_dir>/picorv32_top.v

```
`timescale 1 ns / 1 ps

module picorv32_top #(
    parameter [0:0] ENABLE_COUNTERS = 1,
    parameter [0:0] ENABLE_COUNTERS64 = 1,
    parameter [0:0] ENABLE_REGS_16_31 = 1,
    parameter [0:0] ENABLE_REGS_DUALPORT = 1,
    parameter [0:0] LATCHED_MEM_RDATA = 0,
    parameter [0:0] TWO_STAGE_SHIFT = 1,
    parameter [0:0] BARREL_SHIFTER = 0,
    parameter [0:0] TWO_CYCLE_COMPARE = 0,
    parameter [0:0] TWO_CYCLE_ALU = 0,
    parameter [0:0] COMPRESSED_ISA = 0,
    parameter [0:0] CATCH_MISALIGN = 1,
    parameter [0:0] CATCH_ILLINSN = 1,
    parameter [0:0] ENABLE_PCPI = 0,
    parameter [0:0] ENABLE_MUL = 0,
    parameter [0:0] ENABLE_FAST_MUL = 0,
    parameter [0:0] ENABLE_DIV = 0,
    parameter [0:0] ENABLE_IRQ = 0,
    parameter [0:0] ENABLE_IRQ_QREGS = 1,
    parameter [0:0] ENABLE_IRQ_TIMER = 1,
    parameter [0:0] ENABLE_TRACE = 0,
```

(continues on next page)

(continued from previous page)

```
    parameter [0:0] REGS_INIT_ZERO = 0,
    parameter [31:0] MASKED_IRQ = 32'h0000_0000,
    parameter [31:0] LATCHED_IRQ = 32'hffff_ffff,
    parameter [31:0] PROGADDR_RESET = 32'h0000_0000,
    parameter [31:0] PROGADDR_IRQ = 32'h0000_0010,
    parameter [31:0] STACKADDR = 32'hffff_ffff
) (
    input clk,
    resetn,
    output reg trap,

    // Look-Ahead Interface
    output            mem_la_read,
    output            mem_la_write,
    output     [31:0] mem_la_addr,
    output reg [31:0] mem_la_wdata,
    output reg [ 3:0] mem_la_wstrb,

    // Pico Co-Processor Interface (PCPI)
    output reg        pcpi_valid,
    output reg [31:0] pcpi_insn,
    output     [31:0] pcpi_rs1,
    output     [31:0] pcpi_rs2,
    input             pcpi_wr,
    input      [31:0] pcpi_rd,
    input             pcpi_wait,
    input             pcpi_ready,

    // IRQ Interface
    input      [31:0] irq,
    output reg [31:0] eoi,

`ifdef RISCV_FORMAL
    output reg        rvfi_valid,
    output reg [63:0] rvfi_order,
    output reg [31:0] rvfi_insn,
    output reg        rvfi_trap,
    output reg        rvfi_halt,
    output reg        rvfi_intr,
    output reg [ 1:0] rvfi_mode,
    output reg [ 1:0] rvfi_ixl,
    output reg [ 4:0] rvfi_rs1_addr,
    output reg [ 4:0] rvfi_rs2_addr,
    output reg [31:0] rvfi_rs1_rdata,
    output reg [31:0] rvfi_rs2_rdata,
    output reg [ 4:0] rvfi_rd_addr,
    output reg [31:0] rvfi_rd_wdata,
    output reg [31:0] rvfi_pc_rdata,
    output reg [31:0] rvfi_pc_wdata,
    output reg [31:0] rvfi_mem_addr,
    output reg [ 3:0] rvfi_mem_rmask,
    output reg [ 3:0] rvfi_mem_wmask,
```

(continues on next page)

(continued from previous page)

```
    output reg [31:0] rvfi_mem_rdata,
    output reg [31:0] rvfi_mem_wdata,

    output reg [63:0] rvfi_csr_mcycle_rmask,
    output reg [63:0] rvfi_csr_mcycle_wmask,
    output reg [63:0] rvfi_csr_mcycle_rdata,
    output reg [63:0] rvfi_csr_mcycle_wdata,

    output reg [63:0] rvfi_csr_minstret_rmask,
    output reg [63:0] rvfi_csr_minstret_wmask,
    output reg [63:0] rvfi_csr_minstret_rdata,
    output reg [63:0] rvfi_csr_minstret_wdata,
`endif

    // Trace Interface
    output reg        trace_valid,
    output reg [35:0] trace_data
);

    // Memory signals.
    reg mem_valid, mem_instr, mem_ready;
    reg [31:0] mem_addr;
    reg [31:0] mem_wdata;
    reg [ 3:0] mem_wstrb;
    reg [31:0] mem_rdata;

    // No 'ready' signal in sky130 SRAM macro; presumably it is single-cycle?
    always @(posedge clk) mem_ready <= mem_valid;

    // (Signals have the same name as the picorv32 module: use '.*')
    picorv32 rv32_soc (.*);

    // SRAM with always-active chip select and write control bits.
    sky130_sram_2kbyte_1rw1r_32x512_8 sram (
        .clk0  (clk),
        .csb0  ('b0),
        .web0  (!(mem_wstrb != 0)),
        .wmask0(mem_wstrb),
        .addr0 (mem_addr),
        .din0  (mem_wdata),
        .dout0 (mem_rdata),
        .clk1  (clk),
        .csb1  ('b1),
        .addr1 ('b0),
        .dout1 ()
    );
endmodule
```

Finally, your core build script will need to be updated to import the new SRAM Library, and specify some extra parameters such as die size and macro placement:

Listing 8: <project_dir>/picorv32_ram.py

```
#!/usr/bin/env python3

import os
import siliconcompiler


def build_top():
    # Core settings.
    design = 'picorv32_top'
    target = 'skywater130_demo'
    die_w = 1000
    die_h = 1000

    # Create Chip object.
    chip = siliconcompiler.Chip(design)

    # Set default Skywater130 PDK / standard cell lib / flow.
    chip.load_target(target)

    # Set design source files.
    chip.register_package_source(name='picorv32',
                                 path='git+https://github.com/YosysHQ/picorv32.git',
                                 ref='c0acaebf0d50afc6e4d15ea9973b60f5f4d03c42')
    chip.input(os.path.join(os.path.dirname(__file__), f"{design}.v"))
    chip.input("picorv32.v", package='picorv32')

    # Optional: Relax linting and/or silence each task's output in the terminal.
    chip.set('option', 'relax', True)
    chip.set('option', 'quiet', True)

    # Set die outline and core area.
    margin = 10
    chip.set('constraint', 'outline', [(0, 0), (die_w, die_h)])
    chip.set('constraint', 'corearea', [(margin, margin),
                                        (die_w - margin, die_h - margin)])

    # Setup SRAM macro library.
    import sky130_sram_2k
    chip.use(sky130_sram_2k)
    chip.add('asic', 'macrolib', 'sky130_sram_2k')

    # SRAM pins are inside the macro boundary; no routing blockage padding is needed.
    chip.set('tool', 'openroad', 'task', 'route', 'var', 'grt_macro_extension', '0')
    # Disable CDL file generation until we can find a CDL file for the SRAM block.
    chip.set('tool', 'openroad', 'task', 'export', 'var', 'write_cdl', 'false')
    # Reduce placement density a bit to ease routing congestion and to speed up the␣
↪route step.
    chip.set('tool', 'openroad', 'task', 'place', 'var', 'place_density', '0.5')

    # Place macro instance.
    chip.set('constraint', 'component', 'sram', 'placement', (500.0, 250.0, 0.0))
```

(continues on next page)

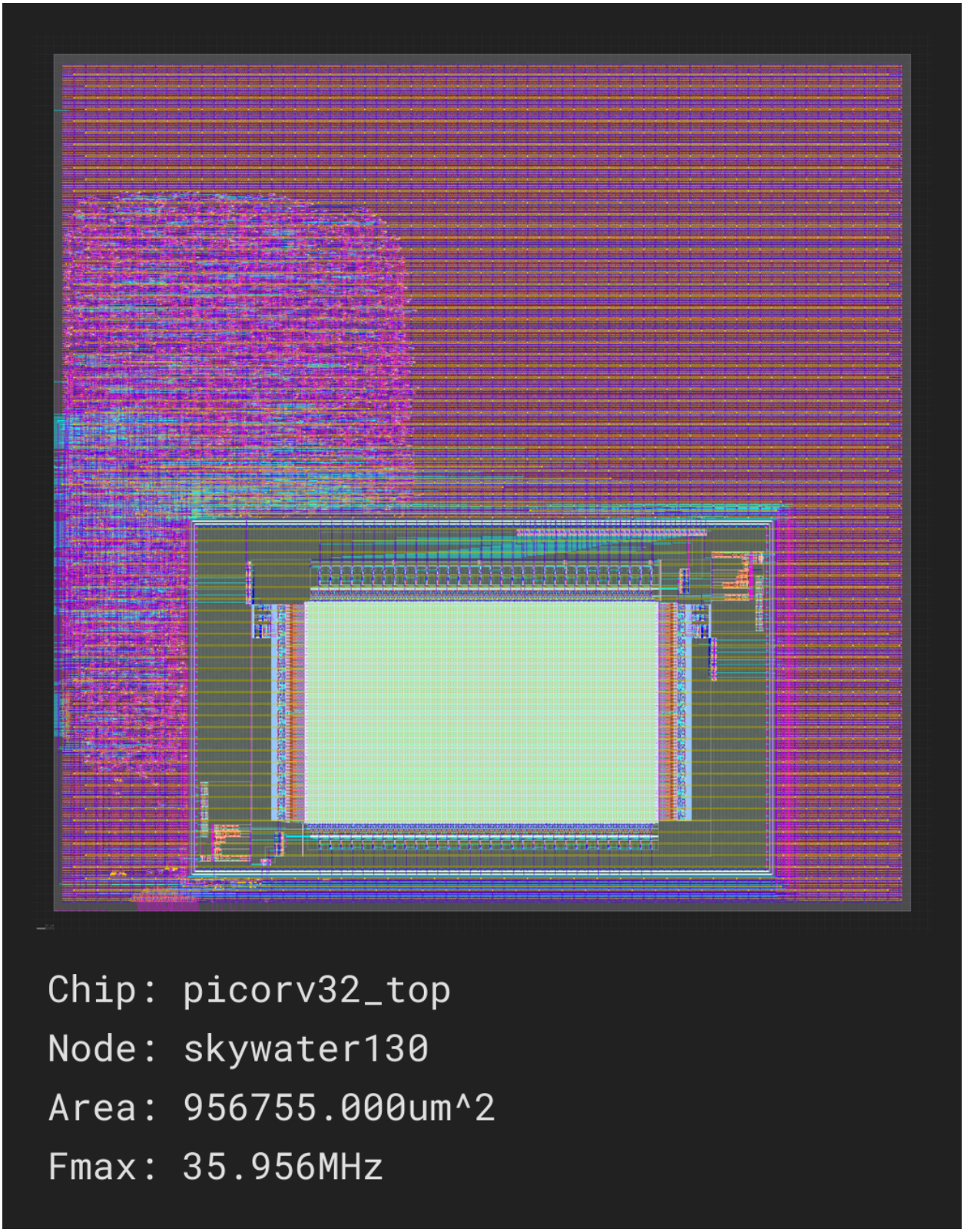
(continued from previous page)

```
    chip.set('constraint', 'component', 'sram', 'rotation', 180)

    # Set clock period, so that we won't need to provide an SDC constraints file.
    chip.clock('clk', period=25)

    # Run the build.
    chip.set('option', 'remote', False)
    chip.set('option', 'quiet', False)

    chip.run()

    # Print results.
    chip.summary()

    return chip


if __name__ == '__main__':
    build_top()
```

With all of that done, your project directory tree should look something like this:

```
<rundir>
├── sky130_sram_2k.bb.v
├── sky130_sram_2k.py
├── picorv32.py
├── picorv32_ram.py
└── picorv32_top.v
```

Your `picorv32_ram.py` build script should take about 20 minutes to run on the cloud servers if they are not too busy, with most of that time spent in the routing task. As with the previous designs, you should see updates on its progress printed every 30 seconds, and you should receive a screenshot once the job is complete and a report in the build directory:


Chip: picorv32_top
Node: skywater130
Area: 956755.000um^2
Fmax: 35.956MHz

1.6.4 Extending your design

Now that you have a basic understanding of how to assemble modular designs using SiliconCompiler, why not try building a design of your own creation, or adding a custom accelerator to your new CPU core?

1.7 Hardware Design Frontends

In addition to traditional hardware description languages like Verilog and SystemVerilog, SiliconCompiler also supports higher-level languages, for ease of hardware design prototyping.

Here are some tutorials of a few different types of high-level hardware design frontends:

- Python-based frontends (Migen)
- *Chisel frontend*
- *C HLS Frontend (Bambu)*
- *Bluespec frontend*

1.7.1 Python-based frontends

Since SC itself is a Python library, it can be used as-is in an end-to-end build script with any Python-based HDL that can be scripted to export designs to Verilog.

For example, if you have SC installed already, you can quickly get started building designs written in the Migen HDL. To install Migen, run `pip install migen`. Then, paste the following into a file called "heartbeat_migen.py".

```
from migen import Module, Signal, Cat, Replicate
from migen.fhdl.verilog import convert

import siliconcompiler


class Heartbeat(Module):
    def __init__(self, N=8):
        self.out = Signal()
        self.counter_reg = Signal(N)

        ###

        self.sync += self.counter_reg.eq(self.counter_reg + 1)
        self.sync += self.out.eq(self.counter_reg == Cat(Replicate(0, N - 1), 1))


def main():
    heartbeat = Heartbeat()
    convert(heartbeat, ios={heartbeat.out}, name='heartbeat').write('heartbeat.v')

    chip = siliconcompiler.Chip('heartbeat')
    chip.input('heartbeat.v')
    # default Migen clock pin is named 'sys_clk'
    chip.clock(pin='sys_clk', period=1)
    chip.load_target("freepdk45_demo")
```

(continues on next page)

(continued from previous page)

```
    chip.run()
    chip.summary()
    chip.show()


if __name__ == '__main__':
    main()
```

Run this file with `python heartbeat_migen.py` to compile your Migen design down to a GDS and automatically display it in KLayout.

In this example, the `Heartbeat` class describes a design as a Migen module, and the `main()` function implements the build flow. The flow begins by using Migen's built-in functionality for exporting the design as a file named "heartbeat.v". The rest of the flow uses SC's core API to take this Verilog file and feed it into a basic *asicflow* build, as described in the *Quickstart guide*. For more info on how Migen works, see the Migen docs.

Although we wrote this example using Migen in particular, the concepts apply to other Python-based HDLs, such as MyHDL or Amaranth (note that Amaranth's Verilog backend requires Yosys installed locally).

1.7.2 Chisel frontend

SiliconCompiler has a *Chisel* frontend that enables you to build Chisel designs for any supported SC target. To get started using Chisel with SC, ensure that SC is installed following the directions from the *Installation* section, and install sbt. See for links to helpful build *scripts*.

To build a Chisel design, the only things you need to do differently from a configuration perspective are:

1) Add all required Scala files as sources. Keep in mind that other frontend-specific features such as include or library directories are not yet supported for the Chisel frontend.

2) Set the *['option', 'frontend']* parameter to 'chisel'.

Otherwise, you can configure the build as normal.

For example, to build the GCD example from the Chisel project template repo, first copy the following code into a file called "GCD.scala".

```
import chisel3._

/**
  * Compute GCD using subtraction method.
  * Subtracts the smaller from the larger until register y is zero.
  * value in register x is then the GCD
  */
class GCD extends Module {
  val io = IO(new Bundle {
    val value1        = Input(UInt(16.W))
    val value2        = Input(UInt(16.W))
    val loadingValues = Input(Bool())
    val outputGCD     = Output(UInt(16.W))
    val outputValid   = Output(Bool())
  })

  val x  = Reg(UInt())
  val y  = Reg(UInt())
```

(continues on next page)

(continued from previous page)

```
  when(x > y) { x := x - y }
    .otherwise { y := y - x }

  when(io.loadingValues) {
    x := io.value1
    y := io.value2
  }

  io.outputGCD := x
  io.outputValid := y === 0.U
}
```

Note: SC's Chisel driver script selects the module to build based on the 'design' parameter. You must ensure that top-level module's class name matches the 'design' parameter you have set, and that this module does not include a `package` statement.

This design can then be quickly compiled to a GDS using the command line:

```
#!/bin/bash

sc GCD.scala -frontend chisel
sc-show -design GCD
```

Or using Python:

```
#!/usr/bin/env python3

import siliconcompiler
import os


def main():
    root = os.path.dirname(__file__)
    chip = siliconcompiler.Chip('GCD')
    chip.input(os.path.join(root, "GCD.scala"))
    chip.set('option', 'frontend', 'chisel')
    # default Chisel clock pin is 'clock'
    chip.clock(pin='clock', period=5)
    chip.load_target("freepdk45_demo")
    chip.run()
    chip.summary()
    chip.show()


if __name__ == '__main__':
    main()
```

For more information on creating designs using Chisel, see the Chisel docs.

1.7.3 C HLS frontend

SiliconCompiler supports high-level synthesis of C code to any supported SC target, implemented using the *Bambu* HLS tool. To get started compiling C code with SC, ensure that SC is installed following the directions from the *Installation* section, and build Bambu from source. See for links to helpful build *scripts*.

To build a C design, the only things you need to do differently from a configuration perspective are:

1) Add all required C files as inputs.

2) Set the *['option', 'frontend']* parameter to 'c'.

Otherwise, you can configure the build as normal.

For example, to implement a GCD function as a circuit, first copy the following into a file called "gcd.c".

```c
#include <stdio.h>

short gcd(short a, short b) {
    if (b > a) {
        short tmp = a;
        a = b;
        b = tmp;
    }

    while (a != 0 && b != 0) {
        short r = a % b;
        a = b;
        b = r;
    }

    if (a == 0)
        return b;
    else
        return a;
}

int main() {
    printf("gcd(4, 4) = %d\n", gcd(4, 4));
    printf("gcd(27, 36) = %d\n", gcd(27, 36));
    printf("gcd(270, 192) = %d\n", gcd(270, 192));

    return 0;
}
```

Note: SC's C frontend driver script selects a function to implement as a Verilog module using the 'design' parameter. Ensure that your C code includes a function that matches the value stored in 'design'.

This design can then be quickly compiled to a GDS using the command line:

```
sc gcd.c -frontend c
sc-show -design gcd
```

Or using Python:

```
import siliconcompiler
import os


def main():
    root = os.path.dirname(__file__)
    chip = siliconcompiler.Chip('gcd')
    chip.input(os.path.join(root, "gcd.c"))
    chip.set('option', 'frontend', 'c')
    # default Bambu clock pin is 'clock'
    chip.clock(pin='clock', period=5)
    chip.load_target("freepdk45_demo")
    chip.run()
    chip.summary()
    chip.show()


if __name__ == '__main__':
    main()
```

For more information on the Bambu project used for implementing this frontend, see their docs.

1.7.4 Bluespec frontend

SiliconCompiler has a Bluespec frontend that enables you to build *bluespec* designs for any supported SC target. To get started using Bluespec with SC, ensure that SC is installed following the directions from the *Installation* section, and download bsc or install it from source following the directions here. See for links to helpful build *scripts*.

To build a Bluespec design, the only things you need to do differently from a configuration perspective are:

1) Add the Bluespec top-level package as an 'input', and add all directories containing imported modules as entries in *['option', 'ydir']*. Keep in mind that the Bluespec integration only supports specifying a single top-level source file, so you must use *['option', 'ydir']* for all other sources.

2) Set the *['option', 'frontend']* parameter to 'bluespec'.

Otherwise, you can configure the build as normal.

For example, to build this fibonacci example adapted from the bsc smoke test, first copy the following code into a file called "FibOne.bsv".

```
interface FibOne_IFC;
   method Action nextFib;
   method ActionValue #(int)  getFib;
endinterface

(* synthesize *)
module mkFibOne(FibOne_IFC);
   // register containing the current Fibonacci value
   Reg#(int) this_fib <- mkReg (0);
   Reg#(int) next_fib <- mkReg (1);

   method Action nextFib;
      this_fib <= next_fib;
```

(continues on next page)

(continued from previous page)

```
      next_fib <= this_fib + next_fib;  // note that this uses stale this_fib
  endmethod

  method ActionValue#(int) getFib;
    return this_fib;
  endmethod

endmodule: mkFibOne
```

Note: SC's Bluespec driver script selects the module to build based on the 'design' parameter. You must ensure that the single file passed in via the 'source' parameter contains a module name that matches the value in 'design'.

This design can then be quickly compiled to a GDS using the command line:

```
sc FibOne.bsv -design mkFibOne -frontend bluespec
sc-show -design mkFibOne
```

Or using Python:

```
import siliconcompiler
import os


def main():
    root = os.path.dirname(__file__)
    chip = siliconcompiler.Chip('mkFibOne')
    chip.input(os.path.join(root, "FibOne.bsv"))
    chip.set('option', 'frontend', 'bluespec')
    # default Bluespec clock pin is 'CLK'
    chip.clock(pin='CLK', period=5)
    chip.load_target("freepdk45_demo")
    chip.run()
    chip.summary()
    chip.show()


if __name__ == '__main__':
    main()
```

For more information on creating designs using Bluespec, see the Bluespec docs.

1.8 Multi-Job Flows and Automation

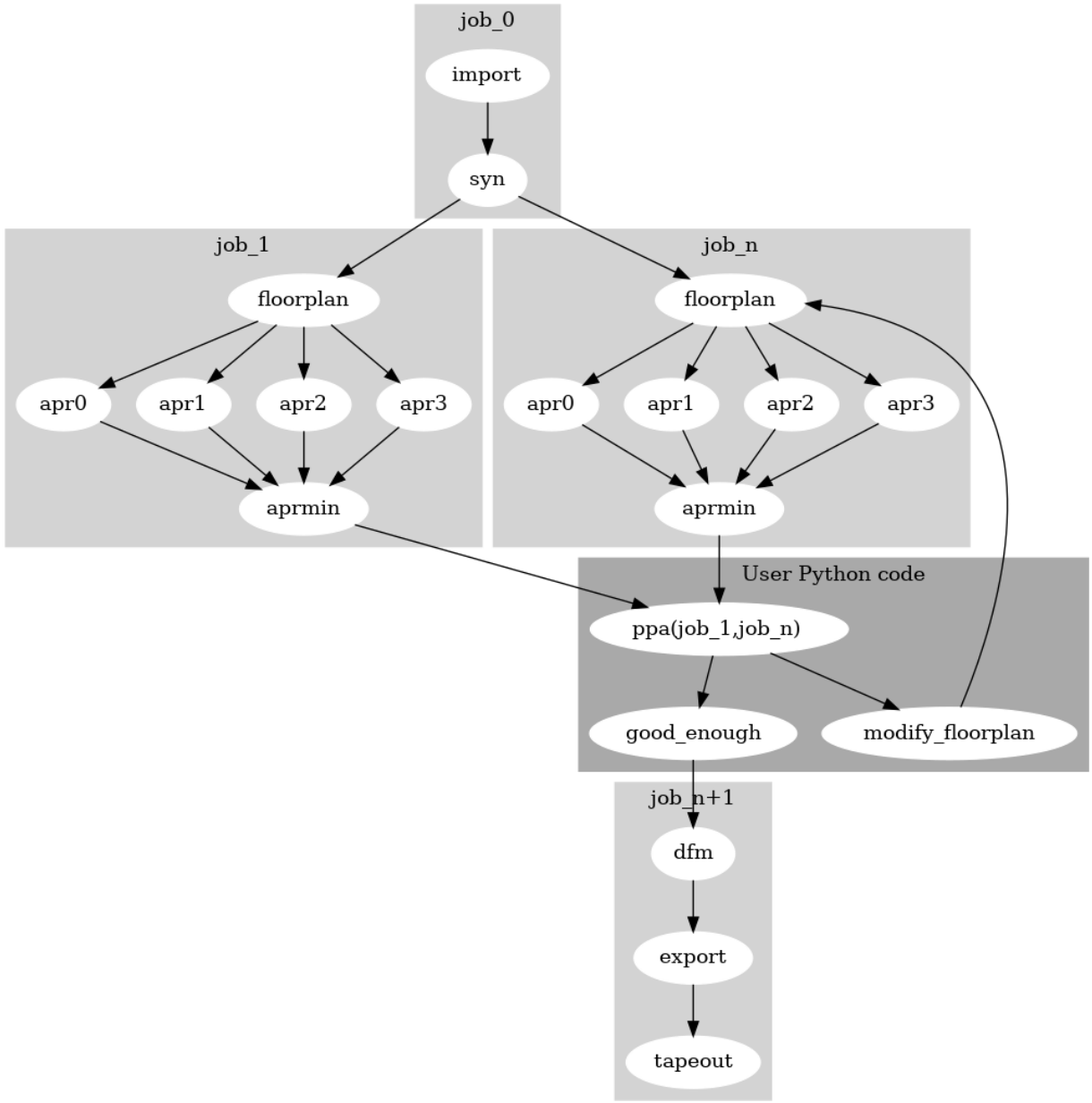
As an extension of *Compilation Process*, which describes setting up only one job, you can link together different jobs and Python manipulation code for your own purposes.

At the end of each *run()* call, the current in-memory job schema entries are copied into a job history dictionary for reference later. The user can access these to create more complex, non-linear flows that take into account run history and gradients. The code snippet below shows a minimal sequence leveraging the multi-job feature.:

```
chip.run()
chip.set('option', 'jobname', 'newname')
chip.set('some parameter..')
chip.run()
```

Complex iterative compilation flows can be created with Python programs that:

1. Calls run() multiple times using a different jobname, and
2. Leverages Python logic to query per job metrics to control the compilation flow decision, for automation



Note: **[In Progress]** This tutorial requires a more detailed step-by-step guide.

1.9 Parallel Job Execution

Single threaded program performance has saturated, so if we want to make hardware compilation fast, we need to figure out how to make effective use of massively parallel hardware.

Working in our favor is the fact that 1.) the data to compute ratio for the most compute intensive compilation steps is very high, and 2.) some of those steps can be easily partitioned into embarrassingly parallel problems.

In this tutorial, we show how the SiliconCompiler flowgraph execution model can be used to achieve an order of magnitude speedup on a single workstation compared to single threaded loops. Speedups within cloud execution is even higher.

The tutorial runs the same design with three different approaches:

1.) Completely serial (two nested for loops N * M).

2.) One blocking for loop (N) to launch runs with parallel index launches for synthesis, placement, cts, and routing steps.

3.) One asynchronous for loop leveraging the Python multiprocessing package to launch N independent flows.

Run the program on your machine to see what kind of speedup you get! Here is example code.

```
#!/usr/bin/env python3

# Copyright 2020 Silicon Compiler Authors. All Rights Reserved.

import multiprocessing
import siliconcompiler
import time
import os


# Shared setup routine
def run_design(design, M, job):
    root = os.path.dirname(__file__)

    chip = siliconcompiler.Chip(design, loglevel='INFO')
    chip.input(os.path.join(root, f"{design}.v"))
    chip.input(os.path.join(root, f"{design}.sdc"))
    chip.set('option', 'jobname', job)
    chip.set('option', 'relax', True)
    chip.set('option', 'quiet', True)
    asic_flow_args = {
        'syn_np': M,
        'place_np': M,
        'cts_np': M,
        'route_np': M
    }
    chip.load_target("freepdk45_demo", **asic_flow_args)
    chip.run()


def all_serial(design='heartbeat', N=2, M=2):
    serial_start = time.time()
    for i in range(N):
```

(continues on next page)

(continued from previous page)

```
        for j in range(M):
            job = f"serial_{i}_{j}"
            run_design(design, 1, job)
    serial_end = time.time()

    return serial_start, serial_end


def parallel_steps(design='heartbeat', N=2, M=2):
    parastep_start = time.time()
    for i in range(M):
        job = f"parasteps_{i}"
        run_design(design, M, job)
    parastep_end = time.time()

    return parastep_start, parastep_end


def parallel_flows(design='heartbeat', N=2, M=2):
    paraflow_start = time.time()

    processes = []

    for i in range(N):
        job = f"paraflows_{i}"
        processes.append(multiprocessing.Process(target=run_design,
                                                 args=(design,
                                                       M,
                                                       job)))

    # Boiler plate start and join
    for p in processes:
        p.start()
    for p in processes:
        p.join()

    paraflow_end = time.time()

    return paraflow_start, paraflow_end


def main():

    ####################################
    design = 'heartbeat'
    N = 2  # parallel flows, change based on your machine
    M = 2  # parallel indices, change based on your machine

    ####################################
    # 1. All serial

    serial_start, serial_end = all_serial(design=design, N=N, M=M)
```

(continues on next page)

(continued from previous page)

```
    ###################################
    # 2. Parallel steps

    parastep_start, parastep_end = parallel_steps(design=design, N=N, M=M)

    ###################################
    # 3. Parallel flows

    paraflow_start, paraflow_end = parallel_flows(design=design, N=N, M=M)

    ###################################
    # Benchmark calculation

    paraflow_time = round(paraflow_end - paraflow_start, 2)
    parastep_time = round(parastep_end - parastep_start, 2)
    serial_time = round(serial_end - serial_start, 2)

    print(f" Serial = {serial_time}s\n",
          f"Parallel steps = {parastep_time}s\n",
          f"Parallel flows = {paraflow_time}s\n")


if __name__ == '__main__':
    main()
```

1.10 Dashboard Tutorial

At this point, you should have built a chip on your computer.

To start, run the command:

```
sc-dashboard -cfg <path to manifest>
```

You can specify the port by adding a port flag:

```
sc-dashboard -cfg <path to manifest> -port <port number>
```

And/or you can include extra chips by adding a `-graph_cfg` flag:

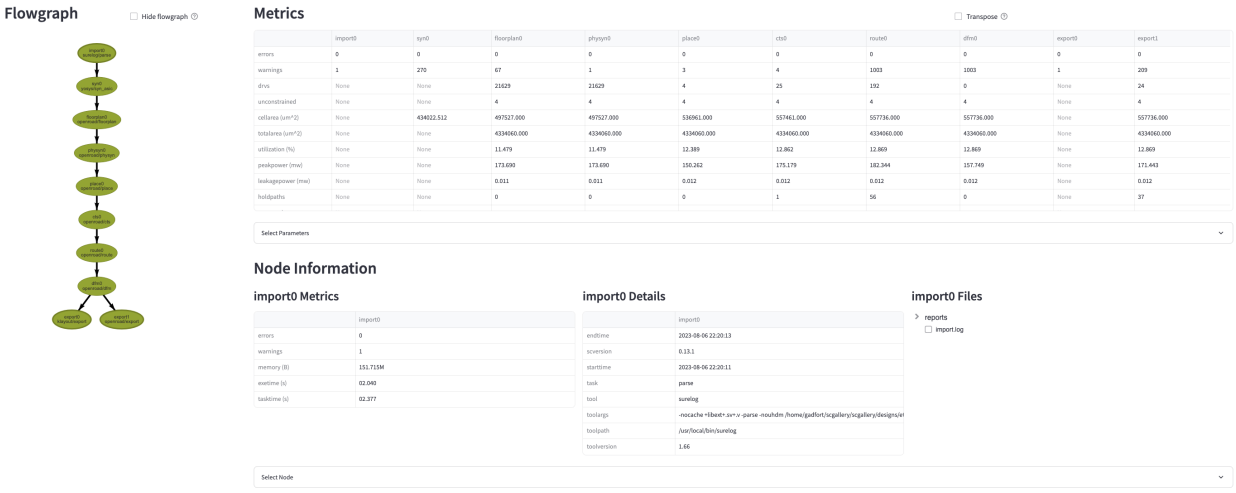
```
sc-dashboard -cfg <path to manifest> -graph_cfg <manifest name> <path to manifest> -
↪graph_cfg <manifest name> <path to manifest>
```

1.10.1 Header

In the header, you can select any of the chips that are part of the history of the chip loaded in and any of the other chips loaded in through the `-graph_cfg` flag (see arrow below).

1.10.2 Metrics Tab

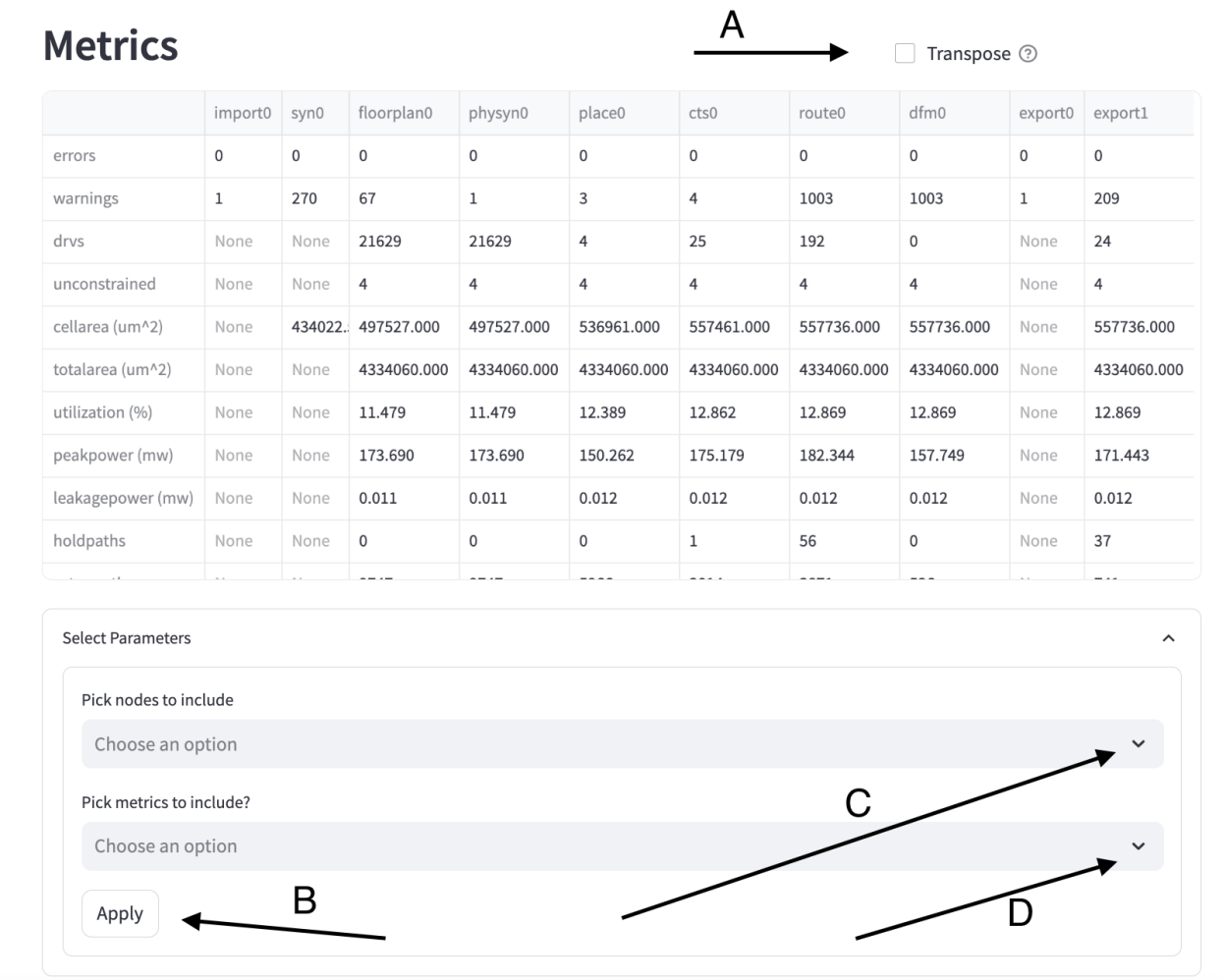
You will load into the metrics tab similar to the one below upon running *sc-dashboard*.



Metrics Section

The metrics section displays an overview of each value for each metric tracked for each node. Click “Transpose” to switch the axes (arrow A in image below).

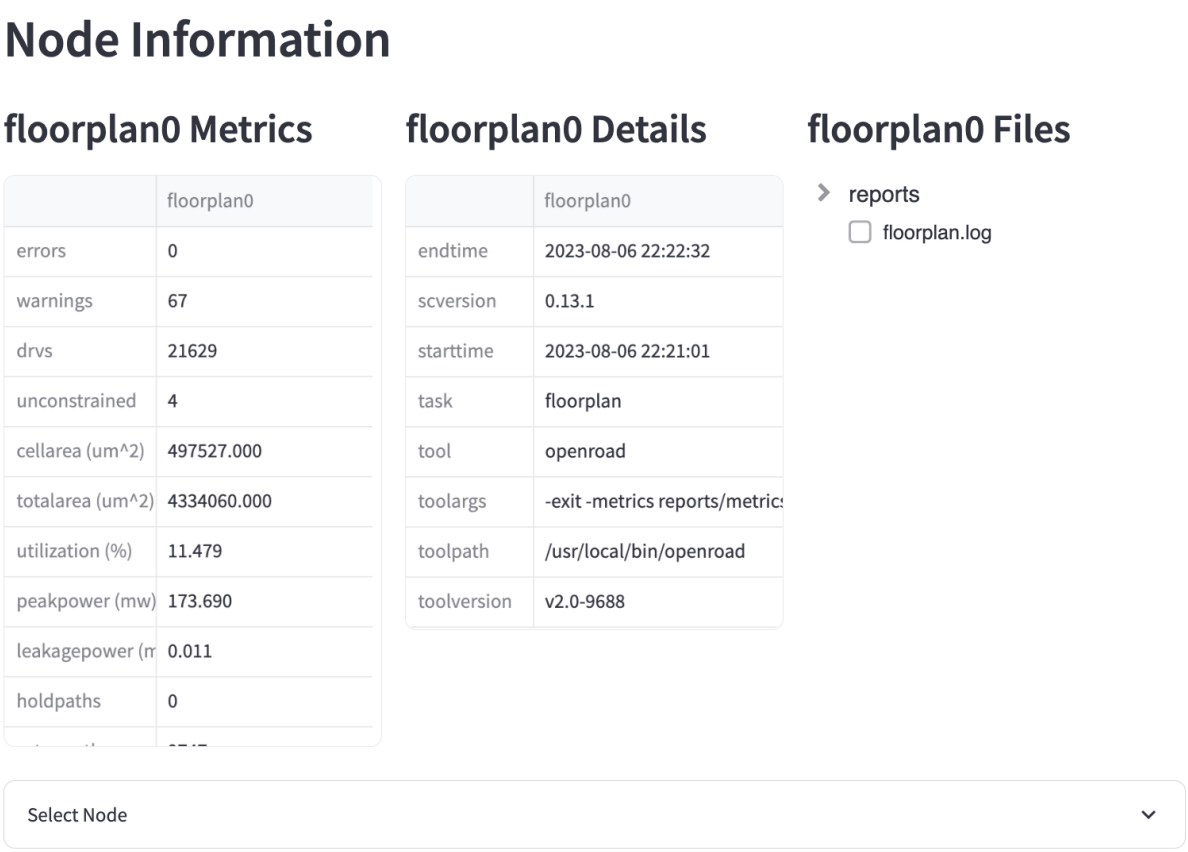
You may use the “Select Parameter” expander to specify certain nodes and/or metrics (arrow C and D in image below). Click “Apply” to make those changes (arrow B in image below).



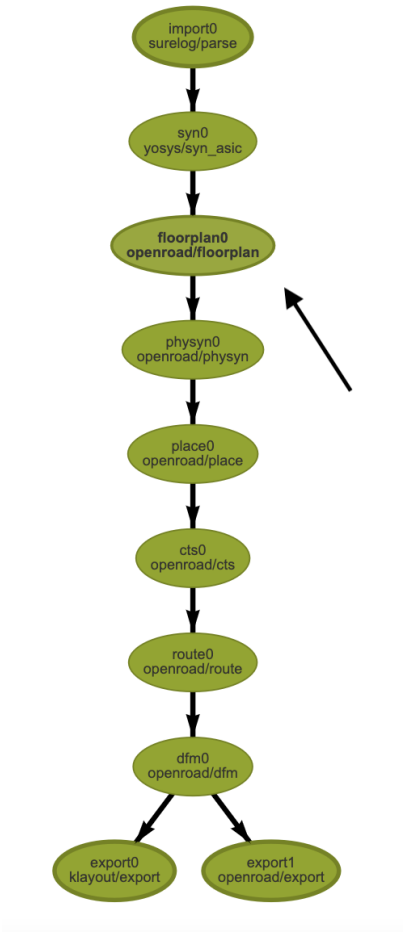
Node Information Section

Below is the node information section. It consists of three subsections - node metrics, node details, and node files.

You can select a node using the “Select Node” expander as seen with arrow’s A and B below. Click “Apply” to make the change.



Alternatively, you can double click on the flowgraph node. Nodes that are selected will bolden.



Node Metrics Subsection

The node metrics subsection consists of all of not `None` values recorded for each of the metrics recorded for the selected node.

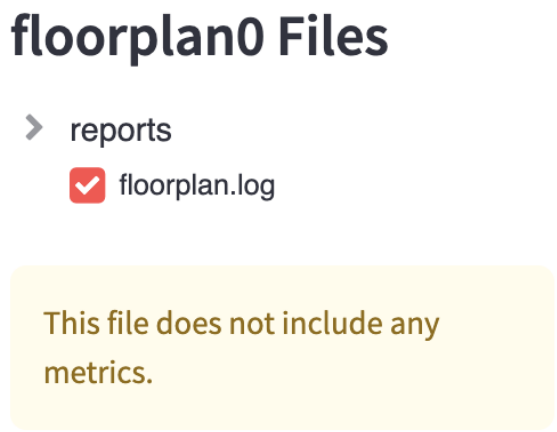
Node Details Subsection

The node details subsection consists of all of the characteristics about this node that are not reflected in the metrics section.

Node Files Subsection

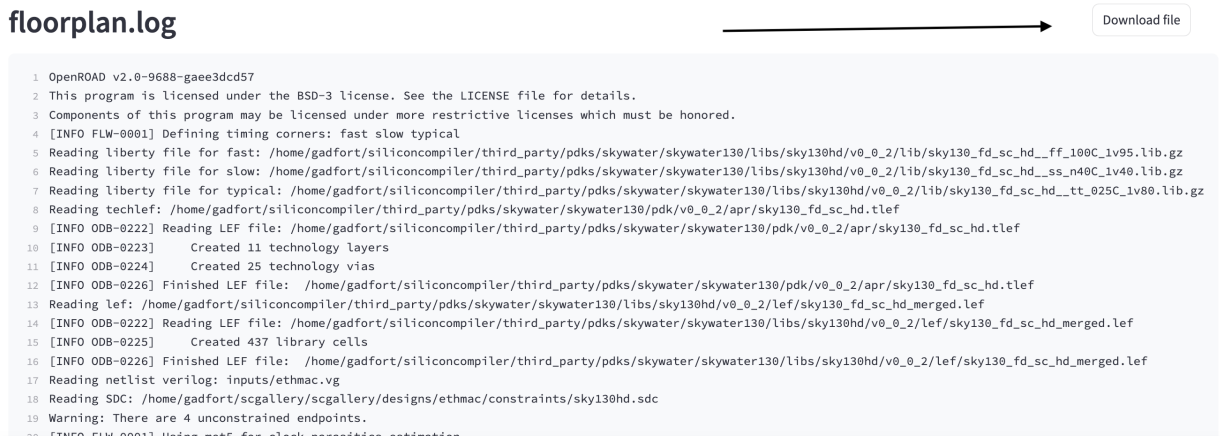
The node files subsection consists of all of the files for a given node that are in the build directory.

Selecting a node will display a list of the metrics that the file informs below the file tree.

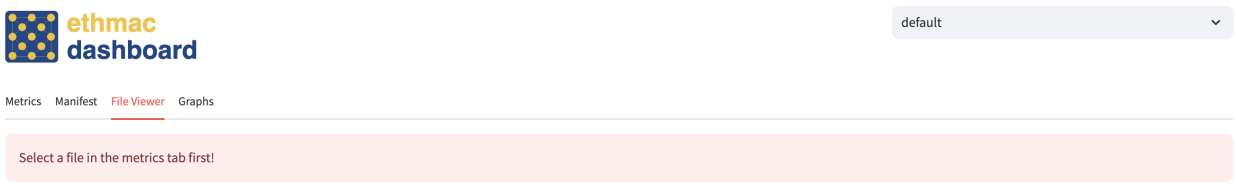


1.10.3 File Viewer Tab

The selected node you clicked in the *Node Files Subsection* will appear here. You may download the file by clicking the download button as shown below.



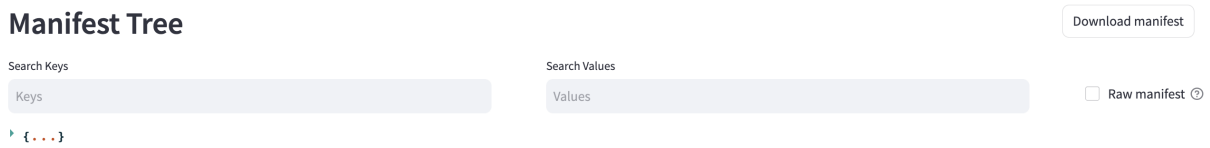
If no file is selected, the error message below will be displayed telling you to select a file first.



1.10.4 Manifest Tab

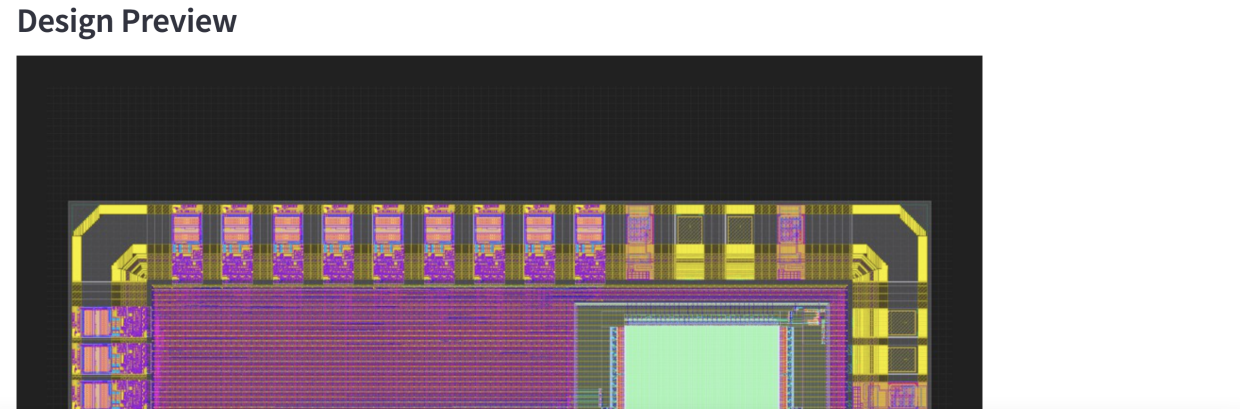
The next tab you can select is the manifest tab. This displays the manifest after it has been filtered through to make it more readable.

To view the manifest, click the arrow on the dictionary (arrow A). The search bars will return partial matches for either the keys (arrow B in image below) or the values (arrow C in image below). You may download the JSON as you view it at any point (arrow D in image below). You can view the raw manifest by clicking the checkbox to the right of the search bar (arrow E in image below).



1.10.5 Display Preview Tab

This displays the preview image of the chip if there is one in the directory (example given below). If not, this tab will not be included.

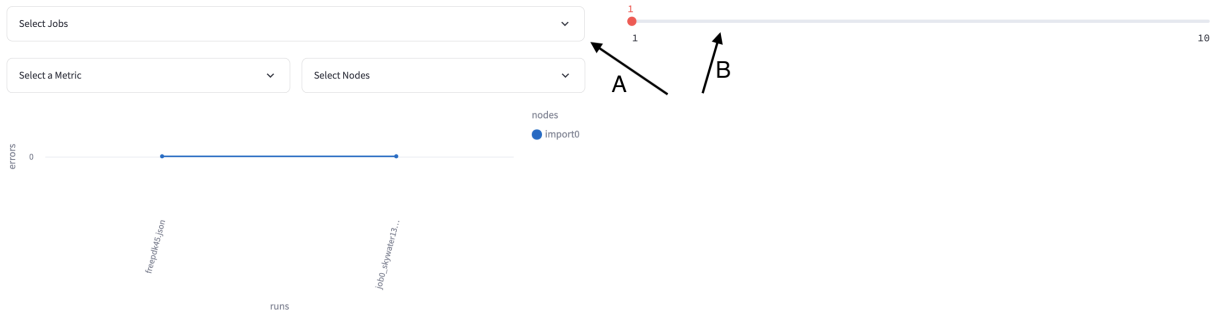


1.10.6 Graphs Tab

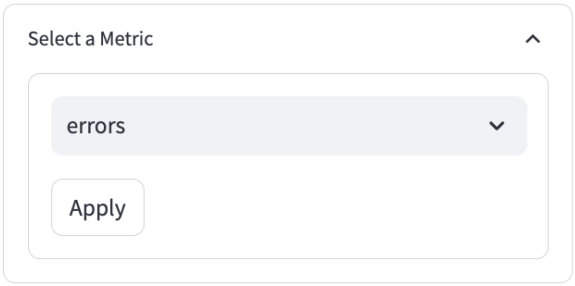
This tab is meant to make comparisons between nodes for a given metric over many chip objects.

At the top of the panel, select which runs/jobs to include for all the graphs (arrow A in image below). T

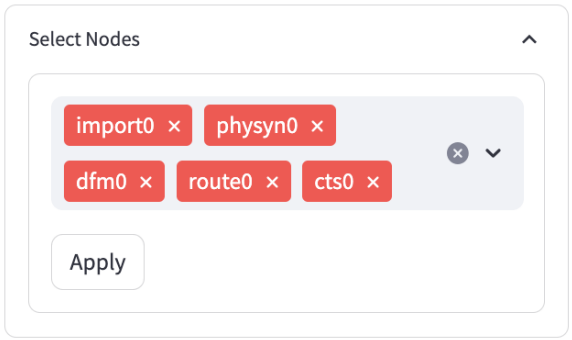
Move the slider to add more graphs or remove old ones (arrow B in image below).



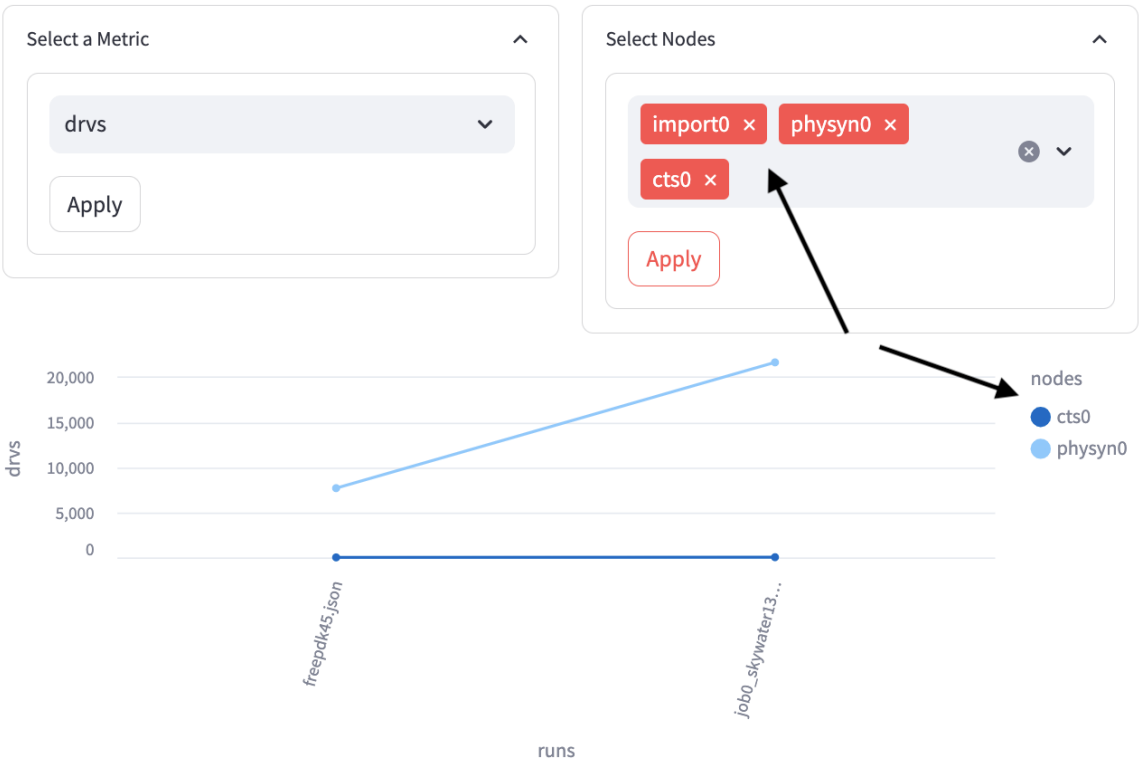
For each graph, you must select one metric (show in image below).



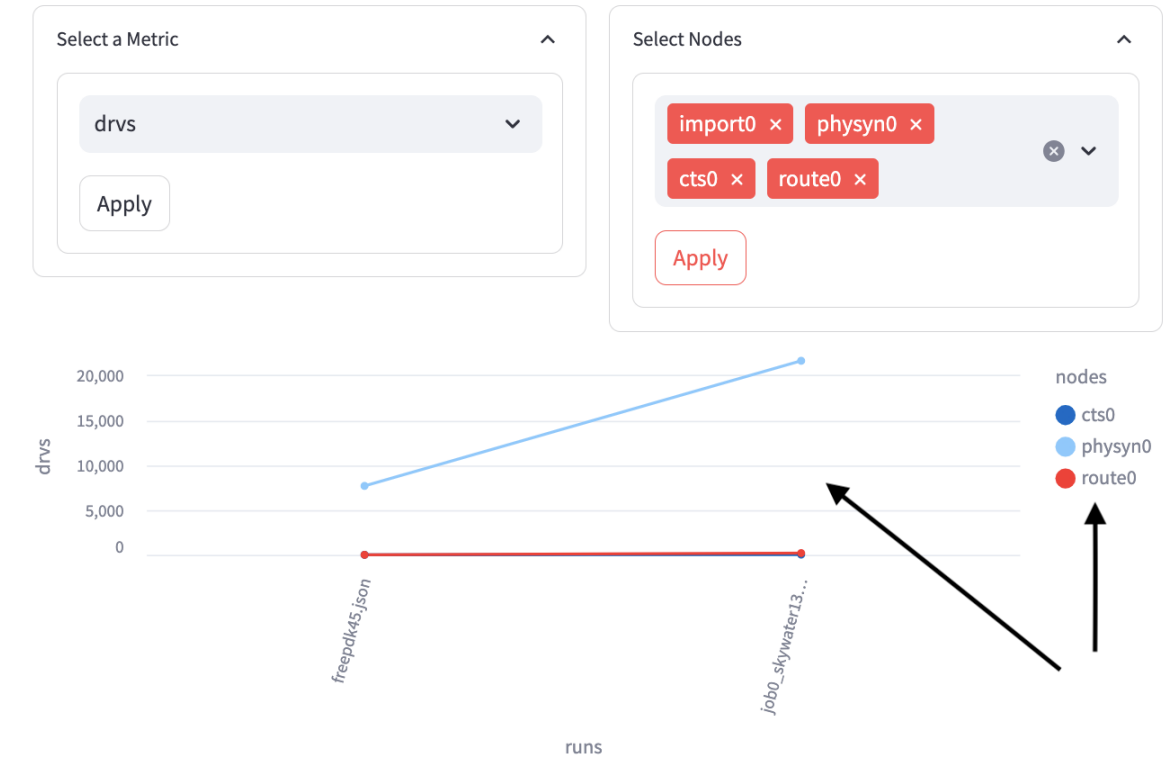
You may select any amount of nodes (show in image below).



Sometimes nodes may not have values for a metric, in which case they will not be included in the graph. In the image below, import0 is not in the legend.



Sometimes nodes that are in the legend are not visible on the graph. What has happened is that they have the exact same values as some other node. Consider deselecting other nodes in this case. In the image below, cts0 is barely visible on the graph.



1.11 Glossary

The following set of terms represents fundamental SiliconCompiler definitions used throughout the documentation.

chip
: Instance of SiliconCompiler Chip() class used to compile a design.

default
: Reserved SiliconCompiler schema key that can be replaced by any legal string.

dictionary
: Associative array, ie. a collection of key-value pairs.

edge
: A directed connection between a tail node and head nodes in a flowgraph.

flowgraph
: A directed acyclic graph specification of the hardware compilation.

index
: A compilation step scenario operating on input data.

job
: Execution of complete or partial compilation flowgraph.

keys
: Immutable strings used as index into dictionary.

keypath
: Ordered list of keys used to access schema parameters.

keywords
Reserved strings that cannot be used as key names.

list
An ordered and mutable sequence of elements.

manifest
JSON file representation of the SiliconCompiler schema.

node
An task in the flowgraph.

parameter
Schema leaf cell with a set of pre-defined key/value pairs.

program
User specified program with one (or more) chip instances.

schema
Nested dictionary of parameters.

step
A discrete function in a flowgraph.

task
An atomic (step, index) task to be executed.

tool
Executable associated with a task in a flowgraph.

1.12 FAQ

This is a list of Frequently Asked Questions about SiliconCompiler. Feel free to suggest new entries!

1.12.1 How do I...

... set up a new tool?

See *Pre-Defined Tools*

... set up a new flow?

See *Flows*

... set up a new pdk?

See *Pre-Defined PDKs*

... set up a new library?

See *Libraries*

... set up a new target?

See *Pre-Defined Targets*

... create a chip object?

```
import siliconcompiler
chip = siliconcompiler.Chip('<design>')
```

... run a compilation?

```
chip.run()
```

... display my layout?

```
chip.show()
```

... display a previous run from the command line?

```
sc-show -design <name>
```

... change the logger level?

```
chip = siliconcompiler.Chip('<design>', loglevel=<INFO|DEBUG|WARNING|ERROR>)
chip.set('option', 'loglevel', <level>)
```

... check my setup before running?

```
chip.check_manifest()
```

... relax the parse constraints on import?

```
chip.set('option', 'relax', True)
```

... change the build directory?

```
chip.set('option', 'builddir', <dirpath>)
```

... use the setup json file from a previous run?

```
chip.read_manifest(<filepath>)
```

... drive custom TCL code into the a target EDA flow?

```
chip.set('tool', <tool>, 'task', <task>, 'prescript', <file>, step=<step>,␣
↪index=<index>)
chip.set('tool', <tool>, 'task', <task>, 'postscript', <file>, step=<step>,␣
↪index=<index>)
```

... control the thread parallelism for a tool?

```
chip.set('tool', <tool>, 'task', <task>, 'threads', <n>, step=<step>, index=
↪<index>)
```

... resume a previous run?

```
chip.set('option', 'resume', True)
```

... print the description of a parameter?

```
print(chip.help(keypath))
```

CHAPTER

TWO

ADVANCED GUIDE

The following sections describe how to build your own custom modules in SiliconCompiler so that you can customize your own flow. If you don't plan to build your own modules and just want to use SiliconCompiler with pre-defined modules, see the *Reference Manual*.

2.1 Contributing modules

The SiliconCompiler project was designed to encourage contribution. Theoretically, the project could support 100's of process PDKs, 100's of tools, and countless flows, but the project maintainers couldn't possibly manage all of them without community help.

Note: Before making a PR, make sure you have the right to do so and you are not violating any potential NDAs and copyright law. In general, PDK modules should only be published by foundries and tool modules should only be published by the tool owners.

The process for target contributions is as follows:

1.) Clone the SiliconCompiler project from the GitHub Repository and follow the *Installation* instructions.

2.) Create a *flow*, *pdk*, *library*, *target*, or *tool* using the existing modules as guides. Place the module file in the appropriate location per the directory structure shown below:

```
.
├── flows
│   ├── asicflow.py
│   ├── dvflow.py
│   └── fpgaflow.py
│   └── ...
├── libs
│   ├── asap7sc7p5t.py
│   ├── nangate45.py
│   └── sky130hd.py
│   └── ...
├── pdks
│   ├── asap7.py
│   ├── freepdk45.py
│   └── skywater130.py
│   └── ...
├── targets
│   ├── asap7_demo.py
```

(continues on next page)

(continued from previous page)

```
    ├── freepdk45_demo.py
    └── skywater130_demo.py
    └── ...
└── tools
    ├── klayout
    │   ├── klayout.py
    │   └── ...
    ├── openroad
    │   ├── openroad.py
    │   ├── sc_apr.tcl
    │   └── ...
    ├── verilator
    │   └── verilator.py
    ├── yosys
    │   ├── yosys.py
    │   ├── sc_syn.tcl
    │   └── ...
    └── <...>
```

3.) Test the new target by calling *use()*. For example.

```
import <newpdk>
chip.use(<newpdk>)
```

4.) Read the CONTRIBUTING guide to learn how to submit a pull request.

2.2 Targets

To facilitate encapsulation and reuse of schema parameters related to design targets, SiliconCompiler implements a `Chip.load_target()` function which can run scripts that set up common combinations of *siliconcompiler.Flow*, *siliconcompiler.PDK*, *siliconcompiler.Library*, and *siliconcompiler.Checklist* modules.

SiliconCompiler comes with a set of built-in targets, which can be pulled in using the *load_target()* function. A full list of built-in targets can be found on the *Pre-Defined Targets* page.

All target modules must contain a function called `setup()`, which takes in a *siliconcompiler.Chip* object and can modify the Chip's schema parameters in any way. It's common for targets to load at least one flow, a PDK and at least one standard cell library if the design is being built as an ASIC. They can also set up default design parameters and tool options. Targets should also include a `make_docs()` function which provides a descriptive docstring and returns a *siliconcompiler.Chip* object with the target loaded.

SC supports additional levels of encapsulation through PDK, library, and flow modules. Unlike targets, these modules are imported as Python libraries and pulled into the *siliconcompiler.Chip* object with the *use()* method. See the *PDK*, *Library*, and *Flow* User Guide pages to learn more about what is expected to be configured in each of these modules.

Generally, these functions will be called by targets, and then a user will only have to call *load_target()* in their build script. However, the `run()` function requires all mandatory flowgraph, pdk, and tool settings to be defined prior to execution, so if a partial target is loaded, additional setup may be required.

The following example calls the *load_target()* function to load the built-in *freepdk45_demo* target.

```
chip.load_target('freepdk45_demo')
```

The following example demonstrates the functional equivalent at the command line:

```
sc hello.v -target "freepdk45_demo"
```

Targets can also be dedicated to individual projects or use cases, rather than general-purpose processing. For example, we ship a self-test target with SiliconCompiler, which builds a simple 8-bit counter to verify that everything is installed and configured correctly:

```
sc -target "asic_demo"
```

A full list of built-in demo targets can be found on the *Pre-Defined Targets* page.

2.3 Flows

SiliconCompiler flows are created by configuring the *['flowgraph', ...]* parameters within the schema. To simplify reuse of complex flows, the project includes standardized interfaces for bundling flowgraph settings as reusable named modules.

Similar to other types of SiliconCompiler modules, flows are loaded by passing a *siliconcompiler.Flow* object into the *use()* function before a run is started. *siliconcompiler.Flow* objects typically use the *node()* and *edge()* functions to configure a "flowgraph" which represents a hierarchical collection of tasks to execute.

2.3.1 setup(chip)

A SiliconCompiler flowgraph consists of a set of connected nodes and edges, where a node is an executable tool performing some ("task"), and an edge is the connection between those tasks. The first task in the flowgraph must be named 'import'.

```
flow.node(flow, 'import', <import_tool>, 'import')
flow.node(flow, <next_step>, <next_tool>, <next_task>)
flow.edge(flow, 'import', <next_step>)
```

In addition, the setup needs to define the compilation mode.

```
flow.set('option', 'mode', 'asic')
```

Flows that support SiliconCompiler metric functions (minimum, maximum, summary) should also set appropriate metric weights and goals for correct behavior.

```
for metric in ('errors','drvs','holdwns','setupwns','holdtns','setuptns'):
  flow.set('flowgraph', flow, step, index, 'goal', metric, 0)
for metric in ('cellarea', 'peakpower', 'standbypower'):
  flow.set('flowgraph', flow, step, index, 'weight', metric, 1.0)
```

For a complete working example, see the asicflow and fpgaflow source code.

2.3.2 make_docs()

The `make_docs()` function is used by the projects auto-doc generation. The function should include a descriptive docstring and a call to the setup function that populates the appropriate schema settings.

```
def make_docs(chip):
  '''
  A configurable ASIC compilation flow.
  '''

  setup(chip)
  return chip
```

2.3.3 Flow Modules

The table below shows the function interfaces for setting up Flow objects.

Function	Description	Arg	Returns	Used by	Required
setup	Flow setup function	*Chip*	*siliconcompile* *Flow*	*use()*	yes
make_docs	Doc generator	*Chip*	*siliconcompile* *Flow*	sphinx	no

A complete set of supported open flows can be found in *Flows*.

2.4 Tools

SiliconCompiler execution depends on implementing adapter code ("drivers") for each tool which gets called in a flowgraph. Tools are referenced as named modules by the flowgraph, and searched using the *find_files()* method. A complete set of supported tools can be found in *Pre-Defined Tools*.

Each tool can support multiple "tasks". Each node in the flowgraph is associated with both a tool and a task. Shared configurations such as the tool's minimum version number go in the tool modules, and task-specific settings go in the task modules.

For example, the KLayout tool can be used to export GDS files, display results in a GUI window, or take screenshots of a design. Each of those three functions is associated with a different task name: `export`, `show`, and `screenshot` respectively.

Each task has its own Python module and `setup()` method in the associated tool's directory. Task modules also support the same `pre_process()` and `post_process()` functionality as tool scripts.

2.4.1 setup(chip)

Tool and task setup is performed for each step and index within the *run()* function, prior to launching each individual task. Tools can be configured independently for different tasks (ie. the place task is different from the route task), so we need a method for passing information about the current step and index to the setup function. This is accomplished with the reserved parameters shown below.

```
step = chip.get('arg','step')
index = chip.get('arg','index')
```

Each node in the flowgraph has a step name, and an index. The step name is linked to a task type by the *node()* function, which is usually called in a *siliconcompiler.Flow*'s `setup()` function. The indices are used to allow multiple instances of a task to run in parallel with slightly different parameters. When you are not performing a parameter sweep, the "index" value will usually be set to "0".

All tools are required to bind the tool name to an executable name and to define any required command line options.

```
chip.set('tool', <toolname>, 'exe', <exename>)
chip.set('tool', <toolname>, 'task', <taskname> 'option', <option>)
```

For tools such as TCL based EDA tools, we also need to define the entry script and any associated script directories.

```
chip.set('tool', <toolname>, 'task', <taskname>, 'script', <entry_script>)
chip.set('tool', <toolname>, 'task', <taskname>, 'refdir', <scriptdir>)
chip.set('tool', <toolname>, 'task', <taskname>, 'format', <scriptformat>)
```

To leverage the *run()* function's internal setup checking logic, it is highly recommend to define the parameter requirements, required inputs, expected output, version switch, and supported version numbers using the commands below:

```
chip.set('tool', <toolname>, 'task', <taskname>, 'input', <list[file]>)
chip.set('tool', <toolname>, 'task', <taskname>, 'output', <list[file]>)
chip.set('tool', <toolname>, 'task', <taskname>, 'require' <list[string]>)
chip.set('tool', <toolname>, 'task', <taskname>, 'report', <list[file]>)
chip.set('tool', <toolname>, 'version' <list[string]>)
chip.set('tool', <toolname>, 'vswitch', <string>)
```

2.4.2 parse_version(stdout)

The *run()* function includes built in executable version checking, which can be disabled with the *['option', 'novercheck']* parameter. The executable option to use for printing out the version number is specified with the *['tool', <tool>, 'vswitch']* parameter within the `setup()` function. Commonly used options include '-v', '--version', '-version'. The executable output varies widely, so we need a parsing function that processes the output and returns a single uniform version string. The example shows how this function is implemented for the Yosys tool.

```
def parse_version(stdout):
    # Yosys 0.9+3672 (git sha1 014c7e26, gcc 7.5.0-3ubuntu1~18.04 -fPIC -Os)
    return stdout.split()[1]
```

The *run()* function compares the returned parsed version against the *['tool', <tool>, 'version']* parameter specified in the setup() function to ensure that a qualified executable version is being used.

2.4.3 normalize_version(version)

SC's version checking logic is based on Python's PEP-440 standard. In order to perform version checking for tools that do not natively provide PEP-440 compatible version numbers, this function must be implemented to convert the tool-specific versions to a PEP-440 compatible equivalent.

Note that a raw version number may parse as a valid PEP-440 version but not be semantically correct. normalize _version() must be implemented in these cases to ensure version comparisons make sense. For example, we have to do this for Yosys.

```
def normalize_version(version):
    # Replace '+', which represents a "local version label", with '-', which is
    # an "implicit post release number".
    return version.replace('+', '-')
```

2.4.4 pre_process(chip)

For certain tools and tasks, we may need to set some Schema parameters immediately before task execution. For example, we may want to set the die and core area before the floorplan step based on the area result from the synthesis step.

2.4.5 post_process(chip)

The post process step is required to extract metrics from the tool log files. At a minimum the post process step should extract the number of warnings and errors from the tool log file and insert the value into the Schema. The post_process() logic is straight forward, but the regular expression logic can get involved for complex log files. Perhaps some day, EDA tools will produce SiliconCompiler compatible JSON metrics files.

The post_process function can also be used to post process the output data in the case of command line executable to produce an output that can be ingested by the SiliconCompiler framework. The Surelog `post_process()` implementation illustrates the power of the this functionality.

```
def post_process(chip):
  ''' Tool specific function to run after step execution
  '''
  design = chip.top()
  step = chip.get('arg', 'step')

  if step != 'import':
      return 0

  # Look in slpp_all/file_elab.lst for list of Verilog files included in
  # design, read these and concatenate them into one pickled output file.
  with open('slpp_all/file_elab.lst', 'r') as filelist, \
          open(f'outputs/{design}.v', 'w') as outfile:
      for path in filelist.read().split('\n'):
          if not path:
              # skip empty lines
              continue
          with open(path, 'r') as infile:
              outfile.write(infile.read())
          # in case end of file is missing a newline
          outfile.write('\n')
```

2.4.6 runtime_options(chip)

The distributed execution model of SiliconCompiler mandates that absolute paths be resolved at task run time. The setup() function is run at *run()* launch to check flow validity, so we need a second function interface (runtime_options) to create the final commandline options. The runtime_options() function inspects the Schema and returns a cmdlist to be used by the 'exe' during task execution. The sequence of items used to generate the final command line invocation is as follows:

```
<'tool',...,'exe'> <'tool',...,'option'> <'tool',...,'script'> <runtime_options()>
```

The Surelog example below illustrates the process of defining a runtime_options function.

```
def runtime_options(chip):

  ''' Custom runtime options, returns list of command line options.
  '''

  step = chip.get('arg','step')
  index = chip.get('arg','index')

  cmdlist = []

  # source files
  for value in chip.find_files('option', 'ydir'):
      cmdlist.append('-y ' + value)
  for value in chip.find_files('option', 'vlib'):
      cmdlist.append('-v ' + value)
  for value in chip.find_files('option', 'idir'):
      cmdlist.append('-I' + value)
  for value in chip.get('option', 'define'):
      cmdlist.append('-D' + value)
  for value in chip.find_files('option', 'cmdfile'):
      cmdlist.append('-f ' + value)
  for value in chip.find_files('option', 'source'):
      cmdlist.append(value)

  cmdlist.append('-top ' + chip.top())
  # make sure we can find .sv files in ydirs
  cmdlist.append('+libext+.sv')

  # Set up user-provided parameters to ensure we elaborate the correct modules
  for param in chip.getkeys('option', 'param'):
      value = chip.get('option', 'param', param)
      cmdlist.append(f'-P{param}={value}')

  return cmdlist
```

2.4.7 make_docs()

The SiliconCompiler includes automated document generators that search all tool modules for functions called `make_docs()`. It is highly recommended for all tools to include a `make_docs()` function. The function docstring is used for general narrative, while the body of the function is used to auto-generate a settings table based on the manifest created. At a minimum, the docstring should include a short description and links to the Documentation, Sources, and Installation. The example below shows the make_docs function for surelog.

```
def make_docs():
  '''
  Surelog is a SystemVerilog pre-processor, parser, elaborator,
  and UHDM compiler that provides IEEE design and testbench
  C/C++ VPI and a Python AST API.

  Documentation: https://github.com/chipsalliance/Surelog

  Sources: https://github.com/chipsalliance/Surelog

  Installation: https://github.com/chipsalliance/Surelog

  '''

  chip = siliconcompiler.Chip('<design>')
  chip.set('arg','step','import')
  chip.set('arg','index','0')
  setup(chip)
  return chip
```

2.4.8 run(chip)

SiliconCompiler supports pure-Python tools that execute a Python function rather than an executable. To define a pure-Python tool, add a function called `run()` in your tool driver, which takes in a Chip object and implements your tool's desired functionality. This function should return an integer exit code, with zero indicating success.

Note that pure-Python tool drivers still require a `setup()` function, but most *['tool', ...]* fields will not be meaningful. At the moment, pure-Python tools do not support the following features:

- Version checking
- Replay scripts
- Task timeout
- Memory usage tracking
- Breakpoints
- Output redirection/regex-based logfile parsing

2.4.9 TCL interface

Note: SiliconCompiler configuration settings are communicated to all script based tools as TCL nested dictionaries.

Schema configuration handoff from SiliconCompiler to script based tools is accomplished within the *run()* function by using the *write_manifest()* function to write out the complete schema as a nested TCL dictionary. A snippet of the resulting TCL dictionary is shown below.

```
dict set sc_cfg asic logiclib [list  NangateOpenCellLibrary ]
dict set sc_cfg asic macrolib [list ]
dict set sc_cfg design [list  gcd ]
dict set sc_cfg option frontend [list "verilog"]
```

This generated manifest also includes a helper function, `sc_top`, that handles the logic for determining the name of the design's top-level module (mirroring the logic of *top()*).

It is the responsibility of the tool reference flow developer to bind the standardized SiliconCompiler TCL schema to the tool specific TCL commands and variables. The TCL snippet below shows how the OpenRoad TCL reference flow remaps the TCL nested dictionary to simple lists and scalars at the beginning of the flow for the sake of clarity.

```
#Design
set sc_design     [sc_top]
set sc_tool       <toolname>
set sc_optmode    [sc_cfg_get optmode]

# APR Parameters
set sc_mainlib     [lindex [sc_cfg_get asic logiclib] 0]
set sc_stackup     [sc_cfg_get option stackup]
set sc_targetlibs  [sc_cfg_get asic logiclib]
set sc_density     [sc_cfg_get constraint density]
set sc_pdk         [sc_cfg_get option pdk]
set sc_hpinmetal   [lindex [sc_cfg_get pdk $sc_pdk {var} $sc_tool pin_layer_horizontal
↪$sc_stackup] 0]
set sc_vpinmetal   [lindex [sc_cfg_get pdk $sc_pdk {var} $sc_tool pin_layer_vertical $sc_
↪stackup] 0]
```

2.4.10 Tool and Task Modules

The table below shows the function interfaces supported in setting up tool and task logic.

Function	Description	Arg	Returns	Used by	Required
setup	Configures tool	*Chip*	n/a	run()	yes
runtime _options	Resolves paths at runtime	*Chip*	list	run()	no
parse_version	Returns executable version	stdout	version	run()	no
normalize _version	Returns executable version	tool version	normalized version	run()	no
pre_process	Pre-executable logic	*Chip*	n/a	run()	no
post_process	Post-executable logic	*Chip*	n/a	run()	no
make_docs	Doc generator	None	*Chip*	sphinx	no
run	Pure Python tool	*Chip*	exit code	run()	no

For a complete example of a tool setup module, see OpenROAD. For more in depth information about the various *['tool', ...]* parameters, see the *Schema* section of the reference manual.

2.5 PDKs

Process Design Kits (PDKs) for leading process nodes generally include hundreds of files, documents, and configuration parameters, resulting in significant startup times in porting a design to a new node. The SiliconCompiler project minimizes per design PDK setup efforts by offering a way to package PDKs as standardized reusable objects, and making them available as named modules which can be loaded by the *use()* function.

A complete set of supported open PDKs can be found in *pdks <pdks>*. The table below shows the function interfaces supported in setting up PDKs.

2.5.1 setup(chip)

A minimally viable PDK will include a simulation device model and a set of codified manufacturing rules ("drc"). For an example setup, see the Freepdk45 source code. An example of some of the fundamental settings are shown below.

```
chip.set('option', 'mode', 'asic')
process = '<process_name>'
chip.set('pdk', process, 'foundry', <foundry_name>)
chip.set('pdk', process, 'node', <node_geometry>)
chip.set('pdk', process, 'version', <version>)
chip.set('pdk', process, 'stackup', <stackuplist>)
chip.set('pdk', process, 'drc', 'runset', <tool>, <stackup>, <runset_type>, <file>)
chip.set('pdk', process, 'lvs', 'runset', <tool>, <stackup>, <runset_type> <file>)
chip.set('pdk', process, 'devmodel', <tool>, <modeltype>, <stackup>, <file>)
```

To support standard RTL2GDS flows, the PDK setup will also need to specify pointers to routing technology rules, layout abstractions, layer maps, and routing grids as shown in the below example. For a complete set of available PDK parameters, see the *['pdk', ...]* section of the *Schema*.

```
chip.set('pdk', process, 'aprtech', <tool>, <stackup>, <libtype>, 'lef', <file>)
chip.set('pdk', process, 'layermap', <tool>, 'def', 'gds', <stackup>, <file>)
```

2.5.2 make_docs()

The `make_docs()` function is used by the projects auto-doc generation. The function should include a descriptive docstring and a call to the setup function to populate the schema with all settings:

```
def make_docs(chip):
  '''
  PDK description
  '''

  setup(chip)
  return chip
```

2.5.3 PDK Modules

The table below shows the function interfaces for setting up PDK objects.

Function	Description	Arg	Returns	Used by	Required
setup	PDK setup function	*Chip*	*siliconcompile PDK*	*use()*	yes
make_docs	Doc generator	*Chip*	*siliconcompile PDK*	sphinx	no

2.6 Libraries

Efficient hardware and software development demands a robust ecosystem of reusable high quality components. In SiliconCompiler, you can add new IP to your design by creating a *siliconcompiler.Library* object which can be passed into the *use()* function. The *siliconcompiler.Library* class contains its own Schema dictionary, which can describe a macro block or standard cell library.

The general flow to create and import a library is to instantiate a *siliconcompiler.Library* object, set up any required sources (in the case of a soft library), or models and outputs (in case of a hardened library), and then import it into a parent design *siliconcompiler.Chip* object.

To select which standard cell libraries to use during compilation, add their names to the *['asic', 'logiclib']* parameter, and to select macro libraries, add their names to the *['asic', 'macrolib']* parameter.

Here's an example of setting up and importing a macro block as a *siliconcompiler.Library* object:

```
chip = siliconcompiler.Chip('mydesign')
chip.add('input', 'verilog', 'mydesign.v')

lib = siliconcompiler.Library(chip, 'mymacro')
lib.add('output', stackup, 'lef', 'mymacro.lef')
lib.add('output', stackup, 'gds', 'mymacro.gds')
# ... add more library sources

chip.use(lib)
chip.add('asic', 'macrolib', 'mymacro')
chip.set('constraint', 'component', 'macro_instance1', 'placement', (20.0, 20.0, 0.0))
chip.set('constraint', 'component', 'macro_instance2', 'placement', (40.0, 20.0, 0.0))
```

(continues on next page)

(continued from previous page)

```
chip.set('constraint', 'component', 'macro_instance2', 'rotation', 180)
# ... perform more build configuration

chip.run()
```

This example assumes that the theoretical *mydesign.v* file contains at least two instances of a block named *mymacro*, named *macro_instance1* and *macro_instance2*. The physical design flow will be able to find the library's LEF and GDS files during the build process, and the macro placements for the named instances will be fixed.

The same approach is used to configure standard cell libraries, which are primarily defined using *['output', ...]* settings. In leading edge process nodes, it's not uncommon to have 10's to 100's of STA signoff corners, with each corner consuming gigabytes of disk space. The SiliconCompiler schema is designed to enable efficient setup of all library file pointers and to allow designers to easily select which libraries and which corners to use in any single run. The run time selection of corners, libraries, and timing enables an agile approach to design, wherein the designer can choose the level of accuracy and performance based on need. For example, early in the architecture exploration phase, speed matters and the right choice for synthesis may be to compile using a single stdcell library, using a NLDM model, at a single timing corner. As the design is fine tuned, and the team closes in on tapeout of a mass produced device, the compilation and signoff verification may use many Vt libraries, CCS timing models, and hundreds of timing scenarios. Being able to make these trade-offs with a unified library setup and a couple of lines of designer Python settings, greatly reduces physical design speed and risk.

SiliconCompiler also supports referencing soft libraries (RTL, C-code, etc), in which case many of the physical IP parameters can be omitted.

2.6.1 Library Modules

To enable simple 'target' based access, it is recommended that fundamental physical foundry sponsored IP (stdcells, GPIO, memory macros) are set up as part of reusable library modules.

Similarly to *PDKs*, library modules must implement the following functions.

Function	Description	Args	Returns	Used by	Required
setup	Library setup function	*Chip*	*siliconcompile* *Library*	*use()*	yes
make_docs	Doc generator	*Chip*	*siliconcompile* *Library*	sphinx	no

A complete set of supported standard cell libraries for SC's included open PDKs can be found in the *libraries <libraries>*.

2.7 Metrics

The SiliconCompiler schema includes a *['metric', ...]* dictionary with a large number of parameters to be tracked on a per step and per index basis.

The metric values are used in the `minimum()` and `maximum()` functions to select the best compilation based on the associated *['flowgraph', <flow>, <step>, <index>, 'goal', ...]* and *['flowgraph', <flow>, <step>, <index>, 'weight', ...]* set for the step and index within the flowgraph. For a complete description of the `minimum()` function, see the *Core API* section of the reference manual.

The default *asicflow* demonstrates a traditional ASIC optimization function, with hard requirements set up for hold, setup, warnings, and errors and soft requirements for area and power.

```
if metric in ('errors', 'warnings', 'drvs', 'holdwns', 'setupwns', 'holdtns', 'setuptns
↪'):
    chip.set('flowgraph', flow, step, index, 'weight', metric, 1.0)
    chip.set('flowgraph', flow, step, index, metric, 'goal', 0)
elif metric in ('cellarea', 'peakpower', 'standbypower'):
    chip.set('flowgraph', flow, step, index, 'weight', metric, 1.0)
else:
    chip.set('flowgraph', flow, step, index, 'weight', metric, 0.001)
```

In addition to step wise minimization, metrics are used by the `Chip.summary()` function to present a dashboard view of the compilation results, and can be accessed through `Chip.set()` / `Chip.get()` by the user to create custom reporting and optimization loops. The metrics are cleared before each step/index run and then updated by the `post_process()` function for each tool. For an example of post_process setup, see the openroad module.

The following table shows a summary of all the available metrics. For a complete descriptions, refer to the *Schema* section of the reference manual.

parameter	description
['metric', 'errors']	Metric: total errors
['metric', 'warnings']	Metric: total warnings
['metric', 'drvs']	Metric: total drvs
['metric', 'unconstrained']	Metric: total unconstrained
['metric', 'coverage']	Metric: coverage
['metric', 'security']	Metric: security
['metric', 'luts']	Metric: FPGA LUTs used
['metric', 'dsps']	Metric: FPGA DSP slices used
['metric', 'brams']	Metric: FPGA BRAM tiles used
['metric', 'cellarea']	Metric: cellarea
['metric', 'totalarea']	Metric: totalarea
['metric', 'utilization']	Metric: area utilization
['metric', 'logicdepth']	Metric: logic depth
['metric', 'peakpower']	Metric: peakpower
['metric', 'averagepower']	Metric: averagepower
['metric', 'dozepower']	Metric: dozepower
['metric', 'idlepower']	Metric: idlepower
['metric', 'leakagepower']	Metric: leakagepower
['metric', 'sleeppower']	Metric: sleeppower
['metric', 'irdrop']	Metric: peak IR drop
['metric', 'holdpaths']	Metric: holdpaths
['metric', 'setuppaths']	Metric: setuppaths
['metric', 'holdslack']	Metric: holdslack
['metric', 'holdwns']	Metric: holdwns
['metric', 'holdtns']	Metric: holdtns
['metric', 'setupslack']	Metric: setupslack
['metric', 'setupwns']	Metric: setupwns
['metric', 'setuptns']	Metric: setuptns
['metric', 'fmax']	Metric: fmax
['metric', 'macros']	Metric: macros
['metric', 'cells']	Metric: cells
['metric', 'registers']	Metric: registers

continues on next page

Table 1 – continued from previous page

['metric', 'buffers']	Metric: buffers
['metric', 'transistors']	Metric: transistors
['metric', 'pins']	Metric: pins
['metric', 'nets']	Metric: nets
['metric', 'vias']	Metric: vias
['metric', 'wirelength']	Metric: wirelength
['metric', 'overflow']	Metric: overflow
['metric', 'memory']	Metric: memory
['metric', 'exetime']	Metric: exetime
['metric', 'tasktime']	Metric: tasktime
['metric', 'totaltime']	Metric: totaltime

2.8 Records

To support hardware provenance, SiliconCompiler supports automated tracking of a number of execution and place of origin related parameters. SiliconCompiler will record the SiliconCompiler version, tool version and options, and task start and end times by default. Additional tracking is off by default in SiliconCompiler, but can be turned on with the ['option', 'track'] parameter.

```
chip.set('option', 'track', True)
```

This will enable tracking of the user and machine information, which may be considered sensitive, so please use caution when enabling all tracking.

Records are kept on a per step, and index basis. Records must be stored for each task in the flowgraph to ensure unbroken traceability from the beginning to the end in the chain of custody.

parameter	description
['record', 'userid']	Record: userid
['record', 'publickey']	Record: public key
['record', 'machine']	Record: machine name
['record', 'macaddr']	Record: MAC address
['record', 'ipaddr']	Record: IP address
['record', 'platform']	Record: platform name
['record', 'distro']	Record: distro name
['record', 'arch']	Record: hardware architecture
['record', 'starttime']	Record: start time
['record', 'endtime']	Record: end time
['record', 'region']	Record: cloud region
['record', 'scversion']	Record: software version
['record', 'toolversion']	Record: tool version
['record', 'toolpath']	Record: tool path
['record', 'toolargs']	Record: tool CLI arguments
['record', 'osversion']	Record: O/S version
['record', 'kernelversion']	Record: O/S kernel version
['record', 'remoteid']	Record: remote job ID

2.9 Remote processing

The SiliconCompiler project supports a remote processing model that leverages the cloud to provide access to:

1. Pre-configured tool installations.
2. Warehouse scale elastic compute.
3. NDA encumbered IPs, PDKs, and EDA tools.

Note: Note that our public beta currently only supports open-source tools and PDKs. You can access the public beta without a signup or login, and it is designed to delete your data after your jobs finish, but it is not intended to process proprietary or restricted intellectual property! Please review our terms of service, and do not submit IP which you are not allowed to distribute.

Currently, our public beta servers will only return a report containing a rendering and metrics for your build results. For the full GDS results, you can build and install the open-source tools and run the job on your local machine. See *Local Run* for more instructions.

In the event that our servers are busy processing a large number of jobs, your job may get queued and experience delays in processing. We appreciate your patience during this public beta period.

Even though our publicly-available servers only support open-source IP and tools, the remote API is capable of supporting any SiliconCompiler modules which the server operators wish to install. If you are interested in creating a custom server implementation, we provide a minimal example development server which can be used as a starting point. You can also find descriptions of the core remote API calls in the *remote API* section.

See the *Quickstart guide* for instructions on running a simple example on our public servers.

2.9.1 Configuring a Different Remote Server

If you have a custom remote endpoint that you wish to use with SiliconCompiler, you can run the *sc-remote* command to set that up with your SiliconCompiler installation.

Public Server

If your remote server does not require authentication, you can simply pass its address in as a command-line argument:

```
sc-remote -configure -server https://server.siliconcompiler.com
```

If a previous credentials file already exists, you will be prompted to overwrite it. Your credentials file will be placed in `$HOME/.sc/`, if you want to back it up or delete it. SiliconCompiler will default to using our public beta address if you have not configured anything, and it will remind you that your design is being uploaded to a public service for processing before starting each remote job.

Private Server

If your custom remote server requires authentication, you can run `sc-remote -configure` with no additional arguments and fill in the address, username, and password fields that it prompts you for.

SiliconCompiler also supports private servers which require authentication to access. If you have such a server to connect to, you will need a credentials text file located at `~/.sc/credentials` on Linux or macOS, or at `C:\\Users\\USERNAME\\.sc\\credentials` on Windows. The credentials file is a JSON formatted file containing information about the remote server address, username, and password.

```
{
"address": "your-server",
"username": "your-username",
"password": "your-key"
}
```

Use a text editor to create the credentials file. Alternatively you can use *sc-remote* app to generate it from the command line.

```
(venv) sc-remote -configure
Remote server address (leave blank to use default server): your-server
Remote username (leave blank for no username): your-username
Remote password (leave blank for no password): your-key
Remote configuration saved to: /home/<USER>/.sc/credentials
```

To verify that your credentials file and server is configured correctly, run the *sc-remote* command.

```
(venv) sc-remote
User myname validated successfully!
Remaining compute time: 1440.00 minutes
Remaining results bandwidth: 5242880 KiB
```

Once you've configured SiliconCompiler to run on your remote endpoint, see the *Quickstart guide* for instructions on running a simple example, along with expected outputs.

2.9.2 Troubleshooting

Our public beta servers do not prune or pre-process Schema parameters, in order to make the remote processing environment as close to a local environment as possible. The jobs will be run in isolated environments with limited communication interfaces, however, so some network and filesystem calls may not work properly.

Any changes that you make to SiliconCompiler's built-in tool setup scripts on your local machine will not be reflected in jobs which are run on a remote server. Likewise, any changes that you make to the built-in open-source PDKs and standard cell libraries will not be sent to the remote servers. If you have suggestions for improving the open-source modules, check out our contributing guidelines.

Please report any issues that you encounter with the remote workflow on the SiliconCompiler repository's issue page.

CHAPTER

THREE

REFERENCES

The following sections provides details on the functions, modules and objects included in SiliconCompiler. To learn how to use SiliconCompiler, see the *User Guide*.

3.1 Pre-Defined Targets

The following are examples are pre-built targets that come with SiliconCompiler which you can use for your own builds. These are typically "demo" targets which use a specific combination of predefined *pdk*, *library*, *flow* and more configurations.

3.1.1 asap7_demo

ASAP7 Demo Target

Setup file: asap7_demo.py

Data sources

Package	Specifications
lambdapdk	• Path: `https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/` • Reference: `v0.1.19`

Flows

- *asicflow*

PDK

- *asap7*

Libraries

- *asap7sc7p5t_rvt*
- *lambdalib_asap7sc7p5t_rvt*
- *asap7sc7p5t_lvt*
- *lambdalib_asap7sc7p5t_lvt*
- *asap7sc7p5t_slvt*
- *lambdalib_asap7sc7p5t_slvt*

Configuration

Keypath	Value
['asic', 'logiclib']	asap7sc7p5t_rvt
['asic', 'delaymodel']	nldm
['constraint', 'timing', 'slow', 'libcorner']	slow
['constraint', 'timing', 'slow', 'pexcorner']	typical
['constraint', 'timing', 'slow', 'mode']	func
['constraint', 'timing', 'slow', 'check']	• setup • hold
['constraint', 'timing', 'fast', 'libcorner']	fast
['constraint', 'timing', 'fast', 'pexcorner']	typical
['constraint', 'timing', 'fast', 'mode']	func
['constraint', 'timing', 'fast', 'check']	• setup • hold
['constraint', 'timing', 'typical', 'libcorner']	typical
['constraint', 'timing', 'typical', 'pexcorner']	typical
['constraint', 'timing', 'typical', 'mode']	func
['constraint', 'timing', 'typical', 'check']	power
['constraint', 'coremargin']	0.27
['constraint', 'density']	10.0
['option', 'mode']	asic
['option', 'pdk']	asap7
['option', 'stackup']	10M
['option', 'flow']	asicflow
['option', 'showtool', 'gds']	klayout
['option', 'showtool', 'lef']	klayout
['option', 'showtool', 'def']	openroad
['option', 'showtool', 'odb']	openroad

continues on next page

Table 2 – continued from previous page

['option', 'showtool', 'route']	vpr
['option', 'showtool', 'place']	vpr

3.1.2 asic_demo

"Self-test" target which builds a small 8-bit counter design as an ASIC, targeting the Skywater130 PDK.

This target is intended for testing purposes only, to verify that SiliconCompiler is installed and configured correctly.

Setup file: asic_demo.py

Data sources

Package	Specifications
siliconcompiler	• Path: python://siliconcompiler
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

Flows

- *asicflow*
- *asictopflow*
- *signoffflow*

PDK

- *skywater130*

Libraries

- *sky130hd*
- *lambdalib_sky130hd*
- *sky130hdll*
- *lambdalib_sky130hdll*

Checklists

- *oh_tapeout*

Configuration

Keypath	Value
['asic', 'logiclib']	sky130hd
['asic', 'delaymodel']	nldm
['constraint', 'timing', 'slow', 'libcorner']	slow
['constraint', 'timing', 'slow', 'pexcorner']	maximum
['constraint', 'timing', 'slow', 'mode']	func
['constraint', 'timing', 'slow', 'check']	• setup • hold
['constraint', 'timing', 'fast', 'libcorner']	fast
['constraint', 'timing', 'fast', 'pexcorner']	minimum
['constraint', 'timing', 'fast', 'mode']	func
['constraint', 'timing', 'fast', 'check']	• setup • hold
['constraint', 'timing', 'typical', 'libcorner']	typical
['constraint', 'timing', 'typical', 'pexcorner']	typical
['constraint', 'timing', 'typical', 'mode']	func
['constraint', 'timing', 'typical', 'check']	power
['constraint', 'outline']	• (0.0, 0.0) • (50.0, 50.0)
['constraint', 'corearea']	• (5.0, 5.0) • (45.0, 45.0)
['constraint', 'coremargin']	4.6
['constraint', 'density']	10.0
['option', 'mode']	asic
['option', 'target']	siliconcompiler.targets.skywater130_demo
['option', 'pdk']	skywater130
['option', 'stackup']	5M1LI
['option', 'flow']	asicflow
['option', 'showtool', 'gds']	klayout
['option', 'showtool', 'lef']	klayout
['option', 'showtool', 'def']	openroad
['option', 'showtool', 'odb']	openroad
['option', 'showtool', 'route']	vpr
['option', 'showtool', 'place']	vpr
['option', 'quiet']	True

3.1.3 fpgaflow_demo

Demonstration target for running the open-source fpgaflow.

Setup file: fpgaflow_demo.py

Data sources

Package	Specifications
siliconcompiler_data	• Path: git+https://github.com/siliconcompiler/siliconcompiler • Reference: v0.21.11

Flows

- *fpgaflow*

Configuration

Keypath	Value
['option', 'mode']	fpga
['option', 'flow']	fpgaflow
['option', 'showtool', 'gds']	klayout
['option', 'showtool', 'lef']	klayout
['option', 'showtool', 'def']	openroad
['option', 'showtool', 'odb']	openroad
['option', 'showtool', 'route']	vpr
['option', 'showtool', 'place']	vpr

3.1.4 freepdk45_demo

FreePDK45 demo target

Setup file: freepdk45_demo.py

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

Flows

- *lintflow*
- *asicflow*
- *asictopflow*

PDK

- *freepdk45*

Libraries

- *nangate45*
- *lambdalib_nangate45*

Configuration

Keypath	Value
['asic', 'logiclib']	nangate45
['asic', 'delaymodel']	nldm
['constraint', 'timing', 'worst', 'libcorner']	typical
['constraint', 'timing', 'worst', 'pexcorner']	typical
['constraint', 'timing', 'worst', 'mode']	func
['constraint', 'timing', 'worst', 'check']	• setup • hold
['constraint', 'coremargin']	1.9
['constraint', 'density']	10.0
['option', 'mode']	asic
['option', 'pdk']	freepdk45
['option', 'stackup']	10M
['option', 'flow']	asicflow
['option', 'showtool', 'gds']	klayout
['option', 'showtool', 'lef']	klayout
['option', 'showtool', 'def']	openroad
['option', 'showtool', 'odb']	openroad
['option', 'showtool', 'route']	vpr
['option', 'showtool', 'place']	vpr

3.1.5 gf180_demo

Global foundries 180 Demo Target

Setup file: gf180_demo.py

Data sources

Package	Specifications
lambdapdk	• Path: `https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/` • Reference: `v0.1.19`

Flows

- *asicflow*
- *asictopflow*
- *signoffflow*

PDK

- *gf180*

Libraries

- *gf180mcu_fd_sc_mcu7t5v0*
- *lambdalib_gf180mcu_fd_sc_mcu7t5v0*
- *gf180mcu_fd_sc_mcu9t5v0*
- *lambdalib_gf180mcu_fd_sc_mcu9t5v0*

Checklists

- *oh_tapeout*

Configuration

Keypath	Value
['asic', 'logiclib']	gf180mcu_fd_sc_mcu9t5v0
['asic', 'delaymodel']	nldm
['constraint', 'timing', 'slow', 'libcorner']	slow
['constraint', 'timing', 'slow', 'pexcorner']	wst
['constraint', 'timing', 'slow', 'mode']	func
['constraint', 'timing', 'slow', 'check']	• setup • hold
['constraint', 'timing', 'fast', 'libcorner']	fast
['constraint', 'timing', 'fast', 'pexcorner']	bst
['constraint', 'timing', 'fast', 'mode']	func
['constraint', 'timing', 'fast', 'check']	• setup • hold
['constraint', 'timing', 'typical', 'libcorner']	typical
['constraint', 'timing', 'typical', 'pexcorner']	typ
['constraint', 'timing', 'typical', 'mode']	func
['constraint', 'timing', 'typical', 'check']	power
['constraint', 'coremargin']	4.6
['constraint', 'density']	10.0
['option', 'mode']	asic
['option', 'pdk']	gf180
['option', 'stackup']	5LM_1TM_9K
['option', 'flow']	asicflow
['option', 'showtool', 'gds']	klayout
['option', 'showtool', 'lef']	klayout
['option', 'showtool', 'def']	openroad
['option', 'showtool', 'odb']	openroad
['option', 'showtool', 'route']	vpr
['option', 'showtool', 'place']	vpr

3.1.6 skywater130_demo

Skywater130 Demo Target

Setup file: skywater130_demo.py

Data sources

Package	Specifications
lambdapdk	• Path: `https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/` • Reference: `v0.1.19`

Flows

- *asicflow*
- *asictopflow*
- *signoffflow*

PDK

- *skywater130*

Libraries

- *sky130hd*
- *lambdalib_sky130hd*
- *sky130hdll*
- *lambdalib_sky130hdll*

Checklists

- *oh_tapeout*

Configuration

Keypath	Value
['asic', 'logiclib']	sky130hd
['asic', 'delaymodel']	nldm
['constraint', 'timing', 'slow', 'libcorner']	slow
['constraint', 'timing', 'slow', 'pexcorner']	maximum
['constraint', 'timing', 'slow', 'mode']	func
['constraint', 'timing', 'slow', 'check']	• setup • hold
['constraint', 'timing', 'fast', 'libcorner']	fast

continues on next page

Table 12 – continued from previous page

['constraint', 'timing', 'fast', 'pexcorner']	minimum
['constraint', 'timing', 'fast', 'mode']	func
['constraint', 'timing', 'fast', 'check']	• setup • hold
['constraint', 'timing', 'typical', 'libcorner']	typical
['constraint', 'timing', 'typical', 'pexcorner']	typical
['constraint', 'timing', 'typical', 'mode']	func
['constraint', 'timing', 'typical', 'check']	power
['constraint', 'coremargin']	4.6
['constraint', 'density']	10.0
['option', 'mode']	asic
['option', 'pdk']	skywater130
['option', 'stackup']	5M1LI
['option', 'flow']	asicflow
['option', 'showtool', 'gds']	klayout
['option', 'showtool', 'lef']	klayout
['option', 'showtool', 'def']	openroad
['option', 'showtool', 'odb']	openroad
['option', 'showtool', 'route']	vpr
['option', 'showtool', 'place']	vpr

3.2 Pre-Defined Flows

The following are examples are pre-built flows that come with SiliconCompiler which you can use for your own builds.

See the pre-built *targets* for examples on how these are used in conjunction with *pdks*, *tools* and *libraries*.

3.2.1 asicflow

A configurable ASIC compilation flow.

The 'asicflow' includes the stages below. The steps syn, floorplan, physyn, place, cts, route, and dfm have minimization associated with them. To view the flowgraph, see the .png file.

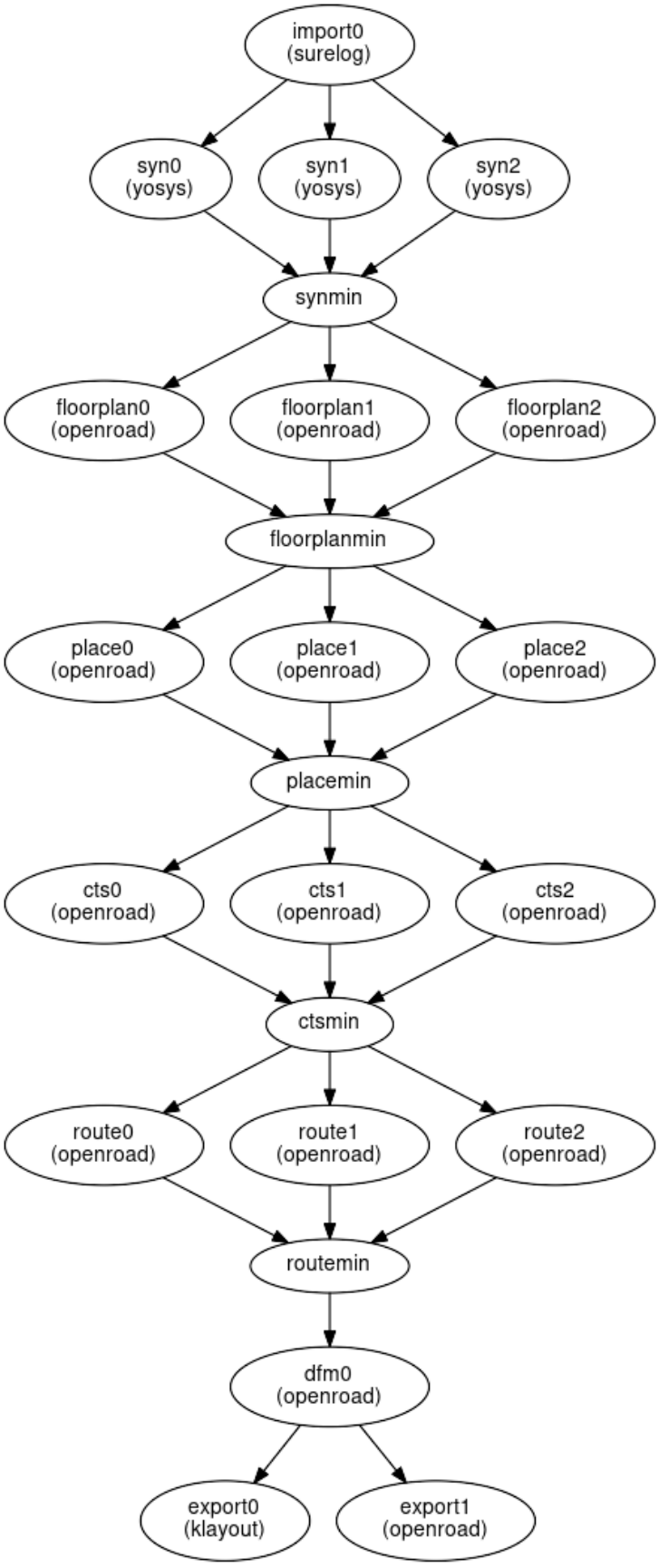
- **import**: Sources are collected and packaged for compilation
- **syn**: Translates RTL to netlist using Yosys
- **floorplan**: Floorplanning
- **physyn**: Physical Synthesis
- **place**: Global and detailed placement
- **cts**: Clock tree synthesis
- **route**: Global and detailed routing
- **dfm**: Metal fill, atenna fixes and any other post routing steps
- **export**: Export design from APR tool and merge with library GDS

- **sta**: Static timing analysis (signoff)
- **lvs**: Layout versus schematic check (signoff)
- **drc**: Design rule check (signoff)

The syn, physyn, place, cts, route steps supports per process options that can be set up by setting '<step>_np' arg to a value > 1, as detailed below:

- syn_np : Number of parallel synthesis jobs to launch
- floorplan_np : Number of parallel floorplan jobs to launch
- physyn_np : Number of parallel physical synthesis jobs to launch
- place_np : Number of parallel place jobs to launch
- cts_np : Number of parallel clock tree synthesis jobs to launch
- route_np : Number of parallel routing jobs to launch

Setup file: asicflow.py


import0
(surelog)
syn0
(yosys)
syn1
(yosys)
syn2
(yosys)
synmin
floorplan0
(openroad)
floorplan1
(openroad)
floorplan2
(openroad)
floorplanmin
place0
(openroad)
place1
(openroad)
place2
(openroad)
placemin
cts0
(openroad)
cts1
(openroad)
cts2
(openroad)
ctsmin
route0
(openroad)
route1
(openroad)
route2
(openroad)
routemin
dfm0
(openroad)
export0
(klayout)
export1
(openroad)

Configuration

import

Keypath	Value
['flowgraph', 'asicflow', 'import', '0', 'tool']	surelog
['flowgraph', 'asicflow', 'import', '0', 'task']	parse
['flowgraph', 'asicflow', 'import', '0', 'taskmodule']	siliconcompiler.tools.surelog.parse

syn

Keypath	Value
['flowgraph', 'asicflow', 'syn', '0', 'input']	('import', '0')
['flowgraph', 'asicflow', 'syn', '0', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'syn', '0', 'tool']	yosys
['flowgraph', 'asicflow', 'syn', '0', 'task']	syn_asic
['flowgraph', 'asicflow', 'syn', '0', 'taskmodule']	siliconcompiler.tools.yosys.syn_asic
['flowgraph', 'asicflow', 'syn', '1', 'input']	('import', '0')
['flowgraph', 'asicflow', 'syn', '1', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'syn', '1', 'tool']	yosys
['flowgraph', 'asicflow', 'syn', '1', 'task']	syn_asic
['flowgraph', 'asicflow', 'syn', '1', 'taskmodule']	siliconcompiler.tools.yosys.syn_asic
['flowgraph', 'asicflow', 'syn', '2', 'input']	('import', '0')
['flowgraph', 'asicflow', 'syn', '2', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'syn', '2', 'tool']	yosys
['flowgraph', 'asicflow', 'syn', '2', 'task']	syn_asic
['flowgraph', 'asicflow', 'syn', '2', 'taskmodule']	siliconcompiler.tools.yosys.syn_asic

synmin

Keypath	Value
['flowgraph', 'asicflow', 'synmin', '0', 'input']	• ('syn', '0') • ('syn', '1') • ('syn', '2')
['flowgraph', 'asicflow', 'synmin', '0', 'tool']	builtin
['flowgraph', 'asicflow', 'synmin', '0', 'task']	minimum

continues on next page

Table 15 – continued from previous page

['flowgraph', 'asicflow', 'synmin', '0', 'taskmodule']	siliconcompiler.tools.builtin.minimum

floorplan

Keypath	Value
['flowgraph', 'asicflow', 'floorplan', '0', 'input']	('synmin', '0')
['flowgraph', 'asicflow', 'floorplan', '0', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'floorplan', '0', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'floorplan', '0', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'floorplan', '0', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'floorplan', '0', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'floorplan', '0', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'floorplan', '0', 'tool']	openroad
['flowgraph', 'asicflow', 'floorplan', '0', 'task']	floorplan
['flowgraph', 'asicflow', 'floorplan', '0', 'taskmodule']	siliconcompiler.tools.openroad.floorplan
['flowgraph', 'asicflow', 'floorplan', '1', 'input']	('synmin', '0')
['flowgraph', 'asicflow', 'floorplan', '1', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'floorplan', '1', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'floorplan', '1', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'floorplan', '1', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'floorplan', '1', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'floorplan', '1', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'floorplan', '1', 'tool']	openroad
['flowgraph', 'asicflow', 'floorplan', '1', 'task']	floorplan
['flowgraph', 'asicflow', 'floorplan', '1', 'taskmodule']	siliconcompiler.tools.openroad.floorplan
['flowgraph', 'asicflow', 'floorplan', '2', 'input']	('synmin', '0')

continues on next page

Table 16 – continued from previous page

['flowgraph', 'asicflow', 'floorplan', '2', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'floorplan', '2', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'floorplan', '2', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'floorplan', '2', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'floorplan', '2', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'floorplan', '2', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'floorplan', '2', 'tool']	openroad
['flowgraph', 'asicflow', 'floorplan', '2', 'task']	floorplan
['flowgraph', 'asicflow', 'floorplan', '2', 'taskmodule']	siliconcompiler.tools.openroad.floorplan

floorplanmin

Keypath	Value
['flowgraph', 'asicflow', 'floorplanmin', '0', 'input']	• ('floorplan', '0') • ('floorplan', '1') • ('floorplan', '2')
['flowgraph', 'asicflow', 'floorplanmin', '0', 'tool']	builtin
['flowgraph', 'asicflow', 'floorplanmin', '0', 'task']	minimum
['flowgraph', 'asicflow', 'floorplanmin', '0', 'taskmodule']	siliconcompiler.tools.builtin.minimum

place

Keypath	Value
['flowgraph', 'asicflow', 'place', '0', 'input']	('floorplanmin', '0')
['flowgraph', 'asicflow', 'place', '0', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'place', '0', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'place', '0', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'place', '0', 'goal', 'errors']	0.0

continues on next page

Table 18 – continued from previous page

['flowgraph', 'asicflow', 'place', '0', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'place', '0', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'place', '0', 'tool']	openroad
['flowgraph', 'asicflow', 'place', '0', 'task']	place
['flowgraph', 'asicflow', 'place', '0', 'taskmodule']	siliconcompiler.tools.openroad.place
['flowgraph', 'asicflow', 'place', '1', 'input']	('floorplanmin', '0')
['flowgraph', 'asicflow', 'place', '1', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'place', '1', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'place', '1', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'place', '1', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'place', '1', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'place', '1', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'place', '1', 'tool']	openroad
['flowgraph', 'asicflow', 'place', '1', 'task']	place
['flowgraph', 'asicflow', 'place', '1', 'taskmodule']	siliconcompiler.tools.openroad.place
['flowgraph', 'asicflow', 'place', '2', 'input']	('floorplanmin', '0')
['flowgraph', 'asicflow', 'place', '2', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'place', '2', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'place', '2', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'place', '2', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'place', '2', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'place', '2', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'place', '2', 'tool']	openroad
['flowgraph', 'asicflow', 'place', '2', 'task']	place
['flowgraph', 'asicflow', 'place', '2', 'taskmodule']	siliconcompiler.tools.openroad.place

placemin

Keypath	Value
['flowgraph', 'asicflow', 'placemin', '0', 'input']	• ('place', '0') • ('place', '1') • ('place', '2')
['flowgraph', 'asicflow', 'placemin', '0', 'tool']	builtin
['flowgraph', 'asicflow', 'placemin', '0', 'task']	minimum
['flowgraph', 'asicflow', 'placemin', '0', 'taskmodule']	siliconcompiler.tools.builtin.minimum

cts

Keypath	Value
['flowgraph', 'asicflow', 'cts', '0', 'input']	('placemin', '0')
['flowgraph', 'asicflow', 'cts', '0', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'cts', '0', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'cts', '0', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'cts', '0', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'cts', '0', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'cts', '0', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'cts', '0', 'tool']	openroad
['flowgraph', 'asicflow', 'cts', '0', 'task']	cts
['flowgraph', 'asicflow', 'cts', '0', 'taskmodule']	siliconcompiler.tools.openroad.cts
['flowgraph', 'asicflow', 'cts', '1', 'input']	('placemin', '0')
['flowgraph', 'asicflow', 'cts', '1', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'cts', '1', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'cts', '1', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'cts', '1', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'cts', '1', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'cts', '1', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'cts', '1', 'tool']	openroad

continues on next page

Table 20 – continued from previous page

['flowgraph', 'asicflow', 'cts', '1', 'task']	cts
['flowgraph', 'asicflow', 'cts', '1', 'taskmodule']	siliconcompiler.tools.openroad.cts
['flowgraph', 'asicflow', 'cts', '2', 'input']	('placemin', '0')
['flowgraph', 'asicflow', 'cts', '2', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'cts', '2', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'cts', '2', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'cts', '2', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'cts', '2', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'cts', '2', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'cts', '2', 'tool']	openroad
['flowgraph', 'asicflow', 'cts', '2', 'task']	cts
['flowgraph', 'asicflow', 'cts', '2', 'taskmodule']	siliconcompiler.tools.openroad.cts

ctsmin

Keypath	Value
['flowgraph', 'asicflow', 'ctsmin', '0', 'input']	• ('cts', '0') • ('cts', '1') • ('cts', '2')
['flowgraph', 'asicflow', 'ctsmin', '0', 'tool']	builtin
['flowgraph', 'asicflow', 'ctsmin', '0', 'task']	minimum
['flowgraph', 'asicflow', 'ctsmin', '0', 'taskmodule']	siliconcompiler.tools.builtin.minimum

route

Keypath	Value
['flowgraph', 'asicflow', 'route', '0', 'input']	('ctsmin', '0')
['flowgraph', 'asicflow', 'route', '0', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'route', '0', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'route', '0', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'route', '0', 'goal', 'errors']	0.0

continues on next page

Table 22 – continued from previous page

['flowgraph', 'asicflow', 'route', '0', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'route', '0', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'route', '0', 'tool']	openroad
['flowgraph', 'asicflow', 'route', '0', 'task']	route
['flowgraph', 'asicflow', 'route', '0', 'taskmodule']	siliconcompiler.tools.openroad.route
['flowgraph', 'asicflow', 'route', '1', 'input']	('ctsmin', '0')
['flowgraph', 'asicflow', 'route', '1', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'route', '1', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'route', '1', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'route', '1', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'route', '1', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'route', '1', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'route', '1', 'tool']	openroad
['flowgraph', 'asicflow', 'route', '1', 'task']	route
['flowgraph', 'asicflow', 'route', '1', 'taskmodule']	siliconcompiler.tools.openroad.route
['flowgraph', 'asicflow', 'route', '2', 'input']	('ctsmin', '0')
['flowgraph', 'asicflow', 'route', '2', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'route', '2', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'route', '2', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'route', '2', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'route', '2', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'route', '2', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'route', '2', 'tool']	openroad
['flowgraph', 'asicflow', 'route', '2', 'task']	route
['flowgraph', 'asicflow', 'route', '2', 'taskmodule']	siliconcompiler.tools.openroad.route

routemin

Keypath	Value
['flowgraph', 'asicflow', 'routemin', '0', 'input']	• ('route', '0') • ('route', '1') • ('route', '2')
['flowgraph', 'asicflow', 'routemin', '0', 'tool']	builtin
['flowgraph', 'asicflow', 'routemin', '0', 'task']	minimum
['flowgraph', 'asicflow', 'routemin', '0', 'taskmodule']	siliconcompiler.tools.builtin.minimum

dfm

Keypath	Value
['flowgraph', 'asicflow', 'dfm', '0', 'input']	('routemin', '0')
['flowgraph', 'asicflow', 'dfm', '0', 'weight', 'cellarea']	1.0
['flowgraph', 'asicflow', 'dfm', '0', 'weight', 'peakpower']	1.0
['flowgraph', 'asicflow', 'dfm', '0', 'weight', 'leakagepower']	1.0
['flowgraph', 'asicflow', 'dfm', '0', 'goal', 'errors']	0.0
['flowgraph', 'asicflow', 'dfm', '0', 'goal', 'setupwns']	0.0
['flowgraph', 'asicflow', 'dfm', '0', 'goal', 'setuptns']	0.0
['flowgraph', 'asicflow', 'dfm', '0', 'tool']	openroad
['flowgraph', 'asicflow', 'dfm', '0', 'task']	dfm
['flowgraph', 'asicflow', 'dfm', '0', 'taskmodule']	siliconcompiler.tools.openroad.dfm

export

Keypath	Value
['flowgraph', 'asicflow', 'export', '0', 'input']	('dfm', '0')
['flowgraph', 'asicflow', 'export', '0', 'tool']	klayout
['flowgraph', 'asicflow', 'export', '0', 'task']	export
['flowgraph', 'asicflow', 'export', '0', 'taskmodule']	siliconcompiler.tools.klayout.export

continues on next page

Table 25 – continued from previous page

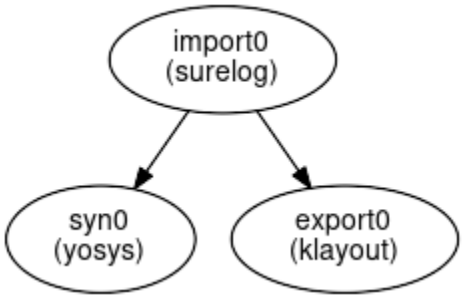
['flowgraph', 'asicflow', 'export', '1', 'input']	('dfm', '0')
['flowgraph', 'asicflow', 'export', '1', 'tool']	openroad
['flowgraph', 'asicflow', 'export', '1', 'task']	export
['flowgraph', 'asicflow', 'export', '1', 'taskmodule']	siliconcompiler.tools.openroad.export

3.2.2 asictopflow

A flow for stitching together hardened blocks without doing any automated place-and-route.

This flow generates a GDS and a netlist for passing to a verification/signoff flow.

Setup file: asictopflow.py



Configuration

import

Keypath	Value
['flowgraph', 'asictopflow', 'import', '0', 'goal', 'errors']	0.0
['flowgraph', 'asictopflow', 'import', '0', 'tool']	surelog
['flowgraph', 'asictopflow', 'import', '0', 'task']	parse
['flowgraph', 'asictopflow', 'import', '0', 'taskmodule']	siliconcompiler.tools.surelog.parse

syn

Keypath	Value
['flowgraph', 'asictopflow', 'syn', '0', 'input']	('import', '0')
['flowgraph', 'asictopflow', 'syn', '0', 'goal', 'errors']	0.0

continues on next page

Table 27 – continued from previous page

['flowgraph', 'asictopflow', 'syn', '0', 'tool']	yosys
['flowgraph', 'asictopflow', 'syn', '0', 'task']	syn_asic
['flowgraph', 'asictopflow', 'syn', '0', 'taskmodule']	siliconcompiler.tools.yosys.syn_asic

export

Keypath	Value
['flowgraph', 'asictopflow', 'export', '0', 'input']	('import', '0')
['flowgraph', 'asictopflow', 'export', '0', 'goal', 'errors']	0.0
['flowgraph', 'asictopflow', 'export', '0', 'tool']	klayout
['flowgraph', 'asictopflow', 'export', '0', 'task']	export
['flowgraph', 'asictopflow', 'export', '0', 'taskmodule']	siliconcompiler.tools.klayout.export

3.2.3 dvflow

A configurable constrained random stimulus DV flow.

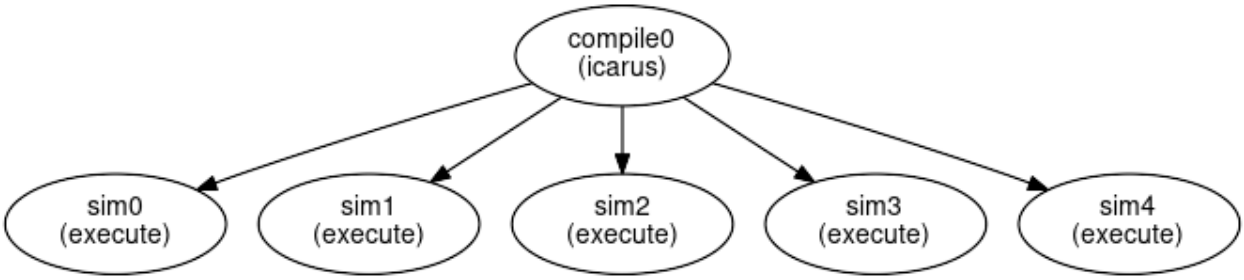
The verification pipeline includes the followins teps:

- **compile**: RTL sources are compiled into object form (once)
- **sim**: Compiled RTL is exercised using generated test

The dvflow can be parametrized using a single 'np' parameter. Setting 'np' > 1 results in multiple independent verificaiton pipelines to be launched.

This flow is a WIP

Setup file: dvflow.py



Configuration

compile

Keypath	Value
['flowgraph', 'dvflow', 'compile', '0', 'tool']	icarus
['flowgraph', 'dvflow', 'compile', '0', 'task']	compile
['flowgraph', 'dvflow', 'compile', '0', 'taskmodule']	siliconcompiler.tools.icarus.compile

sim

Keypath	Value
['flowgraph', 'dvflow', 'sim', '0', 'input']	('compile', '0')
['flowgraph', 'dvflow', 'sim', '0', 'tool']	execute
['flowgraph', 'dvflow', 'sim', '0', 'task']	exec_input
['flowgraph', 'dvflow', 'sim', '0', 'taskmodule']	siliconcompiler.tools.execute.exec_input
['flowgraph', 'dvflow', 'sim', '1', 'input']	('compile', '0')
['flowgraph', 'dvflow', 'sim', '1', 'tool']	execute
['flowgraph', 'dvflow', 'sim', '1', 'task']	exec_input
['flowgraph', 'dvflow', 'sim', '1', 'taskmodule']	siliconcompiler.tools.execute.exec_input
['flowgraph', 'dvflow', 'sim', '2', 'input']	('compile', '0')
['flowgraph', 'dvflow', 'sim', '2', 'tool']	execute
['flowgraph', 'dvflow', 'sim', '2', 'task']	exec_input
['flowgraph', 'dvflow', 'sim', '2', 'taskmodule']	siliconcompiler.tools.execute.exec_input
['flowgraph', 'dvflow', 'sim', '3', 'input']	('compile', '0')
['flowgraph', 'dvflow', 'sim', '3', 'tool']	execute
['flowgraph', 'dvflow', 'sim', '3', 'task']	exec_input
['flowgraph', 'dvflow', 'sim', '3', 'taskmodule']	siliconcompiler.tools.execute.exec_input
['flowgraph', 'dvflow', 'sim', '4', 'input']	('compile', '0')
['flowgraph', 'dvflow', 'sim', '4', 'tool']	execute
['flowgraph', 'dvflow', 'sim', '4', 'task']	exec_input
['flowgraph', 'dvflow', 'sim', '4', 'taskmodule']	siliconcompiler.tools.execute.exec_input

3.2.4 fpgaflow

A configurable FPGA compilation flow.

The 'fpgaflow' module is a configurable FPGA flow with support for open source and commercial tool flows.

The following step convention is recommended for VPR.

- **import**: Sources are collected and packaged for compilation
- **syn**: Synthesize RTL into an device specific netlist

- **place**: FPGA specific placement step
- **route**: FPGA specific routing step
- **bitstream**: Bitstream generation

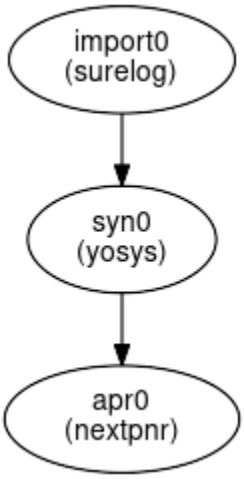
Note that nextpnr does not appear to support breaking placement, routing, and bitstream generation into individual steps, leading to the following recommended step convention

- **import**: Sources are collected and packaged for compilation
- **syn**: Synthesize RTL into an device specific netlist
- **apr**: One-step execution of place, route, bitstream with nextpnr

Args:

- fpgaflow_type (str): this parameter can be used to select a specific fpga flow instead of one selected from the partname.
- partname (str): this parameter can be sued to selecte a specific fpga flow instead of one sleected from the partname set in the schema.

Setup file: fpgaflow.py



Configuration

import

Keypath	Value
['flowgraph', 'fpgaflow', 'import', '0', 'weight', 'luts']	1.0
['flowgraph', 'fpgaflow', 'import', '0', 'weight', 'dsps']	1.0
['flowgraph', 'fpgaflow', 'import', '0', 'weight', 'brams']	1.0
['flowgraph', 'fpgaflow', 'import', '0', 'weight', 'registers']	1.0
['flowgraph', 'fpgaflow', 'import', '0', 'weight', 'pins']	1.0

continues on next page

Table 31 – continued from previous page

['flowgraph', 'fpgaflow', 'import', '0', 'goal', 'errors']	0.0
['flowgraph', 'fpgaflow', 'import', '0', 'goal', 'warnings']	0.0
['flowgraph', 'fpgaflow', 'import', '0', 'goal', 'drvs']	0.0
['flowgraph', 'fpgaflow', 'import', '0', 'goal', 'unconstrained']	0.0
['flowgraph', 'fpgaflow', 'import', '0', 'goal', 'holdwns']	0.0
['flowgraph', 'fpgaflow', 'import', '0', 'goal', 'holdtns']	0.0
['flowgraph', 'fpgaflow', 'import', '0', 'goal', 'holdpaths']	0.0
['flowgraph', 'fpgaflow', 'import', '0', 'goal', 'setupwns']	0.0
['flowgraph', 'fpgaflow', 'import', '0', 'goal', 'setuptns']	0.0
['flowgraph', 'fpgaflow', 'import', '0', 'goal', 'setuppaths']	0.0
['flowgraph', 'fpgaflow', 'import', '0', 'tool']	surelog
['flowgraph', 'fpgaflow', 'import', '0', 'task']	parse
['flowgraph', 'fpgaflow', 'import', '0', 'taskmodule']	siliconcompiler.tools.surelog.parse

syn

Keypath	Value
['flowgraph', 'fpgaflow', 'syn', '0', 'input']	('import', '0')
['flowgraph', 'fpgaflow', 'syn', '0', 'weight', 'luts']	1.0
['flowgraph', 'fpgaflow', 'syn', '0', 'weight', 'dsps']	1.0
['flowgraph', 'fpgaflow', 'syn', '0', 'weight', 'brams']	1.0
['flowgraph', 'fpgaflow', 'syn', '0', 'weight', 'registers']	1.0
['flowgraph', 'fpgaflow', 'syn', '0', 'weight', 'pins']	1.0
['flowgraph', 'fpgaflow', 'syn', '0', 'goal', 'errors']	0.0
['flowgraph', 'fpgaflow', 'syn', '0', 'goal', 'warnings']	0.0
['flowgraph', 'fpgaflow', 'syn', '0', 'goal', 'drvs']	0.0
['flowgraph', 'fpgaflow', 'syn', '0', 'goal', 'unconstrained']	0.0
['flowgraph', 'fpgaflow', 'syn', '0', 'goal', 'holdwns']	0.0

continues on next page

Table 32 – continued from previous page

['flowgraph', 'fpgaflow', 'syn', '0', 'goal', 'holdtns']	0.0
['flowgraph', 'fpgaflow', 'syn', '0', 'goal', 'holdpaths']	0.0
['flowgraph', 'fpgaflow', 'syn', '0', 'goal', 'setupwns']	0.0
['flowgraph', 'fpgaflow', 'syn', '0', 'goal', 'setuptns']	0.0
['flowgraph', 'fpgaflow', 'syn', '0', 'goal', 'setuppaths']	0.0
['flowgraph', 'fpgaflow', 'syn', '0', 'tool']	yosys
['flowgraph', 'fpgaflow', 'syn', '0', 'task']	syn_fpga
['flowgraph', 'fpgaflow', 'syn', '0', 'taskmodule']	siliconcompiler.tools.yosys.syn_fpga

apr

Keypath	Value
['flowgraph', 'fpgaflow', 'apr', '0', 'input']	('syn', '0')
['flowgraph', 'fpgaflow', 'apr', '0', 'weight', 'luts']	1.0
['flowgraph', 'fpgaflow', 'apr', '0', 'weight', 'dsps']	1.0
['flowgraph', 'fpgaflow', 'apr', '0', 'weight', 'brams']	1.0
['flowgraph', 'fpgaflow', 'apr', '0', 'weight', 'registers']	1.0
['flowgraph', 'fpgaflow', 'apr', '0', 'weight', 'pins']	1.0
['flowgraph', 'fpgaflow', 'apr', '0', 'goal', 'errors']	0.0
['flowgraph', 'fpgaflow', 'apr', '0', 'goal', 'warnings']	0.0
['flowgraph', 'fpgaflow', 'apr', '0', 'goal', 'drvs']	0.0
['flowgraph', 'fpgaflow', 'apr', '0', 'goal', 'unconstrained']	0.0
['flowgraph', 'fpgaflow', 'apr', '0', 'goal', 'holdwns']	0.0
['flowgraph', 'fpgaflow', 'apr', '0', 'goal', 'holdtns']	0.0
['flowgraph', 'fpgaflow', 'apr', '0', 'goal', 'holdpaths']	0.0
['flowgraph', 'fpgaflow', 'apr', '0', 'goal', 'setupwns']	0.0
['flowgraph', 'fpgaflow', 'apr', '0', 'goal', 'setuptns']	0.0
['flowgraph', 'fpgaflow', 'apr', '0', 'goal', 'setuppaths']	0.0

continues on next page

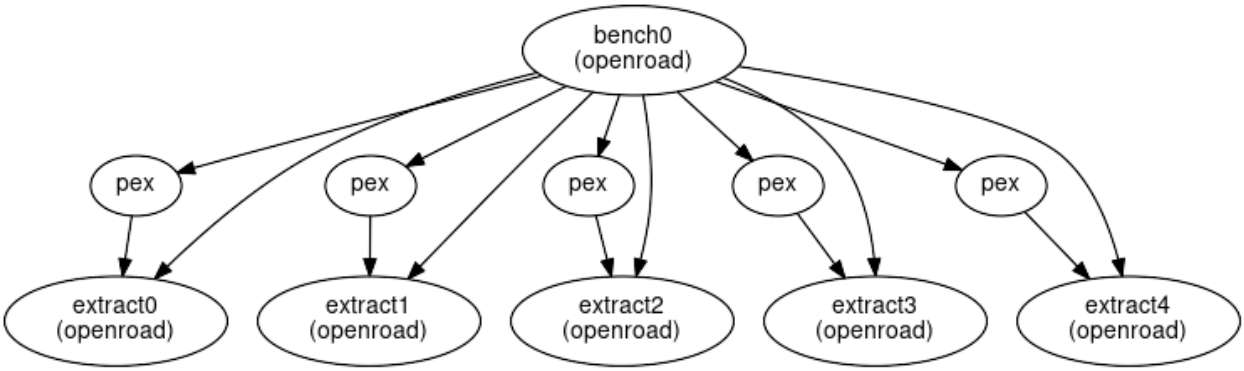
Table 33 – continued from previous page

['flowgraph', 'fpgaflow', 'apr', '0', 'tool']	nextpnr
['flowgraph', 'fpgaflow', 'apr', '0', 'task']	apr
['flowgraph', 'fpgaflow', 'apr', '0', 'taskmodule']	siliconcompiler.tools.nextpnr.apr

3.2.5 generate_openroad_rcx

Flow to generate the OpenRCX decks needed by OpenROAD to do parasitic extraction.

Setup file: generate_openroad_rcx.py



Configuration

bench

Keypath	Value
['flowgraph', 'generate_rcx', 'bench', '0', 'tool']	openroad
['flowgraph', 'generate_rcx', 'bench', '0', 'task']	rcx_bench
['flowgraph', 'generate_rcx', 'bench', '0', 'taskmodule']	siliconcompiler.tools.openroad.rcx_bench

pex

Keypath	Value
['flowgraph', 'generate_rcx', 'pex', '0', 'input']	('bench', '0')
['flowgraph', 'generate_rcx', 'pex', '0', 'tool']	builtin
['flowgraph', 'generate_rcx', 'pex', '0', 'task']	nop
['flowgraph', 'generate_rcx', 'pex', '0', 'taskmodule']	siliconcompiler.tools.builtin.nop

continues on next page

Table 35 – continued from previous page

['flowgraph', 'generate_rcx', 'pex', '1', 'input']	('bench', '0')
['flowgraph', 'generate_rcx', 'pex', '1', 'tool']	builtin
['flowgraph', 'generate_rcx', 'pex', '1', 'task']	nop
['flowgraph', 'generate_rcx', 'pex', '1', 'taskmodule']	siliconcompiler.tools.builtin.nop
['flowgraph', 'generate_rcx', 'pex', '2', 'input']	('bench', '0')
['flowgraph', 'generate_rcx', 'pex', '2', 'tool']	builtin
['flowgraph', 'generate_rcx', 'pex', '2', 'task']	nop
['flowgraph', 'generate_rcx', 'pex', '2', 'taskmodule']	siliconcompiler.tools.builtin.nop
['flowgraph', 'generate_rcx', 'pex', '3', 'input']	('bench', '0')
['flowgraph', 'generate_rcx', 'pex', '3', 'tool']	builtin
['flowgraph', 'generate_rcx', 'pex', '3', 'task']	nop
['flowgraph', 'generate_rcx', 'pex', '3', 'taskmodule']	siliconcompiler.tools.builtin.nop
['flowgraph', 'generate_rcx', 'pex', '4', 'input']	('bench', '0')
['flowgraph', 'generate_rcx', 'pex', '4', 'tool']	builtin
['flowgraph', 'generate_rcx', 'pex', '4', 'task']	nop
['flowgraph', 'generate_rcx', 'pex', '4', 'taskmodule']	siliconcompiler.tools.builtin.nop

extract

Keypath	Value
['flowgraph', 'generate_rcx', 'extract', '0', 'input']	• ('pex', '0') • ('bench', '0')
['flowgraph', 'generate_rcx', 'extract', '0', 'tool']	openroad
['flowgraph', 'generate_rcx', 'extract', '0', 'task']	rcx_extract
['flowgraph', 'generate_rcx', 'extract', '0', 'taskmodule']	siliconcompiler.tools.openroad.rcx_extract

continues on next page

Table 36 – continued from previous page

['flowgraph', 'generate_rcx', 'extract', '1', 'input']	• ('pex', '1') • ('bench', '0')
['flowgraph', 'generate_rcx', 'extract', '1', 'tool']	openroad
['flowgraph', 'generate_rcx', 'extract', '1', 'task']	rcx_extract
['flowgraph', 'generate_rcx', 'extract', '1', 'taskmodule']	siliconcompiler.tools.openroad.rcx_extract
['flowgraph', 'generate_rcx', 'extract', '2', 'input']	• ('pex', '2') • ('bench', '0')
['flowgraph', 'generate_rcx', 'extract', '2', 'tool']	openroad
['flowgraph', 'generate_rcx', 'extract', '2', 'task']	rcx_extract
['flowgraph', 'generate_rcx', 'extract', '2', 'taskmodule']	siliconcompiler.tools.openroad.rcx_extract
['flowgraph', 'generate_rcx', 'extract', '3', 'input']	• ('pex', '3') • ('bench', '0')
['flowgraph', 'generate_rcx', 'extract', '3', 'tool']	openroad
['flowgraph', 'generate_rcx', 'extract', '3', 'task']	rcx_extract
['flowgraph', 'generate_rcx', 'extract', '3', 'taskmodule']	siliconcompiler.tools.openroad.rcx_extract
['flowgraph', 'generate_rcx', 'extract', '4', 'input']	• ('pex', '4') • ('bench', '0')
['flowgraph', 'generate_rcx', 'extract', '4', 'tool']	openroad
['flowgraph', 'generate_rcx', 'extract', '4', 'task']	rcx_extract
['flowgraph', 'generate_rcx', 'extract', '4', 'taskmodule']	siliconcompiler.tools.openroad.rcx_extract

3.2.6 lintflow

An RTL linting flow.

Setup file: lintflow.py

Configuration

lint

Keypath	Value
['flowgraph', 'lintflow', 'lint', '0', 'tool']	verilator
['flowgraph', 'lintflow', 'lint', '0', 'task']	lint
['flowgraph', 'lintflow', 'lint', '0', 'taskmodule']	siliconcompiler.tools.verilator.lint

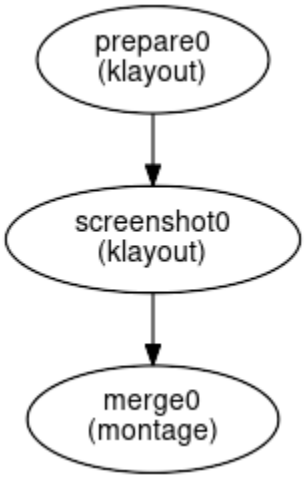
3.2.7 screenshotflow

Flow to generate a high resolution design image from a GDS or OAS file.

The 'screenshotflow' includes the stages below.

- **prepare**: Prepare the stream file, such as flattening design, removing layers, and merging shapes
- **screenshot**: Generate a set of screenshots tiled across the design
- **merge**: Merge tiled images into a single image

Setup file: screenshotflow.py



Configuration

prepare

Keypath	Value
['flowgraph', 'screenshotflow', 'prepare', '0', 'tool']	klayout
['flowgraph', 'screenshotflow', 'prepare', '0', 'task']	operations

continues on next page

Table 38 – continued from previous page

['flowgraph', 'screenshotflow', 'prepare', '0', 'taskmodule']	siliconcompiler.tools.klayout.operations

screenshot

Keypath	Value
['flowgraph', 'screenshotflow', 'screenshot', '0', 'input']	('prepare', '0')
['flowgraph', 'screenshotflow', 'screenshot', '0', 'tool']	klayout
['flowgraph', 'screenshotflow', 'screenshot', '0', 'task']	screenshot
['flowgraph', 'screenshotflow', 'screenshot', '0', 'taskmodule']	siliconcompiler.tools.klayout.screenshot

merge

Keypath	Value
['flowgraph', 'screenshotflow', 'merge', '0', 'input']	('screenshot', '0')
['flowgraph', 'screenshotflow', 'merge', '0', 'tool']	montage
['flowgraph', 'screenshotflow', 'merge', '0', 'task']	tile
['flowgraph', 'screenshotflow', 'merge', '0', 'taskmodule']	siliconcompiler.tools.montage.tile

3.2.8 showflow

A flow to show the output files generated from other flows.

Required settings for this flow are below:

- filetype : Type of file to show

Optional settings for this flow are below:

- np : Number of parallel show jobs to launch
- screenshot : true/false, indicate if this should be configured as a screenshot

Setup file: showflow.py

Configuration

show

Keypath	Value
['flowgraph', 'showflow', 'show', '0', 'tool']	klayout
['flowgraph', 'showflow', 'show', '0', 'task']	show
['flowgraph', 'showflow', 'show', '0', 'taskmodule']	siliconcompiler.tools.klayout.show
['flowgraph', 'showflow', 'show', '1', 'tool']	klayout
['flowgraph', 'showflow', 'show', '1', 'task']	show
['flowgraph', 'showflow', 'show', '1', 'taskmodule']	siliconcompiler.tools.klayout.show
['flowgraph', 'showflow', 'show', '2', 'tool']	klayout
['flowgraph', 'showflow', 'show', '2', 'task']	show
['flowgraph', 'showflow', 'show', '2', 'taskmodule']	siliconcompiler.tools.klayout.show

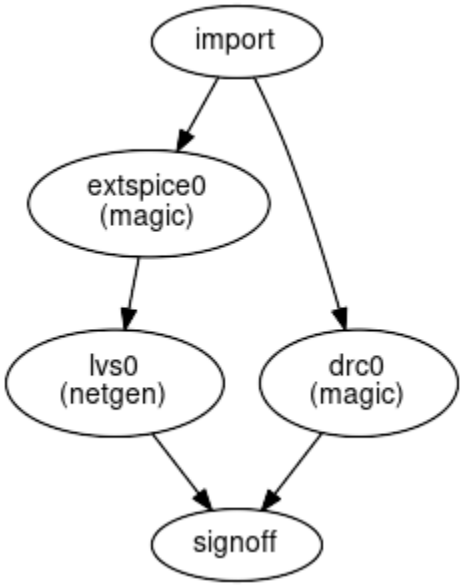
3.2.9 signoffflow

A flow for running LVS/DRC signoff on a GDS layout.

Inputs must be passed to this flow as follows:

```
flow.input('<path-to-layout>.gds')
flow.input('<path-to-netlist>.vg')
```

Setup file: signoffflow.py



Configuration

import

Keypath	Value
['flowgraph', 'signoffflow', 'import', '0', 'goal', 'errors']	0.0
['flowgraph', 'signoffflow', 'import', '0', 'tool']	builtin
['flowgraph', 'signoffflow', 'import', '0', 'task']	nop
['flowgraph', 'signoffflow', 'import', '0', 'taskmodule']	siliconcompiler.tools.builtin.nop

extspice

Keypath	Value
['flowgraph', 'signoffflow', 'extspice', '0', 'input']	('import', '0')
['flowgraph', 'signoffflow', 'extspice', '0', 'goal', 'errors']	0.0
['flowgraph', 'signoffflow', 'extspice', '0', 'tool']	magic
['flowgraph', 'signoffflow', 'extspice', '0', 'task']	extspice
['flowgraph', 'signoffflow', 'extspice', '0', 'taskmodule']	siliconcompiler.tools.magic.extspice

drc

Keypath	Value
['flowgraph', 'signoffflow', 'drc', '0', 'input']	('import', '0')
['flowgraph', 'signoffflow', 'drc', '0', 'goal', 'errors']	0.0
['flowgraph', 'signoffflow', 'drc', '0', 'tool']	magic
['flowgraph', 'signoffflow', 'drc', '0', 'task']	drc
['flowgraph', 'signoffflow', 'drc', '0', 'taskmodule']	siliconcompiler.tools.magic.drc

lvs

Keypath	Value
['flowgraph', 'signoffflow', 'lvs', '0', 'input']	('extspice', '0')
['flowgraph', 'signoffflow', 'lvs', '0', 'goal', 'errors']	0.0
['flowgraph', 'signoffflow', 'lvs', '0', 'tool']	netgen
['flowgraph', 'signoffflow', 'lvs', '0', 'task']	lvs
['flowgraph', 'signoffflow', 'lvs', '0', 'taskmodule']	siliconcompiler.tools.netgen.lvs

signoff

Keypath	Value
['flowgraph', 'signoffflow', 'signoff', '0', 'input']	• ('lvs', '0') • ('drc', '0')
['flowgraph', 'signoffflow', 'signoff', '0', 'goal', 'errors']	0.0
['flowgraph', 'signoffflow', 'signoff', '0', 'tool']	builtin
['flowgraph', 'signoffflow', 'signoff', '0', 'task']	join
['flowgraph', 'signoffflow', 'signoff', '0', 'taskmodule']	siliconcompiler.tools.builtin.join

3.3 Pre-Defined Tools

The following are examples of pre-built tool drivers that come with SiliconCompiler which you can use for your own builds.

See the pre-built *targets* for examples on how these are used in conjunction with *pdks*, *flows* and *libraries*.

3.3.1 bambu

The primary objective of the PandA project is to develop a usable framework that will enable the research of new ideas in the HW-SW Co-Design field.

The PandA framework includes methodologies supporting the research on high-level synthesis of hardware accelerators, on parallelism extraction for embedded systems, on hardware/software partitioning and mapping, on metrics for performance estimation of embedded software applications and on dynamic reconfigurable devices.

Documentation: https://github.com/ferrandi/PandA-bambu

Sources: https://github.com/ferrandi/PandA-bambu

Installation: https://panda.dei.polimi.it/?page_id=88

Setup file: bambu.py

Data sources

Package	Specifications
siliconcompiler	• Path: `python://siliconcompiler`

Keypath	Value
['tool', 'bambu', 'exe']	bambu
['tool', 'bambu', 'vswitch']	--version
['tool', 'bambu', 'version']	>=0.9.6

convert

Performs high level synthesis to generate a verilog output

Setup file: convert.py

3.3.2 bluespec

Bluespec is a high-level hardware description language. It has a variety of advanced features including a powerful type system that can prevent errors prior to synthesis time, and its most distinguishing feature, Guarded Atomic Actions, allow you to define hardware components in a modular manner based on their invariants, and let the compiler pick a scheduler.

Documentation: https://github.com/B-Lang-org/bsc#documentation

Sources: https://github.com/B-Lang-org/bsc

Installation: https://github.com/B-Lang-org/bsc#download

Setup file: bluespec.py

Data sources

Package	Specifications
siliconcompiler	• Path: `python://siliconcompiler`

Keypath	Value
['tool', 'bluespec', 'exe']	bsc

continues on next page

Table 50 – continued from previous page

['tool', 'bluespec', 'vswitch']	-v
['tool', 'bluespec', 'version']	>=2021.07

convert

Performs high level synthesis to generate a verilog output

Setup file: convert.py

3.3.3 builtin

Builtin tools for SiliconCompiler

Setup file: builtin.py

join

Merges outputs from a list of input tasks.

Setup file: join.py

maximum

Selects the task with the maximum metric score from a list of inputs.

Sequence of operation:

1. Check list of input tasks to see if all metrics meets goals
2. Check list of input tasks to find global min/max for each metric
3. Select MAX value if all metrics are met.
4. Normalize the min value as sel = (val - MIN) / (MAX - MIN)
5. Return normalized value and task name

Meeting metric goals takes precedence over compute metric scores. Only goals with values set and metrics with weights set are considered in the calculation.

Setup file: maximum.py

minimum

Selects the task with the minimum metric score from a list of inputs.

Sequence of operation:

1. Check list of input tasks to see if all metrics meets goals
2. Check list of input tasks to find global min/max for each metric
3. Select MIN value if all metrics are met.
4. Normalize the min value as sel = (val - MIN) / (MAX - MIN)
5. Return normalized value and task name

Meeting metric goals takes precedence over compute metric scores. Only goals with values set and metrics with weights set are considered in the calculation.

Setup file: minimum.py

mux

Selects a task from a list of inputs.

The selector criteria provided is used to create a custom function for selecting the best step/index pair from the inputs. Metrics and weights are passed in and used to select the step/index based on the minimum or maximum score depending on the 'op' argument from ['flowgraph', flow, step, index, 'args'] in the form 'minimum(metric)' or 'maximum(metric)'.

The function can be used to bypass the flows weight functions for the purpose of conditional flow execution and verification.

Setup file: mux.py

nop

A no-operation that passes inputs to outputs.

Setup file: nop.py

verify

Tests an assertion on an input task.

The input to this task is verified to ensure that all assertions are True. If any of the assertions fail, False is returned. Assertions are passed in using ['flowgraph', flow, step, index, 'args'] in the form 'metric==0.0'. The allowed conditional operators are: >, <, >=, <=, ==

Setup file: verify.py

3.3.4 chisel

Chisel is a hardware design language that facilitates advanced circuit generation and design reuse for both ASIC and FPGA digital logic designs. Chisel adds hardware construction primitives to the Scala programming language, providing designers with the power of a modern programming language to write complex, parameterizable circuit generators that produce synthesizable Verilog.

Documentation: https://www.chisel-lang.org/docs

Sources: https://github.com/chipsalliance/chisel

Installation: The Chisel plugin relies on having the Scala Build Tool (sbt) installed. Instructions: https://www.scala-sbt.org/download.html.

Setup file: chisel.py

Data sources

Package	Specifications
siliconcompiler	• Path: `python://siliconcompiler`

Keypath	Value
['tool', 'chisel', 'exe']	`sbt`
['tool', 'chisel', 'vswitch']	`--version`
['tool', 'chisel', 'version']	`>=1.5.5`

convert

Performs high level synthesis to generate a verilog output

Setup file: convert.py

3.3.5 execute

This tool is used to execute the output of a previous step. For example, if the flow contains a compile step which generates the next executable needed in the flow.

Setup file: execute.py

Keypath	Value
['tool', 'execute', 'exe']	`:exe:`

exec_input

Execute the output of the previous step directly. This only works if the task receives a single file.

Setup file: exec_input.py

Configuration

Keypath	Value
['tool', 'execute', 'task', 'exec_input', 'option']	

3.3.6 ghdl

GHDL is an open-source analyzer, compiler, simulator and (experimental) synthesizer for VHDL. It allows you to analyse and elaborate sources for generating machine code from your design. Native program execution is the only way for high speed simulation.

Documentation: https://ghdl.readthedocs.io/en/latest

Sources: https://github.com/ghdl/ghdl

Installation: https://github.com/ghdl/ghdl

Setup file: ghdl.py

Keypath	Value
['tool', 'ghdl', 'exe']	ghdl
['tool', 'ghdl', 'vswitch']	--version
['tool', 'ghdl', 'version']	>=4.0.0-dev

convert

Imports VHDL and converts it to verilog

Setup file: convert.py

3.3.7 klayout

Klayout is a production grade viewer and editor of GDSII and Oasis data with customizable Python and Ruby interfaces.

Documentation: https://www.klayout.de

Sources: https://github.com/KLayout/klayout

Installation: https://www.klayout.de/build.html

Setup file: klayout.py

Data sources

Package	Specifications
siliconcompiler	• Path: python://siliconcompiler
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

Keypath	Value

continues on next page

Table 57 – continued from previous page

['tool', 'klayout', 'exe']	klayout
['tool', 'klayout', 'vswitch']	• -zz • -v
['tool', 'klayout', 'version']	>=0.28.0
['tool', 'klayout', 'format']	json

export

Generate a GDSII file from an input DEF file

Setup file: export.py

Built using target: *freepdk45_demo*

Configuration

Keypath	Value
['tool', 'klayout', 'task', 'export', 'regex', 'warnings']	(WARNING|warning)
['tool', 'klayout', 'task', 'export', 'regex', 'errors']	ERROR
['tool', 'klayout', 'task', 'export', 'option']	• -z • -nc • -rx • -r
['tool', 'klayout', 'task', 'export', 'var', 'stream']	gds
['tool', 'klayout', 'task', 'export', 'var', 'timestamps']	true
['tool', 'klayout', 'task', 'export', 'var', 'screenshot']	true
['tool', 'klayout', 'task', 'export', 'var', 'show_exit']	true
['tool', 'klayout', 'task', 'export', 'var', 'show_horizontal_resolution']	4096
['tool', 'klayout', 'task', 'export', 'var', 'show_vertical_resolution']	4096
['tool', 'klayout', 'task', 'export', 'var', 'xbins']	1
['tool', 'klayout', 'task', 'export', 'var', 'ybins']	1
['tool', 'klayout', 'task', 'export', 'var', 'margin']	10
['tool', 'klayout', 'task', 'export', 'var', 'linewidth']	0

continues on next page

Table 58 – continued from previous page

['tool', 'klayout', 'task', 'export', 'var', 'oversampling']	2
['tool', 'klayout', 'task', 'export', 'input']	<design>.def
['tool', 'klayout', 'task', 'export', 'output']	• <design>.gds • <design>.lyt • <design>.png
['tool', 'klayout', 'task', 'export', 'require']	• asic,logiclib • option,stackup • pdk,freepdk45,layermap,klayout,def,klayout,10M • library,nangate45,output,10M,gds • library,nangate45,output,10M,lef
['tool', 'klayout', 'task', 'export', 'refdir']	tools/klayout, siliconcompiler
['tool', 'klayout', 'task', 'export', 'script']	klayout_export.py

Variables

Parameters	Help
[..., 'var', 'stream']	Extension to use for stream generation ((‘gds’, ‘oas’))
[..., 'var', 'timestamps']	Export GDSII with timestamps
[..., 'var', 'screenshot']	true/false: true will cause KLayout to generate a screenshot of the layout
[..., 'var', 'show_exit']	true/false: true will cause kLayout to exit when complete
[..., 'var', 'hide_layers']	List of layers to hide
[..., 'var', 'show_filepath']	File to open
[..., 'var', 'show_filetype']	File type to look for in the inputs
[..., 'var', 'show_horizontal_resolution']	Horizontal resolution in pixels
[..., 'var', 'show_vertical_resolution']	Vertical resolution in pixels
[..., 'var', 'xbins']	If greater than 1, splits the image into multiple segments along x-axis
[..., 'var', 'ybins']	If greater than 1, splits the image into multiple segments along y-axis
[..., 'var', 'margin']	Margin around design in microns
[..., 'var', 'linewidth']	Width of lines in detailed screenshots
[..., 'var', 'oversampling']	Image oversampling used in detailed screenshots

operations

Perform unit operations on stream files. Currently supports:

- rotating (rotate)
- renaming (rename)
- merging streams (merge)
- adding streams together (add)
- adding outline to top (outline)
- swapping cells (swap)
- adding new top cell (add_top)
- renaming cells (rename_cell)
- flatten
- deleting layers
- merging shapes
- writing (write)
- converting properties into text labels on design (convert_property)

To rotate:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', 'rotate')
```

To rename:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'rename:tool,klayout,task,operations,var,new_name')
>>> chip.set('tool', 'klayout, 'task', 'operations', 'var', 'new_name', \
    'chip_top')
```

To merge streams:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'merge:tool,klayout,task,operations,file,fill_stream')
>>> chip.set('tool', 'klayout, 'task', 'operations', 'file', 'fill_stream', \
    './fill.gds')
```

or to get it from the inputs to this task:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'merge:fill.gds')
```

To add streams:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'add:tool,klayout,task,operations,file,fill_stream')
>>> chip.set('tool', 'klayout, 'task', 'operations', 'file', 'fill_stream', \
    './fill.gds')
```

or to get it from the inputs to this task:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'add:fill.gds')
```

To add outline:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'outline:tool,klayout,task,operations,var,outline')
>>> chip.set('tool', 'klayout, 'task', 'operations', 'var', 'outline', \
    ['10', '1'])  # layer / purpose pair
```

To swap layout cells:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'swap:tool,klayout,task,operations,var,cell_swap')
>>> chip.set('tool', 'klayout, 'task', 'operations', 'var', 'cell_swap', \
    ['dummy_ANDX2=ANDX2', 'dummy_NANDX2=NANDX2'])
```

To rename cells:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'rename_cell:tool,klayout,task,operations,var,rename_cell')
>>> chip.set('tool', 'klayout, 'task', 'operations', 'var', 'rename_cell', \
    ['dummy_ANDX2=ANDX2', 'dummy_NANDX2=NANDX2'])
```

To add new top cell:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'add_top:tool,klayout,task,operations,var,new_name')
>>> chip.set('tool', 'klayout, 'task', 'operations', 'var', 'new_name', \
    'chip_top')
```

To write out a new file:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'write:combined.gds')
```

To convert stream properties to text labels:

```
>>> chip.add('tool', 'klayout, 'task', 'operations', 'var', 'operations', \
    'convert_property:tool,klayout,task,operations,var,convert_c4_bumps')
>>> chip.set('tool', 'klayout, 'task', 'operations', 'var', 'convert_c4_bumps', \
    ['10', '2', \  # layer / purpose pair for the source of the labels
     '3' \  # stream property number
     '85', '5'])  #  (optional) destination layer / purpose pair, if not provided
                  # the source pair will be used instead.
```

Setup file: operations.py

Built using target: *freepdk45_demo*

Configuration

Keypath	Value
['tool', 'klayout', 'task', 'operations', 'regex', 'warnings']	(WARNING|warning)
['tool', 'klayout', 'task', 'operations', 'regex', 'errors']	ERROR
['tool', 'klayout', 'task', 'operations', 'option']	• -z • -nc • -rx • -r
['tool', 'klayout', 'task', 'operations', 'var', 'stream']	gds
['tool', 'klayout', 'task', 'operations', 'var', 'timestamps']	true
['tool', 'klayout', 'task', 'operations', 'output']	<design>.gds
['tool', 'klayout', 'task', 'operations', 'require']	input,layout,gds
['tool', 'klayout', 'task', 'operations', 'refdir']	tools/klayout, siliconcompiler
['tool', 'klayout', 'task', 'operations', 'script']	klayout_operations.py

Variables

Parameters	Help
[..., 'var', 'stream']	Extension to use for stream generation ((‘gds’, ‘oas’))
[..., 'var', 'timestamps']	Export GDSII with timestamps

screenshot

Generate a PNG file from a layout file

Setup file: screenshot.py

Built using target: *freepdk45_demo*

Configuration

Keypath	Value
['tool', 'klayout', 'task', 'screenshot', 'regex', 'warnings']	(WARNING|warning)
['tool', 'klayout', 'task', 'screenshot', 'regex', 'errors']	ERROR
['tool', 'klayout', 'task', 'screenshot', 'option']	• -nc • -z • -rm
['tool', 'klayout', 'task', 'screenshot', 'var', 'show_filepath']	<path>
['tool', 'klayout', 'task', 'screenshot', 'var', 'show_exit']	true
['tool', 'klayout', 'task', 'screenshot', 'var', 'show_horizontal_resolution']	4096
['tool', 'klayout', 'task', 'screenshot', 'var', 'show_vertical_resolution']	4096
['tool', 'klayout', 'task', 'screenshot', 'var', 'xbins']	1
['tool', 'klayout', 'task', 'screenshot', 'var', 'ybins']	1
['tool', 'klayout', 'task', 'screenshot', 'var', 'margin']	10
['tool', 'klayout', 'task', 'screenshot', 'var', 'linewidth']	0
['tool', 'klayout', 'task', 'screenshot', 'var', 'oversampling']	2
['tool', 'klayout', 'task', 'screenshot', 'output']	<design>.png
['tool', 'klayout', 'task', 'screenshot', 'require']	tool,klayout,task,screenshot,var,show_filepath
['tool', 'klayout', 'task', 'screenshot', 'refdir']	tools/klayout, siliconcompiler
['tool', 'klayout', 'task', 'screenshot', 'script']	klayout_show.py

Variables

Parameters	Help
[..., 'var', 'show_filepath']	File to open
[..., 'var', 'show_exit']	true/false: true will cause kLayout to exit when complete
[..., 'var', 'hide_layers']	List of layers to hide
[..., 'var', 'show_filetype']	File type to look for in the inputs
[..., 'var', 'show_horizontal_resolution']	Horizontal resolution in pixels
[..., 'var', 'show_vertical_resolution']	Vertical resolution in pixels

continues on next page

Table 63 – continued from previous page

[..., 'var', 'xbins']	If greater than 1, splits the image into multiple segments along x-axis
[..., 'var', 'ybins']	If greater than 1, splits the image into multiple segments along y-axis
[..., 'var', 'margin']	Margin around design in microns
[..., 'var', 'linewidth']	Width of lines in detailed screenshots
[..., 'var', 'oversampling']	Image oversampling used in detailed screenshots

show

Show a layout in kLayout

Setup file: show.py

Built using target: *freepdk45_demo*

Configuration

Keypath	Value
['tool', 'klayout', 'task', 'show', 'regex', 'warnings']	(WARNING|warning)
['tool', 'klayout', 'task', 'show', 'regex', 'errors']	ERROR
['tool', 'klayout', 'task', 'show', 'option']	• -nc • -rm
['tool', 'klayout', 'task', 'show', 'var', 'show_filepath']	<path>
['tool', 'klayout', 'task', 'show', 'var', 'show_exit']	false
['tool', 'klayout', 'task', 'show', 'require']	tool,klayout,task,show,var,show_filepath
['tool', 'klayout', 'task', 'show', 'refdir']	tools/klayout, siliconcompiler
['tool', 'klayout', 'task', 'show', 'script']	klayout_show.py

Variables

Parameters	Help
[..., 'var', 'show_filepath']	File to open
[..., 'var', 'show_exit']	true/false: true will cause kLayout to exit when complete
[..., 'var', 'hide_layers']	List of layers to hide
[..., 'var', 'show_filetype']	File type to look for in the inputs

3.3.8 magic

Magic is a chip layout viewer, editor, and circuit verifier with built in DRC and LVS engines.

Documentation: http://opencircuitdesign.com/magic/userguide.html

Installation: https://github.com/RTimothyEdwards/magic

Sources: https://github.com/RTimothyEdwards/magic

Setup file: magic.py

Data sources

Package	Specifications
siliconcompiler	• Path: `python://siliconcompiler`
lambdapdk	• Path: `https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/` • Reference: `v0.1.19`

Keypath	Value
['tool', 'magic', 'exe']	`magic`
['tool', 'magic', 'vswitch']	`--version`
['tool', 'magic', 'version']	`>=8.3.196`
['tool', 'magic', 'format']	`tcl`

drc

Perform DRC checks

Setup file: drc.py

Built using target: *freepdk45_demo*

Configuration

Keypath	Value
['tool', 'magic', 'task', 'drc', 'regex', 'errors']	`^Error`
['tool', 'magic', 'task', 'drc', 'regex', 'warnings']	`warning`

continues on next page

Table 68 – continued from previous page

['tool', 'magic', 'task', 'drc', 'option']	• -noc • -dnull
['tool', 'magic', 'task', 'drc', 'input']	<design>.gds
['tool', 'magic', 'task', 'drc', 'refdir']	tools/magic, siliconcompiler
['tool', 'magic', 'task', 'drc', 'script']	sc_magic.tcl
['tool', 'magic', 'task', 'drc', 'threads']	2

extspice

Extract spice netlists from a GDS file for simulation use

Setup file: extspice.py

Built using target: *freepdk45_demo*

Configuration

Keypath	Value
['tool', 'magic', 'task', 'extspice', 'regex', 'errors']	^Error
['tool', 'magic', 'task', 'extspice', 'regex', 'warnings']	warning
['tool', 'magic', 'task', 'extspice', 'option']	• -noc • -dnull
['tool', 'magic', 'task', 'extspice', 'input']	<design>.gds
['tool', 'magic', 'task', 'extspice', 'output']	<design>.spice
['tool', 'magic', 'task', 'extspice', 'refdir']	tools/magic, siliconcompiler
['tool', 'magic', 'task', 'extspice', 'script']	sc_magic.tcl
['tool', 'magic', 'task', 'extspice', 'threads']	2

3.3.9 montage

ImageMagick® is a free and open-source software suite for displaying, converting, and editing raster image and vector image files. It can read and write over 200 image file formats, and can support a wide range of image manipulation operations, such as resizing, cropping, and color correction. Use the montage program to create a composite image by combining several separate images. The images are tiled on the composite image optionally adorned with a border, frame, image name, and more

Documentation: https://imagemagick.org/

Sources: https://github.com/ImageMagick/ImageMagick

Installation: https://github.com/ImageMagick/ImageMagick

Setup file: montage.py

Keypath	Value
['tool', 'montage', 'exe']	montage
['tool', 'montage', 'vswitch']	-version
['tool', 'montage', 'version']	>=6.9.0

tile

Tiles input images into a single output image.

Notes: Need to make ensure that /etc/ImageMagick-6/policy.xml

> <policy domain="resource" name="memory" value="8GiB"/> <policy domain="resource" name="map" value="8GiB"/> <policy domain="resource" name="width" value="32KP"/> <policy domain="resource" name="height" value="32KP"/> <policy domain="resource" name="area" value="1GP"/> <policy domain="resource" name="disk" value="8GiB"/>

This ensures there are enough resources available to generate the final image.

Setup file: tile.py

Configuration

Keypath	Value
['tool', 'montage', 'task', 'tile', 'option']	• inputs/<design>_X0_Y0.png • inputs/<design>_X1_Y0.png • inputs/<design>_X0_Y1.png • inputs/<design>_X1_Y1.png • -tile • 2x2 • -geometry • +0+0 • outputs/<design>.png
['tool', 'montage', 'task', 'tile', 'var', 'xbins']	2
['tool', 'montage', 'task', 'tile', 'var', 'ybins']	2
['tool', 'montage', 'task', 'tile', 'input']	• <design>_X0_Y0.png • <design>_X0_Y1.png • <design>_X1_Y0.png • <design>_X1_Y1.png
['tool', 'montage', 'task', 'tile', 'output']	<design>.png

Variables

Parameters	Help
[..., 'var', 'xbins']	Number of bins along the x-axis
[..., 'var', 'ybins']	Number of bins along the y-axis

3.3.10 openfpgaloader

The OpenFPGALoader is a universal utility for programming FPGAs. Compatible with many boards, cables and FPGA from major manufacturers (Xilinx, Altera/Intel, Lattice, Gowin, Efinix, Anlogic). openFPGALoader works on Linux, Windows and macOS.

Documentation: https://github.com/trabucayre/openFPGALoader

Sources: https://github.com/trabucayre/openFPGALoader

Installation: https://github.com/trabucayre/openFPGALoader

Status: SC integration WIP

Setup file: openfpgaloader.py

Keypath	Value
['tool', 'openfpgaloader', 'exe']	`openfpgaloader`
['tool', 'openfpgaloader', 'vswitch']	`--Version`
['tool', 'openfpgaloader', 'version']	`0.5.0`

3.3.11 openroad

OpenROAD is an automated physical design platform for integrated circuit design with a complete set of features needed to translate a synthesized netlist to a tapeout ready GDSII.

Documentation: https://openroad.readthedocs.io/

Sources: https://github.com/The-OpenROAD-Project/OpenROAD

Installation: https://github.com/The-OpenROAD-Project/OpenROAD

Setup file: openroad.py

Data sources

Package	Specifications
siliconcompiler	• Path: `python://siliconcompiler`

continues on next page

Table 74 – continued from previous page

lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

Keypath	Value
['tool', 'openroad', 'exe']	openroad
['tool', 'openroad', 'vswitch']	-version
['tool', 'openroad', 'version']	>=v2.0-13145
['tool', 'openroad', 'format']	tcl

cts

Perform clock tree synthesis and timing repair

Setup file: cts.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'cts', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'cts', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'cts', 'option']	-exit -metrics reports/metrics.json
['tool', 'openroad', 'task', 'cts', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'cts', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'cts', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'cts', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'cts', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'cts', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'cts', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'cts', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'cts', 'var', 'power_corner']	typical

continues on next page

Table 76 – continued from previous page

['tool', 'openroad', 'task', 'cts', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'cts', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'cts', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'cts', 'var', 'macro_place_channel']	• 12 • 12
['tool', 'openroad', 'task', 'cts', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'cts', 'var', 'pdn_enable']	true
['tool', 'openroad', 'task', 'cts', 'var', 'psm_enable']	true
['tool', 'openroad', 'task', 'cts', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'cts', 'var', 'pad_global_place']	2
['tool', 'openroad', 'task', 'cts', 'var', 'gpl_routability_driven']	true
['tool', 'openroad', 'task', 'cts', 'var', 'gpl_timing_driven']	true
['tool', 'openroad', 'task', 'cts', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'cts', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'cts', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'cts', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'cts', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'cts', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'cts', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'cts', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'cts', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'cts', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'cts', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'cts', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'cts', 'var', 'cts_obstruction_aware']	true

continues on next page

Table 76 – continued from previous page

['tool', 'openroad', 'task', 'cts', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'cts', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'cts', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'cts', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'cts', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'cts', 'var', 'rsz_buffer_outputs']	false
['tool', 'openroad', 'task', 'cts', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'cts', 'var', 'rsz_skip_gate_cloning']	true
['tool', 'openroad', 'task', 'cts', 'var', 'rsz_repair_tns']	100
['tool', 'openroad', 'task', 'cts', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'cts', 'var', 'grt_overflow_iter']	100
['tool', 'openroad', 'task', 'cts', 'var', 'grt_macro_extension']	0
['tool', 'openroad', 'task', 'cts', 'var', 'grt_allow_congestion']	false
['tool', 'openroad', 'task', 'cts', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'cts', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'cts', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'cts', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'cts', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'cts', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'cts', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'cts', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'cts', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'cts', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'cts', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'cts', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'cts', 'var', 'pex_corners']	typical

continues on next page

Table 76 – continued from previous page

['tool', 'openroad', 'task', 'cts', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'cts', 'input']	<design>.def
['tool', 'openroad', 'task', 'cts', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb

continues on next page

Table 76 – continued from previous page

['tool', 'openroad', 'task', 'cts', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,cts,var,ord_abstract_lef_bloat_factor • tool,openroad,task,cts,var,ord_abstract_lef_bloat_layers • tool,openroad,task,cts,var,ord_enable_images • tool,openroad,task,cts,var,ord_heatmap_bins_x • tool,openroad,task,cts,var,ord_heatmap_bins_y • tool,openroad,task,cts,var,sta_early_timing_derate • tool,openroad,task,cts,var,sta_late_timing_derate • tool,openroad,task,cts,var,sta_top_n_paths • tool,openroad,task,cts,var,ifp_tie_separation • tool,openroad,task,cts,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,cts,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,cts,var,macro_place_channel • tool,openroad,task,cts,var,rtlmp_enable • tool,openroad,task,cts,var,pdn_enable • tool,openroad,task,cts,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density • tool,openroad,task,cts,var,place_density

Table 76 – continued from previous page

['tool', 'openroad', 'task', 'cts', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'cts', 'script']	sc_apr.tcl
['tool', 'openroad', 'task', 'cts', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on
[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites
[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design
[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design

continues on next page

Table 77 – continued from previous page

[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs
[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion
[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow
[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals

continues on next page

Table 77 – continued from previous page

[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to generate a padring using ICeWall in OpenROAD
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

dfm

Design for manufacturing step will insert fill if specified

Setup file: dfm.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'dfm', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'dfm', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'dfm', 'option']	-exit -metrics reports/metrics.json
['tool', 'openroad', 'task', 'dfm', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'dfm', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'dfm', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'dfm', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'dfm', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'dfm', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'dfm', 'var', 'power_corner']	typical
['tool', 'openroad', 'task', 'dfm', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'dfm', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'dfm', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'dfm', 'var', 'macro_place_channel']	• 12 • 12
['tool', 'openroad', 'task', 'dfm', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'dfm', 'var', 'pdn_enable']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'psm_enable']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'dfm', 'var', 'pad_global_place']	2
['tool', 'openroad', 'task', 'dfm', 'var', 'gpl_routability_driven']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'gpl_timing_driven']	true

continues on next page

Table 79 – continued from previous page

['tool', 'openroad', 'task', 'dfm', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'dfm', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'dfm', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'dfm', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'dfm', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'dfm', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'dfm', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'dfm', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'dfm', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'dfm', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'cts_obstruction_aware']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'dfm', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'dfm', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'dfm', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'dfm', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'dfm', 'var', 'rsz_buffer_outputs']	false
['tool', 'openroad', 'task', 'dfm', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'rsz_skip_gate_cloning']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'rsz_repair_tns']	100
['tool', 'openroad', 'task', 'dfm', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'dfm', 'var', 'grt_overflow_iter']	100
['tool', 'openroad', 'task', 'dfm', 'var', 'grt_macro_extension']	0
['tool', 'openroad', 'task', 'dfm', 'var', 'grt_allow_congestion']	false

continues on next page

Table 79 – continued from previous page

['tool', 'openroad', 'task', 'dfm', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'dfm', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'dfm', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'dfm', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'dfm', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'dfm', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'dfm', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'dfm', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'dfm', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'dfm', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'dfm', 'var', 'pex_corners']	typical
['tool', 'openroad', 'task', 'dfm', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'dfm', 'input']	<design>.def
['tool', 'openroad', 'task', 'dfm', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb

continues on next page

Table 79 – continued from previous page

['tool', 'openroad', 'task', 'dfm', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,dfm,var,ord_abstract_lef_bloat_factor • tool,openroad,task,dfm,var,ord_abstract_lef_bloat_layers • tool,openroad,task,dfm,var,ord_enable_images • tool,openroad,task,dfm,var,ord_heatmap_bins_x • tool,openroad,task,dfm,var,ord_heatmap_bins_y • tool,openroad,task,dfm,var,sta_early_timing_derate • tool,openroad,task,dfm,var,sta_late_timing_derate • tool,openroad,task,dfm,var,sta_top_n_paths • tool,openroad,task,dfm,var,ifp_tie_separation • tool,openroad,task,dfm,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,dfm,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,dfm,var,macro_place_channel • tool,openroad,task,dfm,var,rtlmp_enable • tool,openroad,task,dfm,var,pdn_enable • tool,openroad,task,dfm,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density • tool,openroad,task,dfm,var,place_density

Table 79 – continued from previous page

['tool', 'openroad', 'task', 'dfm', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'dfm', 'script']	sc_apr.tcl
['tool', 'openroad', 'task', 'dfm', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on
[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites
[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design
[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design

continues on next page

Table 80 – continued from previous page

[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs
[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion
[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow
[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals

continues on next page

Table 80 – continued from previous page

[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to generate a padring using ICeWall in OpenROAD
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

export

Generate abstract views (LEF), timing libraries (liberty files), circuit descriptions (CDL), and parasitic annotation files (SPEF)

Setup file: export.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'export', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'export', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'export', 'option']	-exit -metrics reports/metrics.json
['tool', 'openroad', 'task', 'export', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'export', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'export', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'export', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'export', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'export', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'export', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'export', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'export', 'var', 'power_corner']	typical
['tool', 'openroad', 'task', 'export', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'export', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'export', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'export', 'var', 'macro_place_channel']	• 12 • 12
['tool', 'openroad', 'task', 'export', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'export', 'var', 'pdn_enable']	true
['tool', 'openroad', 'task', 'export', 'var', 'psm_enable']	true
['tool', 'openroad', 'task', 'export', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'export', 'var', 'pad_global_place']	2
['tool', 'openroad', 'task', 'export', 'var', 'gpl_routability_driven']	true
['tool', 'openroad', 'task', 'export', 'var', 'gpl_timing_driven']	true

continues on next page

Table 82 – continued from previous page

['tool', 'openroad', 'task', 'export', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'export', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'export', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'export', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'export', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'export', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'export', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'export', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'export', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'export', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'export', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'export', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'export', 'var', 'cts_obstruction_aware']	true
['tool', 'openroad', 'task', 'export', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'export', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'export', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'export', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'export', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'export', 'var', 'rsz_buffer_outputs']	false
['tool', 'openroad', 'task', 'export', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'export', 'var', 'rsz_skip_gate_cloning']	true
['tool', 'openroad', 'task', 'export', 'var', 'rsz_repair_tns']	100
['tool', 'openroad', 'task', 'export', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'export', 'var', 'grt_overflow_iter']	100
['tool', 'openroad', 'task', 'export', 'var', 'grt_macro_extension']	0
['tool', 'openroad', 'task', 'export', 'var', 'grt_allow_congestion']	false

continues on next page

Table 82 – continued from previous page

['tool', 'openroad', 'task', 'export', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'export', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'export', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'export', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'export', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'export', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'export', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'export', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'export', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'export', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'export', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'export', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'export', 'var', 'pex_corners']	typical
['tool', 'openroad', 'task', 'export', 'var', 'write_cdl']	false
['tool', 'openroad', 'task', 'export', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'export', 'input']	<design>.def
['tool', 'openroad', 'task', 'export', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb • <design>.lef • <design>.typical.spef

continues on next page

Table 82 – continued from previous page

['tool', 'openroad', 'task', 'export', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,export,var,ord_abstract_lef_bloat_factor • tool,openroad,task,export,var,ord_abstract_lef_bloat_layers • tool,openroad,task,export,var,ord_enable_images • tool,openroad,task,export,var,ord_heatmap_bins_x • tool,openroad,task,export,var,ord_heatmap_bins_y • tool,openroad,task,export,var,sta_early_timing_derate • tool,openroad,task,export,var,sta_late_timing_derate • tool,openroad,task,export,var,sta_top_n_paths • tool,openroad,task,export,var,ifp_tie_separation • tool,openroad,task,export,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,export,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,export,var,macro_place_channel • tool,openroad,task,export,var,rtlmp_enable • tool,openroad,task,export,var,pdn_enable • tool,openroad,task,export,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density

Table 82 – continued from previous page

['tool', 'openroad', 'task', 'export', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'export', 'script']	sc_apr.tcl
['tool', 'openroad', 'task', 'export', 'threads']	1

Variables

Parameters	Help
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on
[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites
[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design

continues on next page

Table 83 – continued from previous page

[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design
[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs
[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion
[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow

continues on next page

Table 83 – continued from previous page

[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals
[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use
[..., 'var', 'write_cdl']	true/false, when true enables writing the CDL file for the design

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to generate a padring using ICeWall in OpenROAD
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

floorplan

Perform floorplanning, pin placements, macro placements and power grid generation

Setup file: floorplan.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'floorplan', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'floorplan', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'floorplan', 'option']	-exit -metrics reports/metrics.json
['tool', 'openroad', 'task', 'floorplan', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'floorplan', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'floorplan', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'floorplan', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'floorplan', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'floorplan', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'floorplan', 'var', 'power_corner']	typical
['tool', 'openroad', 'task', 'floorplan', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'floorplan', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'floorplan', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'floorplan', 'var', 'macro_place_channel']	• 12 • 12
['tool', 'openroad', 'task', 'floorplan', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'floorplan', 'var', 'pdn_enable']	true

continues on next page

Table 85 – continued from previous page

['tool', 'openroad', 'task', 'floorplan', 'var', 'psm_enable']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'floorplan', 'var', 'pad_global_place']	2
['tool', 'openroad', 'task', 'floorplan', 'var', 'gpl_routability_driven']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'gpl_timing_driven']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'floorplan', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'floorplan', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'floorplan', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'floorplan', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'floorplan', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'floorplan', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'floorplan', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'floorplan', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'floorplan', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'cts_obstruction_aware']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'floorplan', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'floorplan', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'floorplan', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'floorplan', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'floorplan', 'var', 'rsz_buffer_outputs']	false
['tool', 'openroad', 'task', 'floorplan', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'rsz_skip_gate_cloning']	true

continues on next page

Table 85 – continued from previous page

['tool', 'openroad', 'task', 'floorplan', 'var', 'rsz_repair_tns']	100
['tool', 'openroad', 'task', 'floorplan', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'floorplan', 'var', 'grt_overflow_iter']	100
['tool', 'openroad', 'task', 'floorplan', 'var', 'grt_macro_extension']	0
['tool', 'openroad', 'task', 'floorplan', 'var', 'grt_allow_congestion']	false
['tool', 'openroad', 'task', 'floorplan', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'floorplan', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'floorplan', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'floorplan', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'floorplan', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'floorplan', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'floorplan', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'floorplan', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'floorplan', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'floorplan', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'floorplan', 'var', 'pex_corners']	typical
['tool', 'openroad', 'task', 'floorplan', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'floorplan', 'input']	<design>.vg
['tool', 'openroad', 'task', 'floorplan', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb

continues on next page

Table 85 – continued from previous page

['tool', 'openroad', 'task', 'floorplan', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,floorplan,var,ord_abstract_lef_bloat_factor • tool,openroad,task,floorplan,var,ord_abstract_lef_bloat_layers • tool,openroad,task,floorplan,var,ord_enable_images • tool,openroad,task,floorplan,var,ord_heatmap_bins_x • tool,openroad,task,floorplan,var,ord_heatmap_bins_y • tool,openroad,task,floorplan,var,sta_early_timing_derate • tool,openroad,task,floorplan,var,sta_late_timing_derate • tool,openroad,task,floorplan,var,sta_top_n_paths • tool,openroad,task,floorplan,var,ifp_tie_separation • tool,openroad,task,floorplan,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,floorplan,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,floorplan,var,macro_place_channel • tool,openroad,task,floorplan,var,rtlmp_enable • tool,openroad,task,floorplan,var,pdn_enable • tool,openroad,task,floorplan,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density

Table 85 – continued from previous page

['tool', 'openroad', 'task', 'floorplan', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'floorplan', 'script']	sc_apr.tcl
['tool', 'openroad', 'task', 'floorplan', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on
[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites

continues on next page

Table 86 – continued from previous page

[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design
[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design
[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs
[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion

continues on next page

Table 86 – continued from previous page

[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow
[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals
[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to insert the padring
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

physyn

Not implemented yet

Setup file: physyn.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'physyn', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'physyn', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'physyn', 'option']	-exit -metrics reports/metrics.json
['tool', 'openroad', 'task', 'physyn', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'physyn', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'physyn', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'physyn', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'physyn', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'physyn', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'physyn', 'var', 'power_corner']	typical
['tool', 'openroad', 'task', 'physyn', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'physyn', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'physyn', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'physyn', 'var', 'macro_place_channel']	• 12 • 12
['tool', 'openroad', 'task', 'physyn', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'physyn', 'var', 'pdn_enable']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'psm_enable']	true

continues on next page

Table 88 – continued from previous page

['tool', 'openroad', 'task', 'physyn', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'physyn', 'var', 'pad_global_place']	2
['tool', 'openroad', 'task', 'physyn', 'var', 'gpl_routability_driven']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'gpl_timing_driven']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'physyn', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'physyn', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'physyn', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'physyn', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'physyn', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'physyn', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'physyn', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'physyn', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'physyn', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'cts_obstruction_aware']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'physyn', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'physyn', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'physyn', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'physyn', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'physyn', 'var', 'rsz_buffer_outputs']	false
['tool', 'openroad', 'task', 'physyn', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'rsz_skip_gate_cloning']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'rsz_repair_tns']	100

continues on next page

Table 88 – continued from previous page

['tool', 'openroad', 'task', 'physyn', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'physyn', 'var', 'grt_overflow_iter']	100
['tool', 'openroad', 'task', 'physyn', 'var', 'grt_macro_extension']	0
['tool', 'openroad', 'task', 'physyn', 'var', 'grt_allow_congestion']	false
['tool', 'openroad', 'task', 'physyn', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'physyn', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'physyn', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'physyn', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'physyn', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'physyn', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'physyn', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'physyn', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'physyn', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'physyn', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'physyn', 'var', 'pex_corners']	typical
['tool', 'openroad', 'task', 'physyn', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'physyn', 'input']	<design>.def
['tool', 'openroad', 'task', 'physyn', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb

continues on next page

Table 88 – continued from previous page

['tool', 'openroad', 'task', 'physyn', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,physyn,var,ord_abstract_lef_bloat_factor • tool,openroad,task,physyn,var,ord_abstract_lef_bloat_layers • tool,openroad,task,physyn,var,ord_enable_images • tool,openroad,task,physyn,var,ord_heatmap_bins_x • tool,openroad,task,physyn,var,ord_heatmap_bins_y • tool,openroad,task,physyn,var,sta_early_timing_derate • tool,openroad,task,physyn,var,sta_late_timing_derate • tool,openroad,task,physyn,var,sta_top_n_paths • tool,openroad,task,physyn,var,ifp_tie_separation • tool,openroad,task,physyn,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,physyn,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,physyn,var,macro_place_channel • tool,openroad,task,physyn,var,rtlmp_enable • tool,openroad,task,physyn,var,pdn_enable • tool,openroad,task,physyn,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density

Table 88 – continued from previous page

['tool', 'openroad', 'task', 'physyn', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'physyn', 'script']	sc_apr.tcl
['tool', 'openroad', 'task', 'physyn', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on
[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites
[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design

continues on next page

Table 89 – continued from previous page

[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design
[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs
[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion
[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow

continues on next page

Table 89 – continued from previous page

[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals
[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to generate a padring using ICeWall in OpenROAD
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

place

Perform global and detail placements along with design violation repairs

Setup file: place.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'place', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'place', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'place', 'option']	-exit -metrics reports/metrics.json
['tool', 'openroad', 'task', 'place', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'place', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'place', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'place', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'place', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'place', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'place', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'place', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'place', 'var', 'power_corner']	typical
['tool', 'openroad', 'task', 'place', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'place', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'place', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'place', 'var', 'macro_place_channel']	• 12 • 12
['tool', 'openroad', 'task', 'place', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'place', 'var', 'pdn_enable']	true
['tool', 'openroad', 'task', 'place', 'var', 'psm_enable']	true
['tool', 'openroad', 'task', 'place', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'place', 'var', 'pad_global_place']	2
['tool', 'openroad', 'task', 'place', 'var', 'gpl_routability_driven']	true
['tool', 'openroad', 'task', 'place', 'var', 'gpl_timing_driven']	true

continues on next page

Table 91 – continued from previous page

['tool', 'openroad', 'task', 'place', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'place', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'place', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'place', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'place', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'place', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'place', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'place', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'place', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'place', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'place', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'place', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'place', 'var', 'cts_obstruction_aware']	true
['tool', 'openroad', 'task', 'place', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'place', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'place', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'place', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'place', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'place', 'var', 'rsz_buffer_outputs']	false
['tool', 'openroad', 'task', 'place', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'place', 'var', 'rsz_skip_gate_cloning']	true
['tool', 'openroad', 'task', 'place', 'var', 'rsz_repair_tns']	100
['tool', 'openroad', 'task', 'place', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'place', 'var', 'grt_overflow_iter']	100
['tool', 'openroad', 'task', 'place', 'var', 'grt_macro_extension']	0
['tool', 'openroad', 'task', 'place', 'var', 'grt_allow_congestion']	false

continues on next page

Table 91 – continued from previous page

['tool', 'openroad', 'task', 'place', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'place', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'place', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'place', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'place', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'place', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'place', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'place', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'place', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'place', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'place', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'place', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'place', 'var', 'pex_corners']	typical
['tool', 'openroad', 'task', 'place', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'place', 'input']	<design>.def
['tool', 'openroad', 'task', 'place', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb

continues on next page

Table 91 – continued from previous page

['tool', 'openroad', 'task', 'place', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,place,var,ord_abstract_lef_bloat_factor • tool,openroad,task,place,var,ord_abstract_lef_bloat_layers • tool,openroad,task,place,var,ord_enable_images • tool,openroad,task,place,var,ord_heatmap_bins_x • tool,openroad,task,place,var,ord_heatmap_bins_y • tool,openroad,task,place,var,sta_early_timing_derate • tool,openroad,task,place,var,sta_late_timing_derate • tool,openroad,task,place,var,sta_top_n_paths • tool,openroad,task,place,var,ifp_tie_separation • tool,openroad,task,place,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,place,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,place,var,macro_place_channel • tool,openroad,task,place,var,rtlmp_enable • tool,openroad,task,place,var,pdn_enable • tool,openroad,task,place,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density

Table 91 – continued from previous page

['tool', 'openroad', 'task', 'place', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'place', 'script']	sc_apr.tcl
['tool', 'openroad', 'task', 'place', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on
[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites
[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design
[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design

continues on next page

Table 92 – continued from previous page

[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs
[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion
[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow
[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals

continues on next page

Table 92 – continued from previous page

[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to generate a padring using ICeWall in OpenROAD
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

rcx_bench

Helper method for configs specific to extraction tasks.

Setup file: rcx_bench.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'rcx_bench', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'rcx_bench', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'rcx_bench', 'option']	-exit -metrics reports/metrics.json
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'power_corner']	typical
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'macro_place_channel']	• 12 • 12
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'pdn_enable']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'psm_enable']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'pad_global_place']	2
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'gpl_routability_driven']	true

continues on next page

Table 94 – continued from previous page

['tool', 'openroad', 'task', 'rcx_bench', 'var', 'gpl_timing_driven']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'cts_obstruction_aware']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'rsz_buffer_outputs']	false
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'rsz_skip_gate_cloning']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'rsz_repair_tns']	100
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'grt_overflow_iter']	100
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'grt_macro_extension']	0

continues on next page

Table 94 – continued from previous page

['tool', 'openroad', 'task', 'rcx_bench', 'var', 'grt_allow_congestion']	false
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'pex_corners']	typical
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'libtype']	7p5t
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'max_layer']	M7
['tool', 'openroad', 'task', 'rcx_bench', 'var', 'bench_length']	100
['tool', 'openroad', 'task', 'rcx_bench', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'rcx_bench', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb • <design>.def • <design>.vg

continues on next page

Table 94 – continued from previous page

['tool', 'openroad', 'task', 'rcx_bench', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,rcx_bench,var,ord_abstract_lef_bloat_factor • tool,openroad,task,rcx_bench,var,ord_abstract_lef_bloat_layers • tool,openroad,task,rcx_bench,var,ord_enable_images • tool,openroad,task,rcx_bench,var,ord_heatmap_bins_x • tool,openroad,task,rcx_bench,var,ord_heatmap_bins_y • tool,openroad,task,rcx_bench,var,sta_early_timing_derate • tool,openroad,task,rcx_bench,var,sta_late_timing_derate • tool,openroad,task,rcx_bench,var,sta_top_n_paths • tool,openroad,task,rcx_bench,var,ifp_tie_separation • tool,openroad,task,rcx_bench,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,rcx_bench,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,rcx_bench,var,macro_place_channel • tool,openroad,task,rcx_bench,var,rtlmp_enable • tool,openroad,task,rcx_bench,var,pdn_enable • tool,openroad,task,rcx_bench,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density

Table 94 – continued from previous page

['tool', 'openroad', 'task', 'rcx_bench', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'rcx_bench', 'script']	sc_rcx.tcl
['tool', 'openroad', 'task', 'rcx_bench', 'threads']	1

Variables

Parameters	Help
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on
[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites

continues on next page

Table 95 – continued from previous page

[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design
[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design
[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs
[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion

continues on next page

Table 95 – continued from previous page

[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow
[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals
[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use
[..., 'var', 'libtype']	Library type used to select the lef file
[..., 'var', 'max_layer']	Maximum layer to generate extraction bench for
[..., 'var', 'bench_length']	Length of bench wires

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to generate a padring using ICeWall in OpenROAD
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

rcx_extract

Helper method for configs specific to extraction tasks.

Setup file: rcx_extract.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'rcx_extract', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'rcx_extract', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'rcx_extract', 'option']	-exit -metrics reports/metrics.json
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'power_corner']	typical
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'macro_place_channel']	• 12 • 12
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'pdn_enable']	true

continues on next page

Table 97 – continued from previous page

['tool', 'openroad', 'task', 'rcx_extract', 'var', 'psm_enable']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'pad_global_place']	2
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'gpl_routability_driven']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'gpl_timing_driven']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'cts_obstruction_aware']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'rsz_buffer_outputs']	false
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'rsz_skip_gate_cloning']	true

continues on next page

Table 97 – continued from previous page

['tool', 'openroad', 'task', 'rcx_extract', 'var', 'rsz_repair_tns']	100
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'grt_overflow_iter']	100
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'grt_macro_extension']	0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'grt_allow_congestion']	false
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'pex_corners']	typical
['tool', 'openroad', 'task', 'rcx_extract', 'var', 'libtype']	7p5t
['tool', 'openroad', 'task', 'rcx_extract', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'rcx_extract', 'input']	• <design>.def • <design>.corner.spef

continues on next page

Table 97 – continued from previous page

['tool', 'openroad', 'task', 'rcx_extract', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb • <design>.corner.rcx

continues on next page

Table 97 – continued from previous page

['tool', 'openroad', 'task', 'rcx_extract', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,rcx_extract,var,ord_abstract_lef_bloat_factor • tool,openroad,task,rcx_extract,var,ord_abstract_lef_bloat_layers • tool,openroad,task,rcx_extract,var,ord_enable_images • tool,openroad,task,rcx_extract,var,ord_heatmap_bins_x • tool,openroad,task,rcx_extract,var,ord_heatmap_bins_y • tool,openroad,task,rcx_extract,var,sta_early_timing_derate • tool,openroad,task,rcx_extract,var,sta_late_timing_derate • tool,openroad,task,rcx_extract,var,sta_top_n_paths • tool,openroad,task,rcx_extract,var,ifp_tie_separation • tool,openroad,task,rcx_extract,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,rcx_extract,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,rcx_extract,var,macro_place_channel • tool,openroad,task,rcx_extract,var,rtlmp_enable • tool,openroad,task,rcx_extract,var,pdn_enable • tool,openroad,task,rcx_extract,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density

Table 97 – continued from previous page

['tool', 'openroad', 'task', 'rcx_extract', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'rcx_extract', 'script']	sc_rcx.tcl
['tool', 'openroad', 'task', 'rcx_extract', 'threads']	1

Variables

Parameters	Help
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on
[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites

continues on next page

Table 98 – continued from previous page

[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design
[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design
[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs
[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion

continues on next page

Table 98 – continued from previous page

[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow
[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals
[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use
[..., 'var', 'libtype']	Library type used to select the lef file
[..., 'var', 'corner']	Parasitic corner to generate RCX file for

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to generate a padring using ICeWall in OpenROAD
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

route

Performs filler insertion, global routing, antenna repair, and detailed routing

Setup file: route.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'route', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'route', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'route', 'option']	-exit -metrics reports/metrics.json
['tool', 'openroad', 'task', 'route', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'route', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'route', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'route', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'route', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'route', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'route', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'route', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'route', 'var', 'power_corner']	typical
['tool', 'openroad', 'task', 'route', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'route', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'route', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'route', 'var', 'macro_place_channel']	• 12 • 12
['tool', 'openroad', 'task', 'route', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'route', 'var', 'pdn_enable']	true
['tool', 'openroad', 'task', 'route', 'var', 'psm_enable']	true

continues on next page

Table 100 – continued from previous page

['tool', 'openroad', 'task', 'route', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'route', 'var', 'pad_global_place']	2
['tool', 'openroad', 'task', 'route', 'var', 'gpl_routability_driven']	true
['tool', 'openroad', 'task', 'route', 'var', 'gpl_timing_driven']	true
['tool', 'openroad', 'task', 'route', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'route', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'route', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'route', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'route', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'route', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'route', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'route', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'route', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'route', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'route', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'route', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'route', 'var', 'cts_obstruction_aware']	true
['tool', 'openroad', 'task', 'route', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'route', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'route', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'route', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'route', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'route', 'var', 'rsz_buffer_outputs']	false
['tool', 'openroad', 'task', 'route', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'route', 'var', 'rsz_skip_gate_cloning']	true
['tool', 'openroad', 'task', 'route', 'var', 'rsz_repair_tns']	100

continues on next page

Table 100 – continued from previous page

['tool', 'openroad', 'task', 'route', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'route', 'var', 'grt_overflow_iter']	100
['tool', 'openroad', 'task', 'route', 'var', 'grt_macro_extension']	0
['tool', 'openroad', 'task', 'route', 'var', 'grt_allow_congestion']	false
['tool', 'openroad', 'task', 'route', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'route', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'route', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'route', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'route', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'route', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'route', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'route', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'route', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'route', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'route', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'route', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'route', 'var', 'pex_corners']	typical
['tool', 'openroad', 'task', 'route', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'route', 'input']	<design>.def
['tool', 'openroad', 'task', 'route', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb

continues on next page

Table 100 – continued from previous page

['tool', 'openroad', 'task', 'route', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,route,var,ord_abstract_lef_bloat_factor • tool,openroad,task,route,var,ord_abstract_lef_bloat_layers • tool,openroad,task,route,var,ord_enable_images • tool,openroad,task,route,var,ord_heatmap_bins_x • tool,openroad,task,route,var,ord_heatmap_bins_y • tool,openroad,task,route,var,sta_early_timing_derate • tool,openroad,task,route,var,sta_late_timing_derate • tool,openroad,task,route,var,sta_top_n_paths • tool,openroad,task,route,var,ifp_tie_separation • tool,openroad,task,route,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,route,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,route,var,macro_place_channel • tool,openroad,task,route,var,rtlmp_enable • tool,openroad,task,route,var,pdn_enable • tool,openroad,task,route,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density

Table 100 – continued from previous page

['tool', 'openroad', 'task', 'route', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'route', 'script']	sc_apr.tcl
['tool', 'openroad', 'task', 'route', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on
[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites
[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design
[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design

continues on next page

Table 101 – continued from previous page

[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs
[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion
[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow
[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals

continues on next page

Table 101 – continued from previous page

[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to generate a padring using ICeWall in OpenROAD
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

screenshot

Generate a PNG file from a layout file

Setup file: screenshot.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'screenshot', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'screenshot', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'screenshot', 'option']	-exit -metrics reports/metrics.json -no_init -gui
['tool', 'openroad', 'task', 'screenshot', 'var', 'show_filepath']	<path>
['tool', 'openroad', 'task', 'screenshot', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'screenshot', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'screenshot', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'screenshot', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'screenshot', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'screenshot', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'screenshot', 'var', 'power_corner']	typical
['tool', 'openroad', 'task', 'screenshot', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'screenshot', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'screenshot', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'screenshot', 'var', 'macro_place_channel']	• 12 • 12
['tool', 'openroad', 'task', 'screenshot', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'screenshot', 'var', 'pdn_enable']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'psm_enable']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'screenshot', 'var', 'pad_global_place']	2

continues on next page

Table 103 – continued from previous page

['tool', 'openroad', 'task', 'screenshot', 'var', 'gpl_routability_driven']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'gpl_timing_driven']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'screenshot', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'screenshot', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'screenshot', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'screenshot', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'screenshot', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'screenshot', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'screenshot', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'screenshot', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'screenshot', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'cts_obstruction_aware']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'screenshot', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'screenshot', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'screenshot', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'screenshot', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'screenshot', 'var', 'rsz_buffer_outputs']	false
['tool', 'openroad', 'task', 'screenshot', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'rsz_skip_gate_cloning']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'rsz_repair_tns']	100
['tool', 'openroad', 'task', 'screenshot', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'screenshot', 'var', 'grt_overflow_iter']	100

continues on next page

Table 103 – continued from previous page

['tool', 'openroad', 'task', 'screenshot', 'var', 'grt_macro_extension']	0
['tool', 'openroad', 'task', 'screenshot', 'var', 'grt_allow_congestion']	false
['tool', 'openroad', 'task', 'screenshot', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'screenshot', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'screenshot', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'screenshot', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'screenshot', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'screenshot', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'screenshot', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'screenshot', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'screenshot', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'screenshot', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'pex_corners']	typical
['tool', 'openroad', 'task', 'screenshot', 'var', 'show_exit']	true
['tool', 'openroad', 'task', 'screenshot', 'var', 'show_vertical_resolution']	1024
['tool', 'openroad', 'task', 'screenshot', 'var', 'include_report_images']	false
['tool', 'openroad', 'task', 'screenshot', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'screenshot', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb • <design>.png

continues on next page

Table 103 – continued from previous page

['tool', 'openroad', 'task', 'screenshot', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,screenshot,var,ord_abstract_lef_bloat_factor • tool,openroad,task,screenshot,var,ord_abstract_lef_bloat_layers • tool,openroad,task,screenshot,var,ord_enable_images • tool,openroad,task,screenshot,var,ord_heatmap_bins_x • tool,openroad,task,screenshot,var,ord_heatmap_bins_y • tool,openroad,task,screenshot,var,sta_early_timing_derate • tool,openroad,task,screenshot,var,sta_late_timing_derate • tool,openroad,task,screenshot,var,sta_top_n_paths • tool,openroad,task,screenshot,var,ifp_tie_separation • tool,openroad,task,screenshot,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,screenshot,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,screenshot,var,macro_place_channel • tool,openroad,task,screenshot,var,rtlmp_enable • tool,openroad,task,screenshot,var,pdn_enable • tool,openroad,task,screenshot,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density

Table 103 – continued from previous page

['tool', 'openroad', 'task', 'screenshot', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'screenshot', 'script']	sc_apr.tcl
['tool', 'openroad', 'task', 'screenshot', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'show_filepath']	Task script variables specified as key value pairs. Variable names and value types must match the name and type of task and reference script consuming the variable.
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on

continues on next page

Table 104 – continued from previous page

[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites
[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design
[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design
[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs

continues on next page

Table 104 – continued from previous page

[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion
[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow
[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals
[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use
[..., 'var', 'show_exit']	Task script variables specified as key value pairs. Variable names and value types must match the name and type of task and reference script consuming the variable.
[..., 'var', 'show_vertical_resolution']	Task script variables specified as key value pairs. Variable names and value types must match the name and type of task and reference script consuming the variable.
[..., 'var', 'include_report_images']	true/false, include the images in reports/

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to generate a padring using ICeWall in OpenROAD
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

show

Show a design in openroad

Setup file: show.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'openroad', 'task', 'show', 'regex', 'warnings']	^\[WARNING|^Warning
['tool', 'openroad', 'task', 'show', 'regex', 'errors']	^\[ERROR
['tool', 'openroad', 'task', 'show', 'option']	-metrics reports/metrics.json -no_init -gui
['tool', 'openroad', 'task', 'show', 'var', 'show_filepath']	<path>
['tool', 'openroad', 'task', 'show', 'var', 'ord_abstract_lef_bloat_factor']	10
['tool', 'openroad', 'task', 'show', 'var', 'ord_abstract_lef_bloat_layers']	true
['tool', 'openroad', 'task', 'show', 'var', 'ord_enable_images']	true
['tool', 'openroad', 'task', 'show', 'var', 'ord_heatmap_bins_x']	16
['tool', 'openroad', 'task', 'show', 'var', 'ord_heatmap_bins_y']	16
['tool', 'openroad', 'task', 'show', 'var', 'sta_early_timing_derate']	0.0
['tool', 'openroad', 'task', 'show', 'var', 'sta_late_timing_derate']	0.0
['tool', 'openroad', 'task', 'show', 'var', 'sta_top_n_paths']	10
['tool', 'openroad', 'task', 'show', 'var', 'power_corner']	typical
['tool', 'openroad', 'task', 'show', 'var', 'ifp_tie_separation']	0
['tool', 'openroad', 'task', 'show', 'var', 'ifp_snap_strategy']	site
['tool', 'openroad', 'task', 'show', 'var', 'macro_place_halo']	• 10 • 10
['tool', 'openroad', 'task', 'show', 'var', 'macro_place_channel']	• 12 • 12

continues on next page

Table 106 – continued from previous page

['tool', 'openroad', 'task', 'show', 'var', 'rtlmp_enable']	false
['tool', 'openroad', 'task', 'show', 'var', 'pdn_enable']	true
['tool', 'openroad', 'task', 'show', 'var', 'psm_enable']	true
['tool', 'openroad', 'task', 'show', 'var', 'place_density']	0.60
['tool', 'openroad', 'task', 'show', 'var', 'pad_global_place']	2
['tool', 'openroad', 'task', 'show', 'var', 'gpl_routability_driven']	true
['tool', 'openroad', 'task', 'show', 'var', 'gpl_timing_driven']	true
['tool', 'openroad', 'task', 'show', 'var', 'gpl_uniform_placement_adjustment']	0.00
['tool', 'openroad', 'task', 'show', 'var', 'gpl_enable_skip_io']	true
['tool', 'openroad', 'task', 'show', 'var', 'pad_detail_place']	1
['tool', 'openroad', 'task', 'show', 'var', 'dpl_max_displacement']	0
['tool', 'openroad', 'task', 'show', 'var', 'dpl_disallow_one_site']	false
['tool', 'openroad', 'task', 'show', 'var', 'dpo_enable']	true
['tool', 'openroad', 'task', 'show', 'var', 'dpo_max_displacement']	0
['tool', 'openroad', 'task', 'show', 'var', 'cts_clock_buffer']	BUFx4_ASAP7_75t_R
['tool', 'openroad', 'task', 'show', 'var', 'cts_distance_between_buffers']	60
['tool', 'openroad', 'task', 'show', 'var', 'cts_cluster_diameter']	100
['tool', 'openroad', 'task', 'show', 'var', 'cts_cluster_size']	30
['tool', 'openroad', 'task', 'show', 'var', 'cts_balance_levels']	true
['tool', 'openroad', 'task', 'show', 'var', 'cts_obstruction_aware']	true
['tool', 'openroad', 'task', 'show', 'var', 'rsz_setup_slack_margin']	0.0
['tool', 'openroad', 'task', 'show', 'var', 'rsz_hold_slack_margin']	0.0
['tool', 'openroad', 'task', 'show', 'var', 'rsz_slew_margin']	0.0
['tool', 'openroad', 'task', 'show', 'var', 'rsz_cap_margin']	0.0
['tool', 'openroad', 'task', 'show', 'var', 'rsz_buffer_inputs']	false
['tool', 'openroad', 'task', 'show', 'var', 'rsz_buffer_outputs']	false

continues on next page

Table 106 – continued from previous page

['tool', 'openroad', 'task', 'show', 'var', 'rsz_skip_pin_swap']	true
['tool', 'openroad', 'task', 'show', 'var', 'rsz_skip_gate_cloning']	true
['tool', 'openroad', 'task', 'show', 'var', 'rsz_repair_tns']	100
['tool', 'openroad', 'task', 'show', 'var', 'grt_use_pin_access']	false
['tool', 'openroad', 'task', 'show', 'var', 'grt_overflow_iter']	100
['tool', 'openroad', 'task', 'show', 'var', 'grt_macro_extension']	0
['tool', 'openroad', 'task', 'show', 'var', 'grt_allow_congestion']	false
['tool', 'openroad', 'task', 'show', 'var', 'grt_allow_overflow']	false
['tool', 'openroad', 'task', 'show', 'var', 'grt_signal_min_layer']	M2
['tool', 'openroad', 'task', 'show', 'var', 'grt_signal_max_layer']	M7
['tool', 'openroad', 'task', 'show', 'var', 'grt_clock_min_layer']	M2
['tool', 'openroad', 'task', 'show', 'var', 'grt_clock_max_layer']	M7
['tool', 'openroad', 'task', 'show', 'var', 'ant_iterations']	3
['tool', 'openroad', 'task', 'show', 'var', 'ant_margin']	0
['tool', 'openroad', 'task', 'show', 'var', 'ant_check']	true
['tool', 'openroad', 'task', 'show', 'var', 'ant_repair']	true
['tool', 'openroad', 'task', 'show', 'var', 'drt_disable_via_gen']	false
['tool', 'openroad', 'task', 'show', 'var', 'drt_via_repair_post_route']	false
['tool', 'openroad', 'task', 'show', 'var', 'fin_add_fill']	true
['tool', 'openroad', 'task', 'show', 'var', 'pex_corners']	typical
['tool', 'openroad', 'task', 'show', 'var', 'show_exit']	false
['tool', 'openroad', 'task', 'show', 'file', 'parasitics']	/home/docs/checkouts/readthedocs.org/user_builds/siliconcompiler/checkouts/latest/docs/build/<design>/job0/<step>/<index>/inputs/sc_parasitics.tcl
['tool', 'openroad', 'task', 'show', 'output']	• <design>.sdc • <design>.vg • <design>.def • <design>.odb

continues on next page

Table 106 – continued from previous page

['tool', 'openroad', 'task', 'show', 'require']	• asic,logiclib • option,stackup • library,asap7sc7p5t_rvt,asic,site,7p5t • pdk,asap7,aprtech,openroad,10M,7p5t,lef • library,asap7sc7p5t_rvt,output,slow,nldm • library,asap7sc7p5t_rvt,output,fast,nldm • library,asap7sc7p5t_rvt,output,typical,nldm • library,asap7sc7p5t_rvt,output,10M,lef • pdk,asap7,var,openroad,rclayer_signal,10M • pdk,asap7,var,openroad,rclayer_clock,10M • pdk,asap7,var,openroad,pin_layer_horizontal,10M • pdk,asap7,var,openroad,pin_layer_vertical,10M • tool,openroad,task,show,var,ord_abstract_lef_bloat_factor • tool,openroad,task,show,var,ord_abstract_lef_bloat_layers • tool,openroad,task,show,var,ord_enable_images • tool,openroad,task,show,var,ord_heatmap_bins_x • tool,openroad,task,show,var,ord_heatmap_bins_y • tool,openroad,task,show,var,sta_early_timing_derate • tool,openroad,task,show,var,sta_late_timing_derate • tool,openroad,task,show,var,sta_top_n_paths • tool,openroad,task,show,var,ifp_tie_separation • tool,openroad,task,show,var,ifp_snap_strategy • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_halo • tool,openroad,task,show,var,macro_place_halo • library,asap7sc7p5t_rvt,option,var,openroad_macro_place_channel • tool,openroad,task,show,var,macro_place_channel • tool,openroad,task,show,var,rtlmp_enable • tool,openroad,task,show,var,pdn_enable • tool,openroad,task,show,var,psm_enable • library,asap7sc7p5t_rvt,option,var,openroad_place_density

Table 106 – continued from previous page

['tool', 'openroad', 'task', 'show', 'refdir']	tools/openroad/scripts, siliconcompiler
['tool', 'openroad', 'task', 'show', 'script']	sc_apr.tcl
['tool', 'openroad', 'task', 'show', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'show_filepath']	Task script variables specified as key value pairs. Variable names and value types must match the name and type of task and reference script consuming the variable.
[..., 'var', 'debug_level']	list of "tool key level" to enable debugging of OpenROAD
[..., 'var', 'ord_abstract_lef_bloat_factor']	Factor to apply when writing the abstract lef
[..., 'var', 'ord_abstract_lef_bloat_layers']	true/false, fill all layers when writing the abstract lef
[..., 'var', 'ord_enable_images']	true/false, enable generating images of the design at the end of the task
[..., 'var', 'ord_heatmap_bins_x']	number of X bins to use for heatmap image generation
[..., 'var', 'ord_heatmap_bins_y']	number of Y bins to use for heatmap image generation
[..., 'var', 'sta_early_timing_derate']	timing derating factor to use for hold corners
[..., 'var', 'sta_late_timing_derate']	timing derating factor to use for setup corners
[..., 'var', 'sta_top_n_paths']	number of paths to report timing for
[..., 'var', 'power_corner']	corner to use for power analysis
[..., 'var', 'sdc_buffer']	buffer cell to use when auto generating timing constraints
[..., 'var', 'ifp_tie_separation']	maximum distance between tie high/low cells in microns
[..., 'var', 'ifp_snap_strategy']	Snapping strategy to use when placing macros. Allowed values: none, site, manufacturing_grid
[..., 'var', 'ppl_arguments']	additional arguments to pass along to the pin placer.
[..., 'var', 'macro_place_halo']	macro halo to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'macro_place_channel']	macro channel to use when performing automated macro placement ([x, y] in microns)
[..., 'var', 'rtlmp_enable']	true/false, enables the RTLMP macro placement
[..., 'var', 'rtlmp_min_instances']	minimum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_max_instances']	maximum number of instances to use while clustering for macro placement
[..., 'var', 'rtlmp_min_macros']	minimum number of macros to use while clustering for macro placement
[..., 'var', 'rtlmp_max_macros']	maximum number of macros to use while clustering for macro placement
[..., 'var', 'pdn_enable']	true/false, when true enables power grid generation
[..., 'var', 'psm_enable']	true/false, when true enables IR drop analysis
[..., 'var', 'psm_skip_nets']	list of nets to skip power grid analysis on
[..., 'var', 'place_density']	global placement density (0.0 - 1.0)
[..., 'var', 'pad_global_place']	global placement cell padding in number of sites

continues on next page

Table 107 – continued from previous page

[..., 'var', 'gpl_routability_driven']	true/false, when true global placement will consider the routability of the design
[..., 'var', 'gpl_timing_driven']	true/false, when true global placement will consider the timing performance of the design
[..., 'var', 'gpl_uniform_placement_adjustment']	percent of remaining area density to apply above uniform density (0.00 - 0.99)
[..., 'var', 'gpl_enable_skip_io']	true/false, when enabled a global placement is performed without considering the impact of the pin placements
[..., 'var', 'pad_detail_place']	detailed placement cell padding in number of sites
[..., 'var', 'dpl_max_displacement']	maximum cell movement in detailed placement in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'dpl_disallow_one_site']	true/false, disallow single site gaps in detail placement
[..., 'var', 'dpo_enable']	true/false, when true the detailed placement optimization will be performed
[..., 'var', 'dpo_max_displacement']	maximum cell movement in detailed placement optimization in microns, 0 will result in the tool default maximum displacement
[..., 'var', 'cts_clock_buffer']	buffer to use during clock tree synthesis
[..., 'var', 'cts_distance_between_buffers']	maximum distance between buffers during clock tree synthesis in microns
[..., 'var', 'cts_cluster_diameter']	clustering distance to use during clock tree synthesis in microns
[..., 'var', 'cts_cluster_size']	number of instances in a cluster to use during clock tree synthesis
[..., 'var', 'cts_balance_levels']	perform level balancing in clock tree synthesis
[..., 'var', 'cts_obstruction_aware']	make clock tree synthesis aware of obstructions
[..., 'var', 'rsz_setup_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_hold_slack_margin']	specifies the margin to apply when performing setup repair in library timing units
[..., 'var', 'rsz_slew_margin']	specifies the amount of margin to apply to max slew repairs in percent (0 - 100)
[..., 'var', 'rsz_cap_margin']	specifies the amount of margin to apply to max capacitance repairs in percent (0 - 100)
[..., 'var', 'rsz_buffer_inputs']	true/false, when true enables adding buffers to the input ports
[..., 'var', 'rsz_buffer_outputs']	true/false, when true enables adding buffers to the output ports
[..., 'var', 'rsz_skip_pin_swap']	true/false, skip pin swap optimization
[..., 'var', 'rsz_skip_gate_cloning']	true/false, skip gate cloning optimization
[..., 'var', 'rsz_repair_tns']	percentage of violating nets to attempt to repair (0 - 100)
[..., 'var', 'grt_use_pin_access']	true/false, when true perform pin access before global routing
[..., 'var', 'grt_overflow_iter']	maximum number of iterations to use in global routing when attempting to solve overflow
[..., 'var', 'grt_macro_extension']	macro extension distance in number of gcells, this can be useful when the detailed router needs additional space to avoid DRCs
[..., 'var', 'grt_allow_congestion']	true/false, when true allow global routing to finish with congestion

continues on next page

Table 107 – continued from previous page

[..., 'var', 'grt_allow_overflow']	true/false, when true allow global routing to finish with overflow
[..., 'var', 'grt_signal_min_layer']	minimum layer to use for global routing of signals
[..., 'var', 'grt_signal_max_layer']	maximum layer to use for global routing of signals
[..., 'var', 'grt_clock_min_layer']	minimum layer to use for global routing of clock nets
[..., 'var', 'grt_clock_max_layer']	maximum layer to use for global routing of clock nets
[..., 'var', 'ant_iterations']	maximum number of repair iterations to use during antenna repairs
[..., 'var', 'ant_margin']	adds a margin to the antenna ratios (0 - 100)
[..., 'var', 'ant_check']	true/false, flag to indicate whether to check for antenna violations
[..., 'var', 'ant_repair']	true/false, flag to indicate whether to repair antenna violations
[..., 'var', 'drt_disable_via_gen']	true/false, when true turns off via generation in detailed router and only uses the specified tech vias
[..., 'var', 'drt_process_node']	when set this specifies to the detailed router the specific process node
[..., 'var', 'drt_via_in_pin_bottom_layer']	TODO
[..., 'var', 'drt_via_in_pin_top_layer']	TODO
[..., 'var', 'drt_repair_pdn_vias']	TODO
[..., 'var', 'drt_via_repair_post_route']	true/false, when true performs a via ripup step after detailed routing to remove power vias that are causing DRC violations
[..., 'var', 'detailed_route_default_via']	list of default vias to use for detail routing
[..., 'var', 'detailed_route_unidirectional_layer']	list of layers to treat as unidirectional regardless of what the tech lef specifies
[..., 'var', 'fin_add_fill']	true/false, when true enables adding fill, if enabled by the PDK, to the design
[..., 'var', 'pex_corners']	list of parasitic extraction corners to use
[..., 'var', 'show_exit']	Task script variables specified as key value pairs. Variable names and value types must match the name and type of task and reference script consuming the variable.

Files

Parameters	Help
[..., 'file', 'global_connect']	list of files to use for specifying global connections
[..., 'file', 'ifp_tapcell']	tap cell insertion script
[..., 'file', 'padring']	script to generate a padring using ICeWall in OpenROAD
[..., 'file', 'ppl_constraints']	script constrain pin placement
[..., 'file', 'pdn_config']	list of files to use for power grid generation
[..., 'file', 'parasitics']	file used to specify the parasitics for estimation

3.3.12 surelog

Surelog is a SystemVerilog pre-processor, parser, elaborator, and UHDM compiler that provdes IEEE design and testbench C/C++ VPI and a Python AST API.

Documentation: https://github.com/chipsalliance/Surelog

Sources: https://github.com/chipsalliance/Surelog

Installation: https://github.com/chipsalliance/Surelog

Setup file: surelog.py

Keypath	Value
['tool', 'surelog', 'exe']	surelog
['tool', 'surelog', 'vswitch']	--version
['tool', 'surelog', 'version']	>=1.51

parse

Import verilog files

Setup file: parse.py

Configuration

Keypath	Value
['tool', 'surelog', 'task', 'parse', 'regex', 'warnings']	^\[WRN:
['tool', 'surelog', 'task', 'parse', 'regex', 'errors']	^\[(ERR|FTL|SNT):
['tool', 'surelog', 'task', 'parse', 'option']	• -nocache • -parse • -nouhdm
['tool', 'surelog', 'task', 'parse', 'var', 'enable_lowmem']	false
['tool', 'surelog', 'task', 'parse', 'var', 'disable_write_cache']	false
['tool', 'surelog', 'task', 'parse', 'var', 'disable_info']	false
['tool', 'surelog', 'task', 'parse', 'var', 'disable_note']	false
['tool', 'surelog', 'task', 'parse', 'output']	<design>.v
['tool', 'surelog', 'task', 'parse', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'enable_lowmem']	true/false, when true instructs Surelog to minimize its maximum memory usage.
[..., 'var', 'disable_write_cache']	true/false, when true instructs Surelog to not write to its cache.
[..., 'var', 'disable_info']	true/false, when true instructs Surelog to not log infos.
[..., 'var', 'disable_note']	true/false, when true instructs Surelog to not log notes.

3.3.13 sv2v

sv2v converts SystemVerilog (IEEE 1800-2017) to Verilog (IEEE 1364-2005), with an emphasis on supporting synthesizable language constructs. The primary goal of this project is to create a completely free and open-source tool for converting SystemVerilog to Verilog. While methods for performing this conversion already exist, they generally either rely on commercial tools, or are limited in scope.

Documentation: https://github.com/zachjs/sv2v

Sources: https://github.com/zachjs/sv2v

Installation: https://github.com/zachjs/sv2v

Setup file: sv2v.py

Keypath	Value
['tool', 'sv2v', 'exe']	sv2v
['tool', 'sv2v', 'vswitch']	--numeric-version
['tool', 'sv2v', 'version']	>=0.0.9

convert

Convert SystemVerilog to verilog

Setup file: convert.py

Configuration

Keypath	Value
['tool', 'sv2v', 'task', 'convert', 'option']	• inputs/<design>.v • --write=outputs/<design>.v
['tool', 'sv2v', 'task', 'convert', 'input']	<design>.v
['tool', 'sv2v', 'task', 'convert', 'output']	<design>.v
['tool', 'sv2v', 'task', 'convert', 'threads']	2

3.3.14 verilator

Verilator is a free and open-source software tool which converts Verilog (a hardware description language) to a cycle-accurate behavioral model in C++ or SystemC.

All Verilator tasks may consume input either from a single pickled Verilog file (`inputs/<design>.v`) generated by a preceding task, or if that file does not exist, through the following keypaths:

- *['input', 'rtl', 'verilog']*
- *['option', 'ydir']*
- *['option', 'vlib']*
- *['option', 'idir']*
- *['option', 'cmdfile']*

For all tasks, this driver runs Verilator using the `-sv` switch to enable parsing a subset of SystemVerilog features. All tasks also support using *['option', 'relax']* to make warnings nonfatal.

Documentation: https://verilator.org/guide/latest

Sources: https://github.com/verilator/verilator

Installation: https://verilator.org/guide/latest/install.html

Setup file: verilator.py

Data sources

Package	Specifications
lambdapdk	• Path: `https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/` • Reference: `v0.1.19`

Keypath	Value
['tool', 'verilator', 'exe']	`verilator`
['tool', 'verilator', 'vswitch']	`--version`
['tool', 'verilator', 'version']	`>=4.034`

compile

Compiles Verilog and C/C++ sources into an executable. In addition to the standard RTL inputs, this task reads C/C++ sources from *['input', 'hll', 'c']*. Outputs an executable in `outputs/<design>.vexe`.

This task supports using the *['option', 'trace']* parameter to enable Verilator's `--trace` flag.

Setup file: compile.py

Built using target: *freepdk45_demo*

Configuration

Keypath	Value
['tool', 'verilator', 'task', 'compile', 'regex', 'warnings']	^\%Warning
['tool', 'verilator', 'task', 'compile', 'regex', 'errors']	^\%Error
['tool', 'verilator', 'task', 'compile', 'var', 'enable_assert']	false
['tool', 'verilator', 'task', 'compile', 'var', 'mode']	cc
['tool', 'verilator', 'task', 'compile', 'var', 'trace_type']	vcd
['tool', 'verilator', 'task', 'compile', 'require']	input,hll,c
['tool', 'verilator', 'task', 'compile', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'enable_assert']	true/false, when true assertions are enabled in Verilator.
[..., 'var', 'mode']	defines compilation mode for Verilator. Valid options are 'cc' for C++, or 'systemc' for SystemC.
[..., 'var', 'trace_type']	specifies type of wave file to create when [option, trace] is set. Valid options are 'vcd' or 'fst'. Defaults to 'vcd'.
[..., 'var', 'cflags']	flags to provide to the C++ compiler invoked by Verilator
[..., 'var', 'ldflags']	flags to provide to the linker invoked by Verilator
[..., 'var', 'pins_bv']	controls datatypes used to represent SystemC inputs/outputs. See –pins-bv in Verilator docs for more info.

Files

Parameters	Help
[..., 'file', 'config']	Verilator configuration file

lint

Lints Verilog source. Results of linting can be programmatically queried using errors/warnings metrics.

Setup file: lint.py

Built using target: *freepdk45_demo*

Configuration

Keypath	Value
['tool', 'verilator', 'task', 'lint', 'regex', 'warnings']	^\%Warning
['tool', 'verilator', 'task', 'lint', 'regex', 'errors']	^\%Error
['tool', 'verilator', 'task', 'lint', 'var', 'enable_assert']	false
['tool', 'verilator', 'task', 'lint', 'threads']	2

Variables

Parameters	Help
[..., 'var', 'enable_assert']	true/false, when true assertions are enabled in Verilator.

Files

Parameters	Help
[..., 'file', 'config']	Verilator configuration file

3.3.15 vivado

Vivado is an FPGA programming tool suite from Xilinx used to program Xilinx devices.

Documentation: https://www.xilinx.com/products/design-tools/vivado.html

Setup file: vivado.py

Data sources

Package	Specifications

continues on next page

Table 122 – continued from previous page

siliconcompiler_data	• Path: git+https://github.com/siliconcompiler/siliconcompiler • Reference: v0.21.11

Keypath	Value
['tool', 'vivado', 'exe']	vivado
['tool', 'vivado', 'vswitch']	-version
['tool', 'vivado', 'vendor']	xilinx
['tool', 'vivado', 'version']	>=2021
['tool', 'vivado', 'format']	tcl

bitstream

Generates bitstream of implemented design.

Setup file: bitstream.py

Built using target: *fpgaflow_demo*

Configuration

Keypath	Value
['tool', 'vivado', 'task', 'bitstream', 'regex', 'errors']	^ERROR:
['tool', 'vivado', 'task', 'bitstream', 'regex', 'warnings']	^(CRITICAL)?WARNING:
['tool', 'vivado', 'task', 'bitstream', 'option']	• -nolog • -nojournal • -mode • batch • -source
['tool', 'vivado', 'task', 'bitstream', 'input']	<design>_checkpoint.dcp
['tool', 'vivado', 'task', 'bitstream', 'output']	<design>.bit
['tool', 'vivado', 'task', 'bitstream', 'refdir']	tools/vivado/scripts, siliconcompiler
['tool', 'vivado', 'task', 'bitstream', 'script']	sc_run.tcl
['tool', 'vivado', 'task', 'bitstream', 'threads']	2

place

Performs placement.

Setup file: place.py

Built using target: *fpgaflow_demo*

Configuration

Keypath	Value
['tool', 'vivado', 'task', 'place', 'regex', 'errors']	^ERROR:
['tool', 'vivado', 'task', 'place', 'regex', 'warnings']	^(CRITICAL)?WARNING:
['tool', 'vivado', 'task', 'place', 'option']	• -nolog • -nojournal • -mode • batch • -source
['tool', 'vivado', 'task', 'place', 'input']	<design>_checkpoint.dcp
['tool', 'vivado', 'task', 'place', 'output']	<design>_checkpoint.dcp
['tool', 'vivado', 'task', 'place', 'refdir']	tools/vivado/scripts, siliconcompiler
['tool', 'vivado', 'task', 'place', 'script']	sc_run.tcl
['tool', 'vivado', 'task', 'place', 'threads']	2

route

Performs routing.

Setup file: route.py

Built using target: *fpgaflow_demo*

Configuration

Keypath	Value
['tool', 'vivado', 'task', 'route', 'regex', 'errors']	^ERROR:
['tool', 'vivado', 'task', 'route', 'regex', 'warnings']	^(CRITICAL)?WARNING:

continues on next page

Table 126 – continued from previous page

['tool', 'vivado', 'task', 'route', 'option']	• -nolog • -nojournal • -mode • batch • -source
['tool', 'vivado', 'task', 'route', 'input']	<design>_checkpoint.dcp
['tool', 'vivado', 'task', 'route', 'output']	<design>_checkpoint.dcp
['tool', 'vivado', 'task', 'route', 'refdir']	tools/vivado/scripts, siliconcompiler
['tool', 'vivado', 'task', 'route', 'script']	sc_run.tcl
['tool', 'vivado', 'task', 'route', 'threads']	2

syn_fpga

Performs FPGA synthesis.

Setup file: syn_fpga.py

Built using target: *fpgaflow_demo*

Configuration

Keypath	Value
['tool', 'vivado', 'task', 'syn_fpga', 'regex', 'errors']	^ERROR:
['tool', 'vivado', 'task', 'syn_fpga', 'regex', 'warnings']	^(CRITICAL)?WARNING:
['tool', 'vivado', 'task', 'syn_fpga', 'option']	• -nolog • -nojournal • -mode • batch • -source
['tool', 'vivado', 'task', 'syn_fpga', 'input']	<design>.v
['tool', 'vivado', 'task', 'syn_fpga', 'output']	<design>_checkpoint.dcp
['tool', 'vivado', 'task', 'syn_fpga', 'refdir']	tools/vivado/scripts, siliconcompiler
['tool', 'vivado', 'task', 'syn_fpga', 'script']	sc_run.tcl
['tool', 'vivado', 'task', 'syn_fpga', 'threads']	2

3.3.16 vpr

VPR (Versatile Place and Route) is an open source CAD tool designed for the exploration of new FPGA architectures and CAD algorithms, at the packing, placement and routing phases of the CAD flow. VPR takes, as input, a description of an FPGA architecture along with a technology-mapped user circuit. It then performs packing, placement, and routing to map the circuit onto the FPGA. The output of VPR includes the FPGA configuration needed to implement the circuit and statistics about the final mapped design (eg. critical path delay, area, etc).

Documentation: https://docs.verilogtorouting.org/en/latest

Sources: https://github.com/verilog-to-routing/vtr-verilog-to-routing

Installation: https://github.com/verilog-to-routing/vtr-verilog-to-routing

Setup file: vpr.py

Data sources

Package	Specifications
siliconcompiler_data	• Path: `git+https://github.com/siliconcompiler/siliconcompiler` • Reference: `v0.21.11`

Keypath	Value
['tool', 'vpr', 'exe']	`vpr`
['tool', 'vpr', 'vswitch']	`--version`
['tool', 'vpr', 'version']	`>=8.1.0`

place

Perform automated place and route with VPR

Setup file: place.py

Built using target: *fpgaflow_demo*

route

Perform automated place and route with VPR

Setup file: route.py

Built using target: *fpgaflow_demo*

screenshot

Screenshot placed and/or routed designs

Setup file: screenshot.py

Built using target: *fpgaflow_demo*

show

Show placed and/or routed designs in VPR GUI

Setup file: show.py

Built using target: *fpgaflow_demo*

3.3.17 xyce

Xyce is a high performance SPICE-compatible circuit simulator capable capable of solving extremely large circuit problems by supporting large-scale parallel computing platforms. It also supports serial execution on all common desktop platforms, and small-scale parallel runs on Unix-like systems.

Documentation: https://xyce.sandia.gov/documentation-tutorials/

Sources: https://github.com/Xyce/Xyce

Installation: https://xyce.sandia.gov/documentation-tutorials/building-guide/

Status: SC integration WIP

Setup file: xyce.py

Keypath	Value
['tool', 'xyce', 'exe']	xyce
['tool', 'xyce', 'version']	0.0

3.3.18 yosys

Yosys is a framework for RTL synthesis that takes synthesizable Verilog-2005 design and converts it to BLIF, EDIF, BTOR, SMT, Verilog netlist etc. The tool supports logical synthesis and tech mapping to ASIC standard cell libraries, FPGA architectures. In addition it has built in formal methods for property and equivalence checking.

Documentation: https://yosyshq.readthedocs.io/projects/yosys/en/latest/

Sources: https://github.com/YosysHQ/yosys

Installation: https://github.com/YosysHQ/yosys

Setup file: yosys.py

Data sources

Package	Specifications
siliconcompiler	• Path: python://siliconcompiler
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

Keypath	Value
['tool', 'yosys', 'exe']	yosys
['tool', 'yosys', 'vswitch']	--version
['tool', 'yosys', 'version']	>=0.38+92
['tool', 'yosys', 'format']	tcl

lec

Perform logical equivalence checks

Setup file: lec.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'yosys', 'task', 'lec', 'regex', 'warnings']	Warning:
['tool', 'yosys', 'task', 'lec', 'regex', 'errors']	^ERROR
['tool', 'yosys', 'task', 'lec', 'option']	-c
['tool', 'yosys', 'task', 'lec', 'var', 'flatten']	true
['tool', 'yosys', 'task', 'lec', 'var', 'hier_iterations']	10
['tool', 'yosys', 'task', 'lec', 'var', 'hier_threshold']	1000
['tool', 'yosys', 'task', 'lec', 'var', 'autoname']	true
['tool', 'yosys', 'task', 'lec', 'var', 'add_buffers']	true
['tool', 'yosys', 'task', 'lec', 'var', 'map_adders']	true
['tool', 'yosys', 'task', 'lec', 'var', 'synthesis_corner']	slow

continues on next page

Table 133 – continued from previous page

['tool', 'yosys', 'task', 'lec', 'var', 'abc_constraint_driver']	BUFx2_ASAP7_75t_R
['tool', 'yosys', 'task', 'lec', 'var', 'abc_constraint_load']	2.026fF
['tool', 'yosys', 'task', 'lec', 'file', 'synth_extra_map']	tools/yosys/techmaps/lcu_kogge_stone.v, siliconcompiler
['tool', 'yosys', 'task', 'lec', 'input']	<design>.vg
['tool', 'yosys', 'task', 'lec', 'require']	• asic,logiclib • library,asap7sc7p5t_rvt,output,slow,nldm • tool,yosys,task,lec,var,flatten • tool,yosys,task,lec,var,hier_iterations • tool,yosys,task,lec,var,hier_threshold • tool,yosys,task,lec,var,autoname • tool,yosys,task,lec,var,add_buffers • library,asap7sc7p5t_rvt,option,var,yosys_buffer_input • library,asap7sc7p5t_rvt,option,var,yosys_buffer_output • tool,yosys,task,lec,var,map_adders • library,asap7sc7p5t_rvt,option,file,yosys_addermap • tool,yosys,task,lec,var,synthesis_corner • library,asap7sc7p5t_rvt,option,var,yosys_abc_clock_multiplier • library,asap7sc7p5t_rvt,option,var,yosys_abc_constraint_load • library,asap7sc7p5t_rvt,option,var,yosys_abc_constraint_load • tool,yosys,task,lec,var,abc_constraint_load
['tool', 'yosys', 'task', 'lec', 'refdir']	tools/yosys, siliconcompiler
['tool', 'yosys', 'task', 'lec', 'script']	sc_lec.tcl

Variables

Parameters	Help
[..., 'var', 'flatten']	true/false, invoke synth with the -flatten option
[..., 'var', 'hier_iterations']	Number of iterations to attempt to determine the hierarchy to flatten
[..., 'var', 'hier_threshold']	Instance limit for the number of cells in a module to preserve.
[..., 'var', 'autoname']	true/false, call autoname to rename wires based on registers

continues on next page

Table 134 – continued from previous page

[..., 'var', 'add_buffers']	true/false, flag to indicate whether to add buffers or not.
[..., 'var', 'map_adders']	true/false, techmap adders in Yosys
[..., 'var', 'synthesis_corner']	Timing corner to use for synthesis
[..., 'var', 'abc_constraint_driver']	Buffer that drives the abc techmapping, defaults to first buffer specified
[..., 'var', 'abc_constraint_load']	Capacitive load for the abc techmapping in fF, if not specified it will not be used
[..., 'var', 'preserve_modules']	List of modules in input files to prevent flatten from "flattening"
[..., 'var', 'blackbox_modules']	List of modules in input files to exclude from synthesis by replacing them with empty blackboxes"
[..., 'var', 'abc_clock_period']	Clock period to use for synthesis in ps, if more than one clock is specified, the smallest period is used.
[..., 'var', 'abc_clock_derating']	Used to derate the clock period to further constrain the clock, values between 0 and 1
[..., 'var', 'strategy']	ABC synthesis strategy. Allowed values are DELAY0-4, AREA0-3, or if the strategy starts with a + it is assumed to be actual commands for ABC.

Files

Parameters	Help
[..., 'file', 'dff_liberty']	Liberty file to use for flip-flop mapping, if not specified the first in the logiclib is used
[..., 'file', 'abc_constraint_file']	File used to pass in constraints to abc
[..., 'file', 'techmap']	File to use for techmapping in Yosys
[..., 'file', 'dff_liberty_file']	File to use for the DFF mapping stage of Yosys
[..., 'file', 'memory_libmap']	File used to map memories with yosys
[..., 'file', 'memory_techmap']	File used to techmap memories with yosys
[..., 'file', 'synth_extra_map']	Files used in synthesis to perform additional techmapping

syn_asic

Perform ASIC synthesis

Setup file: syn_asic.py

Built using target: *asap7_demo*

Configuration

Keypath	Value
['tool', 'yosys', 'task', 'syn_asic', 'regex', 'warnings']	Warning:
['tool', 'yosys', 'task', 'syn_asic', 'regex', 'errors']	^ERROR
['tool', 'yosys', 'task', 'syn_asic', 'option']	-c
['tool', 'yosys', 'task', 'syn_asic', 'var', 'flatten']	true
['tool', 'yosys', 'task', 'syn_asic', 'var', 'hier_iterations']	10
['tool', 'yosys', 'task', 'syn_asic', 'var', 'hier_threshold']	1000
['tool', 'yosys', 'task', 'syn_asic', 'var', 'autoname']	true
['tool', 'yosys', 'task', 'syn_asic', 'var', 'add_buffers']	true
['tool', 'yosys', 'task', 'syn_asic', 'var', 'map_adders']	true
['tool', 'yosys', 'task', 'syn_asic', 'var', 'synthesis_corner']	slow
['tool', 'yosys', 'task', 'syn_asic', 'var', 'abc_constraint_driver']	BUFx2_ASAP7_75t_R
['tool', 'yosys', 'task', 'syn_asic', 'var', 'abc_constraint_load']	2.026fF
['tool', 'yosys', 'task', 'syn_asic', 'file', 'synth_extra_map']	tools/yosys/techmaps/lcu_kogge_stone.v, siliconcompiler
['tool', 'yosys', 'task', 'syn_asic', 'input']	<design>.v
['tool', 'yosys', 'task', 'syn_asic', 'output']	<design>.vg

continues on next page

Table 136 – continued from previous page

['tool', 'yosys', 'task', 'syn_asic', 'require']	• asic,logiclib • library,asap7sc7p5t_rvt,output,slow,nldm • tool,yosys,task,syn_asic,var,flatten • tool,yosys,task,syn_asic,var,hier_iterations • tool,yosys,task,syn_asic,var,hier_threshold • tool,yosys,task,syn_asic,var,autoname • tool,yosys,task,syn_asic,var,add_buffers • library,asap7sc7p5t_rvt,option,var,yosys_buffer_input • library,asap7sc7p5t_rvt,option,var,yosys_buffer_output • tool,yosys,task,syn_asic,var,map_adders • library,asap7sc7p5t_rvt,option,file,yosys_addermap • tool,yosys,task,syn_asic,var,synthesis_corner • library,asap7sc7p5t_rvt,option,var,yosys_abc_clock_multiplier • library,asap7sc7p5t_rvt,option,var,yosys_abc_constraint_load • library,asap7sc7p5t_rvt,option,var,yosys_abc_constraint_load • tool,yosys,task,syn_asic,var,abc_constraint_load
['tool', 'yosys', 'task', 'syn_asic', 'refdir']	tools/yosys, siliconcompiler
['tool', 'yosys', 'task', 'syn_asic', 'script']	sc_syn.tcl

Variables

Parameters	Help
[..., 'var', 'flatten']	true/false, invoke synth with the -flatten option
[..., 'var', 'hier_iterations']	Number of iterations to attempt to determine the hierarchy to flatten
[..., 'var', 'hier_threshold']	Instance limit for the number of cells in a module to preserve.
[..., 'var', 'autoname']	true/false, call autoname to rename wires based on registers
[..., 'var', 'add_buffers']	true/false, flag to indicate whether to add buffers or not.
[..., 'var', 'map_adders']	true/false, techmap adders in Yosys
[..., 'var', 'synthesis_corner']	Timing corner to use for synthesis
[..., 'var', 'abc_constraint_driver']	Buffer that drives the abc techmapping, defaults to first buffer specified

continues on next page

Table 137 – continued from previous page

[..., 'var', 'abc_constraint_load']	Capacitive load for the abc techmapping in fF, if not specified it will not be used
[..., 'var', 'preserve_modules']	List of modules in input files to prevent flatten from "flattening"
[..., 'var', 'blackbox_modules']	List of modules in input files to exclude from synthesis by replacing them with empty blackboxes"
[..., 'var', 'abc_clock_period']	Clock period to use for synthesis in ps, if more than one clock is specified, the smallest period is used.
[..., 'var', 'abc_clock_derating']	Used to derate the clock period to further constrain the clock, values between 0 and 1
[..., 'var', 'strategy']	ABC synthesis strategy. Allowed values are DELAY0-4, AREA0-3, or if the strategy starts with a + it is assumed to be actual commands for ABC.

Files

Parameters	Help
[..., 'file', 'dff_liberty']	Liberty file to use for flip-flop mapping, if not specified the first in the logiclib is used
[..., 'file', 'abc_constraint_file']	File used to pass in constraints to abc
[..., 'file', 'techmap']	File to use for techmapping in Yosys
[..., 'file', 'dff_liberty_file']	File to use for the DFF mapping stage of Yosys
[..., 'file', 'memory_libmap']	File used to map memories with yosys
[..., 'file', 'memory_techmap']	File used to techmap memories with yosys
[..., 'file', 'synth_extra_map']	Files used in synthesis to perform additional techmapping

syn_fpga

Perform FPGA synthesis

Setup file: syn_fpga.py

Built using target: *fpgaflow_demo*

Configuration

Keypath	Value
['tool', 'yosys', 'task', 'syn_fpga', 'regex', 'warnings']	`Warning:`
['tool', 'yosys', 'task', 'syn_fpga', 'regex', 'errors']	`^ERROR`
['tool', 'yosys', 'task', 'syn_fpga', 'option']	`-c`
['tool', 'yosys', 'task', 'syn_fpga', 'input']	`<design>.v`

continues on next page

Table 139 – continued from previous page

['tool', 'yosys', 'task', 'syn_fpga', 'output']	• <design>.vg • <design>_netlist.json • <design>.blif
['tool', 'yosys', 'task', 'syn_fpga', 'require']	fpga,ice40up5k-sg48,lutsize
['tool', 'yosys', 'task', 'syn_fpga', 'refdir']	tools/yosys, siliconcompiler
['tool', 'yosys', 'task', 'syn_fpga', 'script']	sc_syn.tcl

3.4 Pre-Defined PDKs

The following are examples are pre-built pdks that come with SiliconCompiler which you can use for your own builds.

See the pre-built *targets* for examples on how these are used in conjunction with *librarires*, *tools* and *libraries*.

3.4.1 asap7

The asap7 PDK was developed at ASU in collaboration with ARM Research. With funding from the DARPA IDEA program, the PDK was released a permissive open source PDK in 2021. The PDK contains SPICE-compatible FinFET device models (BSIM-CMG), Technology files for Cadence Virtuoso, Design Rule Checker (DRC), Layout vs Schematic Checker (LVS) and Extraction Deck for the 7nm technology node. For more details regarding the technical specifications of the PDK, please refer the PDK documentation and associated publication. Note that this process design kit is provided as an academic and research aid only and the resulting designs are not manufacturable.

PDK content:

- open source DRM
- device primitive library (virtuoso)
- spice (hspice)
- extraction runsets (calibre)
- drc runsets (calibre)
- APR technology files
- 7.5 track multi-vt standard cell libraries

More information:

- https://asap.asu.edu/
- L.T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthya, and G. Yeric, "ASAP7: A 7-nm FinFET Predictive Process Design Kit," Microelectronics Journal, vol. 53, pp. 105-115, July 2016.

Sources: https://github.com/The-OpenROAD-Project/asap7

Warning: Work in progress (not ready for use)

Setup file: asap7.py

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

Configuration

pdk, asap7

Keypath	Value
['pdk', 'asap7', 'foundry']	virtual
['pdk', 'asap7', 'node']	7.0
['pdk', 'asap7', 'version']	r1p7
['pdk', 'asap7', 'stackup']	10M
['pdk', 'asap7', 'wafersize']	300.0

pdk, asap7, minlayer

Keypath	Value
['pdk', 'asap7', 'minlayer', '10M']	M2

pdk, asap7, maxlayer

Keypath	Value
['pdk', 'asap7', 'maxlayer', '10M']	M7

pdk, asap7, devmodel, xyce, hspice

Keypath	Value
['pdk', 'asap7', 'devmodel', 'xyce', 'hspice', '10M']	lambdapdk/asap7/base/spice/hspice/7nm.lib, lambdapdk

pdk, asap7, pexmodel, openroad, 10M

Keypath	Value
['pdk', 'asap7', 'pexmodel', 'openroad', '10M', 'typical']	lambdapdk/asap7/base/pex/openroad/typical.tcl, lambdapdk

pdk, asap7, pexmodel, openroad-openrcx, 10M

Keypath	Value
['pdk', 'asap7', 'pexmodel', 'openroad-openrcx', '10M', 'typical']	lambdapdk/asap7/base/pex/openroad/typical.rules, lambdapdk

pdk, asap7, layermap, klayout, def, klayout

Keypath	Value
['pdk', 'asap7', 'layermap', 'klayout', 'def', 'klayout', '10M']	lambdapdk/asap7/base/setup/klayout/asap7.lyt, lambdapdk

pdk, asap7, layermap, klayout, def, gds

Keypath	Value
['pdk', 'asap7', 'layermap', 'klayout', 'def', 'gds', '10M']	lambdapdk/asap7/base/apr/asap7.layermap, lambdapdk

pdk, asap7, display, klayout

Keypath	Value
['pdk', 'asap7', 'display', 'klayout', '10M']	lambdapdk/asap7/base/setup/klayout/asap7.lyp, lambdapdk

pdk, asap7, aprtech, openroad, 10M, 7p5t

Keypath	Value
['pdk', 'asap7', 'aprtech', 'openroad', '10M', '7p5t', 'lef']	lambdapdk/asap7/base/apr/asap7_tech.lef, lambdapdk

pdk, asap7, aprtech, klayout, 10M, 7p5t

Keypath	Value
['pdk', 'asap7', 'aprtech', 'klayout', '10M', '7p5t', 'lef']	lambdapdk/asap7/base/apr/asap7_tech.lef, lambdapdk

pdk, asap7, aprtech, magic, 10M, 7p5t

Keypath	Value
['pdk', 'asap7', 'aprtech', 'magic', '10M', '7p5t', 'lef']	lambdapdk/asap7/base/apr/asap7_tech.lef, lambdapdk

pdk, asap7, var, openroad, M1_adjustment

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'M1_adjustment', '10M']	1.0

pdk, asap7, var, openroad, M2_adjustment

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'M2_adjustment', '10M']	0.8

pdk, asap7, var, openroad, M3_adjustment

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'M3_adjustment', '10M']	0.7

pdk, asap7, var, openroad, M4_adjustment

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'M4_adjustment', '10M']	0.4

pdk, asap7, var, openroad, M5_adjustment

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'M5_adjustment', '10M']	0.4

pdk, asap7, var, openroad, M6_adjustment

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'M6_adjustment', '10M']	0.4

pdk, asap7, var, openroad, M7_adjustment

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'M7_adjustment', '10M']	0.4

pdk, asap7, var, openroad, M8_adjustment

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'M8_adjustment', '10M']	0.4

pdk, asap7, var, openroad, M9_adjustment

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'M9_adjustment', '10M']	0.4

pdk, asap7, var, openroad, Pad_adjustment

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'Pad_adjustment', '10M']	1.0

pdk, asap7, var, openroad, rclayer_signal

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'rclayer_signal', '10M']	M3

pdk, asap7, var, openroad, rclayer_clock

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'rclayer_clock', '10M']	M3

pdk, asap7, var, openroad, pin_layer_vertical

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'pin_layer_vertical', '10M']	M5

pdk, asap7, var, openroad, pin_layer_horizontal

Keypath	Value
['pdk', 'asap7', 'var', 'openroad', 'pin_layer_horizontal', '10M']	M4

pdk, asap7, var, klayout, hide_layers

Keypath	Value
['pdk', 'asap7', 'var', 'klayout', 'hide_layers', '10M']	• 235/0 • 235/5 • 100/0 • 97/0 • 98/0

3.4.2 freepdk45

The freepdk45 PDK is a virtual PDK derived from the work done at NCSU (NCSU_TechLib_FreePDK45.) It supplies techfiles, display resources, design rules and scripts to permit layout design and rule checking for a generic 45 nanometer process. The technology information contained in this kit has been compiled from published papers, predictive technology models and rule scaling. This information may be freely used, modified, and distributed under the open-source Apache License (see the file APACHE-LICENSE-2.0.txt in the root install directory for the complete text). This technology is intended to work with the 45nm BSIM4 Predictive Technology Model from Arizona State University (https://ptm.asu.edu/). See the HSPICE Models section of this file for details about these models.

More information:

- https://eda.ncsu.edu/freepdk/freepdk45/

Setup file: freepdk45.py

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

Configuration

pdk, freepdk45

Keypath	Value
['pdk', 'freepdk45', 'foundry']	virtual
['pdk', 'freepdk45', 'node']	45.0
['pdk', 'freepdk45', 'version']	r1p0
['pdk', 'freepdk45', 'stackup']	10M
['pdk', 'freepdk45', 'wafersize']	300.0
['pdk', 'freepdk45', 'd0']	1.25
['pdk', 'freepdk45', 'hscribe']	0.1
['pdk', 'freepdk45', 'vscribe']	0.1
['pdk', 'freepdk45', 'edgemargin']	2.0

pdk, freepdk45, minlayer

Keypath	Value
['pdk', 'freepdk45', 'minlayer', '10M']	metal2

pdk, freepdk45, maxlayer

Keypath	Value
['pdk', 'freepdk45', 'maxlayer', '10M']	metal10

pdk, freepdk45, pexmodel, openroad, 10M

Keypath	Value
['pdk', 'freepdk45', 'pexmodel', 'openroad', '10M', 'typical']	lambdapdk/freepdk45/base/pex/openroad/typical.tcl, lambdapdk

pdk, freepdk45, pexmodel, openroad-openrcx, 10M

Keypath	Value
['pdk', 'freepdk45', 'pexmodel', 'openroad-openrcx', '10M', 'typical']	lambdapdk/freepdk45/base/pex/openroad/typical.rules, lambdapdk

pdk, freepdk45, layermap, klayout, def, klayout

Keypath	Value
['pdk', 'freepdk45', 'layermap', 'klayout', 'def', 'klayout', '10M']	lambdapdk/freepdk45/base/setup/klayout/freepdk45.lyt, lambdapdk

pdk, freepdk45, display, klayout

Keypath	Value
['pdk', 'freepdk45', 'display', 'klayout', '10M']	lambdapdk/freepdk45/base/setup/klayout/freepdk45.lyp, lambdapdk

pdk, freepdk45, aprtech, openroad, 10M, 10t

Keypath	Value
['pdk', 'freepdk45', 'aprtech', 'openroad', '10M', '10t', 'lef']	lambdapdk/freepdk45/base/apr/freepdk45.tech.lef, lambdapdk

pdk, freepdk45, aprtech, klayout, 10M, 10t

Keypath	Value
['pdk', 'freepdk45', 'aprtech', 'klayout', '10M', '10t', 'lef']	lambdapdk/freepdk45/base/apr/freepdk45.tech.lef, lambdapdk

pdk, freepdk45, aprtech, magic, 10M, 10t

Keypath	Value
['pdk', 'freepdk45', 'aprtech', 'magic', '10M', '10t', 'lef']	lambdapdk/freepdk45/base/apr/freepdk45.tech.lef, lambdapdk

pdk, freepdk45, var, openroad, metal1_adjustment

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'metal1_adjustment', '10M']	1.0

pdk, freepdk45, var, openroad, metal2_adjustment

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'metal2_adjustment', '10M']	0.5

pdk, freepdk45, var, openroad, metal3_adjustment

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'metal3_adjustment', '10M']	0.5

pdk, freepdk45, var, openroad, metal4_adjustment

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'metal4_adjustment', '10M']	0.25

pdk, freepdk45, var, openroad, metal5_adjustment

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'metal5_adjustment', '10M']	0.25

pdk, freepdk45, var, openroad, metal6_adjustment

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'metal6_adjustment', '10M']	0.25

pdk, freepdk45, var, openroad, metal7_adjustment

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'metal7_adjustment', '10M']	0.25

pdk, freepdk45, var, openroad, metal8_adjustment

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'metal8_adjustment', '10M']	0.25

pdk, freepdk45, var, openroad, metal9_adjustment

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'metal9_adjustment', '10M']	0.25

pdk, freepdk45, var, openroad, metal10_adjustment

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'metal10_adjustment', '10M']	0.25

pdk, freepdk45, var, openroad, rclayer_signal

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'rclayer_signal', '10M']	metal3

pdk, freepdk45, var, openroad, rclayer_clock

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'rclayer_clock', '10M']	metal5

pdk, freepdk45, var, openroad, pin_layer_vertical

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'pin_layer_vertical', '10M']	metal6

pdk, freepdk45, var, openroad, pin_layer_horizontal

Keypath	Value
['pdk', 'freepdk45', 'var', 'openroad', 'pin_layer_horizontal', '10M']	metal5

3.4.3 gf180

The 'gf180' Open Source PDK is a collaboration between Google and Global Foundries to provide a fully open source Process Design Kit and related resources, which can be used to create manufacturable designs at Global Foundries facility.

... GF180 Process Highlights:

- 180nm process
- 11 metal stack options from 3 to 6 metal levels

PDK content:

- multiple standard digital cell libraries
- primitive cell libraries and models for creating analog designs
- EDA support files for multiple open source and proprietary flows

More information:

- https://gf180mcu-pdk.readthedocs.io/

Sources:

- https://github.com/google/gf180mcu-pdk

Setup file: gf180.py

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

Configuration

pdk, gf180

Keypath	Value
['pdk', 'gf180', 'foundry']	globalfoundries
['pdk', 'gf180', 'node']	180.0
['pdk', 'gf180', 'stackup']	• 3LM_1TM_6K • 3LM_1TM_9K • 3LM_1TM_11K • 3LM_1TM_30K • 4LM_1TM_6K • 4LM_1TM_9K • 4LM_1TM_11K • 4LM_1TM_30K • 5LM_1TM_9K • 5LM_1TM_11K • 6LM_1TM_9K
['pdk', 'gf180', 'wafersize']	200.0

pdk, gf180, minlayer

Keypath	Value
['pdk', 'gf180', 'minlayer', '3LM_1TM_6K']	Metal1
['pdk', 'gf180', 'minlayer', '3LM_1TM_9K']	Metal1
['pdk', 'gf180', 'minlayer', '3LM_1TM_11K']	Metal1
['pdk', 'gf180', 'minlayer', '3LM_1TM_30K']	Metal1
['pdk', 'gf180', 'minlayer', '4LM_1TM_6K']	Metal1
['pdk', 'gf180', 'minlayer', '4LM_1TM_9K']	Metal1
['pdk', 'gf180', 'minlayer', '4LM_1TM_11K']	Metal1
['pdk', 'gf180', 'minlayer', '4LM_1TM_30K']	Metal1
['pdk', 'gf180', 'minlayer', '5LM_1TM_9K']	Metal1
['pdk', 'gf180', 'minlayer', '5LM_1TM_11K']	Metal1
['pdk', 'gf180', 'minlayer', '6LM_1TM_9K']	Metal1

pdk, gf180, maxlayer

Keypath	Value
['pdk', 'gf180', 'maxlayer', '3LM_1TM_6K']	Metal3
['pdk', 'gf180', 'maxlayer', '3LM_1TM_9K']	Metal3
['pdk', 'gf180', 'maxlayer', '3LM_1TM_11K']	Metal3
['pdk', 'gf180', 'maxlayer', '3LM_1TM_30K']	Metal3
['pdk', 'gf180', 'maxlayer', '4LM_1TM_6K']	Metal4
['pdk', 'gf180', 'maxlayer', '4LM_1TM_9K']	Metal4

continues on next page

Table 196 – continued from previous page

['pdk', 'gf180', 'maxlayer', '4LM_1TM_11K']	Metal4
['pdk', 'gf180', 'maxlayer', '4LM_1TM_30K']	Metal4
['pdk', 'gf180', 'maxlayer', '5LM_1TM_9K']	Metal5
['pdk', 'gf180', 'maxlayer', '5LM_1TM_11K']	Metal5
['pdk', 'gf180', 'maxlayer', '6LM_1TM_9K']	Metal6

pdk, gf180, devmodel, xyce, spice

Keypath	Value
['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '3LM_1TM_6K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk
['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '3LM_1TM_9K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk
['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '3LM_1TM_11K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk
['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '3LM_1TM_30K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk
['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '4LM_1TM_6K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk

continues on next page

Table 197 – continued from previous page

['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '4LM_1TM_9K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk
['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '4LM_1TM_11K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk
['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '4LM_1TM_30K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk
['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '5LM_1TM_9K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk
['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '5LM_1TM_11K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk
['pdk', 'gf180', 'devmodel', 'xyce', 'spice', '6LM_1TM_9K']	• lambdapdk/gf180/base//spice/xyce/design.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/sm141064.xyce, lambdapdk • lambdapdk/gf180/base//spice/xyce/smbb000149.xyce, lambdapdk

pdk, gf180, pexmodel, openroad, 4LM_1TM_9K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad', '4LM_1TM_9K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_9k_sp_smim_OPTB_bst.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '4LM_1TM_9K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_9k_sp_smim_OPTB_typ.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '4LM_1TM_9K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_9k_sp_smim_OPTB_wst.tcl, lambdapdk

pdk, gf180, pexmodel, openroad, 4LM_1TM_11K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad', '4LM_1TM_11K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_11k_sp_smim_OPTB_bst.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '4LM_1TM_11K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_11k_sp_smim_OPTB_typ.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '4LM_1TM_11K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_11k_sp_smim_OPTB_wst.tcl, lambdapdk

pdk, gf180, pexmodel, openroad, 4LM_1TM_30K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad', '4LM_1TM_30K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_30k_sp_smim_OPTB_bst.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '4LM_1TM_30K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_30k_sp_smim_OPTB_typ.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '4LM_1TM_30K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_30k_sp_smim_OPTB_wst.tcl, lambdapdk

pdk, gf180, pexmodel, openroad, 5LM_1TM_9K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad', '5LM_1TM_9K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_9k_sp_smim_OPTB_bst.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '5LM_1TM_9K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_9k_sp_smim_OPTB_typ.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '5LM_1TM_9K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_9k_sp_smim_OPTB_wst.tcl, lambdapdk

pdk, gf180, pexmodel, openroad, 5LM_1TM_11K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad', '5LM_1TM_11K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_11k_sp_smim_OPTB_bst.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '5LM_1TM_11K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_11k_sp_smim_OPTB_typ.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '5LM_1TM_11K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_11k_sp_smim_OPTB_wst.tcl, lambdapdk

pdk, gf180, pexmodel, openroad, 6LM_1TM_9K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad', '6LM_1TM_9K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p6m_1tm_9k_sp_smim_OPTB_bst.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '6LM_1TM_9K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p6m_1tm_9k_sp_smim_OPTB_typ.tcl, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad', '6LM_1TM_9K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p6m_1tm_9k_sp_smim_OPTB_wst.tcl, lambdapdk

pdk, gf180, pexmodel, openroad-openrcx, 4LM_1TM_9K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '4LM_1TM_9K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_9k_sp_smim_OPTB_bst.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '4LM_1TM_9K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_9k_sp_smim_OPTB_typ.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '4LM_1TM_9K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_9k_sp_smim_OPTB_wst.rules, lambdapdk

pdk, gf180, pexmodel, openroad-openrcx, 4LM_1TM_11K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '4LM_1TM_11K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_11k_sp_smim_OPTB_bst.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '4LM_1TM_11K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_11k_sp_smim_OPTB_typ.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '4LM_1TM_11K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_11k_sp_smim_OPTB_wst.rules, lambdapdk

pdk, gf180, pexmodel, openroad-openrcx, 4LM_1TM_30K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '4LM_1TM_30K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_30k_sp_smim_OPTB_bst.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '4LM_1TM_30K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_30k_sp_smim_OPTB_typ.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '4LM_1TM_30K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p4m_1tm_30k_sp_smim_OPTB_wst.rules, lambdapdk

pdk, gf180, pexmodel, openroad-openrcx, 5LM_1TM_9K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '5LM_1TM_9K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_9k_sp_smim_OPTB_bst.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '5LM_1TM_9K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_9k_sp_smim_OPTB_typ.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '5LM_1TM_9K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_9k_sp_smim_OPTB_wst.rules, lambdapdk

pdk, gf180, pexmodel, openroad-openrcx, 5LM_1TM_11K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '5LM_1TM_11K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_11k_sp_smim_OPTB_bst.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '5LM_1TM_11K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_11k_sp_smim_OPTB_typ.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '5LM_1TM_11K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p5m_1tm_11k_sp_smim_OPTB_wst.rules, lambdapdk

pdk, gf180, pexmodel, openroad-openrcx, 6LM_1TM_9K

Keypath	Value
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '6LM_1TM_9K', 'bst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p6m_1tm_9k_sp_smim_OPTB_bst.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '6LM_1TM_9K', 'typ']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p6m_1tm_9k_sp_smim_OPTB_typ.rules, lambdapdk
['pdk', 'gf180', 'pexmodel', 'openroad-openrcx', '6LM_1TM_9K', 'wst']	lambdapdk/gf180/base//pex/openroad/gf180mcu_1p6m_1tm_9k_sp_smim_OPTB_wst.rules, lambdapdk

pdk, gf180, layermap, klayout, def, klayout

Keypath	Value
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '3LM_1TM_6K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '3LM_1TM_9K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '3LM_1TM_11K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '3LM_1TM_30K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '4LM_1TM_6K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '4LM_1TM_9K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '4LM_1TM_11K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '4LM_1TM_30K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '5LM_1TM_9K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '5LM_1TM_11K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'klayout', '6LM_1TM_9K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyt, lambdapdk

pdk, gf180, layermap, klayout, def, gds

Keypath	Value
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'gds', '5LM_1TM_9K']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_9K_9t_edi2gds.layermap, lambdapdk
['pdk', 'gf180', 'layermap', 'klayout', 'def', 'gds', '6LM_1TM_9K']	lambdapdk/gf180/base//apr/gf180mcu_6LM_1TM_9K_9t_edi2gds.layermap, lambdapdk

pdk, gf180, display, klayout

Keypath	Value
['pdk', 'gf180', 'display', 'klayout', '3LM_1TM_6K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk
['pdk', 'gf180', 'display', 'klayout', '3LM_1TM_9K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk
['pdk', 'gf180', 'display', 'klayout', '3LM_1TM_11K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk
['pdk', 'gf180', 'display', 'klayout', '3LM_1TM_30K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk

continues on next page

Table 212 – continued from previous page

['pdk', 'gf180', 'display', 'klayout', '4LM_1TM_6K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk
['pdk', 'gf180', 'display', 'klayout', '4LM_1TM_9K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk
['pdk', 'gf180', 'display', 'klayout', '4LM_1TM_11K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk
['pdk', 'gf180', 'display', 'klayout', '4LM_1TM_30K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk
['pdk', 'gf180', 'display', 'klayout', '5LM_1TM_9K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk
['pdk', 'gf180', 'display', 'klayout', '5LM_1TM_11K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk
['pdk', 'gf180', 'display', 'klayout', '6LM_1TM_9K']	lambdapdk/gf180/base//setup/klayout/tech/gf180mcu.lyp, lambdapdk

pdk, gf180, aprtech, openroad, 3LM_1TM_6K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '3LM_1TM_6K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_6K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 3LM_1TM_6K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '3LM_1TM_6K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_6K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 3LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '3LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 3LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '3LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 3LM_1TM_11K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '3LM_1TM_11K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_11K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 3LM_1TM_11K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '3LM_1TM_11K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_11K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 3LM_1TM_30K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '3LM_1TM_30K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_30K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 3LM_1TM_30K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '3LM_1TM_30K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_30K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 4LM_1TM_6K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '4LM_1TM_6K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_6K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 4LM_1TM_6K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '4LM_1TM_6K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_6K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 4LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '4LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 4LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '4LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 4LM_1TM_11K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '4LM_1TM_11K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_11K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 4LM_1TM_11K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '4LM_1TM_11K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_11K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 4LM_1TM_30K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '4LM_1TM_30K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_30K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 4LM_1TM_30K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '4LM_1TM_30K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_30K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 5LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '5LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 5LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '5LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 5LM_1TM_11K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '5LM_1TM_11K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_11K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 5LM_1TM_11K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '5LM_1TM_11K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_11K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 6LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '6LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_6LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, openroad, 6LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'openroad', '6LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_6LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 3LM_1TM_6K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '3LM_1TM_6K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_6K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 3LM_1TM_6K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '3LM_1TM_6K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_6K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 3LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '3LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 3LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '3LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 3LM_1TM_11K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '3LM_1TM_11K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_11K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 3LM_1TM_11K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '3LM_1TM_11K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_11K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 3LM_1TM_30K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '3LM_1TM_30K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_30K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 3LM_1TM_30K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '3LM_1TM_30K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_30K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 4LM_1TM_6K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '4LM_1TM_6K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_6K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 4LM_1TM_6K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '4LM_1TM_6K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_6K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 4LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '4LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 4LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '4LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 4LM_1TM_11K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '4LM_1TM_11K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_11K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 4LM_1TM_11K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '4LM_1TM_11K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_11K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 4LM_1TM_30K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '4LM_1TM_30K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_30K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 4LM_1TM_30K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '4LM_1TM_30K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_30K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 5LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '5LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 5LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '5LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 5LM_1TM_11K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '5LM_1TM_11K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_11K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 5LM_1TM_11K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '5LM_1TM_11K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_11K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 6LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '6LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_6LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, klayout, 6LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'klayout', '6LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_6LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 3LM_1TM_6K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '3LM_1TM_6K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_6K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 3LM_1TM_6K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '3LM_1TM_6K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_6K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 3LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '3LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 3LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '3LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 3LM_1TM_11K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '3LM_1TM_11K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_11K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 3LM_1TM_11K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '3LM_1TM_11K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_11K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 3LM_1TM_30K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '3LM_1TM_30K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_30K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 3LM_1TM_30K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '3LM_1TM_30K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_3LM_1TM_30K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 4LM_1TM_6K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '4LM_1TM_6K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_6K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 4LM_1TM_6K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '4LM_1TM_6K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_6K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 4LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '4LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 4LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '4LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 4LM_1TM_11K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '4LM_1TM_11K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_11K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 4LM_1TM_11K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '4LM_1TM_11K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_11K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 4LM_1TM_30K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '4LM_1TM_30K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_30K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 4LM_1TM_30K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '4LM_1TM_30K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_4LM_1TM_30K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 5LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '5LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 5LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '5LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 5LM_1TM_11K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '5LM_1TM_11K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_11K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 5LM_1TM_11K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '5LM_1TM_11K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_5LM_1TM_11K_9t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 6LM_1TM_9K, 7t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '6LM_1TM_9K', '7t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_6LM_1TM_9K_7t_tech.lef, lambdapdk

pdk, gf180, aprtech, magic, 6LM_1TM_9K, 9t

Keypath	Value
['pdk', 'gf180', 'aprtech', 'magic', '6LM_1TM_9K', '9t', 'lef']	lambdapdk/gf180/base//apr/gf180mcu_6LM_1TM_9K_9t_tech.lef, lambdapdk

pdk, gf180, var, openroad, Metal1_adjustment

Keypath	Value
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '3LM_1TM_6K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '3LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '3LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '3LM_1TM_30K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '4LM_1TM_6K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '4LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '4LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '4LM_1TM_30K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '5LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '5LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal1_adjustment', '6LM_1TM_9K']	0.0

pdk, gf180, var, openroad, Metal2_adjustment

Keypath	Value
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '3LM_1TM_6K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '3LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '3LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '3LM_1TM_30K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '4LM_1TM_6K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '4LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '4LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '4LM_1TM_30K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '5LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '5LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal2_adjustment', '6LM_1TM_9K']	0.0

pdk, gf180, var, openroad, Metal3_adjustment

Keypath	Value
['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '3LM_1TM_6K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '3LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '3LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '3LM_1TM_30K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '4LM_1TM_6K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '4LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '4LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '4LM_1TM_30K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '5LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '5LM_1TM_11K']	0.0

continues on next page

Table 281 – continued from previous page

['pdk', 'gf180', 'var', 'openroad', 'Metal3_adjustment', '6LM_1TM_9K']	0.0

pdk, gf180, var, openroad, rclayer_signal

Keypath	Value
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '3LM_1TM_6K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '3LM_1TM_9K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '3LM_1TM_11K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '3LM_1TM_30K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '4LM_1TM_6K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '4LM_1TM_9K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '4LM_1TM_11K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '4LM_1TM_30K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '5LM_1TM_9K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '5LM_1TM_11K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'rclayer_signal', '6LM_1TM_9K']	Metal3

pdk, gf180, var, openroad, rclayer_clock

Keypath	Value
['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '3LM_1TM_6K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '3LM_1TM_9K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '3LM_1TM_11K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '3LM_1TM_30K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '4LM_1TM_6K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '4LM_1TM_9K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '4LM_1TM_11K']	Metal3

continues on next page

Table 283 – continued from previous page

['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '4LM_1TM_30K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '5LM_1TM_9K']	Metal4
['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '5LM_1TM_11K']	Metal4
['pdk', 'gf180', 'var', 'openroad', 'rclayer_clock', '6LM_1TM_9K']	Metal4

pdk, gf180, var, openroad, pin_layer_vertical

Keypath	Value
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '3LM_1TM_6K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '3LM_1TM_9K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '3LM_1TM_11K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '3LM_1TM_30K']	Metal2
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '4LM_1TM_6K']	Metal4
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '4LM_1TM_9K']	Metal4
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '4LM_1TM_11K']	Metal4
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '4LM_1TM_30K']	Metal4
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '5LM_1TM_9K']	Metal4
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '5LM_1TM_11K']	Metal4
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_vertical', '6LM_1TM_9K']	Metal4

pdk, gf180, var, openroad, pin_layer_horizontal

Keypath	Value
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '3LM_1TM_6K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '3LM_1TM_9K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '3LM_1TM_11K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '3LM_1TM_30K']	Metal3

continues on next page

Table 285 – continued from previous page

['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '4LM_1TM_6K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '4LM_1TM_9K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '4LM_1TM_11K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '4LM_1TM_30K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '5LM_1TM_9K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '5LM_1TM_11K']	Metal3
['pdk', 'gf180', 'var', 'openroad', 'pin_layer_horizontal', '6LM_1TM_9K']	Metal3

pdk, gf180, var, openroad, Metal4_adjustment

Keypath	Value
['pdk', 'gf180', 'var', 'openroad', 'Metal4_adjustment', '4LM_1TM_6K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal4_adjustment', '4LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal4_adjustment', '4LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal4_adjustment', '4LM_1TM_30K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal4_adjustment', '5LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal4_adjustment', '5LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal4_adjustment', '6LM_1TM_9K']	0.0

pdk, gf180, var, openroad, Metal5_adjustment

Keypath	Value
['pdk', 'gf180', 'var', 'openroad', 'Metal5_adjustment', '5LM_1TM_9K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal5_adjustment', '5LM_1TM_11K']	0.0
['pdk', 'gf180', 'var', 'openroad', 'Metal5_adjustment', '6LM_1TM_9K']	0.0

pdk, gf180, var, openroad, Metal6_adjustment

Keypath	Value
['pdk', 'gf180', 'var', 'openroad', 'Metal6_adjustment', '6LM_1TM_9K']	0.0

pdk, gf180, var, klayout, hide_layers

Keypath	Value
['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '3LM_1TM_6K']	• Dualgate • V5_XTOR • PR_bndry
['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '3LM_1TM_9K']	• Dualgate • V5_XTOR • PR_bndry
['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '3LM_1TM_11K']	• Dualgate • V5_XTOR • PR_bndry
['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '3LM_1TM_30K']	• Dualgate • V5_XTOR • PR_bndry
['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '4LM_1TM_6K']	• Dualgate • V5_XTOR • PR_bndry
['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '4LM_1TM_9K']	• Dualgate • V5_XTOR • PR_bndry
['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '4LM_1TM_11K']	• Dualgate • V5_XTOR • PR_bndry

continues on next page

Table 289 – continued from previous page

['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '4LM_1TM_30K']	• Dualgate • V5_XTOR • PR_bndry
['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '5LM_1TM_9K']	• Dualgate • V5_XTOR • PR_bndry
['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '5LM_1TM_11K']	• Dualgate • V5_XTOR • PR_bndry
['pdk', 'gf180', 'var', 'klayout', 'hide_layers', '6LM_1TM_9K']	• Dualgate • V5_XTOR • PR_bndry

3.4.4 skywater130

The 'skywater130' Open Source PDK is a collaboration between Google and SkyWater Technology Foundry to provide a fully open source Process Design Kit and related resources, which can be used to create manufacturable designs at SkyWater's facility.

Skywater130 Process Highlights:

- 130nm process
- support for internal 1.8V with 5.0V I/Os (operable at 2.5V)
- 1 level of local interconnect
- 5 levels of metal

PDK content:

- An open source design rule manual
- multiple standard digital cell libraries
- primitive cell libraries and models for creating analog designs
- EDA support files for multiple open source and proprietary flows

More information:

- https://skywater-pdk.readthedocs.io/

Sources:

- https://github.com/google/skywater-pdk

Setup file: skywater130.py

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

Configuration

pdk, skywater130

Keypath	Value
['pdk', 'skywater130', 'foundry']	skywater
['pdk', 'skywater130', 'node']	130.0
['pdk', 'skywater130', 'version']	v0_0_2
['pdk', 'skywater130', 'stackup']	5M1LI
['pdk', 'skywater130', 'wafersize']	300.0
['pdk', 'skywater130', 'hscribe']	0.1
['pdk', 'skywater130', 'vscribe']	0.1
['pdk', 'skywater130', 'edgemargin']	2.0

pdk, skywater130, minlayer

Keypath	Value
['pdk', 'skywater130', 'minlayer', '5M1LI']	met1

pdk, skywater130, maxlayer

Keypath	Value
['pdk', 'skywater130', 'maxlayer', '5M1LI']	met5

pdk, skywater130, pexmodel, openroad, 5M1LI

Keypath	Value
['pdk', 'skywater130', 'pexmodel', 'openroad', '5M1LI', 'minimum']	lambdapdk/sky130/base/pex/openroad/minimum.tcl, lambdapdk
['pdk', 'skywater130', 'pexmodel', 'openroad', '5M1LI', 'typical']	lambdapdk/sky130/base/pex/openroad/typical.tcl, lambdapdk

continues on next page

Table 294 – continued from previous page

['pdk', 'skywater130', 'pexmodel', 'openroad', '5M1LI', 'maximum']	lambdapdk/sky130/base/pex/openroad/maximum.tcl, lambdapdk

pdk, skywater130, pexmodel, openroad-openrcx, 5M1LI

Keypath	Value
['pdk', 'skywater130', 'pexmodel', 'openroad-openrcx', '5M1LI', 'minimum']	lambdapdk/sky130/base/pex/openroad/minimum.rules, lambdapdk
['pdk', 'skywater130', 'pexmodel', 'openroad-openrcx', '5M1LI', 'typical']	lambdapdk/sky130/base/pex/openroad/typical.rules, lambdapdk
['pdk', 'skywater130', 'pexmodel', 'openroad-openrcx', '5M1LI', 'maximum']	lambdapdk/sky130/base/pex/openroad/maximum.rules, lambdapdk

pdk, skywater130, layermap, klayout, def, klayout

Keypath	Value
['pdk', 'skywater130', 'layermap', 'klayout', 'def', 'klayout', '5M1LI']	lambdapdk/sky130/base/setup/klayout/skywater130.lyt, lambdapdk

pdk, skywater130, display, klayout

Keypath	Value
['pdk', 'skywater130', 'display', 'klayout', '5M1LI']	lambdapdk/sky130/base/setup/klayout/sky130A.lyp, lambdapdk

pdk, skywater130, aprtech, openroad, 5M1LI, unithd

Keypath	Value
['pdk', 'skywater130', 'aprtech', 'openroad', '5M1LI', 'unithd', 'lef']	lambdapdk/sky130/base/apr/sky130_fd_sc_hd.tlef, lambdapdk

pdk, skywater130, aprtech, openroad, 5M1LI, hd

Keypath	Value
['pdk', 'skywater130', 'aprtech', 'openroad', '5M1LI', 'hd', 'lef']	lambdapdk/sky130/base/apr/sky130_fd_sc_hd.tlef, lambdapdk

pdk, skywater130, aprtech, openroad, 5M1LI, hdll

Keypath	Value
['pdk', 'skywater130', 'aprtech', 'openroad', '5M1LI', 'hdll', 'lef']	lambdapdk/sky130/base/apr/sky130_fd_sc_hdll.tlef, lambdapdk

pdk, skywater130, aprtech, klayout, 5M1LI, unithd

Keypath	Value
['pdk', 'skywater130', 'aprtech', 'klayout', '5M1LI', 'unithd', 'lef']	lambdapdk/sky130/base/apr/sky130_fd_sc_hd.tlef, lambdapdk

pdk, skywater130, aprtech, klayout, 5M1LI, hd

Keypath	Value
['pdk', 'skywater130', 'aprtech', 'klayout', '5M1LI', 'hd', 'lef']	lambdapdk/sky130/base/apr/sky130_fd_sc_hd.tlef, lambdapdk

pdk, skywater130, aprtech, klayout, 5M1LI, hdll

Keypath	Value
['pdk', 'skywater130', 'aprtech', 'klayout', '5M1LI', 'hdll', 'lef']	lambdapdk/sky130/base/apr/sky130_fd_sc_hdll.tlef, lambdapdk

pdk, skywater130, aprtech, magic, 5M1LI, unithd

Keypath	Value
['pdk', 'skywater130', 'aprtech', 'magic', '5M1LI', 'unithd', 'lef']	lambdapdk/sky130/base/apr/sky130_fd_sc_hd.tlef, lambdapdk

pdk, skywater130, aprtech, magic, 5M1LI, hd

Keypath	Value
['pdk', 'skywater130', 'aprtech', 'magic', '5M1LI', 'hd', 'lef']	lambdapdk/sky130/base/apr/sky130_fd_sc_hd.tlef, lambdapdk

pdk, skywater130, aprtech, magic, 5M1LI, hdll

Keypath	Value
['pdk', 'skywater130', 'aprtech', 'magic', '5M1LI', 'hdll', 'lef']	lambdapdk/sky130/base/apr/sky130_fd_sc_hdll.tlef, lambdapdk

pdk, skywater130, lvs, runset, netgen, 5M1LI

Keypath	Value
['pdk', 'skywater130', 'lvs', 'runset', 'netgen', '5M1LI', 'basic']	lambdapdk/sky130/base/setup/netgen/lvs_setup.tcl, lambdapdk

pdk, skywater130, drc, runset, magic, 5M1LI

Keypath	Value
['pdk', 'skywater130', 'drc', 'runset', 'magic', '5M1LI', 'basic']	lambdapdk/sky130/base/setup/magic/sky130A.tech, lambdapdk

pdk, skywater130, var, openroad, li1_adjustment

Keypath	Value
['pdk', 'skywater130', 'var', 'openroad', 'li1_adjustment', '5M1LI']	1.0

pdk, skywater130, var, openroad, met1_adjustment

Keypath	Value
['pdk', 'skywater130', 'var', 'openroad', 'met1_adjustment', '5M1LI']	0.3

pdk, skywater130, var, openroad, met2_adjustment

Keypath	Value
['pdk', 'skywater130', 'var', 'openroad', 'met2_adjustment', '5M1LI']	0.3

pdk, skywater130, var, openroad, met3_adjustment

Keypath	Value
['pdk', 'skywater130', 'var', 'openroad', 'met3_adjustment', '5M1LI']	0.3

pdk, skywater130, var, openroad, met4_adjustment

Keypath	Value
['pdk', 'skywater130', 'var', 'openroad', 'met4_adjustment', '5M1LI']	0.3

pdk, skywater130, var, openroad, met5_adjustment

Keypath	Value
['pdk', 'skywater130', 'var', 'openroad', 'met5_adjustment', '5M1LI']	0.3

pdk, skywater130, var, openroad, rclayer_signal

Keypath	Value
['pdk', 'skywater130', 'var', 'openroad', 'rclayer_signal', '5M1LI']	met2

pdk, skywater130, var, openroad, rclayer_clock

Keypath	Value
['pdk', 'skywater130', 'var', 'openroad', 'rclayer_clock', '5M1LI']	met5

pdk, skywater130, var, openroad, pin_layer_vertical

Keypath	Value
['pdk', 'skywater130', 'var', 'openroad', 'pin_layer_vertical', '5M1LI']	met2

pdk, skywater130, var, openroad, pin_layer_horizontal

Keypath	Value
['pdk', 'skywater130', 'var', 'openroad', 'pin_layer_horizontal', '5M1LI']	met3

pdk, skywater130, var, klayout, hide_layers

Keypath	Value
['pdk', 'skywater130', 'var', 'klayout', 'hide_layers', '5M1LI']	areaid.standardc

3.5 Pre-Defined Libraries

The following are examples are pre-built libraries that come with SiliconCompiler which you can use for your own builds.

See the pre-built *targets* for examples on how these are used in conjunction with *pdks*, *tools* and *flows*.

3.5.1 asap7sc7p5t

ASAP 7 7.5-track standard cell library.

Setup file: asap7sc7p5t.py

Associated PDK: *asap7*

Data sources

Package	Specifications
lambdapdk	• Path: `https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/` • Reference: `v0.1.19`

asap7sc7p5t_rvt

asic

Keypath	Value
['asic', 'libarch']	7p5t

asic, cells

Keypath	Value
['asic', 'cells', 'decap']	• DECAPx1_ASAP7_75t_R • DECAPx1_ASAP7_75t_R • DECAPx3_ASAP7_75t_R • DECAPx6_ASAP7_75t_R • DECAPx10_ASAP7_75t_R
['asic', 'cells', 'tie']	• TIEHIx1_ASAP7_75t_R • TIELOx1_ASAP7_75t_R
['asic', 'cells', 'hold']	BUFx2_ASAP7_75t_R
['asic', 'cells', 'clkbuf']	BUFx2_ASAP7_75t_R
['asic', 'cells', 'dontuse']	• *x1p*_ASAP7* • *xp*_ASAP7* • SDF* • ICG*
['asic', 'cells', 'filler']	• FILLER_ASAP7_75t_R • FILLERxp5_ASAP7_75t_R
['asic', 'cells', 'tap']	TAPCELL_ASAP7_75t_R
['asic', 'cells', 'endcap']	DECAPx1_ASAP7_75t_R

asic, site

Keypath	Value
['asic', 'site', '7p5t']	asap7sc7p5t

output, typical

Keypath	Value
['output', 'typical', 'nldm']	• lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_AO_RVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_INVBUF_RVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_OA_RVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_SEQ_RVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_SIMPLE_RVT_TT_nldm.lib.gz, lambdapdk
['output', 'typical', 'spice']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/netlist/asap7sc7p5t_28_R.sp, lambdapdk

output, fast

Keypath	Value
['output', 'fast', 'nldm']	• lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_AO_RVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_INVBUF_RVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_OA_RVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_SEQ_RVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_SIMPLE_RVT_FF_nldm.lib.gz, lambdapdk
['output', 'fast', 'spice']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/netlist/asap7sc7p5t_28_R.sp, lambdapdk

output, slow

Keypath	Value
['output', 'slow', 'nldm']	• lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_AO_RVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_INVBUF_RVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_OA_RVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_SEQ_RVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_SIMPLE_RVT_SS_nldm.lib.gz, lambdapdk
['output', 'slow', 'spice']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/netlist/asap7sc7p5t_28_R.sp, lambdapdk

output, 10M

Keypath	Value
['output', '10M', 'lef']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/lef/asap7sc7p5t_28_R.lef, lambdapdk
['output', '10M', 'gds']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/gds/asap7sc7p5t_28_R.gds.gz, lambdapdk
['output', '10M', 'cdl']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/netlist/asap7sc7p5t_28_R.cdl, lambdapdk

option

Keypath	Value
['option', 'pdk']	asap7

option, var

Keypath	Value
['option', 'var', 'openroad_place_density']	0.60
['option', 'var', 'openroad_pad_global_place']	2
['option', 'var', 'openroad_pad_detail_place']	1
['option', 'var', 'openroad_macro_place_halo']	• 10 • 10
['option', 'var', 'openroad_macro_place_channel']	• 12 • 12
['option', 'var', 'openroad_cts_clock_buffer']	BUFx4_ASAP7_75t_R
['option', 'var', 'openroad_cts_distance_between_buffers']	60
['option', 'var', 'yosys_abc_clock_multiplier']	1
['option', 'var', 'yosys_abc_constraint_load']	2.026fF
['option', 'var', 'yosys_driver_cell']	BUFx2_ASAP7_75t_R
['option', 'var', 'yosys_buffer_cell']	BUFx2_ASAP7_75t_R
['option', 'var', 'yosys_buffer_input']	A
['option', 'var', 'yosys_buffer_output']	Y
['option', 'var', 'yosys_tiehigh_cell']	TIEHIx1_ASAP7_75t_R
['option', 'var', 'yosys_tiehigh_port']	H
['option', 'var', 'yosys_tielow_cell']	TIELOx1_ASAP7_75t_R
['option', 'var', 'yosys_tielow_port']	L
['option', 'var', 'openroad_tiehigh_cell']	TIEHIx1_ASAP7_75t_R
['option', 'var', 'openroad_tiehigh_port']	H
['option', 'var', 'openroad_tielow_cell']	TIELOx1_ASAP7_75t_R
['option', 'var', 'openroad_tielow_port']	L

option, file

Keypath	Value
['option', 'file', 'yosys_techmap']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/techmap/yosys/cells_latch.v, lambdapdk
['option', 'file', 'yosys_addermap']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/techmap/yosys/cells_adders.v, lambdapdk
['option', 'file', 'yosys_dff_liberty']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/nldm/asap7sc7p5t_SEQ_RVT_SS_nldm.lib.gz, lambdapdk
['option', 'file', 'openroad_tracks']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/apr/openroad/tracks.tcl, lambdapdk

continues on next page

Table 330 – continued from previous page

['option', 'file', 'openroad_tapcells']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/apr/openroad/tapcells.tcl, lambdapdk
['option', 'file', 'openroad_pdngen']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/apr/openroad/pdngen.tcl, lambdapdk
['option', 'file', 'openroad_global_connect']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/apr/openroad/global_connect.tcl, lambdapdk

Associated PDK: None

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

lambdalib_asap7sc7p5t_rvt

option

Keypath	Value
['option', 'ydir']	lambdapdk/asap7/libs/asap7sc7p5t_rvt/lambda, lambdapdk

Associated PDK: *asap7*

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

asap7sc7p5t_lvt

asic

Keypath	Value
['asic', 'libarch']	7p5t

asic, cells

Keypath	Value
['asic', 'cells', 'decap']	• DECAPx1_ASAP7_75t_L • DECAPx1_ASAP7_75t_L • DECAPx3_ASAP7_75t_L • DECAPx6_ASAP7_75t_L • DECAPx10_ASAP7_75t_L
['asic', 'cells', 'tie']	• TIEHIx1_ASAP7_75t_L • TIELOx1_ASAP7_75t_L
['asic', 'cells', 'hold']	BUFx2_ASAP7_75t_L
['asic', 'cells', 'clkbuf']	BUFx2_ASAP7_75t_L
['asic', 'cells', 'dontuse']	• *x1p*_ASAP7* • *xp*_ASAP7* • SDF* • ICG*
['asic', 'cells', 'filler']	• FILLER_ASAP7_75t_L • FILLERxp5_ASAP7_75t_L
['asic', 'cells', 'tap']	TAPCELL_ASAP7_75t_L
['asic', 'cells', 'endcap']	DECAPx1_ASAP7_75t_L

asic, site

Keypath	Value
['asic', 'site', '7p5t']	asap7sc7p5t

output, typical

Keypath	Value
['output', 'typical', 'nldm']	• lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_AO_LVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_INVBUF_LVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_OA_LVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_SEQ_LVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_SIMPLE_LVT_TT_nldm.lib.gz, lambdapdk
['output', 'typical', 'spice']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/netlist/asap7sc7p5t_28_L.sp, lambdapdk

output, fast

Keypath	Value
['output', 'fast', 'nldm']	• lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_AO_LVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_INVBUF_LVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_OA_LVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_SEQ_LVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_SIMPLE_LVT_FF_nldm.lib.gz, lambdapdk
['output', 'fast', 'spice']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/netlist/asap7sc7p5t_28_L.sp, lambdapdk

output, slow

Keypath	Value
['output', 'slow', 'nldm']	• lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_AO_LVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_INVBUF_LVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_OA_LVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_SEQ_LVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_SIMPLE_LVT_SS_nldm.lib.gz, lambdapdk
['output', 'slow', 'spice']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/netlist/asap7sc7p5t_28_L.sp, lambdapdk

output, 10M

Keypath	Value
['output', '10M', 'lef']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/lef/asap7sc7p5t_28_L.lef, lambdapdk
['output', '10M', 'gds']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/gds/asap7sc7p5t_28_L.gds.gz, lambdapdk
['output', '10M', 'cdl']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/netlist/asap7sc7p5t_28_L.cdl, lambdapdk

option

Keypath	Value
['option', 'pdk']	asap7

option, var

Keypath	Value
['option', 'var', 'openroad_place_density']	0.60
['option', 'var', 'openroad_pad_global_place']	2
['option', 'var', 'openroad_pad_detail_place']	1
['option', 'var', 'openroad_macro_place_halo']	• 10 • 10
['option', 'var', 'openroad_macro_place_channel']	• 12 • 12
['option', 'var', 'openroad_cts_clock_buffer']	BUFx4_ASAP7_75t_L
['option', 'var', 'openroad_cts_distance_between_buffers']	60
['option', 'var', 'yosys_abc_clock_multiplier']	1
['option', 'var', 'yosys_abc_constraint_load']	2.07924fF
['option', 'var', 'yosys_driver_cell']	BUFx2_ASAP7_75t_L
['option', 'var', 'yosys_buffer_cell']	BUFx2_ASAP7_75t_L
['option', 'var', 'yosys_buffer_input']	A
['option', 'var', 'yosys_buffer_output']	Y
['option', 'var', 'yosys_tiehigh_cell']	TIEHIx1_ASAP7_75t_L
['option', 'var', 'yosys_tiehigh_port']	H
['option', 'var', 'yosys_tielow_cell']	TIELOx1_ASAP7_75t_L
['option', 'var', 'yosys_tielow_port']	L
['option', 'var', 'openroad_tiehigh_cell']	TIEHIx1_ASAP7_75t_L
['option', 'var', 'openroad_tiehigh_port']	H
['option', 'var', 'openroad_tielow_cell']	TIELOx1_ASAP7_75t_L
['option', 'var', 'openroad_tielow_port']	L

option, file

Keypath	Value
['option', 'file', 'yosys_techmap']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/techmap/yosys/cells_latch.v, lambdapdk
['option', 'file', 'yosys_addermap']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/techmap/yosys/cells_adders.v, lambdapdk
['option', 'file', 'yosys_dff_liberty']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/nldm/asap7sc7p5t_SEQ_LVT_SS_nldm.lib.gz, lambdapdk
['option', 'file', 'openroad_tracks']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/apr/openroad/tracks.tcl, lambdapdk

continues on next page

Table 343 – continued from previous page

['option', 'file', 'openroad_tapcells']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/apr/openroad/tapcells.tcl, lambdapdk
['option', 'file', 'openroad_pdngen']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/apr/openroad/pdngen.tcl, lambdapdk
['option', 'file', 'openroad_global_connect']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/apr/openroad/global_connect.tcl, lambdapdk

Associated PDK: None

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

lambdalib_asap7sc7p5t_lvt

option

Keypath	Value
['option', 'ydir']	lambdapdk/asap7/libs/asap7sc7p5t_lvt/lambda, lambdapdk

Associated PDK: *asap7*

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

asap7sc7p5t_slvt

asic

Keypath	Value
['asic', 'libarch']	7p5t

asic, cells

Keypath	Value
['asic', 'cells', 'decap']	• DECAPx1_ASAP7_75t_SL • DECAPx1_ASAP7_75t_SL • DECAPx3_ASAP7_75t_SL • DECAPx6_ASAP7_75t_SL • DECAPx10_ASAP7_75t_SL
['asic', 'cells', 'tie']	• TIEHIx1_ASAP7_75t_SL • TIELOx1_ASAP7_75t_SL
['asic', 'cells', 'hold']	BUFx2_ASAP7_75t_SL
['asic', 'cells', 'clkbuf']	BUFx2_ASAP7_75t_SL
['asic', 'cells', 'dontuse']	• *x1p*_ASAP7* • *xp*_ASAP7* • SDF* • ICG*
['asic', 'cells', 'filler']	• FILLER_ASAP7_75t_SL • FILLERxp5_ASAP7_75t_SL
['asic', 'cells', 'tap']	TAPCELL_ASAP7_75t_SL
['asic', 'cells', 'endcap']	DECAPx1_ASAP7_75t_SL

asic, site

Keypath	Value
['asic', 'site', '7p5t']	asap7sc7p5t

output, typical

Keypath	Value
['output', 'typical', 'nldm']	• lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_AO_SLVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_INVBUF_SLVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_OA_SLVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_SEQ_SLVT_TT_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_SIMPLE_SLVT_TT_nldm.lib.gz, lambdapdk
['output', 'typical', 'spice']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/netlist/asap7sc7p5t_28_SL.sp, lambdapdk

output, fast

Keypath	Value
['output', 'fast', 'nldm']	• lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_AO_SLVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_INVBUF_SLVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_OA_SLVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_SEQ_SLVT_FF_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_SIMPLE_SLVT_FF_nldm.lib.gz, lambdapdk
['output', 'fast', 'spice']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/netlist/asap7sc7p5t_28_SL.sp, lambdapdk

output, slow

Keypath	Value
['output', 'slow', 'nldm']	• lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_AO_SLVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_INVBUF_SLVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_OA_SLVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_SEQ_SLVT_SS_nldm.lib.gz, lambdapdk • lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_SIMPLE_SLVT_SS_nldm.lib.gz, lambdapdk
['output', 'slow', 'spice']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/netlist/asap7sc7p5t_28_SL.sp, lambdapdk

output, 10M

Keypath	Value
['output', '10M', 'lef']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/lef/asap7sc7p5t_28_SL.lef, lambdapdk
['output', '10M', 'gds']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/gds/asap7sc7p5t_28_SL.gds.gz, lambdapdk
['output', '10M', 'cdl']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/netlist/asap7sc7p5t_28_SL.cdl, lambdapdk

option

Keypath	Value
['option', 'pdk']	asap7

option, var

Keypath	Value
['option', 'var', 'openroad_place_density']	0.60
['option', 'var', 'openroad_pad_global_place']	2
['option', 'var', 'openroad_pad_detail_place']	1
['option', 'var', 'openroad_macro_place_halo']	• 10 • 10
['option', 'var', 'openroad_macro_place_channel']	• 12 • 12
['option', 'var', 'openroad_cts_clock_buffer']	BUFx4_ASAP7_75t_SL
['option', 'var', 'openroad_cts_distance_between_buffers']	60
['option', 'var', 'yosys_abc_clock_multiplier']	1
['option', 'var', 'yosys_abc_constraint_load']	2.133644fF
['option', 'var', 'yosys_driver_cell']	BUFx2_ASAP7_75t_SL
['option', 'var', 'yosys_buffer_cell']	BUFx2_ASAP7_75t_SL
['option', 'var', 'yosys_buffer_input']	A
['option', 'var', 'yosys_buffer_output']	Y
['option', 'var', 'yosys_tiehigh_cell']	TIEHIx1_ASAP7_75t_SL
['option', 'var', 'yosys_tiehigh_port']	H
['option', 'var', 'yosys_tielow_cell']	TIELOx1_ASAP7_75t_SL
['option', 'var', 'yosys_tielow_port']	L
['option', 'var', 'openroad_tiehigh_cell']	TIEHIx1_ASAP7_75t_SL
['option', 'var', 'openroad_tiehigh_port']	H
['option', 'var', 'openroad_tielow_cell']	TIELOx1_ASAP7_75t_SL
['option', 'var', 'openroad_tielow_port']	L

option, file

Keypath	Value
['option', 'file', 'yosys_techmap']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/techmap/yosys/cells_latch.v, lambdapdk
['option', 'file', 'yosys_addermap']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/techmap/yosys/cells_adders.v, lambdapdk
['option', 'file', 'yosys_dff_liberty']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/nldm/asap7sc7p5t_SEQ_SLVT_SS_nldm.lib.gz, lambdapdk
['option', 'file', 'openroad_tracks']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/apr/openroad/tracks.tcl, lambdapdk

continues on next page

Table 356 – continued from previous page

['option', 'file', 'openroad_tapcells']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/apr/openroad/tapcells.tcl, lambdapdk
['option', 'file', 'openroad_pdngen']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/apr/openroad/pdngen.tcl, lambdapdk
['option', 'file', 'openroad_global_connect']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/apr/openroad/global_connect.tcl, lambdapdk

Associated PDK: None

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

lambdalib_asap7sc7p5t_slvt

option

Keypath	Value
['option', 'ydir']	lambdapdk/asap7/libs/asap7sc7p5t_slvt/lambda, lambdapdk

3.5.2 gf180mcu

Skywater130 standard cell library.

Setup file: gf180mcu.py

Associated PDK: *gf180*

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

gf180mcu_fd_sc_mcu7t5v0

asic

Keypath	Value
['asic', 'libarch']	7t

asic, cells

Keypath	Value
['asic', 'cells', 'decap']	• gf180mcu_fd_sc_mcu7t5v0__fillcap_4 • gf180mcu_fd_sc_mcu7t5v0__fillcap_8 • gf180mcu_fd_sc_mcu7t5v0__fillcap_16 • gf180mcu_fd_sc_mcu7t5v0__fillcap_32 • gf180mcu_fd_sc_mcu7t5v0__fillcap_64
['asic', 'cells', 'delay']	• gf180mcu_fd_sc_mcu7t5v0__dlya_1 • gf180mcu_fd_sc_mcu7t5v0__dlya_2 • gf180mcu_fd_sc_mcu7t5v0__dlya_4 • gf180mcu_fd_sc_mcu7t5v0__dlyb_1 • gf180mcu_fd_sc_mcu7t5v0__dlyb_2 • gf180mcu_fd_sc_mcu7t5v0__dlyb_4 • gf180mcu_fd_sc_mcu7t5v0__dlyc_1 • gf180mcu_fd_sc_mcu7t5v0__dlyc_2 • gf180mcu_fd_sc_mcu7t5v0__dlyc_4 • gf180mcu_fd_sc_mcu7t5v0__dlyd_1 • gf180mcu_fd_sc_mcu7t5v0__dlyd_2 • gf180mcu_fd_sc_mcu7t5v0__dlyd_4
['asic', 'cells', 'tie']	• gf180mcu_fd_sc_mcu7t5v0__tieh • gf180mcu_fd_sc_mcu7t5v0__tiel
['asic', 'cells', 'hold']	• gf180mcu_fd_sc_mcu7t5v0__dlya_1 • gf180mcu_fd_sc_mcu7t5v0__dlya_2 • gf180mcu_fd_sc_mcu7t5v0__dlya_4 • gf180mcu_fd_sc_mcu7t5v0__dlyb_1 • gf180mcu_fd_sc_mcu7t5v0__dlyb_2 • gf180mcu_fd_sc_mcu7t5v0__dlyb_4 • gf180mcu_fd_sc_mcu7t5v0__dlyc_1 • gf180mcu_fd_sc_mcu7t5v0__dlyc_2 • gf180mcu_fd_sc_mcu7t5v0__dlyc_4 • gf180mcu_fd_sc_mcu7t5v0__dlyd_1 • gf180mcu_fd_sc_mcu7t5v0__dlyd_2 • gf180mcu_fd_sc_mcu7t5v0__dlyd_4

continues on next page

Table 361 – continued from previous page

['asic', 'cells', 'clkbuf']	• gf180mcu_fd_sc_mcu7t5v0__clkbuf_1 • gf180mcu_fd_sc_mcu7t5v0__clkbuf_2 • gf180mcu_fd_sc_mcu7t5v0__clkbuf_3 • gf180mcu_fd_sc_mcu7t5v0__clkbuf_4 • gf180mcu_fd_sc_mcu7t5v0__clkbuf_8 • gf180mcu_fd_sc_mcu7t5v0__clkbuf_12 • gf180mcu_fd_sc_mcu7t5v0__clkbuf_16 • gf180mcu_fd_sc_mcu7t5v0__clkbuf_20
['asic', 'cells', 'dontuse']	*_1
['asic', 'cells', 'filler']	• gf180mcu_fd_sc_mcu7t5v0__fill_1 • gf180mcu_fd_sc_mcu7t5v0__fill_2 • gf180mcu_fd_sc_mcu7t5v0__fill_4 • gf180mcu_fd_sc_mcu7t5v0__fill_8 • gf180mcu_fd_sc_mcu7t5v0__fill_16 • gf180mcu_fd_sc_mcu7t5v0__fill_32 • gf180mcu_fd_sc_mcu7t5v0__fill_64
['asic', 'cells', 'tap']	gf180mcu_fd_sc_mcu7t5v0__filltie
['asic', 'cells', 'endcap']	gf180mcu_fd_sc_mcu7t5v0__endcap
['asic', 'cells', 'antenna']	gf180mcu_fd_sc_mcu7t5v0__antenna

asic, site

Keypath	Value
['asic', 'site', '7t']	GF018hv5v_mcu_sc7

output, slow

Keypath	Value
['output', 'slow', 'nldm']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//nldm/gf180mcu_fd_sc_mcu7t5v0__ss_125C_4v50.lib.gz, lambdapdk
['output', 'slow', 'spice']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//spice/gf180mcu_fd_sc_mcu7t5v0.spice, lambdapdk

output, typical

Keypath	Value
['output', 'typical', 'nldm']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//nldm/gf180mcu_fd_sc_mcu7t5v0__tt_025C_5v00.lib.gz, lambdapdk
['output', 'typical', 'spice']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//spice/gf180mcu_fd_sc_mcu7t5v0.spice, lambdapdk

output, fast

Keypath	Value
['output', 'fast', 'nldm']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//nldm/gf180mcu_fd_sc_mcu7t5v0__ff_n40C_5v50.lib.gz, lambdapdk
['output', 'fast', 'spice']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//spice/gf180mcu_fd_sc_mcu7t5v0.spice, lambdapdk

output, 3LM_1TM_6K

Keypath	Value
['output', '3LM_1TM_6K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '3LM_1TM_6K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk
['output', '3LM_1TM_6K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/3LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

output, 3LM_1TM_9K

Keypath	Value
['output', '3LM_1TM_9K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '3LM_1TM_9K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk

continues on next page

Table 367 – continued from previous page

['output', '3LM_1TM_9K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/3LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

output, 3LM_1TM_11K

Keypath	Value
['output', '3LM_1TM_11K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '3LM_1TM_11K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk
['output', '3LM_1TM_11K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/3LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

output, 3LM_1TM_30K

Keypath	Value
['output', '3LM_1TM_30K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '3LM_1TM_30K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk
['output', '3LM_1TM_30K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/3LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

output, 4LM_1TM_6K

Keypath	Value
['output', '4LM_1TM_6K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '4LM_1TM_6K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk
['output', '4LM_1TM_6K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/4LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

output, 4LM_1TM_9K

Keypath	Value
['output', '4LM_1TM_9K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '4LM_1TM_9K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk
['output', '4LM_1TM_9K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/4LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

output, 4LM_1TM_11K

Keypath	Value
['output', '4LM_1TM_11K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '4LM_1TM_11K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk
['output', '4LM_1TM_11K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/4LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

output, 4LM_1TM_30K

Keypath	Value
['output', '4LM_1TM_30K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '4LM_1TM_30K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk
['output', '4LM_1TM_30K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/4LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

output, 5LM_1TM_9K

Keypath	Value
['output', '5LM_1TM_9K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '5LM_1TM_9K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk
['output', '5LM_1TM_9K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/5LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

output, 5LM_1TM_11K

Keypath	Value
['output', '5LM_1TM_11K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '5LM_1TM_11K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk
['output', '5LM_1TM_11K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/5LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

output, 6LM_1TM_9K

Keypath	Value
['output', '6LM_1TM_9K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lef/gf180mcu_fd_sc_mcu7t5v0.lef, lambdapdk
['output', '6LM_1TM_9K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//cdl/gf180mcu_fd_sc_mcu7t5v0.cdl, lambdapdk
['output', '6LM_1TM_9K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//gds/5LM/gf180mcu_fd_sc_mcu7t5v0.gds.gz, lambdapdk

option

Keypath	Value
['option', 'pdk']	gf180

option, var

Keypath	Value
['option', 'var', 'openroad_place_density']	0.4
['option', 'var', 'openroad_pad_global_place']	1
['option', 'var', 'openroad_pad_detail_place']	0
['option', 'var', 'openroad_macro_place_halo']	• 10 • 10
['option', 'var', 'openroad_macro_place_channel']	• 20.16 • 20.16
['option', 'var', 'openroad_cts_clock_buffer']	gf180mcu_fd_sc_mcu7t5v0__clkbuf_8
['option', 'var', 'openroad_cts_distance_between_buffers']	100
['option', 'var', 'yosys_abc_clock_multiplier']	1000
['option', 'var', 'yosys_abc_constraint_load']	10.084fF
['option', 'var', 'yosys_driver_cell']	gf180mcu_fd_sc_mcu7t5v0__buf_4
['option', 'var', 'yosys_buffer_cell']	gf180mcu_fd_sc_mcu7t5v0__buf_4
['option', 'var', 'yosys_buffer_input']	I
['option', 'var', 'yosys_buffer_output']	Z
['option', 'var', 'yosys_tiehigh_cell']	gf180mcu_fd_sc_mcu7t5v0__tieh
['option', 'var', 'yosys_tiehigh_port']	Z
['option', 'var', 'yosys_tielow_cell']	gf180mcu_fd_sc_mcu7t5v0__tiel
['option', 'var', 'yosys_tielow_port']	ZN
['option', 'var', 'openroad_tiehigh_cell']	gf180mcu_fd_sc_mcu7t5v0__tieh
['option', 'var', 'openroad_tiehigh_port']	Z
['option', 'var', 'openroad_tielow_cell']	gf180mcu_fd_sc_mcu7t5v0__tiel
['option', 'var', 'openroad_tielow_port']	ZN

option, file

Keypath	Value
['option', 'file', 'yosys_techmap']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//techmap/yosys/cells_latch.v, lambdapdk

continues on next page

Table 379 – continued from previous page

['option', 'file', 'yosys_addermap']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//techmap/yosys/cells_adders.v, lambdapdk
['option', 'file', 'openroad_pdngen']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//apr/openroad/pdngen.tcl, lambdapdk
['option', 'file', 'openroad_global_connect']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//apr/openroad/global_connect.tcl, lambdapdk
['option', 'file', 'openroad_tapcells']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//apr/openroad/tapcell.tcl, lambdapdk

Associated PDK: None

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

lambdalib_gf180mcu_fd_sc_mcu7t5v0

option

Keypath	Value
['option', 'ydir']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu7t5v0//lambda, lambdapdk

Associated PDK: *gf180*

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

gf180mcu_fd_sc_mcu9t5v0

asic

Keypath	Value
['asic', 'libarch']	9t

asic, cells

Keypath	Value
['asic', 'cells', 'decap']	• gf180mcu_fd_sc_mcu9t5v0__fillcap_4 • gf180mcu_fd_sc_mcu9t5v0__fillcap_8 • gf180mcu_fd_sc_mcu9t5v0__fillcap_16 • gf180mcu_fd_sc_mcu9t5v0__fillcap_32 • gf180mcu_fd_sc_mcu9t5v0__fillcap_64
['asic', 'cells', 'delay']	• gf180mcu_fd_sc_mcu9t5v0__dlya_1 • gf180mcu_fd_sc_mcu9t5v0__dlya_2 • gf180mcu_fd_sc_mcu9t5v0__dlya_4 • gf180mcu_fd_sc_mcu9t5v0__dlyb_1 • gf180mcu_fd_sc_mcu9t5v0__dlyb_2 • gf180mcu_fd_sc_mcu9t5v0__dlyb_4 • gf180mcu_fd_sc_mcu9t5v0__dlyc_1 • gf180mcu_fd_sc_mcu9t5v0__dlyc_2 • gf180mcu_fd_sc_mcu9t5v0__dlyc_4 • gf180mcu_fd_sc_mcu9t5v0__dlyd_1 • gf180mcu_fd_sc_mcu9t5v0__dlyd_2 • gf180mcu_fd_sc_mcu9t5v0__dlyd_4
['asic', 'cells', 'tie']	• gf180mcu_fd_sc_mcu9t5v0__tieh • gf180mcu_fd_sc_mcu9t5v0__tiel
['asic', 'cells', 'hold']	• gf180mcu_fd_sc_mcu9t5v0__dlya_1 • gf180mcu_fd_sc_mcu9t5v0__dlya_2 • gf180mcu_fd_sc_mcu9t5v0__dlya_4 • gf180mcu_fd_sc_mcu9t5v0__dlyb_1 • gf180mcu_fd_sc_mcu9t5v0__dlyb_2 • gf180mcu_fd_sc_mcu9t5v0__dlyb_4 • gf180mcu_fd_sc_mcu9t5v0__dlyc_1 • gf180mcu_fd_sc_mcu9t5v0__dlyc_2 • gf180mcu_fd_sc_mcu9t5v0__dlyc_4 • gf180mcu_fd_sc_mcu9t5v0__dlyd_1 • gf180mcu_fd_sc_mcu9t5v0__dlyd_2 • gf180mcu_fd_sc_mcu9t5v0__dlyd_4

continues on next page

Table 384 – continued from previous page

['asic', 'cells', 'clkbuf']	• gf180mcu_fd_sc_mcu9t5v0__clkbuf_1 • gf180mcu_fd_sc_mcu9t5v0__clkbuf_2 • gf180mcu_fd_sc_mcu9t5v0__clkbuf_3 • gf180mcu_fd_sc_mcu9t5v0__clkbuf_4 • gf180mcu_fd_sc_mcu9t5v0__clkbuf_8 • gf180mcu_fd_sc_mcu9t5v0__clkbuf_12 • gf180mcu_fd_sc_mcu9t5v0__clkbuf_16 • gf180mcu_fd_sc_mcu9t5v0__clkbuf_20
['asic', 'cells', 'dontuse']	*_1
['asic', 'cells', 'filler']	• gf180mcu_fd_sc_mcu9t5v0__fill_1 • gf180mcu_fd_sc_mcu9t5v0__fill_2 • gf180mcu_fd_sc_mcu9t5v0__fill_4 • gf180mcu_fd_sc_mcu9t5v0__fill_8 • gf180mcu_fd_sc_mcu9t5v0__fill_16 • gf180mcu_fd_sc_mcu9t5v0__fill_32 • gf180mcu_fd_sc_mcu9t5v0__fill_64
['asic', 'cells', 'tap']	gf180mcu_fd_sc_mcu9t5v0__filltie
['asic', 'cells', 'endcap']	gf180mcu_fd_sc_mcu9t5v0__endcap
['asic', 'cells', 'antenna']	gf180mcu_fd_sc_mcu9t5v0__antenna

asic, site

Keypath	Value
['asic', 'site', '9t']	GF018hv5v_green_sc9

output, slow

Keypath	Value
['output', 'slow', 'nldm']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//nldm/gf180mcu_fd_sc_mcu9t5v0__ss_125C_4v50.lib.gz, lambdapdk
['output', 'slow', 'spice']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//spice/gf180mcu_fd_sc_mcu9t5v0.spice, lambdapdk

output, typical

Keypath	Value
['output', 'typical', 'nldm']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//nldm/gf180mcu_fd_sc_mcu9t5v0__tt_025C_5v00.lib.gz, lambdapdk
['output', 'typical', 'spice']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//spice/gf180mcu_fd_sc_mcu9t5v0.spice, lambdapdk

output, fast

Keypath	Value
['output', 'fast', 'nldm']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//nldm/gf180mcu_fd_sc_mcu9t5v0__ff_n40C_5v50.lib.gz, lambdapdk
['output', 'fast', 'spice']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//spice/gf180mcu_fd_sc_mcu9t5v0.spice, lambdapdk

output, 3LM_1TM_6K

Keypath	Value
['output', '3LM_1TM_6K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '3LM_1TM_6K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk
['output', '3LM_1TM_6K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/3LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

output, 3LM_1TM_9K

Keypath	Value
['output', '3LM_1TM_9K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '3LM_1TM_9K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk

continues on next page

Table 390 – continued from previous page

['output', '3LM_1TM_9K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/3LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

output, 3LM_1TM_11K

Keypath	Value
['output', '3LM_1TM_11K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '3LM_1TM_11K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk
['output', '3LM_1TM_11K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/3LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

output, 3LM_1TM_30K

Keypath	Value
['output', '3LM_1TM_30K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '3LM_1TM_30K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk
['output', '3LM_1TM_30K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/3LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

output, 4LM_1TM_6K

Keypath	Value
['output', '4LM_1TM_6K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '4LM_1TM_6K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk
['output', '4LM_1TM_6K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/4LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

output, 4LM_1TM_9K

Keypath	Value
['output', '4LM_1TM_9K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '4LM_1TM_9K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk
['output', '4LM_1TM_9K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/4LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

output, 4LM_1TM_11K

Keypath	Value
['output', '4LM_1TM_11K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '4LM_1TM_11K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk
['output', '4LM_1TM_11K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/4LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

output, 4LM_1TM_30K

Keypath	Value
['output', '4LM_1TM_30K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '4LM_1TM_30K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk
['output', '4LM_1TM_30K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/4LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

output, 5LM_1TM_9K

Keypath	Value
['output', '5LM_1TM_9K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '5LM_1TM_9K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk
['output', '5LM_1TM_9K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/5LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

output, 5LM_1TM_11K

Keypath	Value
['output', '5LM_1TM_11K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '5LM_1TM_11K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk
['output', '5LM_1TM_11K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/5LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

output, 6LM_1TM_9K

Keypath	Value
['output', '6LM_1TM_9K', 'lef']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lef/gf180mcu_fd_sc_mcu9t5v0.lef, lambdapdk
['output', '6LM_1TM_9K', 'cdl']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//cdl/gf180mcu_fd_sc_mcu9t5v0.cdl, lambdapdk
['output', '6LM_1TM_9K', 'gds']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//gds/5LM/gf180mcu_fd_sc_mcu9t5v0.gds.gz, lambdapdk

option

Keypath	Value
['option', 'pdk']	gf180

option, var

Keypath	Value
['option', 'var', 'openroad_place_density']	0.4
['option', 'var', 'openroad_pad_global_place']	1
['option', 'var', 'openroad_pad_detail_place']	0
['option', 'var', 'openroad_macro_place_halo']	• 10 • 10
['option', 'var', 'openroad_macro_place_channel']	• 20.16 • 20.16
['option', 'var', 'openroad_cts_clock_buffer']	gf180mcu_fd_sc_mcu9t5v0__clkbuf_8
['option', 'var', 'openroad_cts_distance_between_buffers']	100
['option', 'var', 'yosys_abc_clock_multiplier']	1000
['option', 'var', 'yosys_abc_constraint_load']	14.692fF
['option', 'var', 'yosys_driver_cell']	gf180mcu_fd_sc_mcu9t5v0__buf_4
['option', 'var', 'yosys_buffer_cell']	gf180mcu_fd_sc_mcu9t5v0__buf_4
['option', 'var', 'yosys_buffer_input']	I
['option', 'var', 'yosys_buffer_output']	Z
['option', 'var', 'yosys_tiehigh_cell']	gf180mcu_fd_sc_mcu9t5v0__tieh
['option', 'var', 'yosys_tiehigh_port']	Z
['option', 'var', 'yosys_tielow_cell']	gf180mcu_fd_sc_mcu9t5v0__tiel
['option', 'var', 'yosys_tielow_port']	ZN
['option', 'var', 'openroad_tiehigh_cell']	gf180mcu_fd_sc_mcu9t5v0__tieh
['option', 'var', 'openroad_tiehigh_port']	Z
['option', 'var', 'openroad_tielow_cell']	gf180mcu_fd_sc_mcu9t5v0__tiel
['option', 'var', 'openroad_tielow_port']	ZN

option, file

Keypath	Value
['option', 'file', 'yosys_techmap']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//techmap/yosys/cells_latch.v, lambdapdk

continues on next page

Table 402 – continued from previous page

['option', 'file', 'yosys_addermap']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//techmap/yosys/cells_adders.v, lambdapdk
['option', 'file', 'openroad_pdngen']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//apr/openroad/pdngen.tcl, lambdapdk
['option', 'file', 'openroad_global_connect']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//apr/openroad/global_connect.tcl, lambdapdk
['option', 'file', 'openroad_tapcells']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//apr/openroad/tapcell.tcl, lambdapdk

Associated PDK: None

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

lambdalib_gf180mcu_fd_sc_mcu9t5v0

option

Keypath	Value
['option', 'ydir']	lambdapdk/gf180/libs/gf180mcu_fd_sc_mcu9t5v0//lambda, lambdapdk

3.5.3 nangate45

Nangate open standard cell library for FreePDK45.

Setup file: nangate45.py

Associated PDK: *freepdk45*

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

nangate45

asic

Keypath	Value
['asic', 'libarch']	10t

asic, cells

Keypath	Value
['asic', 'cells', 'tie']	• LOGIC1_X1 • LOGIC0_X1
['asic', 'cells', 'hold']	BUF_X1
['asic', 'cells', 'clkbuf']	BUF_X4
['asic', 'cells', 'dontuse']	OAI211_X1
['asic', 'cells', 'filler']	• FILLCELL_X1 • FILLCELL_X2 • FILLCELL_X4 • FILLCELL_X8 • FILLCELL_X16 • FILLCELL_X32
['asic', 'cells', 'tap']	TAPCELL_X1
['asic', 'cells', 'endcap']	TAPCELL_X1

asic, site

Keypath	Value
['asic', 'site', '10t']	FreePDK45_38x28_10R_NP_162NW_34O

output, typical

Keypath	Value
['output', 'typical', 'nldm']	lambdapdk/freepdk45/libs/nangate45/nldm/NangateOpenCellLibrary_typical.lib, lambdapdk

output, 10M

Keypath	Value
['output', '10M', 'lef']	lambdapdk/freepdk45/libs/nangate45/lef/NangateOpenCellLibrary.macro.mod.lef, lambdapdk
['output', '10M', 'gds']	lambdapdk/freepdk45/libs/nangate45/gds/NangateOpenCellLibrary.gds, lambdapdk
['output', '10M', 'cdl']	lambdapdk/freepdk45/libs/nangate45/cdl/NangateOpenCellLibrary.cdl, lambdapdk

option

Keypath	Value
['option', 'pdk']	freepdk45
['option', 'stackup']	10M

option, var

Keypath	Value
['option', 'var', 'openroad_place_density']	0.35
['option', 'var', 'openroad_pad_global_place']	0
['option', 'var', 'openroad_pad_detail_place']	0
['option', 'var', 'openroad_macro_place_halo']	• 22.4 • 15.12

continues on next page

Table 412 – continued from previous page

['option', 'var', 'openroad_macro_place_channel']	• 18.8 • 19.95
['option', 'var', 'yosys_abc_clock_multiplier']	1000
['option', 'var', 'yosys_abc_constraint_load']	3.898fF
['option', 'var', 'yosys_driver_cell']	BUF_X4
['option', 'var', 'yosys_buffer_cell']	BUF_X1
['option', 'var', 'yosys_buffer_input']	A
['option', 'var', 'yosys_buffer_output']	Z
['option', 'var', 'yosys_tiehigh_cell']	LOGIC1_X1
['option', 'var', 'yosys_tiehigh_port']	Z
['option', 'var', 'yosys_tielow_cell']	LOGIC0_X1
['option', 'var', 'yosys_tielow_port']	Z
['option', 'var', 'openroad_tiehigh_cell']	LOGIC1_X1
['option', 'var', 'openroad_tiehigh_port']	Z
['option', 'var', 'openroad_tielow_cell']	LOGIC0_X1
['option', 'var', 'openroad_tielow_port']	Z

option, file

Keypath	Value
['option', 'file', 'yosys_techmap']	lambdapdk/freepdk45/libs/nangate45/techmap/yosys/cells_latch.v, lambdapdk
['option', 'file', 'yosys_addermap']	lambdapdk/freepdk45/libs/nangate45/techmap/yosys/cells_adders.v, lambdapdk
['option', 'file', 'openroad_tapcells']	lambdapdk/freepdk45/libs/nangate45/apr/openroad/tapcell.tcl, lambdapdk
['option', 'file', 'openroad_pdngen']	lambdapdk/freepdk45/libs/nangate45/apr/openroad/pdngen.tcl, lambdapdk
['option', 'file', 'openroad_global_connect']	lambdapdk/freepdk45/libs/nangate45/apr/openroad/global_connect.tcl, lambdapdk

Associated PDK: None

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

lambdalib_nangate45

option

Keypath	Value
['option', 'ydir']	lambdapdk/freepdk45/libs/nangate45/lambda, lambdapdk

3.5.4 sky130hd

Skywater130 standard cell library.

Setup file: sky130hd.py

Associated PDK: *skywater130*

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

sky130hd

asic

Keypath	Value
['asic', 'libarch']	hd

asic, cells

Keypath	Value
['asic', 'cells', 'tie']	sky130_fd_sc_hd__conb_1

continues on next page

Table 418 – continued from previous page

['asic', 'cells', 'hold']	• sky130_fd_sc_hd__buf_1 • sky130_fd_sc_hd__buf_2 • sky130_fd_sc_hd__buf_4 • sky130_fd_sc_hd__buf_6 • sky130_fd_sc_hd__buf_8 • sky130_fd_sc_hd__buf_12 • sky130_fd_sc_hd__buf_16
['asic', 'cells', 'clkbuf']	• sky130_fd_sc_hd__clkbuf_1 • sky130_fd_sc_hd__clkbuf_2 • sky130_fd_sc_hd__clkbuf_4 • sky130_fd_sc_hd__clkbuf_6 • sky130_fd_sc_hd__clkbuf_8 • sky130_fd_sc_hd__clkbuf_12 • sky130_fd_sc_hd__clkbuf_16

continues on next page

Table 418 – continued from previous page

['asic', 'cells', 'dontuse']	• sky130_fd_sc_hd__probe_p_8 • sky130_fd_sc_hd__probec_p_8 • sky130_fd_sc_hd__lpflow_bleeder_1 • sky130_fd_sc_hd__lpflow_clkbufkapwr_1 • sky130_fd_sc_hd__lpflow_clkbufkapwr_16 • sky130_fd_sc_hd__lpflow_clkbufkapwr_2 • sky130_fd_sc_hd__lpflow_clkbufkapwr_4 • sky130_fd_sc_hd__lpflow_clkbufkapwr_8 • sky130_fd_sc_hd__lpflow_clkinvkapwr_1 • sky130_fd_sc_hd__lpflow_clkinvkapwr_16 • sky130_fd_sc_hd__lpflow_clkinvkapwr_2 • sky130_fd_sc_hd__lpflow_clkinvkapwr_4 • sky130_fd_sc_hd__lpflow_clkinvkapwr_8 • sky130_fd_sc_hd__lpflow_decapkapwr_12 • sky130_fd_sc_hd__lpflow_decapkapwr_3 • sky130_fd_sc_hd__lpflow_decapkapwr_4 • sky130_fd_sc_hd__lpflow_decapkapwr_6 • sky130_fd_sc_hd__lpflow_decapkapwr_8 • sky130_fd_sc_hd__lpflow_inputiso0n_1 • sky130_fd_sc_hd__lpflow_inputiso0p_1 • sky130_fd_sc_hd__lpflow_inputiso1n_1 • sky130_fd_sc_hd__lpflow_inputiso1p_1 • sky130_fd_sc_hd__lpflow_inputisolatch_1 • sky130_fd_sc_hd__lpflow_isobufsrc_1 • sky130_fd_sc_hd__lpflow_isobufsrc_16 • sky130_fd_sc_hd__lpflow_isobufsrc_2 • sky130_fd_sc_hd__lpflow_isobufsrc_4 • sky130_fd_sc_hd__lpflow_isobufsrc_8 • sky130_fd_sc_hd__lpflow_isobufsrckapwr_16 • sky130_fd_sc_hd__lpflow_lsbuf_lh_hl_isowell_tap_1 • sky130_fd_sc_hd__lpflow_lsbuf_lh_hl_isowell_tap_2 • sky130_fd_sc_hd__lpflow_lsbuf_lh_hl_isowell_tap_4 • sky130_fd_sc_hd__lpflow_lsbuf_lh_isowell_4 • sky130_fd_sc_hd__lpflow_lsbuf_lh_isowell_tap_1 • sky130_fd_sc_hd__lpflow_lsbuf_lh_isowell_tap_2 • sky130_fd_sc_hd__lpflow_lsbuf_lh_isowell_tap_4

continues on next page

Table 418 – continued from previous page

['asic', 'cells', 'filler']	• sky130_fd_sc_hd__fill_1 • sky130_fd_sc_hd__fill_2 • sky130_fd_sc_hd__fill_4 • sky130_fd_sc_hd__fill_8
['asic', 'cells', 'tap']	sky130_fd_sc_hd__tapvpwrvgnd_1
['asic', 'cells', 'endcap']	sky130_fd_sc_hd__decap_4
['asic', 'cells', 'antenna']	sky130_fd_sc_hd__diode_2

asic, site

Keypath	Value
['asic', 'site', 'hd']	• unithd • unithddbl

output, slow

Keypath	Value
['output', 'slow', 'nldm']	lambdapdk/sky130/libs/sky130hd/nldm/sky130_fd_sc_hd__ss_n40C_1v40.lib.gz, lambdapdk

output, typical

Keypath	Value
['output', 'typical', 'nldm']	lambdapdk/sky130/libs/sky130hd/nldm/sky130_fd_sc_hd__tt_025C_1v80.lib.gz, lambdapdk

output, fast

Keypath	Value
['output', 'fast', 'nldm']	lambdapdk/sky130/libs/sky130hd/nldm/sky130_fd_sc_hd__ff_100C_1v95.lib.gz, lambdapdk

output, 5M1LI

Keypath	Value
['output', '5M1LI', 'lef']	lambdapdk/sky130/libs/sky130hd/lef/sky130_fd_sc_hd_merged.lef, lambdapdk
['output', '5M1LI', 'gds']	lambdapdk/sky130/libs/sky130hd/gds/sky130_fd_sc_hd.gds, lambdapdk
['output', '5M1LI', 'cdl']	lambdapdk/sky130/libs/sky130hd/cdl/sky130_fd_sc_hd.cdl, lambdapdk

output, rtl

Keypath	Value
['output', 'rtl', 'verilog']	• lambdapdk/sky130/libs/sky130hd/verilog/sky130_fd_sc_hd.v, lambdapdk • lambdapdk/sky130/libs/sky130hd/verilog/primitives.v, lambdapdk

option

Keypath	Value
['option', 'pdk']	skywater130

option, var

Keypath	Value
['option', 'var', 'openroad_place_density']	0.6
['option', 'var', 'openroad_pad_global_place']	1
['option', 'var', 'openroad_pad_detail_place']	0
['option', 'var', 'openroad_macro_place_halo']	• 40 • 40
['option', 'var', 'openroad_macro_place_channel']	• 80 • 80
['option', 'var', 'openroad_cts_clock_buffer']	sky130_fd_sc_hd__clkbuf_4

continues on next page

Table 426 – continued from previous page

['option', 'var', 'yosys_abc_clock_multiplier']	1000
['option', 'var', 'yosys_abc_constraint_load']	7.488fF
['option', 'var', 'yosys_driver_cell']	sky130_fd_sc_hd__buf_4
['option', 'var', 'yosys_buffer_cell']	sky130_fd_sc_hd__buf_4
['option', 'var', 'yosys_buffer_input']	A
['option', 'var', 'yosys_buffer_output']	X
['option', 'var', 'yosys_tiehigh_cell']	sky130_fd_sc_hd__conb_1
['option', 'var', 'yosys_tiehigh_port']	HI
['option', 'var', 'yosys_tielow_cell']	sky130_fd_sc_hd__conb_1
['option', 'var', 'yosys_tielow_port']	LO
['option', 'var', 'openroad_tiehigh_cell']	sky130_fd_sc_hd__conb_1
['option', 'var', 'openroad_tiehigh_port']	HI
['option', 'var', 'openroad_tielow_cell']	sky130_fd_sc_hd__conb_1
['option', 'var', 'openroad_tielow_port']	LO

option, file

Keypath	Value
['option', 'file', 'yosys_techmap']	lambdapdk/sky130/libs/sky130hd/techmap/yosys/cells_latch.v, lambdapdk
['option', 'file', 'yosys_addermap']	lambdapdk/sky130/libs/sky130hd/techmap/yosys/cells_adders.v, lambdapdk
['option', 'file', 'openroad_pdngen']	lambdapdk/sky130/libs/sky130hd/apr/openroad/pdngen.tcl, lambdapdk
['option', 'file', 'openroad_global_connect']	lambdapdk/sky130/libs/sky130hd/apr/openroad/global_connect.tcl, lambdapdk
['option', 'file', 'openroad_tapcells']	lambdapdk/sky130/libs/sky130hd/apr/openroad/tapcell.tcl, lambdapdk

Associated PDK: None

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

lambdalib_sky130hd

option

Keypath	Value
['option', 'ydir']	lambdapdk/sky130/libs/sky130hd/lambda, lambdapdk

Associated PDK: *skywater130*

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

sky130hdll

asic

Keypath	Value
['asic', 'libarch']	hdll

asic, cells

Keypath	Value
['asic', 'cells', 'tie']	sky130_fd_sc_hdll__conb_1
['asic', 'cells', 'hold']	• sky130_fd_sc_hdll__buf_1 • sky130_fd_sc_hdll__buf_2 • sky130_fd_sc_hdll__buf_4 • sky130_fd_sc_hdll__buf_6 • sky130_fd_sc_hdll__buf_8 • sky130_fd_sc_hdll__buf_12 • sky130_fd_sc_hdll__buf_16

continues on next page

Table 432 – continued from previous page

['asic', 'cells', 'clkbuf']	• sky130_fd_sc_hdll__clkbuf_1 • sky130_fd_sc_hdll__clkbuf_2 • sky130_fd_sc_hdll__clkbuf_4 • sky130_fd_sc_hdll__clkbuf_6 • sky130_fd_sc_hdll__clkbuf_8 • sky130_fd_sc_hdll__clkbuf_12 • sky130_fd_sc_hdll__clkbuf_16
['asic', 'cells', 'dontuse']	• sky130_fd_sc_hdll__probe_p_8 • sky130_fd_sc_hdll__probec_p_8 • sky130_fd_sc_hdll__inputiso0p_1 • sky130_fd_sc_hdll__inputiso0n_1 • sky130_fd_sc_hdll__inputiso1p_1 • sky130_fd_sc_hdll__inputiso1n_1 • sky130_fd_sc_hdll__isobufsrc_1 • sky130_fd_sc_hdll__isobufsrc_2 • sky130_fd_sc_hdll__isobufsrc_4 • sky130_fd_sc_hdll__isobufsrc_8 • sky130_fd_sc_hdll__isobufsrc_16
['asic', 'cells', 'filler']	• sky130_fd_sc_hdll__fill_1 • sky130_fd_sc_hdll__fill_2 • sky130_fd_sc_hdll__fill_4 • sky130_fd_sc_hdll__fill_8
['asic', 'cells', 'tap']	sky130_fd_sc_hdll__tapvpwrvgnd_1
['asic', 'cells', 'endcap']	sky130_fd_sc_hdll__decap_4
['asic', 'cells', 'antenna']	sky130_fd_sc_hdll__diode_2

asic, site

Keypath	Value
['asic', 'site', 'hdll']	• unithd • unithddbl

output, slow

Keypath	Value
['output', 'slow', 'nldm']	lambdapdk/sky130/libs/sky130hdll/nldm/sky130_fd_sc_hdll__ss_n40C_1v44.lib.gz, lambdapdk

output, typical

Keypath	Value
['output', 'typical', 'nldm']	lambdapdk/sky130/libs/sky130hdll/nldm/sky130_fd_sc_hdll__tt_025C_1v80.lib.gz, lambdapdk

output, fast

Keypath	Value
['output', 'fast', 'nldm']	lambdapdk/sky130/libs/sky130hdll/nldm/sky130_fd_sc_hdll__ff_100C_1v95.lib.gz, lambdapdk

output, 5M1LI

Keypath	Value
['output', '5M1LI', 'lef']	lambdapdk/sky130/libs/sky130hdll/lef/sky130_fd_sc_hdll_merged.lef, lambdapdk
['output', '5M1LI', 'gds']	lambdapdk/sky130/libs/sky130hdll/gds/sky130_fd_sc_hdll.gds, lambdapdk
['output', '5M1LI', 'cdl']	lambdapdk/sky130/libs/sky130hdll/cdl/sky130_fd_sc_hdll.cdl, lambdapdk

output, rtl

Keypath	Value
['output', 'rtl', 'verilog']	• lambdapdk/sky130/libs/sky130hdll/verilog/sky130_fd_sc_hdll.v, lambdapdk • lambdapdk/sky130/libs/sky130hdll/verilog/primitives.v, lambdapdk

option

Keypath	Value
['option', 'pdk']	skywater130

option, var

Keypath	Value
['option', 'var', 'openroad_place_density']	0.6
['option', 'var', 'openroad_pad_global_place']	1
['option', 'var', 'openroad_pad_detail_place']	0
['option', 'var', 'openroad_macro_place_halo']	• 40 • 40
['option', 'var', 'openroad_macro_place_channel']	• 80 • 80
['option', 'var', 'openroad_cts_clock_buffer']	sky130_fd_sc_hdll__clkbuf_4
['option', 'var', 'yosys_abc_clock_multiplier']	1000
['option', 'var', 'yosys_abc_constraint_load']	7.644fF
['option', 'var', 'yosys_driver_cell']	sky130_fd_sc_hdll__buf_4
['option', 'var', 'yosys_buffer_cell']	sky130_fd_sc_hdll__buf_4
['option', 'var', 'yosys_buffer_input']	A
['option', 'var', 'yosys_buffer_output']	X
['option', 'var', 'yosys_tiehigh_cell']	sky130_fd_sc_hdll__conb_1
['option', 'var', 'yosys_tiehigh_port']	HI
['option', 'var', 'yosys_tielow_cell']	sky130_fd_sc_hdll__conb_1
['option', 'var', 'yosys_tielow_port']	LO
['option', 'var', 'openroad_tiehigh_cell']	sky130_fd_sc_hdll__conb_1
['option', 'var', 'openroad_tiehigh_port']	HI
['option', 'var', 'openroad_tielow_cell']	sky130_fd_sc_hdll__conb_1
['option', 'var', 'openroad_tielow_port']	LO

option, file

Keypath	Value
['option', 'file', 'yosys_techmap']	lambdapdk/sky130/libs/sky130hdll/techmap/yosys/cells_latch.v, lambdapdk
['option', 'file', 'openroad_pdngen']	lambdapdk/sky130/libs/sky130hdll/apr/openroad/pdngen.tcl, lambdapdk
['option', 'file', 'openroad_global_connect']	lambdapdk/sky130/libs/sky130hdll/apr/openroad/global_connect.tcl, lambdapdk

continues on next page

Table 441 – continued from previous page

['option', 'file', 'openroad_tapcells']	lambdapdk/sky130/libs/sky130hdll/apr/openroad/tapcell.tcl, lambdapdk

Associated PDK: None

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

lambdalib_sky130hdll

option

Keypath	Value
['option', 'ydir']	lambdapdk/sky130/libs/sky130hdll/lambda, lambdapdk

3.5.5 sky130io

Skywater130 I/O library.

Setup file: sky130io.py

Associated PDK: *skywater130*

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

sky130io

asic, cells

Keypath	Value
['asic', 'cells', 'filler']	• sky130_ef_io__com_bus_slice_1um • sky130_ef_io__com_bus_slice_5um • sky130_ef_io__com_bus_slice_10um • sky130_ef_io__com_bus_slice_20um

output, slow

Keypath	Value
['output', 'slow', 'nldm']	lambdapdk/sky130/libs/sky130io/nldm/sky130_dummy_io.lib, lambdapdk

output, typical

Keypath	Value
['output', 'typical', 'nldm']	lambdapdk/sky130/libs/sky130io/nldm/sky130_dummy_io.lib, lambdapdk

output, fast

Keypath	Value
['output', 'fast', 'nldm']	lambdapdk/sky130/libs/sky130io/nldm/sky130_dummy_io.lib, lambdapdk

output, 5M1LI

Keypath	Value
['output', '5M1LI', 'lef']	lambdapdk/sky130/libs/sky130io/lef/sky130_ef_io.lef, lambdapdk

continues on next page

Table 449 – continued from previous page

['output', '5M1LI', 'gds']	• lambdapdk/sky130/libs/sky130io/gds/sky130_ef_io.gds, lambdapdk • lambdapdk/sky130/libs/sky130io/gds/sky130_fd_io.gds, lambdapdk • lambdapdk/sky130/libs/sky130io/gds/sky130_ef_io__gpiov2_pad_wrapped.gds, lambdapdk

output, blackbox

Keypath	Value
['output', 'blackbox', 'verilog']	lambdapdk/sky130/libs/sky130io/bb/sky130_io.blackbox.v, lambdapdk

option

Keypath	Value
['option', 'pdk']	skywater130

Associated PDK: None

Data sources

Package	Specifications
lambdapdk	• Path: https://github.com/siliconcompiler/lambdapdk/archive/refs/tags/ • Reference: v0.1.19

lambdalib_sky130io

option

Keypath	Value
['option', 'ydir']	lambdapdk/sky130/libs/sky130io/lambda, lambdapdk

3.6 Pre-Defined Checklists

The following are examples are pre-built checklists that come with SiliconCompiler which you can use for your own builds.

3.6.1 oh_tapeout

Subset of OH! library tapeout checklist.

https://github.com/aolofsson/oh/blob/master/docs/tapeout_checklist.md

Setup file: oh_tapeout.py

Configuration

drc_clean

Keypath	Value
['checklist', 'oh_tapeout', 'drc_clean', 'description']	Is block DRC clean?
['checklist', 'oh_tapeout', 'drc_clean', 'criteria']	drvs==0

lvs_clean

Keypath	Value
['checklist', 'oh_tapeout', 'lvs_clean', 'description']	Is block LVS clean?
['checklist', 'oh_tapeout', 'lvs_clean', 'criteria']	drvs==0

setup_time

Keypath	Value
['checklist', 'oh_tapeout', 'setup_time', 'description']	Setup time met?
['checklist', 'oh_tapeout', 'setup_time', 'criteria']	setupslack>=0

errors_warnings

Keypath	Value
['checklist', 'oh_tapeout', 'errors_warnings', 'description']	Are all EDA warnings/errors acceptable?
['checklist', 'oh_tapeout', 'errors_warnings', 'criteria']	• errors==0 • warnings==0

spec

Keypath	Value
['checklist', 'oh_tapeout', 'spec', 'description']	Is there a written specification?

3.7 Schema

3.7.1 Keywords

default
Reserved SiliconCompiler schema key that can be replaced by any legal string.

3.7.2 Parameter Fields

copy
Whether to copy files into build directory, applies to files only

enum
List of strings containing the set of legal values for this parameter.

example
List of two strings, the first string containing an example for specifying the parameter using a command line switch, and a second string for setting the value using the core Python API. The examples can be pruned/filtered before the schema is dumped into a JSON file.

hashalgo
Hashing algorithm used to calculate filehash value.

help
Complete parameter help doc string. The help string serves as ground truth for describing the parameter functionality and should be used for long help descriptions in command line interface programs and for automated schema document generation. The long help can be pruned/filtered before the schema is dumped into a JSON file.

lock
Boolean value dictating whether the parameter can be modified by the set/get/add core API methods. A value of True specifiers that the parameter is locked and cannot be modified. Attempts to write to to a locked parameter shall result in an exception/error that blocks compilation progress.

node
Dictionary containing fields whose values may vary on a per-step/index basis. Sub-fields are described in *Per-node Parameter Fields*

notes
User entered 'notes'/'disclaimers' about value being set.

pernode
Enables/disables setting of value on a per node basis. Allowed values are 'never', 'option', or 'required'.

require
String that specifies scenarios, conditions, and modes for which the parameter must return a non-empty value. Valid requirement keywords include 'all' and 'fpga'/'asic'. The 'all' keyword specifies that the parameter must always have a non-empty value before running a flow. The 'fpga'/'asic' keyword specifies that that the parameter must have a non-empty value when the respective mode is being executed. All Boolean values have a valid True/False default value and a requirement of 'all'. The vast majority of schema parameters have requirements of None and empty values which can be overridden by the user based on need.

scope
Scope of parameter in schema

switch
String that specifies the equivalent switch to use in command line interfaces. The switch string must start with a '-' and cannot contain spaces.

shorthelp
Short help string to be used in cases where brevity matters. Use cases include JSON dictionary dumps and command line interface help functions.

type
The parameter type. Supported types include Python compatible types ('int', 'float', 'str', 'enum', and 'bool') and two custom file types ('file' and 'dir'). The 'file' and 'dir' type specify that the parameter is a 'regular' file or directory as described by Posix. All types can be specified as a Python compatible list type by enclosing the type value in brackets. (ie. [str] specifies that the parameter is a list of strings). Types can also be specified as tuples, using the Python-like parentheses syntax (eg. [(float,float)] specifies a list of 2-float tuples). Input arguments and return values of the set/get/add core methods are encoded as native Python types. When exporting the manifest to JSON, values are converted to the equivalent JSON type. Most types have a straightforward mapping, but note that values of "None" get mapped to "null", and both tuples and lists get mapped to arrays. Tuple-type parameters have their values normalized back into tuple form when a JSON manifest is read in.

unit
Implied unit for parameter value.

3.7.3 Per-node Parameter Fields

The following fields are specified inside the `node` dictionary on a per-step/index basis. Default values for each field are stored under the special keys `"default", "default"`, and global values are specified under the special keys `"global", "global"`.

author
File author. The author string records the person/entity that authored/created each item in the list of files within 'value' parameter field. The 'author' field can be used to validate the provenance of the data used for compilation.

date
String containing the data stamp of each item in the list of files within 'value' parameter field. The 'date' field can be used to validate the provenance of the data used for compilation.

filehash
Calculated file hash value for each file in the 'value' field of the parameter.

signature
String recording a unique machine calculated string for each item in the list of files within 'value' parameter field. The 'signature' field can be used to validate the provenance of the data used for compilation.

value
Parameter value

3.7.4 Parameters

arg

index

Description	ARG: Index argument
Type	str
Default Value	None
CLI Switch	• `-arg_index <str>`
Example (CLI)	`-arg_index 0`
Example (API)	`chip.set('arg', 'index', '0')`

Dynamic parameter passed in by the SC runtime as an argument to a runtime task. The parameter enables configuration code (usually TCL) to use control flow that depend on the current 'index'. The parameter is used the run() function and is not intended for external use.

step

Description	ARG: Step argument
Type	str
Default Value	None
CLI Switch	• `-arg_step <str>`
Example (CLI)	`-arg_step 'route'`
Example (API)	`chip.set('arg', 'step', 'route')`

Dynamic parameter passed in by the SC runtime as an argument to a runtime task. The parameter enables configuration code (usually TCL) to use control flow that depend on the current 'step'. The parameter is used the run() function and is not intended for external use.

asic

cells

antenna

Description	ASIC: antenna cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_antenna '<str>'`
Example (CLI)	`-asic_cells_antenna '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'antenna', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

clkbuf

Description	ASIC: clkbuf cell list
Type	[str]
Per step/index	optional
Default Value	[]

continues on next page

Table 462 – continued from previous page

CLI Switch	• `-asic_cells_clkbuf '<str>'`
Example (CLI)	`-asic_cells_clkbuf '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'clkbuf', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

clkdelay

Description	ASIC: clkdelay cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_clkdelay '<str>'`
Example (CLI)	`-asic_cells_clkdelay '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'clkdelay', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

clkgate

Description	ASIC: clkgate cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_clkgate '<str>'`
Example (CLI)	`-asic_cells_clkgate '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'clkgate', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

clkicg

Description	ASIC: clkicg cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_clkicg '<str>'`
Example (CLI)	`-asic_cells_clkicg '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'clkicg', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

clkinv

Description	ASIC: clkinv cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_clkinv '<str>'`
Example (CLI)	`-asic_cells_clkinv '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'clkinv', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

clklogic

Description	ASIC: clklogic cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_clklogic '<str>'`
Example (CLI)	`-asic_cells_clklogic '*eco*'`

continues on next page

Table 467 – continued from previous page

Example (API)	`chip.set('asic', 'cells', 'clklogic', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

decap

Description	ASIC: decap cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_decap '<str>'`
Example (CLI)	`-asic_cells_decap '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'decap', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

delay

Description	ASIC: delay cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_delay '<str>'`
Example (CLI)	`-asic_cells_delay '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'delay', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

dontuse

Description	ASIC: dontuse cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_dontuse '<str>'`
Example (CLI)	`-asic_cells_dontuse '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'dontuse', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

endcap

Description	ASIC: endcap cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_endcap '<str>'`
Example (CLI)	`-asic_cells_endcap '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'endcap', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

filler

Description	ASIC: filler cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_filler '<str>'`
Example (CLI)	`-asic_cells_filler '*eco*'`

continues on next page

Table 472 – continued from previous page

Example (API)	`chip.set('asic', 'cells', 'filler', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

hold

Description	ASIC: hold cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_hold '<str>'`
Example (CLI)	`-asic_cells_hold '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'hold', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

tap

Description	ASIC: tap cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_tap '<str>'`
Example (CLI)	`-asic_cells_tap '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'tap', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

tie

Description	ASIC: tie cell list
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_cells_tie '<str>'`
Example (CLI)	`-asic_cells_tie '*eco*'`
Example (API)	`chip.set('asic', 'cells', 'tie', '*eco*')`

List of cells grouped by a property that can be accessed directly by the designer and tools. The example below shows how all cells containing the string 'eco' could be marked as dont use for the tool.

delaymodel

Description	ASIC: delay model
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-asic_delaymodel <str>`
Example (CLI)	`-asic_delaymodel ccs`
Example (API)	`chip.set('asic', 'delaymodel', 'ccs')`

Delay model to use for the target libs. Supported values are nldm and ccs.

libarch

Description	ASIC: library architecture
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-asic_libarch '<str>'`
Example (CLI)	`-asic_libarch '12track'`
Example (API)	`chip.set('asic', 'libarch', '12track')`

The library architecture (e.g. library height) used to build the design. For example a PDK with support for 9 and 12 track libraries might have 'libarchs' called 9t and 12t.

logiclib

Description	ASIC: logic libraries
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_logiclib <str>`
Example (CLI)	`-asic_logiclib nangate45`
Example (API)	`chip.set('asic', 'logiclib', 'nangate45')`

List of all selected logic libraries libraries to use for optimization for a given library architecture (9T, 11T, etc).

macrolib

Description	ASIC: macro libraries
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_macrolib <str>`
Example (CLI)	`-asic_macrolib sram64x1024`
Example (API)	`chip.set('asic', 'macrolib', 'sram64x1024')`

List of macro libraries to be linked in during synthesis and place and route. Macro libraries are used for resolving instances but are not used as targets for logic synthesis.

site

Description	ASIC: Library sites
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-asic_site 'libarch <str>'`
Example (CLI)	`-asic_site '12track Site_12T'`
Example (API)	`chip.set('asic', 'site', '12track', 'Site_12T')`

Site names for a given library architecture.

checklist

criteria

Description	Checklist: item criteria
Type	[str]
Default Value	[]
CLI Switch	• `-checklist_criteria 'standard item <str>'`
Example (CLI)	`-checklist_criteria 'ISO D000 errors==0'`
Example (API)	`chip.set('checklist', 'ISO', 'D000', 'criteria', 'errors==0')`

Simple list of signoff criteria for checklist item which must all be met for signoff. Each signoff criteria consists of a metric, a relational operator, and a value in the form. 'metric op value'.

dataformat

Description	Checklist: item data format
Type	str
Default Value	None
CLI Switch	• `-checklist_dataformat 'standard item <str>'`
Example (CLI)	`-checklist_dataformat 'ISO D000 dataformat README'`
Example (API)	`chip.set('checklist', 'ISO', 'D000', 'dataformat', 'README')`

Free text description of the type of data files acceptable as checklist signoff validation.

description

Description	Checklist: item description
Type	str
Default Value	None
CLI Switch	• `-checklist_description 'standard item <str>'`
Example (CLI)	`-checklist_description 'ISO D000 A-DESCRIPTION'`
Example (API)	`chip.set('checklist', 'ISO', 'D000', 'description', 'A-DESCRIPTION')`

A short one line description of the checklist item.

ok

Description	Checklist: item ok
Type	bool
Default Value	False
CLI Switch	• `-checklist_ok 'standard item <bool>'`
Example (CLI)	`-checklist_ok 'ISO D000 true'`
Example (API)	`chip.set('checklist', 'ISO', 'D000', 'ok', True)`

Boolean check mark for the checklist item. A value of True indicates a human has inspected the all item dictionary parameters check out.

rationale

Description	Checklist: item rational
Type	[str]
Default Value	[]
CLI Switch	• `-checklist_rationale 'standard item <str>'`
Example (CLI)	`-checklist_rationale 'ISO D000 reliability'`

continues on next page

Table 485 – continued from previous page

Example (API)	`chip.set('checklist', 'ISO', 'D000', 'rationale', 'reliability')`

Rationale for the the checklist item. Rationale should be a unique alphanumeric code used by the standard or a short one line or single word description.

report

Description	Checklist: item report
Type	[file]
Default Value	[]
CLI Switch	• `-checklist_report 'standard item <file>'`
Example (CLI)	`-checklist_report 'ISO D000 my.rpt'`
Example (API)	`chip.set('checklist', 'ISO', 'D000', 'report', 'my.rpt')`

Filepath to report(s) of specified type documenting the successful validation of the checklist item.

requirement

Description	Checklist: item requirement
Type	str
Default Value	None
CLI Switch	• `-checklist_requirement 'standard item <str>'`
Example (CLI)	`-checklist_requirement 'ISO D000 DOCSTRING'`
Example (API)	`chip.set('checklist', 'ISO', 'D000', 'requirement', 'DOCSTRING')`

A complete requirement description of the checklist item entered as a multi-line string.

task

Description	Checklist: item task
Type	[(str,str,str)]
Default Value	[]
CLI Switch	• `-checklist_task 'standard item <(str, str,str)>'`
Example (CLI)	`-checklist_task 'ISO D000 (job0,place,0)'`
Example (API)	`chip.set('checklist', 'ISO', 'D000', 'task', ('job0', 'place', '0'))`

Flowgraph job and task used to verify the checklist item. The parameter should be left empty for manual and for tool flows that bypass the SC infrastructure.

waiver

Description	Checklist: item metric waivers
Type	[file]
Default Value	[]
CLI Switch	• `-checklist_waiver 'standard item metric <file>'`
Example (CLI)	`-checklist_waiver 'ISO D000 bold my.txt'`
Example (API)	`chip.set('checklist', 'ISO', 'D000', 'waiver', 'hold', 'my.txt')`

Filepath to report(s) documenting waivers for the checklist item specified on a per metric basis.

constraint

aspectratio

Description	Constraint: Layout aspect ratio
Type	float
Per step/index	optional
Default Value	1.0
CLI Switch	• `-constraint_aspectratio <float>`
Example (CLI)	`-constraint_aspectratio 2.0`

continues on next page

Table 490 – continued from previous page

Example (API)	`chip.set('constraint', 'aspectratio', '2.0')`

Height to width ratio of the block for automated floorplanning. Values below 0.1 and above 10 should be avoided as they will likely fail to converge during placement and routing. The ideal aspect ratio for most designs is 1. This value is only used when no diearea or floorplan is supplied.

component

flip

Description	Constraint: Component flip option
Type	bool
Per step/index	optional
Default Value	False
CLI Switch	• `-constraint_component_flip 'inst <bool>'`
Example (CLI)	`-constraint_component_flip 'i0 true'`
Example (API)	`chip.set('constraint', 'component', 'i0', 'flip', True)`

Boolean parameter specifying that the instanced library component should be flipped around the vertical axis before being placed on the substrate. The need to flip a component depends on the component footprint. Most dies have pads facing up and so must be flipped when assembled face down (eg. flip-chip, WCSP).

halo

Description	Constraint: Component halo
Type	(float,float)
Per step/index	optional
Unit	um
Default Value	None
CLI Switch	• `-constraint_component_halo 'inst <(float,float)>'`
Example (CLI)	`-constraint_component_halo 'i0 (1,1)'`
Example (API)	`chip.set('constraint', 'component', 'i0', 'halo', (1, 1))`

Placement keepout halo around the named component, specified as a (horizontal, vertical) tuple represented in microns or lambda units.

partname

Description	Constraint: Component part name
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_component_partname 'inst <str>'`
Example (CLI)	`-constraint_component_partname 'i0 filler_x1'`
Example (API)	`chip.set('constraint', 'component', 'i0', 'partname', 'filler_x1')`

Part name of a named instance. The parameter is required for instances that are not contained within the design netlist (ie. physical only cells).

placement

Description	Constraint: Component placement
Type	(float,float,float)
Per step/index	optional
Unit	um
Default Value	None
CLI Switch	• `-constraint_component_placement 'inst <(float,float,float)>'`
Example (CLI)	`-constraint_component_placement 'i0 (2.0, 3.0,0.0)'`
Example (API)	`chip.set('constraint', 'component', 'i0', 'placement', (2.0, 3.0, 0.0))`

Placement location of a named instance, specified as a (x, y, z) tuple of floats. The location refers to the placement of the center/centroid of the component. The 'placement' parameter is a goal/intent, not an exact specification. The compiler and layout system may adjust coordinates to meet competing goals such as manufacturing design rules and grid placement guidelines. The 'z' coordinate shall be set to 0 for planar systems with only (x, y) coordinates. Discretized systems like PCB stacks, package stacks, and breadboards only allow a reduced set of floating point values (0, 1, 2, 3). The user specifying the placement will need to have some understanding of the type of layout system the component is being placed in (ASIC, SIP, PCB) but should not need to know exact manufacturing specifications.

rotation

Description	Constraint: Component rotation
Type	float
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_component_rotation 'inst <float>'`
Example (CLI)	`-constraint_component_rotation 'i0 90'`
Example (API)	`chip.set('constraint', 'component', 'i0', 'rotation', '90')`

Placement rotation of the component specified in degrees. Rotation goes counter-clockwise for all parts on top and clock-wise for parts on the bottom. In both cases, this is from the perspective of looking at the top of the board. Rotation is specified in degrees. Most gridded layout systems (like ASICs) only allow a finite number of rotation values (0, 90, 180, 270).

corearea

Description	Constraint: Layout core area
Type	[(float,float)]
Per step/index	optional
Unit	um
Default Value	[]
CLI Switch	• `-constraint_corearea <(float,float)>`
Example (CLI)	`-constraint_corearea '(0,0)'`
Example (API)	`chip.set('constraint', 'corearea', (0, 0))`

List of (x, y) points that define the outline of the core area for the physical design. Simple rectangle areas can be defined with two points, one for the lower left corner and one for the upper right corner. All values are specified in microns or lambda units.

coremargin

Description	Constraint: Layout core margin
Type	float
Per step/index	optional
Unit	um
Default Value	None

continues on next page

Table 497 – continued from previous page

CLI Switch	• `-constraint_coremargin <float>`
Example (CLI)	`-constraint_coremargin 1`
Example (API)	`chip.set('constraint', 'coremargin', '1')`

Halo/margin between the outline and core area for fully automated layout sizing and floorplanning, specified in microns or lambda units.

density

Description	Constraint: Layout density
Type	float
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_density <float>`
Example (CLI)	`-constraint_density 30`
Example (API)	`chip.set('constraint', 'density', '30')`

Target density based on the total design cells area reported after synthesis/elaboration. This number is used when no outline or floorplan is supplied. Any number between 1 and 100 is legal, but values above 50 may fail due to area/congestion issues during automated place and route.

net

diffpair

Description	Constraint: Net diffpair
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_net_diffpair 'name <str>'`
Example (CLI)	`-constraint_net_diffpair 'clkn clkp'`
Example (API)	`chip.set('constraint', 'net', 'clkn', 'diffpair', 'clkp')`

Differential pair signal of the named net (only used for actual differential pairs).

match

Description	Constraint: Net matched routing
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-constraint_net_match 'name <str>'`
Example (CLI)	`-constraint_net_match 'clk1 clk2'`
Example (API)	`chip.set('constraint', 'net', 'clk1', 'match', 'clk2')`

List of nets whose routing should closely matched the named net in terms of length, layer, width, etc. Wildcards ('*') can be used for net names.

maxlayer

Description	Constraint: Net maximum routing layer
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_net_maxlayer 'name <str>'`
Example (CLI)	`-constraint_net_maxlayer 'nreset m1'`
Example (API)	`chip.set('constraint', 'net', 'nreset', 'maxlayer', 'm1')`

Maximum metal layer to be used for automated place and route specified on a per net basis. Metal names should either be the PDK specific metal stack name or an integer with '1' being the lowest routing layer. Wildcards ('*') can be used for net names.

maxlength

Description	Constraint: Net max length
Type	float
Per step/index	optional
Unit	um
Default Value	None

continues on next page

Table 502 – continued from previous page

CLI Switch	• `-constraint_net_maxlength 'name <float>'`
Example (CLI)	`-constraint_net_maxlength 'nreset 1000'`
Example (API)	`chip.set('constraint', 'net', 'nreset', 'maxlength', '1000')`

Maximum total length of a net, specified in microns or lambda units. Wildcards ('*') can be used for net names.

maxresistance

Description	Constraint: Net max resistance
Type	float
Per step/index	optional
Unit	ohm
Default Value	None
CLI Switch	• `-constraint_net_maxresistance 'name <float>'`
Example (CLI)	`-constraint_net_maxresistance 'nreset 1'`
Example (API)	`chip.set('constraint', 'net', 'nreset', 'maxresistance', '1')`

Maximum resistance of named net between driver and receiver specified in ohms. Wildcards ('*') can be used for net names.

minlayer

Description	Constraint: Net minimum routing layer
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_net_minlayer 'name <str>'`
Example (CLI)	`-constraint_net_minlayer 'nreset m1'`
Example (API)	`chip.set('constraint', 'net', 'nreset', 'minlayer', 'm1')`

Minimum metal layer to be used for automated place and route specified on a per net basis. Metal names should either be the PDK specific metal stack name or an integer with '1' being the lowest routing layer. Wildcards ('*') can be used for net names.

ndr

Description	Constraint: Net routing rule
Type	(float,float)
Per step/index	optional
Unit	um
Default Value	None
CLI Switch	• `-constraint_net_ndr 'name <(float, float)>'`
Example (CLI)	`-constraint_net_ndr 'nreset (0.4,0.4)'`
Example (API)	`chip.set('constraint', 'net', 'nreset', 'ndr', (0.4, 0.4))`

Definitions of non-default routing rule specified on a per net basis. Constraints are entered as a (width, space) tuples specified in microns or lambda units. Wildcards ('*') can be used for net names.

shield

Description	Constraint: Net shielding
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_net_shield 'name <str>'`
Example (CLI)	`-constraint_net_shield 'clk vss'`
Example (API)	`chip.set('constraint', 'net', 'clk', 'shield', 'vss')`

Specifies that the named net should be shielded by the given signal on both sides of the net.

sympair

Description	Constraint: Net sympair
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_net_sympair 'name <str>'`
Example (CLI)	`-constraint_net_sympair 'netA netB'`
Example (API)	`chip.set('constraint', 'net', 'netA', 'sympair', 'netB')`

Symmetrical pair signal to the named net. The two nets should be routed as reflections around the vertical or horizontal axis to minimize on-chip variability.

outline

Description	Constraint: Layout outline
Type	[(float,float)]
Per step/index	optional
Unit	um
Default Value	[]
CLI Switch	• `-constraint_outline <(float,float)>`
Example (CLI)	`-constraint_outline '(0,0)'`
Example (API)	`chip.set('constraint', 'outline', (0, 0))`

List of (x, y) points that define the outline physical layout physical design. Simple rectangle areas can be defined with two points, one for the lower left corner and one for the upper right corner. All values are specified in microns or lambda units.

pin

layer

Description	Constraint: Pin layer
Type	str
Per step/index	optional
Default Value	None

continues on next page

Table 509 – continued from previous page

CLI Switch	• `-constraint_pin_layer 'name <str>'`
Example (CLI)	`-constraint_pin_layer 'nreset m4'`
Example (API)	`chip.set('constraint', 'pin', 'nreset', 'layer', 'm4')`

Pin metal layer specified based on the SC standard layer stack starting with m1 as the lowest routing layer and ending with m<n> as the highest routing layer.

order

Description	Constraint: Pin order
Type	int
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_pin_order 'name <int>'`
Example (CLI)	`-constraint_pin_order 'nreset 1'`
Example (API)	`chip.set('constraint', 'pin', 'nreset', 'order', 1)`

The relative position of the named pin in a vector of pins on the side specified by the 'side' option. Pin order counting is done clockwise. If multiple pins on the same side have the same order number, the actual order is at the discretion of the tool.

placement

Description	Constraint: Pin placement
Type	(float,float,float)
Per step/index	optional
Unit	um
Default Value	None
CLI Switch	• `-constraint_pin_placement 'name <(float,float,float)>'`
Example (CLI)	`-constraint_pin_placement 'nreset (2.0,3.0,0.0)'`
Example (API)	`chip.set('constraint', 'pin', 'nreset', 'placement', (2.0, 3.0, 0.0))`

Placement location of a named pin, specified as a (x, y, z) tuple of floats. The location refers to the placement of the center of the pin. The 'placement' parameter is a goal/intent, not an exact specification. The compiler and layout system may adjust sizes to meet competing goals such as manufacturing design rules and grid placement guidelines. The 'z' coordinate shall be set to 0 for planar components with only (x, y) coordinates. Discretized systems like 3D chips with pins on top and bottom may choose to discretize the top and bottom layer as 0, 1 or use absolute coordinates. Values are specified in microns or lambda units.

side

Description	Constraint: Pin side
Type	int
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_pin_side 'name <int>'`
Example (CLI)	`-constraint_pin_side 'nreset 1'`
Example (API)	`chip.set('constraint', 'pin', 'nreset', 'side', 1)`

Side of block where the named pin should be placed. Sides are enumerated as integers with '1' being the lower left side, with the side index incremented on right turn in a clock wise fashion. In case of conflict between 'lower' and 'left', 'left' has precedence. The side option and order option are orthogonal to the placement option.

timing

check

Description	Constraint: timing checks
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-constraint_timing_check 'scenario <str>'`
Example (CLI)	`-constraint_timing_check 'worst setup'`
Example (API)	`chip.add('constraint', 'timing', 'worst', 'check', 'setup')`

List of checks for to perform for the scenario. The checks must align with the capabilities of the EDA tools and flow being used. Checks generally include objectives like meeting setup and hold goals and minimize power. Standard check names include setup, hold, power, noise, reliability.

file

Description	Constraint: SDC files
Type	[file]
Per step/index	optional
Default Value	[]
CLI Switch	• `-constraint_timing_file 'scenario <file>'`
Example (CLI)	`-constraint_timing_file 'worst hello.sdc'`
Example (API)	`chip.set('constraint', 'timing', 'worst', 'file', 'hello.sdc')`

List of timing constraint files to use for the scenario. The values are combined with any constraints specified by the design 'constraint' parameter. If no constraints are found, a default constraint file is used based on the clock definitions.

libcorner

Description	Constraint: library corner
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-constraint_timing_libcorner 'scenario <str>'`
Example (CLI)	`-constraint_timing_libcorner 'worst ttt'`
Example (API)	`chip.set('constraint', 'timing', 'worst', 'libcorner', 'ttt')`

List of characterization corners used to select timing files for all logiclibs and macrolibs.

mode

Description	Constraint: operating mode
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_timing_mode 'scenario <str>'`

continues on next page

Table 516 – continued from previous page

Example (CLI)	`-constraint_timing_mode 'worst test'`
Example (API)	`chip.set('constraint', 'timing', 'worst', 'mode', 'test')`

Operating mode for the scenario. Operating mode strings can be values such as test, functional, standby.

opcond

Description	Constraint: operating condition
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_timing_opcond 'scenario <str>'`
Example (CLI)	`-constraint_timing_opcond 'worst typical_1.0'`
Example (API)	`chip.set('constraint', 'timing', 'worst', 'opcond', 'typical_1.0')`

Operating condition applied to the scenario. The value can be used to access specific conditions within the library timing models from the 'logiclib' timing models.

pexcorner

Description	Constraint: pex corner
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-constraint_timing_pexcorner 'scenario <str>'`
Example (CLI)	`-constraint_timing_pexcorner 'worst max'`
Example (API)	`chip.set('constraint', 'timing', 'worst', 'pexcorner', 'max')`

Parasitic corner applied to the scenario. The 'pexcorner' string must match a corner found in the pdk pexmodel setup.

temperature

Description	Constraint: temperature
Type	float
Per step/index	optional
Unit	C
Default Value	None
CLI Switch	• `-constraint_timing_temperature 'scenario <float>'`
Example (CLI)	`-constraint_timing_temperature 'worst 125'`
Example (API)	`chip.set('constraint', 'timing', 'worst', 'temperature', '125')`

Chip temperature applied to the scenario specified in degrees C.

voltage

Description	Constraint: pin voltage level
Type	float
Per step/index	optional
Unit	V
Default Value	None
CLI Switch	• `-constraint_timing_voltage 'scenario pin <float>'`
Example (CLI)	`-constraint_timing_voltage 'worst VDD 0.9'`
Example (API)	`chip.set('constraint', 'timing', 'worst', 'voltage', 'VDD', '0.9')`

Operating voltage applied to a specific pin in the scenario.

datasheet

abstraction

Description	Datasheet: abstraction level
Type	[enum]
Default Value	[]

continues on next page

Table 521 – continued from previous page

CLI Switch	• `-datasheet_abstraction '<str>'`
Example (CLI)	`-datasheet_abstraction model`
Example (API)	`chip.set('datasheet', 'abstraction', 'model')`

List of device abstraction levels.

analog

arch

Description	Datasheet: analog architecture
Type	str
Default Value	None
CLI Switch	• `-datasheet_analog_arch 'name <str>'`
Example (CLI)	`-datasheet_analog_arch 'adc0 pipelined'`
Example (API)	`chip.set('datasheet', 'analog', 'adc0', 'arch', 'pipelined')`

Analog component architecture.

channels

Description	Datasheet: Analog parallel channels
Type	int
Default Value	None
CLI Switch	• `-datasheet_analog_channels 'name <int>'`
Example (CLI)	`-datasheet_analog_channels 'i0 8'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'channels', 8)`

Analog 8.

cmrr

Description	Datasheet: Analog common mode rejection ratio
Type	(float,float,float)
Unit	dB
Default Value	None
CLI Switch	• `-datasheet_analog_cmrr 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_cmrr 'i0 (70, 80, 90)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'cmrr', (70, 80, 90))`

Analog (70, 80, 90).

dnl

Description	Datasheet: Analog differential nonlinearity
Type	(float,float,float)
Unit	LSB
Default Value	None
CLI Switch	• `-datasheet_analog_dnl 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_dnl 'i0 (-1.0, 0.0, 1.0)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'dnl', (-1.0, 0.0, 1.0))`

Analog (-1.0, 0.0, 1.0).

enob

Description	Datasheet: Analog effective number of bits
Type	(float,float,float)
Unit	bits
Default Value	None
CLI Switch	• `-datasheet_analog_enob 'name <(float, float,float)>'`

continues on next page

Table 526 – continued from previous page

Example (CLI)	`-datasheet_analog_enob 'i0 (8, 9, 10)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'enob', (8, 9, 10))`

Analog (8, 9, 10).

features

Description	Datasheet: analog features
Type	[str]
Default Value	[]
CLI Switch	• `-datasheet_analog_features 'name <str>'`
Example (CLI)	`-datasheet_analog_features '0 differential input'`
Example (API)	`chip.set('datasheet','analog','adc0', 'features', 'differential input')`

List of maker specified analog features.

gain

Description	Datasheet: Analog gain
Type	(float,float,float)
Unit	dB
Default Value	None
CLI Switch	• `-datasheet_analog_gain 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_gain 'i0 (11.4, 11.4, 11.4)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'gain', (11.4, 11.4, 11.4))`

Analog (11.4, 11.4, 11.4).

hd2

Description	Datasheet: Analog 2nd order harmonic distortion
Type	(float,float,float)
Unit	dBc
Default Value	None
CLI Switch	• `-datasheet_analog_hd2 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_hd2 'i0 (62, 64, 66)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'hd2', (62, 64, 66))`

Analog (62, 64, 66).

hd3

Description	Datasheet: Analog 3rd order harmonic distortion
Type	(float,float,float)
Unit	dBc
Default Value	None
CLI Switch	• `-datasheet_analog_hd3 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_hd3 'i0 (62, 64, 66)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'hd3', (62, 64, 66))`

Analog (62, 64, 66).

hd4

Description	Datasheet: Analog 4th order harmonic distortion
Type	(float,float,float)
Unit	dBc
Default Value	None
CLI Switch	• `-datasheet_analog_hd4 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_hd4 'i0 (62, 64, 66)'`

continues on next page

Table 531 – continued from previous page

Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'hd4', (62, 64, 66))`

Analog (62, 64, 66).

ib1db

Description	Datasheet: Analog rf in band 1 dB compression point
Type	(float,float,float)
Unit	dBm
Default Value	None
CLI Switch	• `-datasheet_analog_ib1db 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_ib1db 'i0 (-1, 1, 1)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'ib1db', (-1, 1, 1))`

Analog (-1, 1, 1).

iip3

Description	Datasheet: Analog rf 3rd order input intercept point
Type	(float,float,float)
Unit	dBm
Default Value	None
CLI Switch	• `-datasheet_analog_iip3 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_iip3 'i0 (3, 3, 3)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'iip3', (3, 3, 3))`

Analog (3, 3, 3).

imd3

Description	Datasheet: Analog 3rd order intermodulation distortion
Type	(float,float,float)
Unit	dBc
Default Value	None
CLI Switch	• `-datasheet_analog_imd3 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_imd3 'i0 (82, 88, 98)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'imd3', (82, 88, 98))`

Analog (82, 88, 98).

inl

Description	Datasheet: Analog integral nonlinearity
Type	(float,float,float)
Unit	LSB
Default Value	None
CLI Switch	• `-datasheet_analog_inl 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_inl 'i0 (-7, 0.0, 7)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'inl', (-7, 0.0, 7))`

Analog (-7, 0.0, 7).

noisefigure

Description	Datasheet: Analog rf noise figure
Type	(float,float,float)
Unit	dB
Default Value	None
CLI Switch	• `-datasheet_analog_noisefigure 'name <(float,float,float)>'`

continues on next page

Table 536 – continued from previous page

Example (CLI)	`-datasheet_analog_noisefigure 'i0 (4.6, 4.6, 4.6)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'noisefigure', (4.6, 4.6, 4.6))`

Analog (4.6, 4.6, 4.6).

nsd

Description	Datasheet: Analog noise spectral density
Type	(float,float,float)
Unit	dBFS/Hz
Default Value	None
CLI Switch	• `-datasheet_analog_nsd 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_nsd 'i0 (-158, -158, -158)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'nsd', (-158, -158, -158))`

Analog (-158, -158, -158).

oob1db

Description	Datasheet: Analog rf out of band 1 dB compression point
Type	(float,float,float)
Unit	dBm
Default Value	None
CLI Switch	• `-datasheet_analog_oob1db 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_oob1db 'i0 (3, 3, 3)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'oob1db', (3, 3, 3))`

Analog (3, 3, 3).

phasenoise

Description	Datasheet: Analog phase noise
Type	(float,float,float)
Unit	dBc/Hz
Default Value	None
CLI Switch	• `-datasheet_analog_phasenoise 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_phasenoise 'i0 (-158, -158, -158)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'phasenoise', (-158, -158, -158))`

Analog (-158, -158, -158).

pout

Description	Datasheet: Analog output power
Type	(float,float,float)
Unit	dBm
Default Value	None
CLI Switch	• `-datasheet_analog_pout 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_pout 'i0 (12.2, 12.2, 12.2)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'pout', (12.2, 12.2, 12.2))`

Analog (12.2, 12.2, 12.2).

pout2

Description	Datasheet: Analog 2nd harmonic power
Type	(float,float,float)
Unit	dBm
Default Value	None

continues on next page

Table 541 – continued from previous page

CLI Switch	• `-datasheet_analog_pout2 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_pout2 'i0 (-14, -14, -14)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'pout2', (-14, -14, -14))`

Analog (-14, -14, -14).

pout3

Description	Datasheet: Analog 3rd harmonic power
Type	(float,float,float)
Unit	dBm
Default Value	None
CLI Switch	• `-datasheet_analog_pout3 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_pout3 'i0 (-28, -28, -28)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'pout3', (-28, -28, -28))`

Analog (-28, -28, -28).

psnr

Description	Datasheet: Analog power supply noise rejection
Type	(float,float,float)
Unit	dB
Default Value	None
CLI Switch	• `-datasheet_analog_psnr 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_psnr 'i0 (61, 61, 61)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'psnr', (61, 61, 61))`

Analog (61, 61, 61).

resolution

Description	Datasheet: Analog architecture resolution
Type	int
Default Value	None
CLI Switch	• `-datasheet_analog_resolution 'name <int>'`
Example (CLI)	`-datasheet_analog_resolution 'i0 8'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'resolution', 8)`

Analog 8.

s11

Description	Datasheet: Analog rf input return loss
Type	(float,float,float)
Unit	dB
Default Value	None
CLI Switch	• `-datasheet_analog_s11 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_s11 'i0 (7, 7, 7)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 's11', (7, 7, 7))`

Analog (7, 7, 7).

s12

Description	Datasheet: Analog rf reverse isolation
Type	(float,float,float)
Unit	dB
Default Value	None
CLI Switch	• `-datasheet_analog_s12 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_s12 'i0 (-20, -20, -20)'`

continues on next page

Table 546 – continued from previous page

Example (API)	`chip.set('datasheet', 'analog', 'abc123', 's12', (-20, -20, -20))`

Analog (-20, -20, -20).

s21

Description	Datasheet: Analog rf gain
Type	(float,float,float)
Unit	dB
Default Value	None
CLI Switch	• `-datasheet_analog_s21 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_s21 'i0 (10, 11, 12)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 's21', (10, 11, 12))`

Analog (10, 11, 12).

s22

Description	Datasheet: Analog rf output return loss
Type	(float,float,float)
Unit	dB
Default Value	None
CLI Switch	• `-datasheet_analog_s22 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_analog_s22 'i0 (10, 10, 10)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 's22', (10, 10, 10))`

Analog (10, 10, 10).

samplerate

Description	Datasheet: Analog sample rate
Type	(float,float,float)
Unit	Hz
Default Value	None
CLI Switch	• `-datasheet_analog_samplerate 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_samplerate 'i0 (1000000000.0, 1000000000.0, 1000000000.0)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'samplerate', (1000000000.0, 1000000000.0, 1000000000.0))`

Analog (1000000000.0, 1000000000.0, 1000000000.0).

sfdr

Description	Datasheet: Analog spurious-free dynamic range
Type	(float,float,float)
Unit	dBc
Default Value	None
CLI Switch	• `-datasheet_analog_sfdr 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_sfdr 'i0 (82, 88, 98)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'sfdr', (82, 88, 98))`

Analog (82, 88, 98).

sinad

Description	Datasheet: Analog signal to noise and distortion ratio
Type	(float,float,float)
Unit	dB
Default Value	None

continues on next page

Table 551 – continued from previous page

CLI Switch	• `-datasheet_analog_sinad 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_sinad 'i0 (71, 72, 73)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'sinad', (71, 72, 73))`

Analog (71, 72, 73).

snr

Description	Datasheet: Analog signal to noise ratio
Type	(float,float,float)
Unit	dB
Default Value	None
CLI Switch	• `-datasheet_analog_snr 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_snr 'i0 (70, 72, 74)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'snr', (70, 72, 74))`

Analog (70, 72, 74).

thd

Description	Datasheet: Analog total harmonic distortion
Type	(float,float,float)
Unit	dB
Default Value	None
CLI Switch	• `-datasheet_analog_thd 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_thd 'i0 (82, 88, 98)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'thd', (82, 88, 98))`

Analog (82, 88, 98).

vgainerror

Description	Datasheet: Analog gain error
Type	(float,float,float)
Unit	mV
Default Value	None
CLI Switch	• `-datasheet_analog_vgainerror 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_vgainerror 'i0 (-1.0, 0.0, 1.0)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'vgainerror', (-1.0, 0.0, 1.0))`

Analog (-1.0, 0.0, 1.0).

vofferror

Description	Datasheet: Analog offset error
Type	(float,float,float)
Unit	mV
Default Value	None
CLI Switch	• `-datasheet_analog_vofferror 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_analog_vofferror 'i0 (-1.0, 0.0, 1.0)'`
Example (API)	`chip.set('datasheet', 'analog', 'abc123', 'vofferror', (-1.0, 0.0, 1.0))`

Analog (-1.0, 0.0, 1.0).

description

Description	Datasheet: description
Type	str
Default Value	None
CLI Switch	• `-datasheet_description '<str>'`
Example (CLI)	`-datasheet_description 'Yet another CPU'`

continues on next page

Table 556 – continued from previous page

Example (API)	`chip.set('datasheet', 'description', 'Yet another CPU')`

Free text device description

doc

Description	Datasheet: part documentation
Type	[file]
Default Value	[]
CLI Switch	• `-datasheet_doc '<file>'`
Example (CLI)	`-datasheet_doc 'za001.pdf'`
Example (API)	`chip.set('datasheet', 'doc', 'za001.pdf)`

Device datasheet document.

features

Description	Datasheet: part features
Type	[str]
Default Value	[]
CLI Switch	• `-datasheet_features '<str>'`
Example (CLI)	`-datasheet_features 'usb3.0'`
Example (API)	`chip.set('datasheet', 'features', 'usb3.0')`

List of manufacturer specified device features

fmax

Description	Datasheet: device maximum frequency
Type	float
Unit	MHz
Default Value	None

continues on next page

Table 559 – continued from previous page

CLI Switch	• `-datasheet_fmax '<float>'`
Example (CLI)	`-datasheet_fmax 100'`
Example (API)	`chip.set('datasheet', 'fmax', 100')`

Device maximum operating frequency.

fpga

arch

Description	Datasheet: fpga architecture
Type	str
Default Value	None
CLI Switch	• `-datasheet_fpga_arch 'name <str>'`
Example (CLI)	`-datasheet_fpga_arch 'i0 openfpga'`
Example (API)	`chip.set('datasheet', 'fpga', 'i0', 'arch', 'openfpga')`

FPGA architecture.

blockram

Description	Datasheet: fpga block ram
Type	int
Unit	Kb
Default Value	None
CLI Switch	• `-datasheet_fpga_blockram 'name <int>'`
Example (CLI)	`-datasheet_fpga_blockram 'i0 128'`
Example (API)	`chip.set('datasheet', 'fpga', 'i0', 'blockram', 128)`

FPGA 128.

distram

Description	Datasheet: fpga distributed ram
Type	int
Unit	Kb
Default Value	None
CLI Switch	• `-datasheet_fpga_distram 'name <int>'`
Example (CLI)	`-datasheet_fpga_distram 'i0 128'`
Example (API)	`chip.set('datasheet', 'fpga', 'i0', 'distram', 128)`

FPGA 128.

luts

Description	Datasheet: fpga LUTs (4 input)
Type	int
Default Value	None
CLI Switch	• `-datasheet_fpga_luts 'name <int>'`
Example (CLI)	`-datasheet_fpga_luts 'i0 32000'`
Example (API)	`chip.set('datasheet', 'fpga', 'i0', 'luts', 32000)`

FPGA 32000.

mults

Description	Datasheet: fpga multiplier/dsp elements
Type	int
Default Value	None
CLI Switch	• `-datasheet_fpga_mults 'name <int>'`
Example (CLI)	`-datasheet_fpga_mults 'i0 100'`
Example (API)	`chip.set('datasheet', 'fpga', 'i0', 'mults', 100)`

FPGA 100.

plls

Description	Datasheet: fpga pll blocks
Type	int
Default Value	None
CLI Switch	• `-datasheet_fpga_plls 'name <int>'`
Example (CLI)	`-datasheet_fpga_plls 'i0 1'`
Example (API)	`chip.set('datasheet', 'fpga', 'i0', 'plls', 1)`

FPGA 1.

registers

Description	Datasheet: fpga registers
Type	int
Default Value	None
CLI Switch	• `-datasheet_fpga_registers 'name <int>'`
Example (CLI)	`-datasheet_fpga_registers 'i0 100'`
Example (API)	`chip.set('datasheet', 'fpga', 'i0', 'registers', 100)`

FPGA 100.

totalram

Description	Datasheet: fpga total ram
Type	int
Unit	Kb
Default Value	None
CLI Switch	• `-datasheet_fpga_totalram 'name <int>'`
Example (CLI)	`-datasheet_fpga_totalram 'i0 128'`
Example (API)	`chip.set('datasheet', 'fpga', 'i0', 'totalram', 128)`

FPGA 128.

grade

Description	Datasheet: part manufacturing grade
Type	enum
Allowed Values	• consumer • industrial • medical • automotive • military • space
Default Value	None
CLI Switch	• -datasheet_grade '<str>'
Example (CLI)	-datasheet_grade 'automotive'
Example (API)	chip.set('datasheet', 'grade', 'automotive')

Device end application qualification grade.

io

arch

Description	Datasheet: io standard
Type	enum

continues on next page

Table 569 – continued from previous page

Allowed Values	• spi • uart • i2c • pwm • qspi • sdio • can • jtag • ddr • hbm • onfi • sram • hdmi • mipi-csi • mipi-dsi • slvs • sata • usb • pcie • cxl • spdif • i2s • gpio • lvds • serdes • pio • ethernet • rmii • rgmii • sgmii • xaui • 10gbase-kr • 25gbase-kr • xfi • cei28g • jesd204 • cpri
Default Value	None
CLI Switch	• -datasheet_io_arch 'name <str>'
Example (CLI)	-datasheet_io_arch 'mif0 ddr'
Example (API)	chip.set('datasheet', 'io', 'mif0', 'arch', 'ddr')

Datasheet: IO standard architecture specified on a per port basis.

channels

Description	Datasheet: io channels
Type	int
Default Value	None
CLI Switch	• `-datasheet_io_channels 'name <int>'`
Example (CLI)	`-datasheet_io_channels 'name 4'`
Example (API)	`chip.set('datasheet', 'io', name, 'channels', 4)`

Datasheet: IO 4 metrics specified on a per port basis.

fmax

Description	Datasheet: io maximum frequency
Type	float
Unit	MHz
Default Value	None
CLI Switch	• `-datasheet_io_fmax 'name <float>'`
Example (CLI)	`-datasheet_io_fmax 'name 100'`
Example (API)	`chip.set('datasheet', 'io', name, 'fmax', 100)`

Datasheet: IO 100 metrics specified on a per port basis.

gen

Description	Datasheet: io generation
Type	[str]
Default Value	[]
CLI Switch	• `-datasheet_io_gen 'name <str>'`
Example (CLI)	`-datasheet_io_gen 'ddr 3'`
Example (API)	`chip.set('datasheet', 'io', 'ddr', 'gen', '3')`

Datasheet: list of IO generations (versions) supported specified on a per port basis.

width

Description	Datasheet: io width
Type	int
Default Value	None
CLI Switch	• `-datasheet_io_width 'name <int>'`
Example (CLI)	`-datasheet_io_width 'name 4'`
Example (API)	`chip.set('datasheet', 'io', name, 'width', 4)`

Datasheet: IO 4 metrics specified on a per port basis.

iobw

Description	Datasheet: total I/O bandwidth
Type	float
Unit	bps
Default Value	None
CLI Switch	• `-datasheet_iobw '<float>'`
Example (CLI)	`-datasheet_iobw 1e18'`
Example (API)	`chip.set('datasheet', 'iobw', 1e18)`

Device peak off-device bandwidth in bits per second.

iocount

Description	Datasheet: total number of I/Os
Type	int
Default Value	None
CLI Switch	• `-datasheet_iocount '<int>'`
Example (CLI)	`-datasheet_iocount 100'`
Example (API)	`chip.set('datasheet', 'iocount', 100)`

Device total number of I/Os (not counting supplies).

limit

seb

Description	Datasheet: limit single event burnout threshold
Type	(float,float)
Unit	MeV-cm2/mg
Default Value	None
CLI Switch	• `-datasheet_limit_seb '<(float, float)>'`
Example (CLI)	`-datasheet_limit_seb '(75, 75)'`
Example (API)	`chip.set('datasheet', 'limit', 'seb', (75, 75)`

Limit single event burnout threshold. Values are tuples of (min, max).

segr

Description	Datasheet: limit single event gate rupture threshold
Type	(float,float)
Unit	MeV-cm2/mg
Default Value	None
CLI Switch	• `-datasheet_limit_segr '<(float, float)>'`
Example (CLI)	`-datasheet_limit_segr '(75, 75)'`
Example (API)	`chip.set('datasheet', 'limit', 'segr', (75, 75)`

Limit single event gate rupture threshold. Values are tuples of (min, max).

sel

Description	Datasheet: limit single event latchup threshold
Type	(float,float)
Unit	MeV-cm2/mg
Default Value	None

continues on next page

Table 578 – continued from previous page

CLI Switch	• `-datasheet_limit_sel '<(float, float)>'`
Example (CLI)	`-datasheet_limit_sel '(75, 75)'`
Example (API)	`chip.set('datasheet', 'limit', 'sel', (75, 75)`

Limit single event latchup threshold. Values are tuples of (min, max).

set

Description	Datasheet: limit single event transient threshold
Type	(float,float)
Unit	MeV-cm2/mg
Default Value	None
CLI Switch	• `-datasheet_limit_set '<(float, float)>'`
Example (CLI)	`-datasheet_limit_set '(75, 75)'`
Example (API)	`chip.set('datasheet', 'limit', 'set', (75, 75)`

Limit single event transient threshold. Values are tuples of (min, max).

seu

Description	Datasheet: limit single event upset threshold
Type	(float,float)
Unit	MeV-cm2/mg
Default Value	None
CLI Switch	• `-datasheet_limit_seu '<(float, float)>'`
Example (CLI)	`-datasheet_limit_seu '(75, 75)'`
Example (API)	`chip.set('datasheet', 'limit', 'seu', (75, 75)`

Limit single event upset threshold. Values are tuples of (min, max).

ta

Description	Datasheet: limit ambient temperature limits
Type	(float,float)
Unit	C
Default Value	None
CLI Switch	• `-datasheet_limit_ta '<(float,float)>'`
Example (CLI)	`-datasheet_limit_ta '(-40, 125)'`
Example (API)	`chip.set('datasheet', 'limit', 'ta', (-40, 125)`

Limit ambient temperature limits. Values are tuples of (min, max).

tid

Description	Datasheet: limit total ionizing dose threshold
Type	(float,float)
Unit	rad
Default Value	None
CLI Switch	• `-datasheet_limit_tid '<(float, float)>'`
Example (CLI)	`-datasheet_limit_tid '(300000.0, 300000.0)'`
Example (API)	`chip.set('datasheet', 'limit', 'tid', (300000.0, 300000.0)`

Limit total ionizing dose threshold. Values are tuples of (min, max).

tj

Description	Datasheet: limit junction temperature limits
Type	(float,float)
Unit	C
Default Value	None
CLI Switch	• `-datasheet_limit_tj '<(float,float)>'`
Example (CLI)	`-datasheet_limit_tj '(-40, 125)'`

continues on next page

Table 583 – continued from previous page

Example (API)	`chip.set('datasheet', 'limit', 'tj', (-40, 125)`

Limit junction temperature limits. Values are tuples of (min, max).

tsolder

Description	Datasheet: limit solder temperature limits
Type	(float,float)
Unit	C
Default Value	None
CLI Switch	• `-datasheet_limit_tsolder '<(float, float)>'`
Example (CLI)	`-datasheet_limit_tsolder '(-40, 125)'`
Example (API)	`chip.set('datasheet', 'limit', 'tsolder', (-40, 125)`

Limit solder temperature limits. Values are tuples of (min, max).

tstorage

Description	Datasheet: limit storage temperature limits
Type	(float,float)
Unit	C
Default Value	None
CLI Switch	• `-datasheet_limit_tstorage '<(float, float)>'`
Example (CLI)	`-datasheet_limit_tstorage '(-40, 125)'`
Example (API)	`chip.set('datasheet', 'limit', 'tstorage', (-40, 125)`

Limit storage temperature limits. Values are tuples of (min, max).

vcdm

Description	Datasheet: limit ESD charge device model voltage level
Type	(float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_limit_vcdm '<(float, float)>'`
Example (CLI)	`-datasheet_limit_vcdm '(150, 150)'`
Example (API)	`chip.set('datasheet', 'limit', 'vcdm', (150, 150)`

Limit ESD charge device model voltage level. Values are tuples of (min, max).

vhbm

Description	Datasheet: limit ESD human body model voltage level
Type	(float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_limit_vhbm '<(float, float)>'`
Example (CLI)	`-datasheet_limit_vhbm '(200, 250)'`
Example (API)	`chip.set('datasheet', 'limit', 'vhbm', (200, 250)`

Limit ESD human body model voltage level. Values are tuples of (min, max).

vmm

Description	Datasheet: limit ESD machine model voltage level
Type	(float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_limit_vmm '<(float, float)>'`
Example (CLI)	`-datasheet_limit_vmm '(125, 125)'`

continues on next page

Table 588 – continued from previous page

Example (API)	`chip.set('datasheet', 'limit', 'vmm', (125, 125)`

Limit ESD machine model voltage level. Values are tuples of (min, max).

manufacturer

Description	Datasheet: part manufacturer
Type	str
Default Value	None
CLI Switch	• `-datasheet_manufacturer '<str>'`
Example (CLI)	`-datasheet_manufacturer 'Acme'`
Example (API)	`chip.set('datasheet', 'manufacturer', 'Acme')`

Device manufacturer/vendor.

memory

banks

Description	Datasheet: memory banks
Type	int
Default Value	None
CLI Switch	• `-datasheet_memory_banks 'name <int>'`
Example (CLI)	`-datasheet_memory_banks 'm0 4'`
Example (API)	`chip.set('datasheet', 'memory', 'm0', 'banks', 4)`

Memory banks.

bits

Description	Datasheet: memory total bits
Type	int
Default Value	None
CLI Switch	• `-datasheet_memory_bits 'name <int>'`
Example (CLI)	`-datasheet_memory_bits 'm0 1024'`
Example (API)	`chip.set('datasheet', 'memory', 'm0', 'bits', 1024)`

Memory total number of bits.

bwrd

Description	Datasheet: memory maximum read bandwidth
Type	(float,float,float)
Unit	bps
Default Value	None
CLI Switch	• `-datasheet_memory_bwrd 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_memory_bwrd 'name (1000000000.0, 1000000000.0, 1000000000.0)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'bwrd', (1000000000.0, 1000000000.0, 1000000000.0))`

Memory (1000000000.0, 1000000000.0, 1000000000.0).

bwwr

Description	Datasheet: memory maximum write bandwidth
Type	(float,float,float)
Unit	bps
Default Value	None
CLI Switch	• `-datasheet_memory_bwwr 'name <(float, float,float)>'`

continues on next page

Table 593 – continued from previous page

Example (CLI)	`-datasheet_memory_bwwr 'name (1000000000.0, 1000000000.0, 1000000000.0)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'bwwr', (1000000000.0, 1000000000.0, 1000000000.0))`

Memory (1000000000.0, 1000000000.0, 1000000000.0).

depth

Description	Datasheet: memory depth
Type	int
Default Value	None
CLI Switch	• `-datasheet_memory_depth 'name <int>'`
Example (CLI)	`-datasheet_memory_depth 'm0 128'`
Example (API)	`chip.set('datasheet', 'memory', 'm0', 'depth', 128)`

Memory depth.

erd

Description	Datasheet: memory read energy
Type	(float,float,float)
Unit	J
Default Value	None
CLI Switch	• `-datasheet_memory_erd 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_memory_erd 'name (1e-12, 2e-12, 3e-12)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'erd', (1e-12, 2e-12, 3e-12))`

Memory (1e-12, 2e-12, 3e-12).

ewr

Description	Datasheet: memory write energy
Type	(float,float,float)
Unit	J
Default Value	None
CLI Switch	• `-datasheet_memory_ewr 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_memory_ewr 'name (1e-12, 2e-12, 3e-12)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'ewr', (1e-12, 2e-12, 3e-12))`

Memory (1e-12, 2e-12, 3e-12).

fmax

Description	Datasheet: memory max frequency
Type	(float,float,float)
Unit	Hz
Default Value	None
CLI Switch	• `-datasheet_memory_fmax 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_memory_fmax 'name (1000000000.0, 1000000000.0, 1000000000.0)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'fmax', (1000000000.0, 1000000000.0, 1000000000.0))`

Memory (1000000000.0, 1000000000.0, 1000000000.0).

tcl

Description	Datasheet: memory column address latency
Type	(int,int,int)
Unit	cycles
Default Value	None

continues on next page

Table 598 – continued from previous page

CLI Switch	• `-datasheet_memory_tcl 'name <(int, int,int)>'`
Example (CLI)	`-datasheet_memory_tcl 'name (100, 100, 100)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'tcl', (100, 100, 100))`

Memory (100, 100, 100).

tcycle

Description	Datasheet: memory access clock cycle
Type	(float,float,float)
Unit	ns
Default Value	None
CLI Switch	• `-datasheet_memory_tcycle 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_memory_tcycle 'name (9.0, 10.0, 11.0)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'tcycle', (9.0, 10.0, 11.0))`

Memory (9.0, 10.0, 11.0).

terase

Description	Datasheet: memory erase time
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_memory_terase 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_memory_terase 'name (1e-06, 1e-06, 1e-06)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'terase', (1e-06, 1e-06, 1e-06))`

Memory (1e-06, 1e-06, 1e-06).

tras

Description	Datasheet: memory row active time latency
Type	(int,int,int)
Unit	cycles
Default Value	None
CLI Switch	• `-datasheet_memory_tras 'name <(int, int,int)>'`
Example (CLI)	`-datasheet_memory_tras 'name (100, 100, 100)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'tras', (100, 100, 100))`

Memory (100, 100, 100).

trcd

Description	Datasheet: memory row address latency
Type	(int,int,int)
Unit	cycles
Default Value	None
CLI Switch	• `-datasheet_memory_trcd 'name <(int, int,int)>'`
Example (CLI)	`-datasheet_memory_trcd 'name (100, 100, 100)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'trcd', (100, 100, 100))`

Memory (100, 100, 100).

trd

Description	Datasheet: memory read clock cycle
Type	(float,float,float)
Unit	ns
Default Value	None
CLI Switch	• `-datasheet_memory_trd 'name <(float, float,float)>'`
Example (CLI)	`-datasheet_memory_trd 'name (0.9, 1, 1.1)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'trd', (0.9, 1, 1.1))`

Memory (0.9, 1, 1.1).

trefresh

Description	Datasheet: memory refresh time
Type	(float,float,float)
Unit	ns
Default Value	None
CLI Switch	• `-datasheet_memory_trefresh 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_memory_trefresh 'name (99, 100, 101)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'trefresh', (99, 100, 101))`

Memory (99, 100, 101).

trp

Description	Datasheet: memory row precharge time latency
Type	(int,int,int)
Unit	cycles
Default Value	None
CLI Switch	• `-datasheet_memory_trp 'name <(int, int,int)>'`

continues on next page

Table 605 – continued from previous page

Example (CLI)	`-datasheet_memory_trp 'name (100, 100, 100)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'trp', (100, 100, 100))`

Memory (100, 100, 100).

twearout

Description	Datasheet: memory write/erase wear-out
Type	(float,float,float)
Unit	cycles
Default Value	None
CLI Switch	• `-datasheet_memory_twearout 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_memory_twearout 'name (100000.0, 1000000.0, 10000000.0)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'twearout', (100000.0, 1000000.0, 10000000.0))`

Memory (100000.0, 1000000.0, 10000000.0).

twr

Description	Datasheet: memory write clock cycle
Type	(float,float,float)
Unit	ns
Default Value	None
CLI Switch	• `-datasheet_memory_twr 'name <(float,float,float)>'`
Example (CLI)	`-datasheet_memory_twr 'name (0.9, 1, 1.1)'`
Example (API)	`chip.set('datasheet', 'memory', name, 'twr', (0.9, 1, 1.1))`

Memory (0.9, 1, 1.1).

width

Description	Datasheet: memory width
Type	int
Default Value	None
CLI Switch	• `-datasheet_memory_width 'name <int>'`
Example (CLI)	`-datasheet_memory_width 'm0 16'`
Example (API)	`chip.set('datasheet', 'memory', 'm0', 'width', 16)`

Memory width.

ops

Description	Datasheet: total device operations per second
Type	float
Default Value	None
CLI Switch	• `-datasheet_ops '<float>'`
Example (CLI)	`-datasheet_ops 1e18'`
Example (API)	`chip.set('datasheet', 'ops', 1e18)`

Device peak total operations per second, describing the total mathematical opereations performed by all on-device processing units.

package

drawing

Description	Datasheet: package drawing
Type	[file]
Default Value	[]
CLI Switch	• `-datasheet_package_drawing '<file>'`
Example (CLI)	`-datasheet_package_drawing 'name.pdf'`
Example (API)	`chip.set('datasheet', 'package', 'drawing', 'p484.pdf')`

Datasheet: package drawing

length

Description	Datasheet: package length
Type	(float,float,float)
Unit	mm
Default Value	None
CLI Switch	• `-datasheet_package_length '<(float, float,float)>'`
Example (CLI)	`-datasheet_package_length '(20, 20, 20)'`
Example (API)	`chip.set('datasheet', 'package', 'length', (20, 20, 20)`

Package specification length. Values are tuples of (min, nominal, max).

name

Description	Datasheet: package name
Type	str
Default Value	None
CLI Switch	• `-datasheet_package_name '<str>'`
Example (CLI)	`-datasheet_package_name 'BGA484'`
Example (API)	`chip.set('datasheet', 'package', 'name', 'BGA484')`

Datasheet: package name

pincount

Description	Datasheet: package pincount
Type	int
Default Value	None
CLI Switch	• `-datasheet_package_pincount '<int>'`
Example (CLI)	`-datasheet_package_pincount '484'`
Example (API)	`chip.set('datasheet', 'package', 'pincount', '484')`

Datasheet: package pincount

pinpitch

Description	Datasheet: package pitch
Type	(float,float,float)
Unit	mm
Default Value	None
CLI Switch	• `-datasheet_package_pinpitch '<(float,float,float)>'`
Example (CLI)	`-datasheet_package_pinpitch '(0.8, 0.85, 0.9)'`
Example (API)	`chip.set('datasheet', 'package', 'pinpitch', (0.8, 0.85, 0.9)`

Package specification pitch. Values are tuples of (min, nominal, max).

thickness

Description	Datasheet: package thickness
Type	(float,float,float)
Unit	mm
Default Value	None
CLI Switch	• `-datasheet_package_thickness '<(float,float,float)>'`
Example (CLI)	`-datasheet_package_thickness '(1.0, 1.1, 1.2)'`
Example (API)	`chip.set('datasheet', 'package', 'thickness', (1.0, 1.1, 1.2)`

Package specification thickness. Values are tuples of (min, nominal, max).

width

Description	Datasheet: package width
Type	(float,float,float)
Unit	mm
Default Value	None

continues on next page

Table 616 – continued from previous page

CLI Switch	• `-datasheet_package_width '<(float, float,float)>'`
Example (CLI)	`-datasheet_package_width '(20, 20, 20)'`
Example (API)	`chip.set('datasheet', 'package', 'width', (20, 20, 20)`

Package specification width. Values are tuples of (min, nominal, max).

partnumber

Description	Datasheet: part number
Type	str
Default Value	None
CLI Switch	• `-datasheet_partnumber '<str>'`
Example (CLI)	`-datasheet_partnumber 'PN101'`
Example (API)	`chip.set('datasheet', 'partnumber', 'PN101')`

A unique device identifier.

peakpower

Description	Datasheet: peak power
Type	float
Unit	W
Default Value	None
CLI Switch	• `-datasheet_peakpower '<float>'`
Example (CLI)	`-datasheet_peakpower 1'`
Example (API)	`chip.set('datasheet', 'peakpower', 1)`

Device total peak power.

pin

cap

Description	Datasheet: pin capacitance
Type	(float,float,float)
Unit	F
Default Value	None
CLI Switch	• `-datasheet_pin_cap 'pin mode <(float, float,float)>'`
Example (CLI)	`-datasheet_pin_cap 'sclk global (1e-12, 1.2e-12, 1.5e-12)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'cap', 'global', (1e-12, 1.2e-12, 1.5e-12)`

Pin capacitance. Values are tuples of (min, typical, max).

complement

Description	Datasheet: pin complement
Type	str
Default Value	None
CLI Switch	• `-datasheet_pin_complement 'name mode <str>'`
Example (CLI)	`-datasheet_pin_complement 'ina global inb'`
Example (API)	`chip.set('datasheet', 'pin', 'ina', 'complement', 'global', 'inb')`

Pin complement specified on a per mode basis for differential signals.

dir

Description	Datasheet: pin direction
Type	enum
Allowed Values	• `input` • `output` • `inout`

continues on next page

Table 621 – continued from previous page

Default Value	None
CLI Switch	• -datasheet_pin_dir 'name mode <str>'
Example (CLI)	-datasheet_pin_dir 'clk global input'
Example (API)	chip.set('datasheet', 'pin', 'clk', 'dir', 'global', 'input')

Pin direction specified on a per mode basis. Acceptable pin directions include: input, output, inout.

ibias

Description	Datasheet: pin bias current
Type	(float,float,float)
Unit	A
Default Value	None
CLI Switch	• -datasheet_pin_ibias 'pin mode <(float,float,float)>'
Example (CLI)	-datasheet_pin_ibias 'sclk global (0.001, 0.0012, 0.0015)'
Example (API)	chip.set('datasheet', 'pin', 'sclk', 'ibias', 'global', (0.001, 0.0012, 0.0015)

Pin bias current. Values are tuples of (min, typical, max).

iinject

Description	Datasheet: pin injection current
Type	(float,float,float)
Unit	A
Default Value	None
CLI Switch	• -datasheet_pin_iinject 'pin mode <(float,float,float)>'
Example (CLI)	-datasheet_pin_iinject 'sclk global (0.001, 0.0012, 0.0015)'
Example (API)	chip.set('datasheet', 'pin', 'sclk', 'iinject', 'global', (0.001, 0.0012, 0.0015)

Pin injection current. Values are tuples of (min, typical, max).

ileakage

Description	Datasheet: pin leakage current
Type	(float,float,float)
Unit	A
Default Value	None
CLI Switch	• `-datasheet_pin_ileakage 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_ileakage 'sclk global (1e-06, 1.2e-06, 1.5e-06)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'ileakage', 'global', (1e-06, 1.2e-06, 1.5e-06)`

Pin leakage current. Values are tuples of (min, typical, max).

interface

Description	Datasheet: pin interface map
Type	[str]
Default Value	[]
CLI Switch	• `-datasheet_pin_interface 'name mode <str>'`
Example (CLI)	`-datasheet_pin_interface 'clk0 ddr4 CLKN'`
Example (API)	`chip.set('datasheet', 'pin', 'clk0', 'interface', 'ddr4', 'CLKN')`

Pin mapping to standardized interface names.

ioffset

Description	Datasheet: pin offset current
Type	(float,float,float)
Unit	A
Default Value	None

continues on next page

Table 626 – continued from previous page

CLI Switch	• `-datasheet_pin_ioffset 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_ioffset 'sclk global (0.001, 0.0012, 0.0015)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'ioffset', 'global', (0.001, 0.0012, 0.0015)`

Pin offset current. Values are tuples of (min, typical, max).

ioh

Description	Datasheet: pin output high current
Type	(float,float,float)
Unit	A
Default Value	None
CLI Switch	• `-datasheet_pin_ioh 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_ioh 'sclk global (0.01, 0.012, 0.015)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'ioh', 'global', (0.01, 0.012, 0.015)`

Pin output high current. Values are tuples of (min, typical, max).

iol

Description	Datasheet: pin output low current
Type	(float,float,float)
Unit	A
Default Value	None
CLI Switch	• `-datasheet_pin_iol 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_iol 'sclk global (0.01, 0.012, 0.015)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'iol', 'global', (0.01, 0.012, 0.015)`

Pin output low current. Values are tuples of (min, typical, max).

ishort

Description	Datasheet: pin short circuit current
Type	(float,float,float)
Unit	A
Default Value	None
CLI Switch	• `-datasheet_pin_ishort 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_ishort 'sclk global (0.001, 0.0012, 0.0015)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'ishort', 'global', (0.001, 0.0012, 0.0015)`

Pin short circuit current. Values are tuples of (min, typical, max).

isupply

Description	Datasheet: pin supply current
Type	(float,float,float)
Unit	A
Default Value	None
CLI Switch	• `-datasheet_pin_isupply 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_isupply 'sclk global (0.001, 0.012, 0.015)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'isupply', 'global', (0.001, 0.012, 0.015)`

Pin supply current. Values are tuples of (min, typical, max).

map

Description	Datasheet: pin map
Type	(float,float)
Unit	um
Default Value	None
CLI Switch	• `-datasheet_pin_map 'name bump <(float,float)>'`
Example (CLI)	`-datasheet_pin_map 'in0 B4 (100.0, 100.0)'`
Example (API)	`chip.set('datasheet', 'pin','in0','map', 'B4',(100.0, 100.0)`

Mapping of signal pin to physical package pin name and location. Power and ground signals usually map to multiple pins/bumps/balls. Pin locations specify the (x,y) center of the pin with respect to the centroid of the design/package.

power

Description	Datasheet: pin power consumption
Type	(float,float,float)
Unit	W
Default Value	None
CLI Switch	• `-datasheet_pin_power 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_power 'sclk global (1, 2, 3)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'power', 'global', (1, 2, 3)`

Pin power consumption. Values are tuples of (min, typical, max).

rdiff

Description	Datasheet: pin differential pair resistance
Type	(float,float,float)
Unit	Ohm
Default Value	None

continues on next page

Table 633 – continued from previous page

CLI Switch	• `-datasheet_pin_rdiff 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_rdiff 'sclk global (45, 50, 55)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'rdiff', 'global', (45, 50, 55)`

Pin differential pair resistance. Values are tuples of (min, typical, max).

rdown

Description	Datasheet: pin output pulldown resistance
Type	(float,float,float)
Unit	Ohm
Default Value	None
CLI Switch	• `-datasheet_pin_rdown 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_rdown 'sclk global (1000, 1200, 3000)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'rdown', 'global', (1000, 1200, 3000)`

Pin output pulldown resistance. Values are tuples of (min, typical, max).

resetvalue

Description	Datasheet: pin reset value
Type	enum
Allowed Values	• `weak1` • `weak0` • `strong0` • `strong1` • `highz`
Default Value	None

continues on next page

Table 635 – continued from previous page

CLI Switch	• `-datasheet_pin_resetvalue 'name mode <str>'`
Example (CLI)	`-datasheet_pin_resetvalue 'clk global weak1'`
Example (API)	`chip.set('datasheet', 'pin', 'clk', 'resetvalue', 'global', 'weak1')`

Pin reset value specified on a per mode basis.

rin

Description	Datasheet: pin input resistance
Type	(float,float,float)
Unit	Ohm
Default Value	None
CLI Switch	• `-datasheet_pin_rin 'pin mode <(float, float,float)>'`
Example (CLI)	`-datasheet_pin_rin 'sclk global (1000, 1200, 3000)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'rin', 'global', (1000, 1200, 3000)`

Pin input resistance. Values are tuples of (min, typical, max).

rup

Description	Datasheet: pin output pullup resistance
Type	(float,float,float)
Unit	Ohm
Default Value	None
CLI Switch	• `-datasheet_pin_rup 'pin mode <(float, float,float)>'`
Example (CLI)	`-datasheet_pin_rup 'sclk global (1000, 1200, 3000)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'rup', 'global', (1000, 1200, 3000)`

Pin output pullup resistance. Values are tuples of (min, typical, max).

rweakdown

Description	Datasheet: pin weak pulldown resistance
Type	(float,float,float)
Unit	Ohm
Default Value	None
CLI Switch	• `-datasheet_pin_rweakdown 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_rweakdown 'sclk global (1000, 1200, 3000)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'rweakdown', 'global', (1000, 1200, 3000)`

Pin weak pulldown resistance. Values are tuples of (min, typical, max).

rweakup

Description	Datasheet: pin weak pullup resistance
Type	(float,float,float)
Unit	Ohm
Default Value	None
CLI Switch	• `-datasheet_pin_rweakup 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_rweakup 'sclk global (1000, 1200, 3000)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'rweakup', 'global', (1000, 1200, 3000)`

Pin weak pullup resistance. Values are tuples of (min, typical, max).

standard

Description	Datasheet: pin standard
Type	[str]
Default Value	[]
CLI Switch	• `-datasheet_pin_standard 'name mode <str>'`
Example (CLI)	`-datasheet_pin_standard 'clk def LVCMOS'`
Example (API)	`chip.set('datasheet', 'pin', 'clk', 'standard', 'def', 'LVCMOS')`

Pin electrical signaling standard (LVDS, LVCMOS, TTL, ...).

tdelayf

Description	Datasheet: pin propagation delay (fall)
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_tdelayf 'pin mode relpin <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_tdelayf 'a glob clock (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'a', 'tdelayf', 'glob', 'ck', (1e-09, 2e-09, 4e-09)`

Pin propagation delay (fall) specified on a per pin, mode, and relpin basis. Values are tuples of (min, typical, max).

tdelayr

Description	Datasheet: pin propagation delay (rise)
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_tdelayr 'pin mode relpin <(float,float,float)>'`

continues on next page

Table 642 – continued from previous page

Example (CLI)	`-datasheet_pin_tdelayr 'a glob clock (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'a', 'tdelayr', 'glob', 'ck', (1e-09, 2e-09, 4e-09)`

Pin propagation delay (rise) specified on a per pin, mode, and relpin basis. Values are tuples of (min, typical, max).

tduty

Description	Datasheet: pin duty cycle
Type	(float,float,float)
Unit	%
Default Value	None
CLI Switch	• `-datasheet_pin_tduty 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_tduty 'sclk global (45, 50, 55)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'tduty', 'global', (45, 50, 55)`

Pin duty cycle. Values are tuples of (min, typical, max).

tfall

Description	Datasheet: pin fall transition
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_tfall 'pin mode relpin <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_tfall 'a glob clock (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'a', 'tfall', 'glob', 'ck', (1e-09, 2e-09, 4e-09)`

Pin fall transition specified on a per pin, mode, and relpin basis. Values are tuples of (min, typical, max).

thigh

Description	Datasheet: pin pulse width high
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_thigh 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_thigh 'sclk global (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'thigh', 'global', (1e-09, 2e-09, 4e-09)`

Pin pulse width high. Values are tuples of (min, typical, max).

thold

Description	Datasheet: pin hold time
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_thold 'pin mode relpin <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_thold 'a glob clock (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'a', 'thold', 'glob', 'ck', (1e-09, 2e-09, 4e-09)`

Pin hold time specified on a per pin, mode, and relpin basis. Values are tuples of (min, typical, max).

tjitter

Description	Datasheet: pin rms jitter
Type	(float,float,float)
Unit	s
Default Value	None

continues on next page

Table 647 – continued from previous page

CLI Switch	• `-datasheet_pin_tjitter 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_tjitter 'sclk global (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'tjitter', 'global', (1e-09, 2e-09, 4e-09)`

Pin rms jitter. Values are tuples of (min, typical, max).

tlow

Description	Datasheet: pin pulse width low
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_tlow 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_tlow 'sclk global (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'tlow', 'global', (1e-09, 2e-09, 4e-09)`

Pin pulse width low. Values are tuples of (min, typical, max).

tperiod

Description	Datasheet: pin minimum period
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_tperiod 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_tperiod 'sclk global (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'tperiod', 'global', (1e-09, 2e-09, 4e-09)`

Pin minimum period. Values are tuples of (min, typical, max).

tpulse

Description	Datasheet: pin pulse width
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_tpulse 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_tpulse 'sclk global (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'tpulse', 'global', (1e-09, 2e-09, 4e-09)`

Pin pulse width. Values are tuples of (min, typical, max).

trise

Description	Datasheet: pin rise transition
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_trise 'pin mode relpin <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_trise 'a glob clock (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'a', 'trise', 'glob', 'ck', (1e-09, 2e-09, 4e-09)`

Pin rise transition specified on a per pin, mode, and relpin basis. Values are tuples of (min, typical, max).

tsetup

Description	Datasheet: pin setup time
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_tsetup 'pin mode relpin <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_tsetup 'a glob clock (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'a', 'tsetup', 'glob', 'ck', (1e-09, 2e-09, 4e-09)`

Pin setup time specified on a per pin, mode, and relpin basis. Values are tuples of (min, typical, max).

tskew

Description	Datasheet: pin timing skew
Type	(float,float,float)
Unit	s
Default Value	None
CLI Switch	• `-datasheet_pin_tskew 'pin mode relpin <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_tskew 'a glob clock (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'a', 'tskew', 'glob', 'ck', (1e-09, 2e-09, 4e-09)`

Pin timing skew specified on a per pin, mode, and relpin basis. Values are tuples of (min, typical, max).

type

Description	Datasheet: pin type
Type	enum

continues on next page

Table 654 – continued from previous page

Allowed Values	• digital • analog • clock • supply • ground
Default Value	None
CLI Switch	• -datasheet_pin_type 'name mode <str>'
Example (CLI)	-datasheet_pin_type 'vdd global supply'
Example (API)	chip.set('datasheet', 'pin', 'vdd', 'type', 'global', 'supply')

Pin type specified on a per mode basis.

vcdm

Description	Datasheet: pin ESD charge device model voltage level
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• -datasheet_pin_vcdm 'pin mode <(float,float,float)>'
Example (CLI)	-datasheet_pin_vcdm 'sclk global (125, 150, 175)'
Example (API)	chip.set('datasheet', 'pin', 'sclk', 'vcdm', 'global', (125, 150, 175)

Pin ESD charge device model voltage level. Values are tuples of (min, typical, max).

vcm

Description	Datasheet: pin common mode voltage
Type	(float,float,float)
Unit	V
Default Value	None

continues on next page

Table 656 – continued from previous page

CLI Switch	• `-datasheet_pin_vcm 'pin mode <(float, float,float)>'`
Example (CLI)	`-datasheet_pin_vcm 'sclk global (0.3, 1.2, 1.6)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vcm', 'global', (0.3, 1.2, 1.6)`

Pin common mode voltage. Values are tuples of (min, typical, max).

vdiff

Description	Datasheet: pin differential voltage
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_pin_vdiff 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_vdiff 'sclk global (0.2, 0.3, 0.9)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vdiff', 'global', (0.2, 0.3, 0.9)`

Pin differential voltage. Values are tuples of (min, typical, max).

vhbm

Description	Datasheet: pin ESD human body model voltage level
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_pin_vhbm 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_vhbm 'sclk global (200, 250, 300)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vhbm', 'global', (200, 250, 300)`

Pin ESD human body model voltage level. Values are tuples of (min, typical, max).

vih

Description	Datasheet: pin high input voltage level
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_pin_vih 'pin mode <(float, float,float)>'`
Example (CLI)	`-datasheet_pin_vih 'sclk global (1.4, 1.8, 2.2)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vih', 'global', (1.4, 1.8, 2.2)`

Pin high input voltage level. Values are tuples of (min, typical, max).

vil

Description	Datasheet: pin low input voltage level
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_pin_vil 'pin mode <(float, float,float)>'`
Example (CLI)	`-datasheet_pin_vil 'sclk global (-0.2, 0, 1.0)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vil', 'global', (-0.2, 0, 1.0)`

Pin low input voltage level. Values are tuples of (min, typical, max).

vmax

Description	Datasheet: pin absolute maximum voltage
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_pin_vmax 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_vmax 'sclk global (0.2, 0.3, 0.9)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vmax', 'global', (0.2, 0.3, 0.9)`

Pin absolute maximum voltage. Values are tuples of (min, typical, max).

vmm

Description	Datasheet: pin ESD machine model voltage level
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_pin_vmm 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_vmm 'sclk global (100, 125, 150)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vmm', 'global', (100, 125, 150)`

Pin ESD machine model voltage level. Values are tuples of (min, typical, max).

vnoise

Description	Datasheet: pin random voltage noise
Type	(float,float,float)
Unit	V
Default Value	None

continues on next page

Table 663 – continued from previous page

CLI Switch	• `-datasheet_pin_vnoise 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_vnoise 'sclk global (0, 0.01, 0.1)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vnoise', 'global', (0, 0.01, 0.1)`

Pin random voltage noise. Values are tuples of (min, typical, max).

vnominal

Description	Datasheet: pin nominal operating voltage
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_pin_vnominal 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_vnominal 'sclk global (1.72, 1.8, 1.92)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vnominal', 'global', (1.72, 1.8, 1.92)`

Pin nominal operating voltage. Values are tuples of (min, typical, max).

voffset

Description	Datasheet: pin offset voltage
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_pin_voffset 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_voffset 'sclk global (0.2, 0.3, 0.9)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'voffset', 'global', (0.2, 0.3, 0.9)`

Pin offset voltage. Values are tuples of (min, typical, max).

voh

Description	Datasheet: pin high output voltage level
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_pin_voh 'pin mode <(float, float,float)>'`
Example (CLI)	`-datasheet_pin_voh 'sclk global (4.6, 4.8, 5.2)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'voh', 'global', (4.6, 4.8, 5.2)`

Pin high output voltage level. Values are tuples of (min, typical, max).

vol

Description	Datasheet: pin low output voltage level
Type	(float,float,float)
Unit	V
Default Value	None
CLI Switch	• `-datasheet_pin_vol 'pin mode <(float, float,float)>'`
Example (CLI)	`-datasheet_pin_vol 'sclk global (-0.2, 0, 0.2)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vol', 'global', (-0.2, 0, 0.2)`

Pin low output voltage level. Values are tuples of (min, typical, max).

vslew

Description	Datasheet: pin slew rate
Type	(float,float,float)
Unit	V/s
Default Value	None
CLI Switch	• `-datasheet_pin_vslew 'pin mode <(float,float,float)>'`
Example (CLI)	`-datasheet_pin_vslew 'sclk global (1e-09, 2e-09, 4e-09)'`
Example (API)	`chip.set('datasheet', 'pin', 'sclk', 'vslew', 'global', (1e-09, 2e-09, 4e-09)`

Pin slew rate. Values are tuples of (min, typical, max).

proc

arch

Description	Datasheet: processor architecture
Type	str
Default Value	None
CLI Switch	• `-datasheet_proc_arch 'name <str>'`
Example (CLI)	`-datasheet_proc_arch '0 RV64GC'`
Example (API)	`chip.set('datasheet', 'proc', name, 'arch', 'openfpga')`

Processor architecture.

archsize

Description	Datasheet: processor architecture size
Type	int
Default Value	None
CLI Switch	• `-datasheet_proc_archsize 'name <int>'`
Example (CLI)	`-datasheet_proc_archsize 'name 64'`

continues on next page

Table 670 – continued from previous page

Example (API)	`chip.set('datasheet', 'proc', name, 'archsize', 64)`

Processor metric: 64.

cores

Description	Datasheet: processor number of cores
Type	int
Default Value	None
CLI Switch	• `-datasheet_proc_cores 'name <int>'`
Example (CLI)	`-datasheet_proc_cores 'name 4'`
Example (API)	`chip.set('datasheet', 'proc', name, 'cores', 4)`

Processor metric: 4.

datatypes

Description	Datasheet: processor datatypes
Type	[enum]
Default Value	[]
CLI Switch	• `-datasheet_proc_datatypes 'name <str>'`
Example (CLI)	`-datasheet_proc_datatypes '0 int8'`
Example (API)	`chip.set('datasheet', 'proc', 'cpu', 'datatypes', 'int8')`

List of datatypes supported by the processor.

dcache

Description	Datasheet: processor l1 dcache size
Type	int
Unit	KB
Default Value	None
CLI Switch	• `-datasheet_proc_dcache 'name <int>'`
Example (CLI)	`-datasheet_proc_dcache 'name 32'`
Example (API)	`chip.set('datasheet', 'proc', name, 'dcache', 32)`

Processor metric: 32.

features

Description	Datasheet: processor features
Type	[str]
Default Value	[]
CLI Switch	• `-datasheet_proc_features 'name <str>'`
Example (CLI)	`-datasheet_proc_features '0 SIMD'`
Example (API)	`chip.set('datasheet','proc','cpu', 'features', 'SIMD')`

List of maker specified processor features.

fmax

Description	Datasheet: processor maximum frequency
Type	int
Unit	MHz
Default Value	None
CLI Switch	• `-datasheet_proc_fmax 'name <int>'`
Example (CLI)	`-datasheet_proc_fmax 'name 100'`
Example (API)	`chip.set('datasheet', 'proc', name, 'fmax', 100)`

Processor metric: 100.

icache

Description	Datasheet: processor l1 icache size
Type	int
Unit	KB
Default Value	None
CLI Switch	• `-datasheet_proc_icache 'name <int>'`
Example (CLI)	`-datasheet_proc_icache 'name 32'`
Example (API)	`chip.set('datasheet', 'proc', name, 'icache', 32)`

Processor metric: 32.

l2cache

Description	Datasheet: processor l2 cache size
Type	int
Unit	KB
Default Value	None
CLI Switch	• `-datasheet_proc_l2cache 'name <int>'`
Example (CLI)	`-datasheet_proc_l2cache 'name 1024'`
Example (API)	`chip.set('datasheet', 'proc', name, 'l2cache', 1024)`

Processor metric: 1024.

l3cache

Description	Datasheet: processor l3 cache size
Type	int
Unit	KB
Default Value	None
CLI Switch	• `-datasheet_proc_l3cache 'name <int>'`
Example (CLI)	`-datasheet_proc_l3cache 'name 1024'`
Example (API)	`chip.set('datasheet', 'proc', name, 'l3cache', 1024)`

Processor metric: 1024.

mults

Description	Datasheet: processor hard multiplier units per core
Type	int
Default Value	None
CLI Switch	• `-datasheet_proc_mults 'name <int>'`
Example (CLI)	`-datasheet_proc_mults 'name 100'`
Example (API)	`chip.set('datasheet', 'proc', name, 'mults', 100)`

Processor metric: 100.

nvm

Description	Datasheet: processor local non-volatile memory
Type	int
Unit	KB
Default Value	None
CLI Switch	• `-datasheet_proc_nvm 'name <int>'`
Example (CLI)	`-datasheet_proc_nvm 'name 128'`
Example (API)	`chip.set('datasheet', 'proc', name, 'nvm', 128)`

Processor metric: 128.

ops

Description	Datasheet: processor operations per cycle per core
Type	int
Default Value	None
CLI Switch	• `-datasheet_proc_ops 'name <int>'`
Example (CLI)	`-datasheet_proc_ops 'name 4'`
Example (API)	`chip.set('datasheet', 'proc', name, 'ops', 4)`

Processor metric: 4.

sram

Description	Datasheet: processor local sram
Type	int
Unit	KB
Default Value	None
CLI Switch	• `-datasheet_proc_sram 'name <int>'`
Example (CLI)	`-datasheet_proc_sram 'name 128'`
Example (API)	`chip.set('datasheet', 'proc', name, 'sram', 128)`

Processor metric: 128.

qual

Description	Datasheet: qualification
Type	[str]
Default Value	[]
CLI Switch	• `-datasheet_qual '<str>'`
Example (CLI)	`-datasheet_qual 'AEC-Q100'`
Example (API)	`chip.set('datasheet', 'qual', 'AEC-Q100')`

List of qualification standards passed by device.

ram

Description	Datasheet: total device RAM
Type	float
Unit	bits
Default Value	None
CLI Switch	• `-datasheet_ram '<float>'`
Example (CLI)	`-datasheet_ram 128'`
Example (API)	`chip.set('datasheet', 'ram', 128)`

Device total RAM.

series

Description	Datasheet: device series
Type	str
Default Value	None
CLI Switch	• `-datasheet_series '<str>'`
Example (CLI)	`-datasheet_series 'ZA0'`
Example (API)	`chip.set('datasheet', 'series', 'ZA0)`

Device series describing a family of devices or a singular device with multiple packages and/or qualification SKUs.

status

Description	Datasheet: product status
Type	enum
Allowed Values	• `preview` • `active` • `deprecated` • `last time buy` • `obsolete`
Default Value	None
CLI Switch	• `-datasheet_status '<str>'`
Example (CLI)	`-datasheet_status 'active'`
Example (API)	`chip.set('datasheet', 'status', 'active')`

Device production status.

thermal

rja

Description	Datasheet: thermal junction to ambient resistance
Type	float
Unit	C/W
Default Value	None

continues on next page

Table 687 – continued from previous page

CLI Switch	• `-datasheet_thermal_rja '<float>'`
Example (CLI)	`-datasheet_thermal_rja '30.4'`
Example (API)	`chip.set('datasheet', 'thermal', 'rja', 30.4)`

Device rja.

rjb

Description	Datasheet: thermal junction to board resistance
Type	float
Unit	C/W
Default Value	None
CLI Switch	• `-datasheet_thermal_rjb '<float>'`
Example (CLI)	`-datasheet_thermal_rjb '30.4'`
Example (API)	`chip.set('datasheet', 'thermal', 'rjb', 30.4)`

Device rjb.

rjcb

Description	Datasheet: thermal junction to case (bottom) resistance
Type	float
Unit	C/W
Default Value	None
CLI Switch	• `-datasheet_thermal_rjcb '<float>'`
Example (CLI)	`-datasheet_thermal_rjcb '30.4'`
Example (API)	`chip.set('datasheet', 'thermal', 'rjcb', 30.4)`

Device rjcb.

rjct

Description	Datasheet: thermal junction to case (top) resistance
Type	float
Unit	C/W
Default Value	None
CLI Switch	• `-datasheet_thermal_rjct '<float>'`
Example (CLI)	`-datasheet_thermal_rjct '30.4'`
Example (API)	`chip.set('datasheet', 'thermal', 'rjct', 30.4)`

Device rjct.

tjb

Description	Datasheet: thermal junction to bottom model
Type	float
Unit	C/W
Default Value	None
CLI Switch	• `-datasheet_thermal_tjb '<float>'`
Example (CLI)	`-datasheet_thermal_tjb '30.4'`
Example (API)	`chip.set('datasheet', 'thermal', 'tjb', 30.4)`

Device tjb.

tjt

Description	Datasheet: thermal junction to top model
Type	float
Unit	C/W
Default Value	None
CLI Switch	• `-datasheet_thermal_tjt '<float>'`
Example (CLI)	`-datasheet_thermal_tjt '30.4'`
Example (API)	`chip.set('datasheet', 'thermal', 'tjt', 30.4)`

Device tjt.

trl

Description	Datasheet: technology readiness level
Type	int
Default Value	None
CLI Switch	• `-datasheet_trl '<int>'`
Example (CLI)	`-datasheet_trl 9`
Example (API)	`chip.set('datasheet', 'trl', 9)`

Technology readiness level (TRL) of device. For more information, see: https://en.wikipedia.org/wiki/Technology_readiness_level

type

Description	Datasheet: part type
Type	enum
Allowed Values	• `digital` • `analog` • `ams` • `passive` • `soc` • `fpga` • `adc` • `dac` • `pmic` • `buck` • `boost` • `ldo` • `sram` • `dram` • `flash` • `rom` • `interface` • `clock` • `amplifier` • `filter` • `mixer` • `modulator` • `lna`
Default Value	None

continues on next page

Table 694 – continued from previous page

CLI Switch	• `-datasheet_type '<str>'`
Example (CLI)	`-datasheet_type 'digital'`
Example (API)	`chip.set('datasheet', 'type', 'digital')`

Part type.

design

Description	Design top module name
Type	str
Default Value	None
CLI Switch	• `-design <str>`
Example (CLI)	`-design hello_world`
Example (API)	`chip.set('design', 'hello_world')`

Name of the top level module or library. Required for all chip objects.

flowgraph

args

Description	Flowgraph: setup arguments
Type	[str]
Default Value	[]
CLI Switch	• `-flowgraph_args 'flow step index <str>'`
Example (CLI)	`-flowgraph_args 'asicflow cts 0 0'`
Example (API)	`chip.add('flowgraph', 'asicflow', 'cts', '0', 'args', '0')`

User specified flowgraph string arguments specified on a per step and per index basis.

goal

Description	Flowgraph: metric goals
Type	float
Default Value	None
CLI Switch	• `-flowgraph_goal 'flow step index metric <float>'`
Example (CLI)	`-flowgraph_goal 'asicflow cts 0 area_cells 1.0'`
Example (API)	`chip.set('flowgraph', 'asicflow', 'cts', '0', 'goal', 'errors', 0)`

Goals specified on a per step and per metric basis used to determine whether a certain task can be considered when merging multiple tasks at a minimum or maximum node. A task is considered failing if the absolute value of any of its metrics are larger than the goal for that metric, if set.

input

Description	Flowgraph: step input
Type	[(str,str)]
Default Value	[]
CLI Switch	• `-flowgraph_input 'flow step index <(str,str)>'`
Example (CLI)	`-flowgraph_input 'asicflow cts 0 (place, 0)'`
Example (API)	`chip.set('flowgraph', 'asicflow', 'cts', '0', 'input', ('place', '0'))`

A list of inputs for the current step and index, specified as a (step, index) tuple.

select

Description	Flowgraph: task select record
Type	[(str,str)]
Default Value	[]
CLI Switch	• `-flowgraph_select 'flow step index <(str,str)>'`

continues on next page

Table 699 – continued from previous page

Example (CLI)	`-flowgraph_select 'asicflow cts 0 (place, 42)'`
Example (API)	`chip.set('flowgraph', 'asicflow', 'cts', '0', 'select', ('place', '42'))`

List of selected inputs for the current step/index specified as (in_step, in_index) tuple.

status

Description	Flowgraph: task status
Type	enum
Allowed Values	• `pending` • `success` • `error`
Default Value	None
CLI Switch	• `-flowgraph_status 'flow step index <str>'`
Example (CLI)	`-flowgraph_status 'asicflow cts 10 success'`
Example (API)	`chip.set('flowgraph', 'asicflow', 'cts', '10', 'status', 'success')`

Parameter that tracks the status of a task. Valid values are:

- ”success”: task ran successfully
- ”error”: task failed with an error

An empty value indicates the task has not yet been completed.

task

Description	Flowgraph: task selection
Type	str
Default Value	None
CLI Switch	• `-flowgraph_task 'flow step index <str>'`
Example (CLI)	`-flowgraph_task 'asicflow myplace 0 place'`

continues on next page

Table 701 – continued from previous page

Example (API)	`chip.set('flowgraph', 'asicflow', 'myplace', '0', 'task', 'place')`

Name of the tool associated task used for step execution. Builtin task names include: minimum, maximum, join, verify, mux.

taskmodule

Description	Flowgraph: task module
Type	str
Default Value	None
CLI Switch	• `-flowgraph_taskmodule 'flow step index <str>'`
Example (CLI)	`-flowgraph_taskmodule 'asicflow place 0 siliconcompiler.tools.openroad.place'`
Example (API)	`chip.set('flowgraph', 'asicflow', 'place', '0', 'taskmodule', 'siliconcompiler.tools.openroad.place')`

Full python module name of the task module used for task setup and execution.

timeout

Description	Flowgraph: task timeout value
Type	float
Unit	s
Default Value	None
CLI Switch	• `-flowgraph_timeout 'flow step index <float>'`
Example (CLI)	`-flowgraph_timeout 'asicflow cts 0 3600'`
Example (API)	`chip.set('flowgraph', 'asicflow', 'cts', '0', 'timeout', 3600)`

Timeout value in seconds specified on a per step and per index basis. The flowgraph timeout value is compared against the wall time tracked by the SC runtime to determine if an operation should continue. Timeout values help in situations where 1.) an operation is stuck and may never finish. 2.) the operation progress has saturated and continued execution has a negative return on investment.

tool

Description	Flowgraph: tool selection
Type	str
Default Value	None
CLI Switch	• `-flowgraph_tool 'flow step index <str>'`
Example (CLI)	`-flowgraph_tool 'asicflow place 0 openroad'`
Example (API)	`chip.set('flowgraph', 'asicflow', 'place', '0', 'tool', 'openroad')`

Name of the tool name used for task execution. The 'tool' parameter is ignored for builtin tasks.

weight

Description	Flowgraph: metric weights
Type	float
Default Value	None
CLI Switch	• `-flowgraph_weight 'flow step index metric <float>'`
Example (CLI)	`-flowgraph_weight 'asicflow cts 0 area_cells 1.0'`
Example (API)	`chip.set('flowgraph', 'asicflow', 'cts', '0', 'weight', 'area_cells', 1.0)`

Weights specified on a per step and per metric basis used to give effective "goodness" score for a step by calculating the sum all step real metrics results by the corresponding per step weights.

fpga

board

Description	FPGA: board name
Type	str
Default Value	None
CLI Switch	• `-fpga_board <str>`

continues on next page

Table 706 – continued from previous page

Example (CLI)	`-fpga_board parallella`
Example (API)	`chip.set('fpga', 'board', 'parallella')`

Complete board name used as a device target by the FPGA compilation tool. The board name must be an exact string match to the partname hard coded within the FPGA EDA tool. The parameter is optional and can be used in place of a partname and pin constraints for some tools.

file

Description	FPGA: file
Type	[file]
Default Value	[]
CLI Switch	• `-fpga_file 'partname key <file>'`
Example (CLI)	`-fpga_file 'fpga64k file archfile my_arch.xml'`
Example (API)	`chip.set('fpga', 'fpga64k', 'file', 'archfile', 'my_arch.xml')`

Specify a file for the FPGA partname.

flash

Description	FPGA: flash enable
Type	bool
Default Value	False
CLI Switch	• `-fpga_flash <bool>`
Example (CLI)	`-fpga_flash`
Example (API)	`chip.set('fpga', 'flash', True)`

Specifies that the bitstream should be flashed in the board/device. The default is to load the bitstream into volatile memory (SRAM).

lutsize

Description	FPGA: lutsize
Type	int
Default Value	None
CLI Switch	• `-fpga_lutsize 'partname <int>'`
Example (CLI)	`-fpga_lutsize 'fpga64k 4'`
Example (API)	`chip.set('fpga', 'fpga64k', 'lutsize', '4')`

Specify the number of inputs in each lookup table (LUT) for the FPGA partname. For architectures with fracturable LUTs, this is the number of inputs of the unfractured LUT.

partname

Description	FPGA: part name
Type	str
Default Value	None
CLI Switch	• `-fpga_partname <str>`
Example (CLI)	`-fpga_partname fpga64k`
Example (API)	`chip.set('fpga', 'partname', 'fpga64k')`

Complete part name used as a device target by the FPGA compilation tool. The part name must be an exact string match to the partname hard coded within the FPGA EDA tool.

program

Description	FPGA: program enable
Type	bool
Default Value	False
CLI Switch	• `-fpga_program <bool>`
Example (CLI)	`-fpga_program`
Example (API)	`chip.set('fpga', 'program', True)`

Specifies that the bitstream should be loaded into an FPGA.

resources

brams

Description	FPGA: list of brams names
Type	[str]
Default Value	[]
CLI Switch	• `-fpga_resources_brams 'partname <str>'`
Example (CLI)	`-fpga_resources_brams 'fpga64k spram_64'`
Example (API)	`chip.set('fpga', 'fpga64k', 'resources', 'brams', 'spram_64')`

List of names for the brams available in the FPGA.

dsps

Description	FPGA: list of dsps names
Type	[str]
Default Value	[]
CLI Switch	• `-fpga_resources_dsps 'partname <str>'`
Example (CLI)	`-fpga_resources_dsps 'fpga64k multiplier'`
Example (API)	`chip.set('fpga', 'fpga64k', 'resources', 'dsps', 'multiplier')`

List of names for the dsps available in the FPGA.

registers

Description	FPGA: list of registers names
Type	[str]
Default Value	[]
CLI Switch	• `-fpga_resources_registers 'partname <str>'`
Example (CLI)	`-fpga_resources_registers 'fpga64k dff'`
Example (API)	`chip.set('fpga', 'fpga64k', 'resources', 'registers', 'dff')`

List of names for the registers available in the FPGA.

var

Description	FPGA: var
Type	[str]
Default Value	[]
CLI Switch	• `-fpga_var 'partname key <str>'`
Example (CLI)	`-fpga_var 'fpga64k channelwidth 100'`
Example (API)	`chip.set('fpga', 'fpga64k', 'var', 'channelwidth', '100')`

Specify a variable value for the FPGA partname.

vendor

Description	FPGA: vendor name
Type	str
Default Value	None
CLI Switch	• `-fpga_vendor 'partname <str>'`
Example (CLI)	`-fpga_vendor 'fpga64k acme'`
Example (API)	`chip.set('fpga', 'fpga64k', 'vendor', 'acme')`

Name of the FPGA vendor for the FPGA partname.

input

Description	Input: files
Type	[file]
Per step/index	optional
Default Value	[]
CLI Switch	• `-input 'fileset filetype <file>'`

continues on next page

Table 717 – continued from previous page

Example (CLI)	`-input 'rtl verilog hello_world.v'`
Example (API)	`chip.set('input', 'rtl', 'verilog', 'hello_world.v')`

List of files of type ('filetype') grouped as a named set ('fileset'). The exact names of filetypes and filesets must match the string names used by the tasks called during flowgraph execution. By convention, the fileset names should match the the name of the flowgraph being executed.

metric

averagepower

Description	Metric: averagepower
Type	float
Per step/index	required
Unit	mw
Default Value	None
CLI Switch	• `-metric_averagepower 'step index <float>'`
Example (CLI)	`-metric_averagepower 'place 0 0.01'`
Example (API)	`chip.set('metric', 'averagepower', 0.01, step='place', index=0)`

Metric tracking the average workload power of the design specified on a per step and index basis. Power metric depend heavily on the method being used for extraction: dynamic vs static, workload specification (vcd vs saif), power models, process/voltage/temperature. The power averagepower metric tries to capture the data that would usually be reflected inside a datasheet given the appropriate footnote conditions.

brams

Description	Metric: FPGA BRAM tiles used
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_brams 'step index <int>'`
Example (CLI)	`-metric_brams 'place 0 100'`
Example (API)	`chip.set('metric', 'brams', 100, step='place', index=0)`

Metric tracking the total FPGA BRAM tiles used used by the design as reported by the implementation tool. There is no standardized definition for this metric across vendors, so metric comparisons can generally only be done between runs on identical tools and device families.

buffers

Description	Metric: buffers
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_buffers 'step index <int>'`
Example (CLI)	`-metric_buffers 'place 0 100'`
Example (API)	`chip.set('metric', 'buffers', 50, step='place', index=0)`

Metric tracking the total number of buffer and inverter instances in the design on a per step and index basis.

cellarea

Description	Metric: cellarea
Type	float
Per step/index	required
Unit	um^2
Default Value	None
CLI Switch	• `-metric_cellarea 'step index <float>'`
Example (CLI)	`-metric_cellarea 'place 0 100.00'`
Example (API)	`chip.set('metric', 'cellarea', 100.00, step='place', index=0)`

Metric tracking the total cell area (ignoring fillers) occupied by the design.

cells

Description	Metric: cells
Type	int
Per step/index	required
Default Value	None

continues on next page

Table 722 – continued from previous page

CLI Switch	• -metric_cells 'step index <int>'
Example (CLI)	-metric_cells 'place 0 100'
Example (API)	chip.set('metric', 'cells', 50, step='place', index=0)

Metric tracking the total number of cell instances in the design on a per step and index basis.

coverage

Description	Metric: coverage
Type	float
Per step/index	required
Unit	%
Default Value	None
CLI Switch	• -metric_coverage 'step index <float>'
Example (CLI)	-metric_coverage 'place 0 99.9'
Example (API)	chip.set('metric', 'coverage', 99.9, step='place', index=0)

Metric tracking the test coverage in the design expressed as a percentage with 100 meaning full coverage. The meaning of the metric depends on the task being executed. It can refer to code coverage, feature coverage, stuck at fault coverage.

dozepower

Description	Metric: dozepower
Type	float
Per step/index	required
Unit	mw
Default Value	None
CLI Switch	• -metric_dozepower 'step index <float>'
Example (CLI)	-metric_dozepower 'place 0 0.01'
Example (API)	chip.set('metric', 'dozepower', 0.01, step='place', index=0)

Metric tracking the power consumed while in low frequency operating mode of the design specified on a per step and index basis. Power metric depend heavily on the method being used for extraction: dynamic vs static, workload

specification (vcd vs saif), power models, process/voltage/temperature. The power dozepower metric tries to capture the data that would usually be reflected inside a datasheet given the appropriate footnote conditions.

drvs

Description	Metric: total drvs
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_drvs 'step index <int>'`
Example (CLI)	`-metric_drvs 'dfm 0 0'`
Example (API)	`chip.set('metric', 'drvs', 0, step='dfm', index=0)`

Metric tracking the total number of design rule violations on a per step and index basis.

dsps

Description	Metric: FPGA DSP slices used
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_dsps 'step index <int>'`
Example (CLI)	`-metric_dsps 'place 0 100'`
Example (API)	`chip.set('metric', 'dsps', 100, step='place', index=0)`

Metric tracking the total FPGA DSP slices used used by the design as reported by the implementation tool. There is no standardized definition for this metric across vendors, so metric comparisons can generally only be done between runs on identical tools and device families.

errors

Description	Metric: total errors
Type	int
Per step/index	required
Default Value	None

continues on next page

Table 727 – continued from previous page

CLI Switch	• `-metric_errors 'step index <int>'`
Example (CLI)	`-metric_errors 'dfm 0 0'`
Example (API)	`chip.set('metric', 'errors', 0, step='dfm', index=0)`

Metric tracking the total number of errors on a per step and index basis.

exetime

Description	Metric: exetime
Type	float
Per step/index	required
Unit	s
Default Value	None
CLI Switch	• `-metric_exetime 'step index <float>'`
Example (CLI)	`-metric_exetime 'dfm 0 10.0'`
Example (API)	`chip.set('metric', 'exetime', 10.0, step='dfm', index=0)`

Metric tracking time spent by the EDA executable 'exe' on a per step and index basis. It does not include the SiliconCompiler runtime overhead or time waiting for I/O operations and inter-processor communication to complete.

fmax

Description	Metric: fmax
Type	float
Per step/index	required
Unit	Hz
Default Value	None
CLI Switch	• `-metric_fmax 'step index <float>'`
Example (CLI)	`-metric_fmax 'place 0 100e6'`
Example (API)	`chip.set('metric', 'fmax', 100e6, step='place', index=0)`

Metric tracking the maximum clock frequency on a per step and index basis.

holdpaths

Description	Metric: holdpaths
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_holdpaths 'step index <int>'`
Example (CLI)	`-metric_holdpaths 'place 0 10'`
Example (API)	`chip.set('metric', 'holdpaths', 10, step='place', index=0)`

Metric tracking the total number of timing paths violating hold constraints.

holdslack

Description	Metric: holdslack
Type	float
Per step/index	required
Unit	ns
Default Value	None
CLI Switch	• `-metric_holdslack 'step index <float>'`
Example (CLI)	`-metric_holdslack 'place 0 0.01'`
Example (API)	`chip.set('metric', 'holdslack', 0.01, step='place', index=0)`

Metric tracking the worst hold slack (positive or negative) on a per step and index basis.

holdtns

Description	Metric: holdtns
Type	float
Per step/index	required
Unit	ns
Default Value	None
CLI Switch	• `-metric_holdtns 'step index <float>'`
Example (CLI)	`-metric_holdtns 'place 0 0.01'`

continues on next page

Table 732 – continued from previous page

Example (API)	`chip.set('metric', 'holdtns', 0.01, step='place', index=0)`

Metric tracking the total negative hold slack (TNS) on a per step and index basis.

holdwns

Description	Metric: holdwns
Type	float
Per step/index	required
Unit	ns
Default Value	None
CLI Switch	• `-metric_holdwns 'step index <float>'`
Example (CLI)	`-metric_holdwns 'place 0 0.01'`
Example (API)	`chip.set('metric', 'holdwns', 0.01, step='place', index=0)`

Metric tracking the worst negative hold slack (positive values truncated to zero) on a per step and index basis.

idlepower

Description	Metric: idlepower
Type	float
Per step/index	required
Unit	mw
Default Value	None
CLI Switch	• `-metric_idlepower 'step index <float>'`
Example (CLI)	`-metric_idlepower 'place 0 0.01'`
Example (API)	`chip.set('metric', 'idlepower', 0.01, step='place', index=0)`

Metric tracking the power while not performing useful work of the design specified on a per step and index basis. Power metric depend heavily on the method being used for extraction: dynamic vs static, workload specification (vcd vs saif), power models, process/voltage/temperature. The power idlepower metric tries to capture the data that would usually be reflected inside a datasheet given the appropriate footnote conditions.

irdrop

Description	Metric: peak IR drop
Type	float
Per step/index	required
Unit	mv
Default Value	None
CLI Switch	• `-metric_irdrop 'step index <float>'`
Example (CLI)	`-metric_irdrop 'place 0 0.05'`
Example (API)	`chip.set('metric', 'irdrop', 0.05, step='place', index=0)`

Metric tracking the peak IR drop in the design based on extracted power and ground rail parasitics, library power models, and switching activity. The switching activity calculated on a per node basis is taken from one of three possible sources, in order of priority: VCD file, SAIF file, 'activityfactor' parameter.

leakagepower

Description	Metric: leakagepower
Type	float
Per step/index	required
Unit	mw
Default Value	None
CLI Switch	• `-metric_leakagepower 'step index <float>'`
Example (CLI)	`-metric_leakagepower 'place 0 0.01'`
Example (API)	`chip.set('metric', 'leakagepower', 0.01, step='place', index=0)`

Metric tracking the leakage power with rails active but without any dynamic switching activity of the design specified on a per step and index basis. Power metric depend heavily on the method being used for extraction: dynamic vs static, workload specification (vcd vs saif), power models, process/voltage/temperature. The power leakagepower metric tries to capture the data that would usually be reflected inside a datasheet given the appropriate footnote conditions.

logicdepth

Description	Metric: logic depth
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_logicdepth step index <int>`
Example (CLI)	`-metric_logicdepth 'place 0 8'`
Example (API)	`chip.set('metric', 'logicdepth', 8, step='place', index=0)`

Metric tracking the logic depth of the design. This is determined by the number of logic gates between the start of the critital timing path to the end of the path.

luts

Description	Metric: FPGA LUTs used
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_luts 'step index <int>'`
Example (CLI)	`-metric_luts 'place 0 100'`
Example (API)	`chip.set('metric', 'luts', 100, step='place', index=0)`

Metric tracking the total FPGA LUTs used used by the design as reported by the implementation tool. There is no standardized definition for this metric across vendors, so metric comparisons can generally only be done between runs on identical tools and device families.

macros

Description	Metric: macros
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_macros 'step index <int>'`
Example (CLI)	`-metric_macros 'place 0 100'`

continues on next page

Table 739 – continued from previous page

Example (API)	`chip.set('metric', 'macros', 50, step='place', index=0)`

Metric tracking the total number of macros in the design on a per step and index basis.

memory

Description	Metric: memory
Type	float
Per step/index	required
Unit	B
Default Value	None
CLI Switch	• `-metric_memory 'step index <float>'`
Example (CLI)	`-metric_memory 'dfm 0 10e9'`
Example (API)	`chip.set('metric', 'memory', 10e9, step='dfm', index=0)`

Metric tracking total peak program memory footprint on a per step and index basis.

nets

Description	Metric: nets
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_nets 'step index <int>'`
Example (CLI)	`-metric_nets 'place 0 100'`
Example (API)	`chip.set('metric', 'nets', 50, step='place', index=0)`

Metric tracking the total number of nets in the design on a per step and index basis.

overflow

Description	Metric: overflow
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_overflow 'step index <int>'`
Example (CLI)	`-metric_overflow 'place 0 0'`
Example (API)	`chip.set('metric', 'overflow', 50, step='place', index=0)`

Metric tracking the total number of overflow tracks for the routing on per step and index basis. Any non-zero number suggests an over congested design. To analyze where the congestion is occurring inspect the router log files for detailed per metal overflow reporting and open up the design to find routing hotspots.

peakpower

Description	Metric: peakpower
Type	float
Per step/index	required
Unit	mw
Default Value	None
CLI Switch	• `-metric_peakpower 'step index <float>'`
Example (CLI)	`-metric_peakpower 'place 0 0.01'`
Example (API)	`chip.set('metric', 'peakpower', 0.01, step='place', index=0)`

Metric tracking the worst case total peak power of the design specified on a per step and index basis. Power metric depend heavily on the method being used for extraction: dynamic vs static, workload specification (vcd vs saif), power models, process/voltage/temperature. The power peakpower metric tries to capture the data that would usually be reflected inside a datasheet given the appropriate footnote conditions.

pins

Description	Metric: pins
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_pins 'step index <int>'`
Example (CLI)	`-metric_pins 'place 0 100'`
Example (API)	`chip.set('metric', 'pins', 50, step='place', index=0)`

Metric tracking the total number of pins in the design on a per step and index basis.

registers

Description	Metric: registers
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_registers 'step index <int>'`
Example (CLI)	`-metric_registers 'place 0 100'`
Example (API)	`chip.set('metric', 'registers', 50, step='place', index=0)`

Metric tracking the total number of register instances in the design on a per step and index basis.

security

Description	Metric: security
Type	float
Per step/index	required
Unit	%
Default Value	None
CLI Switch	• `-metric_security 'step index <float>'`
Example (CLI)	`-metric_security 'place 0 100'`
Example (API)	`chip.set('metric', 'security', 100, step='place', index=0)`

Metric tracking the level of security (1/vulnerability) of the design. A completely secure design would have a score of 100. There is no absolute scale for the security metrics (like with power, area, etc) so the metric will be task and tool dependent.

setuppaths

Description	Metric: setuppaths
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_setuppaths 'step index <int>'`
Example (CLI)	`-metric_setuppaths 'place 0 10'`
Example (API)	`chip.set('metric', 'setuppaths', 10, step='place', index=0)`

Metric tracking the total number of timing paths violating setup constraints.

setupslack

Description	Metric: setupslack
Type	float
Per step/index	required
Unit	ns
Default Value	None
CLI Switch	• `-metric_setupslack 'step index <float>'`
Example (CLI)	`-metric_setupslack 'place 0 0.01'`
Example (API)	`chip.set('metric', 'setupslack', 0.01, step='place', index=0)`

Metric tracking the worst setup slack (positive or negative) on a per step and index basis.

setuptns

Description	Metric: setuptns
Type	float
Per step/index	required
Unit	ns
Default Value	None
CLI Switch	• `-metric_setuptns 'step index <float>'`
Example (CLI)	`-metric_setuptns 'place 0 0.01'`
Example (API)	`chip.set('metric', 'setuptns', 0.01, step='place', index=0)`

Metric tracking the total negative setup slack (TNS) on a per step and index basis.

setupwns

Description	Metric: setupwns
Type	float
Per step/index	required
Unit	ns
Default Value	None
CLI Switch	• `-metric_setupwns 'step index <float>'`
Example (CLI)	`-metric_setupwns 'place 0 0.01'`
Example (API)	`chip.set('metric', 'setupwns', 0.01, step='place', index=0)`

Metric tracking the worst negative setup slack (positive values truncated to zero) on a per step and index basis.

sleeppower

Description	Metric: sleeppower
Type	float
Per step/index	required
Unit	mw
Default Value	None
CLI Switch	• `-metric_sleeppower 'step index <float>'`

continues on next page

Table 751 – continued from previous page

Example (CLI)	`-metric_sleeppower 'place 0 0.01'`
Example (API)	`chip.set('metric', 'sleeppower', 0.01, step='place', index=0)`

Metric tracking the power consumed with some or all power rails gated off of the design specified on a per step and index basis. Power metric depend heavily on the method being used for extraction: dynamic vs static, workload specification (vcd vs saif), power models, process/voltage/temperature. The power sleeppower metric tries to capture the data that would usually be reflected inside a datasheet given the appropriate footnote conditions.

tasktime

Description	Metric: tasktime
Type	float
Per step/index	required
Unit	s
Default Value	None
CLI Switch	• `-metric_tasktime 'step index <float>'`
Example (CLI)	`-metric_tasktime 'dfm 0 10.0'`
Example (API)	`chip.set('metric', 'tasktime', 10.0, step='dfm', index=0)`

Metric tracking the total amount of time spent on a task from beginning to end, including data transfers and pre/post processing.

totalarea

Description	Metric: totalarea
Type	float
Per step/index	required
Unit	um^2
Default Value	None
CLI Switch	• `-metric_totalarea 'step index <float>'`
Example (CLI)	`-metric_totalarea 'place 0 100.00'`
Example (API)	`chip.set('metric', 'totalarea', 100.00, step='place', index=0)`

Metric tracking the total physical die area occupied by the design.

totaltime

Description	Metric: totaltime
Type	float
Per step/index	required
Unit	s
Default Value	None
CLI Switch	• `-metric_totaltime 'step index <float>'`
Example (CLI)	`-metric_totaltime 'dfm 0 10.0'`
Example (API)	`chip.set('metric', 'totaltime', 10.0, step='dfm', index=0)`

Metric tracking the total amount of time spent from the beginning of the run up to and including the current step and index.

transistors

Description	Metric: transistors
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_transistors 'step index <int>'`
Example (CLI)	`-metric_transistors 'place 0 100'`
Example (API)	`chip.set('metric', 'transistors', 50, step='place', index=0)`

Metric tracking the total number of transistors in the design on a per step and index basis.

unconstrained

Description	Metric: total unconstrained
Type	int
Per step/index	required
Default Value	None

continues on next page

Table 756 – continued from previous page

CLI Switch	• `-metric_unconstrained 'step index <int>'`
Example (CLI)	`-metric_unconstrained 'dfm 0 0'`
Example (API)	`chip.set('metric', 'unconstrained', 0, step='dfm', index=0)`

Metric tracking the total number of unconstrained timing paths on a per step and index basis.

utilization

Description	Metric: area utilization
Type	float
Per step/index	required
Unit	%
Default Value	None
CLI Switch	• `-metric_utilization step index <float>`
Example (CLI)	`-metric_utilization 'place 0 50.00'`
Example (API)	`chip.set('metric', 'utilization', 50.00, step='place', index=0)`

Metric tracking the area utilization of the design calculated as 100 * (cellarea/totalarea).

vias

Description	Metric: vias
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_vias 'step index <int>'`
Example (CLI)	`-metric_vias 'place 0 100'`
Example (API)	`chip.set('metric', 'vias', 50, step='place', index=0)`

Metric tracking the total number of vias in the design on a per step and index basis.

warnings

Description	Metric: total warnings
Type	int
Per step/index	required
Default Value	None
CLI Switch	• `-metric_warnings 'step index <int>'`
Example (CLI)	`-metric_warnings 'dfm 0 0'`
Example (API)	`chip.set('metric', 'warnings', 0, step='dfm', index=0)`

Metric tracking the total number of warnings on a per step and index basis.

wirelength

Description	Metric: wirelength
Type	float
Per step/index	required
Unit	um
Default Value	None
CLI Switch	• `-metric_wirelength 'step index <float>'`
Example (CLI)	`-metric_wirelength 'place 0 100.0'`
Example (API)	`chip.set('metric', 'wirelength', 50.0, step='place', index=0)`

Metric tracking the total wirelength of the design on a per step and index basis.

option

autoinstall

Description	Option: auto install packages
Type	bool
Default Value	False
CLI Switch	• `-autoinstall <bool>`
Example (CLI)	`-autoinstall true`

continues on next page

Table 761 – continued from previous page

Example (API)	`chip.set('option', 'autoinstall', True)`

Enables automatic installation of missing dependencies from the registry.

breakpoint

Description	Breakpoint list
Type	bool
Per step/index	optional
Default Value	False
CLI Switch	• `-breakpoint <bool>`
Example (CLI)	`-breakpoint true`
Example (API)	`chip.set('option, 'breakpoint', True)`

Set a breakpoint on specific steps. If the step is a TCL based tool, then the breakpoints stops the flow inside the EDA tool. If the step is a command line tool, then the flow drops into a Python interpreter.

builddir

Description	Build directory
Type	dir
Default Value	build
CLI Switch	• `-builddir <dir>`
Example (CLI)	`-builddir ./build_the_future`
Example (API)	`chip.set('option', 'builddir', './build_the_future')`

The default build directory is in the local ‘./build’ where SC was executed. The ‘builddir’ parameter can be used to set an alternate compilation directory path.

cache

Description	User cache directory
Type	file
Default Value	None
CLI Switch	• `-cache <file>`
Example (CLI)	`-cache /home/user/.sc/cache`
Example (API)	`chip.set('option', 'cache', '/home/user/.sc/cache')`

Filepath to cache used for package data sources. If the cache parameter is empty, ".sc/cache" directory in the user's home directory will be used.

cfg

Description	Configuration manifest
Type	[file]
Default Value	[]
CLI Switch	• `-cfg <file>`
Example (CLI)	`-cfg mypdk.json`
Example (API)	`chip.set('option', 'cfg', 'mypdk.json')`

List of filepaths to JSON formatted schema configuration manifests. The files are read in automatically when using the 'sc' command line application. In Python programs, JSON manifests can be merged into the current working manifest using the read_manifest() method.

clean

Description	Clean up after run
Type	bool
Default Value	False
CLI Switch	• `-clean <bool>`
Example (CLI)	`-clean`
Example (API)	`chip.set('option', 'clean', True)`

Clean up all intermediate and non essential files at the end of a task, leaving the following:

- log file
- replay.sh
- inputs/
- outputs/
- reports/
- autogenerated manifests
- any files generated by schema-specified regexes
- files specified by *['tool', <tool>, 'task', <task>, 'keep']*

cmdfile

Description	Design compilation command file
Type	[file]
Default Value	[]
CLI Switch	• `-f <file>` • `-cmdfile <file>`
Example (CLI)	`-f design.f` `-cmdfile design.f`
Example (API)	`chip.set('option', 'cmdfile', 'design.f')`

Read the specified file, and act as if all text inside it was specified as command line parameters. Supported by most verilog simulators including Icarus and Verilator. The format of the file is not strongly standardized. Support for comments and environment variables within the file varies and depends on the tool used. SC simply passes on the filepath to the tool executable.

continue

Description	Implementation continue-on-error
Type	bool
Per step/index	optional
Default Value	False
CLI Switch	• `-continue <bool>`
Example (CLI)	`-continue`
Example (API)	`chip.set('option', 'continue', True)`

Attempt to continue even when errors are encountered in the SC implementation. If errors are encountered, execution will halt before a run.

copyall

Description	Copy all inputs to build directory
Type	bool
Default Value	False
CLI Switch	• `-copyall <bool>`
Example (CLI)	`-copyall`
Example (API)	`chip.set('option', 'copyall', True)`

Specifies that all used files should be copied into the build directory, overriding the per schema entry copy settings.

credentials

Description	User credentials file
Type	file
Default Value	None
CLI Switch	• `-credentials <file>`
Example (CLI)	`-credentials /home/user/.sc/credentials`
Example (API)	`chip.set('option', 'credentials', '/home/user/.sc/credentials')`

Filepath to credentials used for remote processing. If the credentials parameter is empty, the remote processing client program tries to access the ".sc/credentials" file in the user's home directory. The file supports the following fields:

address=<server address>

port=<server port> (optional)

username=<user id> (optional)

password=<password / key used for authentication> (optional)

define

Description	Design pre-processor symbol
Type	[str]
Default Value	[]
CLI Switch	• `-D<str>` • `-define <str>`

continues on next page

Table 771 – continued from previous page

Example (CLI)	`-DCFG_ASIC=1` `-define CFG_ASIC=1`
Example (API)	`chip.set('option', 'define', 'CFG_ASIC=1')`

Symbol definition for source preprocessor.

dir

Description	Custom directories
Type	[dir]
Default Value	[]
CLI Switch	• `-dir 'key <dir>'`
Example (CLI)	`-dir 'openroad_tapcell ./tapcell.tcl'`
Example (API)	`chip.set('option', 'dir', 'openroad_files', './openroad_support/')`

List of named directories specified. Certain tools and reference flows require special parameters, this parameter should only be used for specifying directories that are not directly supported by the schema.

entrypoint

Description	Program entry point
Type	str
Default Value	None
CLI Switch	• `-entrypoint <str>`
Example (CLI)	`-entrypoint top`
Example (API)	`chip.set('option', 'entrypoint', 'top')`

Alternative entrypoint for compilation and simulation. The default entry point is 'design'.

env

Description	Environment variables
Type	str
Default Value	None
CLI Switch	• `-env 'key <str>'`
Example (CLI)	`-env 'PDK_HOME /disk/mypdk'`
Example (API)	`chip.set('option', 'env', 'PDK_HOME', '/disk/mypdk')`

Certain tools and reference flows require global environment variables to be set. These variables can be managed externally or specified through the env variable.

file

Description	Custom files
Type	[file]
Default Value	[]
CLI Switch	• `-file 'key <file>'`
Example (CLI)	`-file 'openroad_tapcell ./tapcell.tcl'`
Example (API)	`chip.set('option', 'file', 'openroad_tapcell', './tapcell.tcl')`

List of named files specified. Certain tools and reference flows require special parameters, this parameter should only be used for specifying files that are not directly supported by the schema.

flow

Description	Flow target
Type	str
Default Value	None
CLI Switch	• `-flow <str>`
Example (CLI)	`-flow asicflow`
Example (API)	`chip.set('option', 'flow', 'asicflow')`

Sets the flow for the current run. The flow name must match up with a 'flow' in the flowgraph

flowcontinue

Description	Flow continue-on-error
Type	bool
Per step/index	optional
Default Value	False
CLI Switch	• `-flowcontinue <bool>`
Example (CLI)	`-flowcontinue`
Example (API)	`chip.set('option', 'flowcontinue', True)`

Continue executing flow after a tool logs errors. The default behavior is to quit executing the flow if a task ends and the errors metric is greater than 0. Note that the flow will always cease executing if the tool returns a nonzero status code.

from

Description	Start flowgraph execution from
Type	[str]
Default Value	[]
CLI Switch	• `-from <str>`
Example (CLI)	`-from 'import'`
Example (API)	`chip.set('option', 'from', 'import')`

Inclusive list of steps to start execution from. The default is to start at all entry steps in the flow graph.

frontend

Description	Compilation frontend
Type	str
Default Value	verilog
CLI Switch	• `-frontend <str>`
Example (CLI)	`-frontend systemverilog`
Example (API)	`chip.set('option', 'frontend', 'systemverilog')`

Specifies the frontend that flows should use for importing and processing source files. Default option is 'verilog', also supports 'systemverilog' and 'chisel'. When using the Python API, this parameter must be configured before calling load_target().

hash

Description	Enable file hashing
Type	bool
Default Value	False
CLI Switch	• -hash <bool>
Example (CLI)	-hash
Example (API)	chip.set('option', 'hash', True)

Enables hashing of all inputs and outputs during compilation. The hash values are stored in the hashvalue field of the individual parameters.

idir

Description	Design search paths
Type	[dir]
Default Value	[]
CLI Switch	• +incdir+<dir> • -I <dir> • -idir <dir>
Example (CLI)	+incdir+./mylib -I ./mylib -idir ./mylib
Example (API)	chip.set('option', 'idir', './mylib')

Search paths to look for files included in the design using the `include statement.

jobincr

Description	Autoincrement jobname
Type	bool
Default Value	False
CLI Switch	• -jobincr <bool>
Example (CLI)	-jobincr
Example (API)	chip.set('option', 'jobincr', True)

Forces an auto-update of the jobname parameter if a directory matching the jobname is found in the build directory. If the jobname does not include a trailing digit, then the number '1' is added to the jobname before updating the jobname parameter.

jobinput

Description	Input job name
Type	str
Default Value	None
CLI Switch	• `-jobinput 'step index <str>'`
Example (CLI)	`-jobinput 'cts 0 job0'`
Example (API)	`chip.set('option', 'jobinput', 'cts, '0', 'job0')`

Specifies jobname inputs for the current run() on a per step and per index basis. During execution, the default behavior is to copy inputs from the current job.

jobname

Description	Job name
Type	str
Default Value	job0
CLI Switch	• `-jobname <str>`
Example (CLI)	`-jobname may1`
Example (API)	`chip.set('option', 'jobname', 'may1')`

Jobname during invocation of run(). The jobname combined with a defined director structure (<dir>/<design>/<jobname>/<step>/<index>) enables multiple levels of transparent job, step, and index introspection.

libext

Description	Design file extensions
Type	[str]
Default Value	[]
CLI Switch	• `+libext+<str>` • `-libext <str>`

continues on next page

Table 785 – continued from previous page

Example (CLI)	`+libext+sv` `-libext sv`
Example (API)	`chip.set('option', 'libext', 'sv')`

List of file extensions that should be used for finding modules. For example, if -y is specified as ./lib", and '.v' is specified as libext then the files ./lib/*.v ", will be searched for module matches.

loglevel

Description	Logging level
Type	enum
Per step/index	optional
Allowed Values	• `NOTSET` • `INFO` • `DEBUG` • `WARNING` • `ERROR` • `CRITICAL`
Default Value	INFO
CLI Switch	• `-loglevel <str>`
Example (CLI)	`-loglevel INFO`
Example (API)	`chip.set('option', 'loglevel', 'INFO')`

Provides explicit control over the level of debug logging printed. Valid entries include INFO, DEBUG, WARNING, ERROR.

metricoff

Description	Metric summary filter
Type	[str]
Default Value	[]
CLI Switch	• `-metricoff '<str>'`
Example (CLI)	`-metricoff 'wirelength'`
Example (API)	`chip.set('option', 'metricoff', 'wirelength')`

List of metrics to suppress when printing out the run summary.

mode

Description	Compilation mode
Type	enum
Allowed Values	• `asic` • `fpga` • `sim`
Default Value	None
CLI Switch	• `-mode <str>`
Example (CLI)	`-mode asic`
Example (API)	`chip.set('option', 'mode', 'asic')`

Sets the operating mode of the compiler. Valid modes are: asic: RTL to GDS ASIC compilation fpga: RTL to bitstream FPGA compilation sim: simulation to verify design and compilation

nice

Description	Tool execution scheduling priority
Type	int
Per step/index	optional
Default Value	None
CLI Switch	• `-nice <int>`
Example (CLI)	`-nice 5`
Example (API)	`chip.set('option', 'nice', 5)`

Sets the type of execution priority of each individual flowgraph steps. If the parameter is undefined, nice will not be used. For more information see Unix 'nice'.

nodisplay

Description	Headless execution
Type	bool
Default Value	False
CLI Switch	• `-nodisplay <bool>`
Example (CLI)	`-nodisplay`

continues on next page

Table 790 – continued from previous page

Example (API)	`chip.set('option', 'nodisplay', True)`

The '-nodisplay' flag prevents SiliconCompiler from opening GUI windows such as the final metrics report.

novercheck

Description	Disable version checking
Type	bool
Per step/index	optional
Default Value	False
CLI Switch	• `-novercheck <bool>`
Example (CLI)	`-novercheck`
Example (API)	`chip.set('option', 'novercheck', True)`

Disables strict version checking on all invoked tools if True. The list of supported version numbers is defined in the 'version' parameter in the 'tool' dictionary for each tool.

optmode

Description	Optimization mode
Type	str
Per step/index	optional
Default Value	O0
CLI Switch	• `-O<str>` • `-optmode <str>`
Example (CLI)	`-O3` `-optmode O3`
Example (API)	`chip.set('option', 'optmode', 'O3')`

The compiler has modes to prioritize run time and ppa. Modes include.

(O0) = Exploration mode for debugging setup (O1) = Higher effort and better PPA than O0 (O2) = Higher effort and better PPA than O1 (O3) = Signoff quality. Better PPA and higher run times than O2 (O4-O98) = Reserved (compiler/target dependent) (O99) = Experimental highest possible effort, may be unstable

param

Description	Design parameter
Type	str
Default Value	None
CLI Switch	• `-param 'name <str>'`
Example (CLI)	`-param 'N 64'`
Example (API)	`chip.set('option', 'param', 'N', '64')`

Sets a top verilog level design module parameter. The value is limited to basic data literals. The parameter override is passed into tools such as Verilator and Yosys. The parameters support Verilog integer literals (64'h4, 2'b0, 4) and strings. Name of the top level module to compile.

pdk

Description	PDK target
Type	str
Default Value	None
CLI Switch	• `-pdk <str>`
Example (CLI)	`-pdk freepdk45`
Example (API)	`chip.set('option', 'pdk', 'freepdk45')`

Target PDK used during compilation.

prune

Description	Prune flowgraph branches starting with
Type	[(str,str)]
Default Value	[]
CLI Switch	• `-prune 'node <(str,str)>'`
Example (CLI)	`-prune (syn,0)`
Example (API)	`chip.set('option', 'prune', ('syn', '0'))`

List of starting nodes for branches to be pruned. The default is to not prune any nodes/branches.

quiet

Description	Quiet execution
Type	bool
Per step/index	optional
Default Value	False
CLI Switch	• `-quiet <bool>`
Example (CLI)	`-quiet`
Example (API)	`chip.set('option', 'quiet', True)`

The -quiet option forces all steps to print to a log file. This can be useful with Modern EDA tools which print significant content to the screen.

relax

Description	Relax design checking
Type	bool
Default Value	False
CLI Switch	• `-relax <bool>`
Example (CLI)	`-relax`
Example (API)	`chip.set('option', 'relax', True)`

Global option specifying that tools should be lenient and suppress warnings that may or may not indicate real design issues. Extent of leniency is tool/task specific.

remote

Description	Enable remote processing
Type	bool
Default Value	False
CLI Switch	• `-remote <bool>`
Example (CLI)	`-remote`
Example (API)	`chip.set('option', 'remote', True)`

Sends job for remote processing if set to true. The remote option requires a credentials file to be placed in the home directory. Fore more information, see the credentials parameter.

resume

Description	Resume build
Type	bool
Default Value	False
CLI Switch	• `-resume <bool>`
Example (CLI)	`-resume`
Example (API)	`chip.set('option', 'resume', True)`

If results exist for current job, then don't re-run any steps that had at least one index run successfully. Useful for debugging a flow that failed partway through.

scheduler

cores

Description	Option: Scheduler core constraint
Type	int
Per step/index	optional
Default Value	None
CLI Switch	• `-cores <int>`
Example (CLI)	`-cores 48`
Example (API)	`chip.set('option', 'scheduler', 'cores', '48')`

Specifies the number CPU cores required to run the job. For the slurm scheduler, this translates to the '-c' switch. For more information, see the job scheduler documentation

defer

Description	Option: Scheduler start time
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-defer <str>`
Example (CLI)	`-defer 16:00`

continues on next page

Table 801 – continued from previous page

Example (API)	`chip.set('option', 'scheduler', 'defer', '16:00')`

Defer initiation of job until the specified time. The parameter is pass through string for remote job scheduler such as slurm. For more information about the exact format specification, see the job scheduler documentation. Examples of valid slurm specific values include: now+1hour, 16:00, 010-01-20T12:34:00. For more information, see the job scheduler documentation.

memory

Description	Option: Scheduler memory constraint
Type	int
Per step/index	optional
Unit	MB
Default Value	None
CLI Switch	• `-memory <int>`
Example (CLI)	`-memory 8000`
Example (API)	`chip.set('option', 'scheduler', 'memory', '8000')`

Specifies the amount of memory required to run the job, specified in MB. For the slurm scheduler, this translates to the '–mem' switch. For more information, see the job scheduler documentation

msgcontact

Description	Option: Message contact
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-msgcontact <str>`
Example (CLI)	`-msgcontact 'wile.e.coyote@acme.com'`
Example (API)	`chip.set('option', 'scheduler', 'msgcontact', 'wiley@acme.com')`

List of email addresses to message on a 'msgevent'. Support for email messages relies on job scheduler daemon support. For more information, see the job scheduler documentation.

msgevent

Description	Option: Message event trigger
Type	str
Per step/index	optional
Default Value	NONE
CLI Switch	• `-msgevent <str>`
Example (CLI)	`-msgevent ALL`
Example (API)	`chip.set('option', 'scheduler', 'msgevent', 'ALL')`

Directs job scheduler to send a message to the user when certain events occur during a task. Supported data types for SLURM include NONE, BEGIN, END, FAIL, ALL, TIME_LIMIT. For a list of supported event types, see the job scheduler documentation. For more information, see the job scheduler documentation.

name

Description	Option: Scheduler platform
Type	enum
Per step/index	optional
Allowed Values	• `slurm` • `lsf` • `sge`
Default Value	None
CLI Switch	• `-scheduler <str>`
Example (CLI)	`-scheduler slurm`
Example (API)	`chip.set('option', 'scheduler', 'name', 'slurm')`

Sets the type of job scheduler to be used for each individual flowgraph steps. If the parameter is undefined, the steps are executed on the same machine that the SC was launched on. If 'slurm' is used, the host running the 'sc' command must be running a 'slurmctld' daemon managing a Slurm cluster. Additionally, the build directory ('-dir') must be located in shared storage which can be accessed by all hosts in the cluster.

options

Description	Option: Scheduler arguments
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-scheduler_options <str>`
Example (CLI)	`-scheduler_options "--pty"`
Example (API)	`chip.set('option', 'scheduler', 'options', "--pty")`

Advanced/export options passed through unchanged to the job scheduler as-is. (The user specified options must be compatible with the rest of the scheduler parameters entered.(memory etc). For more information, see the job scheduler documentation.

queue

Description	Option: Scheduler queue
Type	str
Per step/index	optional
Default Value	None
CLI Switch	• `-queue <str>`
Example (CLI)	`-queue nightrun`
Example (API)	`chip.set('option', 'scheduler', 'queue', 'nightrun')`

Send the job to the specified queue. With slurm, this translates to 'partition'. The queue name must match the name of an existing job scheduler queue. For more information, see the job scheduler documentation

show

Description	Show layout
Type	bool
Default Value	False
CLI Switch	• `-show <bool>`
Example (CLI)	`-show`
Example (API)	`chip.set('option', 'show', True)`

Specifies that the final hardware layout should be shown after the compilation has been completed. The final layout and tool used to display the layout is flow dependent.

showtool

Description	Select data display tool
Type	str
Default Value	None
CLI Switch	• `-showtool 'filetype <str>'`
Example (CLI)	`-showtool 'gds klayout'`
Example (API)	`chip.set('option', 'showtool', 'gds', 'klayout')`

Selects the tool to use by the show function for displaying the specified filetype.

skipall

Description	Skip all tasks
Type	bool
Default Value	False
CLI Switch	• `-skipall <bool>`
Example (CLI)	`-skipall`
Example (API)	`chip.set('option', 'skipall', True)`

Skips the execution of all tools in run(), enabling a quick check of tool and setup without having to run through each step of a flow to completion.

skipcheck

Description	Skip manifest check
Type	bool
Default Value	False
CLI Switch	• `-skipcheck <bool>`

continues on next page

Table 811 – continued from previous page

Example (CLI)	`-skipcheck`
Example (API)	`chip.set('option', 'skipcheck', True)`

Bypasses the strict runtime manifest check. Can be used for accelerating initial bringup of tool/flow/pdk/libs targets. The flag should not be used for production compilation.

stackup

Description	Stackup target
Type	str
Default Value	None
CLI Switch	• `-stackup <str>`
Example (CLI)	`-stackup 2MA4MB2MC`
Example (API)	`chip.set('option', 'stackup', '2MA4MB2MC')`

Target stackup used during compilation. The stackup is required parameter for PDKs with multiple metal stackups.

strict

Description	Option: Strict checking
Type	bool
Default Value	False
CLI Switch	• `-strict <bool>`
Example (CLI)	`-strict true`
Example (API)	`chip.set('option', 'strict', True)`

Enable additional strict checking in the SC Python API. When this parameter is set to True, users must provide step and index keyword arguments when reading from parameters with the pernode field set to 'optional'.

target

Description	Compilation target
Type	str
Default Value	None
CLI Switch	• `-target <str>`
Example (CLI)	`-target freepdk45_demo`
Example (API)	`chip.set('option', 'target', 'freepdk45_demo')`

Sets a target module to be used for compilation. The target module must set up all parameters needed. The target module may load multiple flows and libraries.

timeout

Description	Option: Timeout value
Type	float
Unit	s
Default Value	None
CLI Switch	• `-timeout <float>`
Example (CLI)	`-timeout 3600`
Example (API)	`chip.set('option', 'timeout', 3600)`

Timeout value in seconds. The timeout value is compared against the wall time tracked by the SC runtime to determine if an operation should continue. The timeout value is also used by the jobscheduler to automatically kill jobs.

to

Description	End flowgraph execution with
Type	[str]
Default Value	[]
CLI Switch	• `-to <str>`
Example (CLI)	`-to 'syn'`
Example (API)	`chip.set('option', 'to', 'syn')`

Inclusive list of steps to end execution with. The default is to go to all exit steps in the flow graph.

trace

Description	Enable debug traces
Type	bool
Per step/index	optional
Default Value	False
CLI Switch	• `-trace <bool>`
Example (CLI)	`-trace`
Example (API)	`chip.set('option', 'trace', True)`

Enables debug tracing during compilation and/or runtime.

track

Description	Enable provenance tracking
Type	bool
Per step/index	optional
Default Value	False
CLI Switch	• `-track <bool>`
Example (CLI)	`-track`
Example (API)	`chip.set('option', 'track', True)`

Turns on tracking of all 'record' parameters during each task, otherwise only tool and runtime information will be recorded. Tracking will result in potentially sensitive data being recorded in the manifest so only turn on this feature if you have control of the final manifest.

uselambda

Description	Use lambda scaling
Type	bool
Default Value	False
CLI Switch	• `-uselambda <bool>`
Example (CLI)	`-uselambda true`
Example (API)	`chip.set('option', 'uselambda', True)`

Turns on lambda scaling of all dimensional constraints. (new value = value * ['pdk', 'lambda']).

var

Description	Custom variables
Type	[str]
Default Value	[]
CLI Switch	• -var 'key <str>'
Example (CLI)	-var 'openroad_place_density 0.4'
Example (API)	chip.set('option', 'var', 'openroad_place_density', '0.4')

List of key/value strings specified. Certain tools and reference flows require special parameters, this should only be used for specifying variables that are not directly supported by the SiliconCompiler schema.

vlib

Description	Design libraries
Type	[file]
Default Value	[]
CLI Switch	• -v <file> • -vlib <file>
Example (CLI)	-v './mylib.v' -vlib './mylib.v'
Example (API)	chip.set('option', 'vlib', './mylib.v')

List of library files to be read in. Modules found in the libraries are not interpreted as root modules.

ydir

Description	Design module search paths
Type	[dir]
Default Value	[]
CLI Switch	• -y <dir> • -ydir <dir>
Example (CLI)	-y './mylib' -ydir './mylib'
Example (API)	chip.set('option', 'ydir', './mylib')

Search paths to look for verilog modules found in the the source list. The import engine will look for modules inside files with the specified +libext+ param suffix.

output

Description	Output: files
Type	[file]
Per step/index	optional
Default Value	[]
CLI Switch	• `-output 'fileset filetype <file>'`
Example (CLI)	`-output 'rtl verilog hello_world.v'`
Example (API)	`chip.set('output', 'rtl', 'verilog', 'hello_world.v')`

List of files of type ('filetype') grouped as a named set ('fileset'). The exact names of filetypes and filesets must match the string names used by the tasks called during flowgraph execution. By convention, the fileset names should match the the name of the flowgraph being executed.

package

author

email

Description	Package: author email
Type	str
Default Value	None
CLI Switch	• `-package_author_email 'userid <str>'`
Example (CLI)	`-package_author_email 'wiley wiley@acme.com'`
Example (API)	`chip.set('package', 'author', 'wiley', 'email', 'wiley@acme.com')`

Package author email provided with full name as key and email as value.

location

Description	Package: author location
Type	str
Default Value	None
CLI Switch	• `-package_author_location 'userid <str>'`
Example (CLI)	`-package_author_location 'wiley wiley@acme.com'`
Example (API)	`chip.set('package', 'author', 'wiley', 'location', 'wiley@acme.com')`

Package author location provided with full name as key and location as value.

name

Description	Package: author name
Type	str
Default Value	None
CLI Switch	• `-package_author_name 'userid <str>'`
Example (CLI)	`-package_author_name 'wiley wiley@acme.com'`
Example (API)	`chip.set('package', 'author', 'wiley', 'name', 'wiley@acme.com')`

Package author name provided with full name as key and name as value.

organization

Description	Package: author organization
Type	str
Default Value	None
CLI Switch	• `-package_author_organization 'userid <str>'`
Example (CLI)	`-package_author_organization 'wiley wiley@acme.com'`

continues on next page

Table 827 – continued from previous page

Example (API)	`chip.set('package', 'author', 'wiley', 'organization', 'wiley@acme.com')`

Package author organization provided with full name as key and organization as value.

publickey

Description	Package: author publickey
Type	str
Default Value	None
CLI Switch	• `-package_author_publickey 'userid <str>'`
Example (CLI)	`-package_author_publickey 'wiley wiley@acme.com'`
Example (API)	`chip.set('package', 'author', 'wiley', 'publickey', 'wiley@acme.com')`

Package author publickey provided with full name as key and publickey as value.

username

Description	Package: author username
Type	str
Default Value	None
CLI Switch	• `-package_author_username 'userid <str>'`
Example (CLI)	`-package_author_username 'wiley wiley@acme.com'`
Example (API)	`chip.set('package', 'author', 'wiley', 'username', 'wiley@acme.com')`

Package author username provided with full name as key and username as value.

description

Description	Package: description
Type	str
Default Value	None
CLI Switch	• -package_description <str>
Example (CLI)	-package_description 'Yet another cpu'
Example (API)	chip.set('package', 'description', 'Yet another cpu')

Package short one line description for package managers and summary reports.

doc

datasheet

Description	Package: datasheet document
Type	[file]
Default Value	[]
CLI Switch	• -package_doc_datasheet <file>
Example (CLI)	-package_doc_datasheet datasheet.pdf
Example (API)	chip.set('package', 'doc', 'datasheet', 'datasheet.pdf')

Package list of datasheet documents.

homepage

Description	Package: documentation homepage
Type	str
Default Value	None
CLI Switch	• -package_doc_homepage <str>
Example (CLI)	-package_doc_homepage index.html
Example (API)	chip.set('package', 'doc', 'homepage', 'index.html')

Package documentation homepage. Filepath to design docs homepage. Complex designs can can include a long non standard list of documents dependent. A single html entry point can be used to present an organized documentation dashboard to the designer.

quickstart

Description	Package: quickstart document
Type	[file]
Default Value	[]
CLI Switch	• `-package_doc_quickstart <file>`
Example (CLI)	`-package_doc_quickstart quickstart.pdf`
Example (API)	`chip.set('package', 'doc', 'quickstart', 'quickstart.pdf')`

Package list of quickstart documents.

reference

Description	Package: reference document
Type	[file]
Default Value	[]
CLI Switch	• `-package_doc_reference <file>`
Example (CLI)	`-package_doc_reference reference.pdf`
Example (API)	`chip.set('package', 'doc', 'reference', 'reference.pdf')`

Package list of reference documents.

releasenotes

Description	Package: releasenotes document
Type	[file]
Default Value	[]
CLI Switch	• `-package_doc_releasenotes <file>`
Example (CLI)	`-package_doc_releasenotes releasenotes.pdf`

continues on next page

Table 835 – continued from previous page

Example (API)	`chip.set('package', 'doc', 'releasenotes', 'releasenotes.pdf')`

Package list of releasenotes documents.

signoff

Description	Package: signoff document
Type	[file]
Default Value	[]
CLI Switch	• `-package_doc_signoff <file>`
Example (CLI)	`-package_doc_signoff signoff.pdf`
Example (API)	`chip.set('package', 'doc', 'signoff', 'signoff.pdf')`

Package list of signoff documents.

testplan

Description	Package: testplan document
Type	[file]
Default Value	[]
CLI Switch	• `-package_doc_testplan <file>`
Example (CLI)	`-package_doc_testplan testplan.pdf`
Example (API)	`chip.set('package', 'doc', 'testplan', 'testplan.pdf')`

Package list of testplan documents.

tutorial

Description	Package: tutorial document
Type	[file]
Default Value	[]
CLI Switch	• -package_doc_tutorial <file>
Example (CLI)	-package_doc_tutorial tutorial.pdf
Example (API)	chip.set('package', 'doc', 'tutorial', 'tutorial.pdf')

Package list of tutorial documents.

userguide

Description	Package: userguide document
Type	[file]
Default Value	[]
CLI Switch	• -package_doc_userguide <file>
Example (CLI)	-package_doc_userguide userguide.pdf
Example (API)	chip.set('package', 'doc', 'userguide', 'userguide.pdf')

Package list of userguide documents.

keyword

Description	Package: keyword
Type	str
Default Value	None
CLI Switch	• -package_keyword <str>
Example (CLI)	-package_keyword cpu
Example (API)	chip.set('package', 'keyword', 'cpu')

Package keyword(s) used to characterize package.

license

Description	Package: license identifiers
Type	[str]
Default Value	[]
CLI Switch	• -package_license <str>
Example (CLI)	-package_license 'Apache-2.0'
Example (API)	chip.set('package', 'license', 'Apache-2.0')

Package list of SPDX license identifiers.

licensefile

Description	Package: license files
Type	[file]
Default Value	[]
CLI Switch	• -package_licensefile <file>
Example (CLI)	-package_licensefile './LICENSE'
Example (API)	chip.set('package', 'licensefile', './LICENSE')

Package list of license files for to be applied in cases when a SPDX identifier is not available. (eg. proprietary licenses).list of SPDX license identifiers.

organization

Description	Package: sponsoring organization
Type	[str]
Default Value	[]
CLI Switch	• -package_organization <str>
Example (CLI)	-package_organization 'humanity'
Example (API)	chip.set('package', 'organization', 'humanity')

Package sponsoring organization. The field can be left blank if not applicable.

source

path

Description	Package data source path
Type	str
Default Value	None
CLI Switch	• -package_source_path 'source <str>'
Example (CLI)	-package_source_path 'freepdk45_data ssh://git@github.com/siliconcompiler/freepdk45/'
Example (API)	chip.set('package', 'source', 'freepdk45_data', 'path', 'ssh://git@github.com/siliconcompiler/freepdk45/')

Package data source path, allowed paths:

- /path/on/network/drive
- file:///path/on/network/drive
- git+https://github.com/xyz/xyz
- git://github.com/xyz/xyz
- git+ssh://github.com/xyz/xyz
- ssh://github.com/xyz/xyz
- https://github.com/xyz/xyz/archive
- https://zeroasic.com/xyz.tar.gz
- python://siliconcompiler

ref

Description	Package data source reference
Type	str
Default Value	None
CLI Switch	• -package_source_ref 'source <str>'
Example (CLI)	-package_source_ref 'freepdk45_data 07ec4aa'
Example (API)	chip.set('package', 'source', 'freepdk45_data', 'ref', '07ec4aa')

Package data source reference

version

Description	Package: version
Type	str
Default Value	None
CLI Switch	• `-package_version <str>`
Example (CLI)	`-package_version 1.0`
Example (API)	`chip.set('package', 'version', '1.0')`

Package version. Can be a branch, tag, commit hash, or a semver compatible version.

pdk

aprtech

Description	PDK: APR technology files
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_aprtech 'pdkname tool stackup libarch filetype <file>'`
Example (CLI)	`-pdk_aprtech 'asap7 openroad M10 12t lef tech.lef'`
Example (API)	`chip.set('pdk', 'asap7', 'aprtech', 'openroad', 'M10', '12t', 'lef', 'tech.lef')`

Technology file containing setup information needed to enable DRC clean APR for the specified stackup, libarch, and format. The 'libarch' specifies the library architecture (e.g. library height). For example a PDK with support for 9 and 12 track libraries might have 'libarchs' called 9t and 12t. The standard filetype for specifying place and route design rules for a process node is through a 'lef' format technology file. The 'filetype' used in the aprtech is used by the tool specific APR TCL scripts to set up the technology parameters. Some tools may require additional files beyond the tech.lef file. Examples of extra file types include antenna, tracks, tapcell, viarules, em.

d0

Description	PDK: process defect density
Type	float
Default Value	None
CLI Switch	• `-pdk_d0 'pdkname <float>'`
Example (CLI)	`-pdk_d0 'asap7 0.1'`
Example (API)	`chip.set('pdk', 'asap7', 'd0', 0.1)`

Process defect density (d0) expressed as random defects per cm^2. The value is used to calculate yield losses as a function of area, which in turn affects the chip full factory costs. Two yield models are supported: Poisson (default), and Murphy. The Poisson based yield is calculated as dy = exp(-area * d0/100). The Murphy based yield is calculated as dy = ((1-exp(-area * d0/100))/(area * d0/100))^2.

density

Description	PDK: transistor density
Type	float
Default Value	None
CLI Switch	• `-pdk_density 'pdkname <float>'`
Example (CLI)	`-pdk_density 'asap7 100e6'`
Example (API)	`chip.set('pdk', 'asap7', 'density', 10e6)`

Approximate logic density expressed as # transistors / mm^2 calculated as: 0.6 * (Nand2 Transistor Count) / (Nand2 Cell Area) + 0.4 * (Register Transistor Count) / (Register Cell Area) The value is specified for a fixed standard cell library within a node and will differ depending on the library vendor, library track height and library type. The value can be used to to normalize the effective density reported for the design across different process nodes. The value can be derived from a variety of sources, including the PDK DRM, library LEFs, conference presentations, and public analysis.

devmodel

Description	PDK: device models
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_devmodel 'pdkname tool simtype stackup <file>'`

continues on next page

Table 850 – continued from previous page

Example (CLI)	`-pdk_devmodel 'asap7 xyce spice M10 asap7.sp'`
Example (API)	`chip.set('pdk', 'asap7', 'devmodel', 'xyce', 'spice', 'M10', 'asap7.sp')`

List of filepaths to PDK device models for different simulation purposes and for different tools. Examples of device model types include spice, aging, electromigration, radiation. An example of a 'spice' tool is xyce. Device models are specified on a per metal stack basis. Process nodes with a single device model across all stacks will have a unique parameter record per metal stack pointing to the same device model file. Device types and tools are dynamic entries that depend on the tool setup and device technology. Pseudo-standardized device types include spice, em (electromigration), and aging.

directory

Description	PDK: special directory
Type	[dir]
Default Value	[]
CLI Switch	• `-pdk_directory 'pdkname tool key stackup <dir>'`
Example (CLI)	`-pdk_directory 'asap7 xyce rfmodel M10 rftechdir'`
Example (API)	`chip.set('pdk', 'asap7', 'directory', 'xyce', 'rfmodel', 'M10', 'rftechdir')`

List of named directories specified on a per tool and per stackup basis. The parameter should only be used for specifying files that are not directly supported by the SiliconCompiler PDK schema.

display

Description	PDK: display file
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_display 'pdkname tool stackup <file>'`
Example (CLI)	`-pdk_display 'asap7 klayout M10 display.lyt'`
Example (API)	`chip.set('pdk', 'asap7', 'display', 'klayout', 'M10', 'display.cfg')`

Display configuration files describing colors and pattern schemes for all layers in the PDK. The display configuration file is entered on a stackup and tool basis.

doc

datasheet

Description	PDK: datasheet
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_doc_datasheet 'pdkname <file>'`
Example (CLI)	`-pdk_doc_datasheet 'asap7 datasheet.pdf'`
Example (API)	`chip.set('pdk', 'asap7', 'doc', 'datasheet', 'datasheet.pdf')`

Filepath to datasheet document.

homepage

Description	PDK: documentation homepage
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_doc_homepage 'pdkname <file>'`
Example (CLI)	`-pdk_doc_homepage 'asap7 index.html'`
Example (API)	`chip.set('pdk', 'asap7', 'doc', 'homepage', 'index.html')`

Filepath to PDK docs homepage. Modern PDKs can include tens or hundreds of individual documents. A single html entry point can be used to present an organized documentation dashboard to the designer.

install

Description	PDK: install
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_doc_install 'pdkname <file>'`

continues on next page

Table 855 – continued from previous page

Example (CLI)	`-pdk_doc_install 'asap7 install.pdf'`
Example (API)	`chip.set('pdk', 'asap7', 'doc', 'install', 'install.pdf')`

Filepath to install document.

quickstart

Description	PDK: quickstart
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_doc_quickstart 'pdkname <file>'`
Example (CLI)	`-pdk_doc_quickstart 'asap7 quickstart.pdf'`
Example (API)	`chip.set('pdk', 'asap7', 'doc', 'quickstart', 'quickstart.pdf')`

Filepath to quickstart document.

reference

Description	PDK: reference
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_doc_reference 'pdkname <file>'`
Example (CLI)	`-pdk_doc_reference 'asap7 reference.pdf'`
Example (API)	`chip.set('pdk', 'asap7', 'doc', 'reference', 'reference.pdf')`

Filepath to reference document.

releasenotes

Description	PDK: releasenotes
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_doc_releasenotes 'pdkname <file>'`
Example (CLI)	`-pdk_doc_releasenotes 'asap7 releasenotes.pdf'`
Example (API)	`chip.set('pdk', 'asap7', 'doc', 'releasenotes', 'releasenotes.pdf')`

Filepath to releasenotes document.

tutorial

Description	PDK: tutorial
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_doc_tutorial 'pdkname <file>'`
Example (CLI)	`-pdk_doc_tutorial 'asap7 tutorial.pdf'`
Example (API)	`chip.set('pdk', 'asap7', 'doc', 'tutorial', 'tutorial.pdf')`

Filepath to tutorial document.

userguide

Description	PDK: userguide
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_doc_userguide 'pdkname <file>'`
Example (CLI)	`-pdk_doc_userguide 'asap7 userguide.pdf'`
Example (API)	`chip.set('pdk', 'asap7', 'doc', 'userguide', 'userguide.pdf')`

Filepath to userguide document.

drc

runset

Description	PDK: DRC runset files
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_drc_runset 'pdkname tool stackup name <file>'`
Example (CLI)	`-pdk_drc_runset 'asap7 magic M10 basic $PDK/drc.rs'`
Example (API)	`chip.set('pdk', 'asap7', 'drc', 'runset', 'magic', 'M10', 'basic', '$PDK/drc.rs')`

Runset files for DRC task.

waiver

Description	PDK: DRC waiver files
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_drc_waiver 'pdkname tool stackup name <file>'`
Example (CLI)	`-pdk_drc_waiver 'asap7 magic M10 basic $PDK/drc.txt'`
Example (API)	`chip.set('pdk', 'asap7', 'drc', 'waiver', 'magic', 'M10', 'basic', '$PDK/drc.txt')`

Waiver files for DRC task.

edgemargin

Description	PDK: wafer edge keep-out margin
Type	float
Unit	mm
Default Value	None
CLI Switch	• `-pdk_edgemargin 'pdkname <float>'`

continues on next page

Table 863 – continued from previous page

Example (CLI)	`-pdk_edgemargin 'asap7 1'`
Example (API)	`chip.set('pdk', 'asap7', 'edgemargin', 1)`

Keep-out distance/margin from the edge inwards. The edge is prone to chipping and need special treatment that preclude placement of designs in this area. The edge value is used to calculate effective units per wafer/panel and full factory cost.

erc

runset

Description	PDK: ERC runset files
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_erc_runset 'pdkname tool stackup name <file>'`
Example (CLI)	`-pdk_erc_runset 'asap7 magic M10 basic $PDK/erc.rs'`
Example (API)	`chip.set('pdk', 'asap7', 'erc', 'runset', 'magic', 'M10', 'basic', '$PDK/erc.rs')`

Runset files for ERC task.

waiver

Description	PDK: ERC waiver files
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_erc_waiver 'pdkname tool stackup name <file>'`
Example (CLI)	`-pdk_erc_waiver 'asap7 magic M10 basic $PDK/erc.txt'`
Example (API)	`chip.set('pdk', 'asap7', 'erc', 'waiver', 'magic', 'M10', 'basic', '$PDK/erc.txt')`

Waiver files for ERC task.

file

Description	PDK: special file
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_file 'pdkname tool key stackup <file>'`
Example (CLI)	`-pdk_file 'asap7 xyce spice M10 asap7.sp'`
Example (API)	`chip.set('pdk', 'asap7', 'file', 'xyce', 'spice', 'M10', 'asap7.sp')`

List of named files specified on a per tool and per stackup basis. The parameter should only be used for specifying files that are not directly supported by the SiliconCompiler PDK schema.

fill

runset

Description	PDK: FILL runset files
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_fill_runset 'pdkname tool stackup name <file>'`
Example (CLI)	`-pdk_fill_runset 'asap7 magic M10 basic $PDK/fill.rs'`
Example (API)	`chip.set('pdk', 'asap7', 'fill', 'runset', 'magic', 'M10', 'basic', '$PDK/fill.rs')`

Runset files for FILL task.

waiver

Description	PDK: FILL waiver files
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_fill_waiver 'pdkname tool stackup name <file>'`

continues on next page

Table 868 – continued from previous page

Example (CLI)	`-pdk_fill_waiver 'asap7 magic M10 basic $PDK/fill.txt'`
Example (API)	`chip.set('pdk', 'asap7', 'fill', 'waiver', 'magic', 'M10', 'basic', '$PDK/fill.txt')`

Waiver files for FILL task.

foundry

Description	PDK: foundry name
Type	str
Default Value	None
CLI Switch	• `-pdk_foundry 'pdkname <str>'`
Example (CLI)	`-pdk_foundry 'asap7 virtual'`
Example (API)	`chip.set('pdk', 'asap7', 'foundry', 'virtual')`

Name of foundry corporation. Examples include intel, gf, tsmc, samsung, skywater, virtual. The 'virtual' keyword is reserved for simulated non-manufacturable processes.

hscribe

Description	PDK: horizontal scribe line width
Type	float
Unit	mm
Default Value	None
CLI Switch	• `-pdk_hscribe 'pdkname <float>'`
Example (CLI)	`-pdk_hscribe 'asap7 0.1'`
Example (API)	`chip.set('pdk', 'asap7', 'hscribe', 0.1)`

Width of the horizontal scribe line used during die separation. The process is generally completed using a mechanical saw, but can be done through combinations of mechanical saws, lasers, wafer thinning, and chemical etching in more advanced technologies. The value is used to calculate effective dies per wafer and full factory cost.

lambda

Description	PDK: Lambda value
Type	float
Default Value	1e-06
CLI Switch	• `-pdk_lambda 'pdkname <float>'`
Example (CLI)	`-pdk_lambda 'asap7 1e-06'`
Example (API)	`chip.set('pdk', 'asap7', 'lambda', 1e-06)`

Elementary distance unit used for scaling user specified physical schema parameters such as layout constraints.

layermap

Description	PDK: layer map file
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_layermap 'pdkname tool src dst stackup <file>'`
Example (CLI)	`-pdk_layermap 'asap7 klayout db gds M10 asap7.map'`
Example (API)	`chip.set('pdk', 'asap7', 'layermap', 'klayout', 'db', 'gds', 'M10', 'asap7.map')`

Files describing input/output mapping for streaming layout data from one format to another. A foundry PDK will include an official layer list for all user entered and generated layers supported in the GDS accepted by the foundry for processing, but there is no standardized layer definition format that can be read and written by all EDA tools. To ensure mask layer matching, key/value type mapping files are needed to convert EDA databases to/from GDS and to convert between different types of EDA databases. Layer maps are specified on a per metal stackup basis. The 'src' and 'dst' can be names of SC supported tools or file formats (like 'gds').

lvs

runset

Description	PDK: LVS runset files
Type	[file]
Default Value	[]

continues on next page

Table 873 – continued from previous page

CLI Switch	• `-pdk_lvs_runset 'pdkname tool stackup name <file>'`
Example (CLI)	`-pdk_lvs_runset 'asap7 magic M10 basic $PDK/lvs.rs'`
Example (API)	`chip.set('pdk', 'asap7', 'lvs', 'runset', 'magic', 'M10', 'basic', '$PDK/lvs.rs')`

Runset files for LVS task.

waiver

Description	PDK: LVS waiver files
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_lvs_waiver 'pdkname tool stackup name <file>'`
Example (CLI)	`-pdk_lvs_waiver 'asap7 magic M10 basic $PDK/lvs.txt'`
Example (API)	`chip.set('pdk', 'asap7', 'lvs', 'waiver', 'magic', 'M10', 'basic', '$PDK/lvs.txt')`

Waiver files for LVS task.

maxlayer

Description	PDK: maximum routing layer
Type	str
Default Value	None
CLI Switch	• `-pdk_maxlayer 'pdk stackup <str>'`
Example (CLI)	`-pdk_maxlayer 'asap7 2MA4MB2MC M8'`
Example (API)	`chip.set('pdk', 'asap7', 'maxlayer', 'MA4MB2MC', 'M8')`

Maximum metal layer to be used for automated place and route specified on a per stackup basis.

minlayer

Description	PDK: minimum routing layer
Type	str
Default Value	None
CLI Switch	• `-pdk_minlayer 'pdk stackup <str>'`
Example (CLI)	`-pdk_minlayer 'asap7 2MA4MB2MC M2'`
Example (API)	`chip.set('pdk', 'asap7', 'minlayer', '2MA4MB2MC', 'M2')`

Minimum metal layer to be used for automated place and route specified on a per stackup basis.

node

Description	PDK: process node
Type	float
Default Value	None
CLI Switch	• `-pdk_node 'pdkname <float>'`
Example (CLI)	`-pdk_node 'asap7 130'`
Example (API)	`chip.set('pdk', 'asap7', 'node', 130)`

Approximate relative minimum dimension of the process target specified in nanometers. The parameter is required for flows and tools that leverage the value to drive technology dependent synthesis and APR optimization. Node examples include 180, 130, 90, 65, 45, 32, 22 14, 10, 7, 5, 3.

panelsize

Description	PDK: panel size
Type	[(float,float)]
Unit	mm
Default Value	[]
CLI Switch	• `-pdk_panelsize 'pdkname <(float, float)>'`
Example (CLI)	`-pdk_panelsize 'asap7 (45.72,60.96)'`
Example (API)	`chip.set('pdk', 'asap7', 'panelsize', (45.72, 60.96))`

List of panel sizes supported in the manufacturing process.

pexmodel

Description	PDK: parasitic TCAD models
Type	[file]
Default Value	[]
CLI Switch	• `-pdk_pexmodel 'pdkname tool stackup corner <file>'`
Example (CLI)	`-pdk_pexmodel 'asap7 fastcap M10 max wire.mod'`
Example (API)	`chip.set('pdk', 'asap7', 'pexmodel', 'fastcap', 'M10', 'max', 'wire.mod')`

List of filepaths to PDK wire TCAD models used during automated synthesis, APR, and signoff verification. Pexmodels are specified on a per metal stack basis. Corner values depend on the process being used, but typically include nomenclature such as min, max, nominal. For exact names, refer to the DRM. Pexmodels are generally not standardized and specified on a per tool basis. An example of pexmodel type is 'fastcap'.

stackup

Description	PDK: metal stackups
Type	[str]
Default Value	[]
CLI Switch	• `-pdk_stackup 'pdkname <str>'`
Example (CLI)	`-pdk_stackup 'asap7 2MA4MB2MC'`
Example (API)	`chip.add('pdk', 'asap7', 'stackup', '2MA4MB2MC')`

List of all metal stackups offered in the process node. Older process nodes may only offer a single metal stackup, while advanced nodes offer a large but finite list of metal stacks with varying combinations of metal line pitches and thicknesses. Stackup naming is unique to a foundry, but is generally a long string or code. For example, a 10 metal stackup with two 1x wide, four 2x wide, and 4x wide metals, might be identified as 2MA4MB2MC, where MA, MB, and MC denote wiring layers with different properties (thickness, width, space). Each stackup will come with its own set of routing technology files and parasitic models specified in the pdk_pexmodel and pdk_aprtech parameters.

thickness

Description	PDK: unit thickness
Type	float
Unit	mm
Default Value	None
CLI Switch	• `-pdk_thickness 'pdkname stackup <float>'`
Example (CLI)	`-pdk_thickness 'asap7 2MA4MB2MC 1.57'`
Example (API)	`chip.set('pdk', 'asap7', 'thickness', '2MA4MB2MC', 1.57)`

Thickness of a manufactured unit specified on a per stackup.

unitcost

Description	PDK: unit cost
Type	float
Unit	USD
Default Value	None
CLI Switch	• `-pdk_unitcost 'pdkname <float>'`
Example (CLI)	`-pdk_unitcost 'asap7 10000'`
Example (API)	`chip.set('pdk', 'asap7', 'unitcost', 10000)`

Raw cost per unit shipped by the factory, not accounting for yield loss.

var

Description	PDK: special variable
Type	[str]
Default Value	[]
CLI Switch	• `-pdk_var 'pdkname tool stackup key <str>'`
Example (CLI)	`-pdk_var 'asap7 xyce modeltype M10 bsim4'`
Example (API)	`chip.set('pdk', 'asap7', 'var', 'xyce', 'modeltype', 'M10', 'bsim4')`

List of key/value strings specified on a per tool and per stackup basis. The parameter should only be used for specifying variables that are not directly supported by the SiliconCompiler PDK schema.

version

Description	PDK: version
Type	str
Default Value	None
CLI Switch	• `-pdk_version 'pdkname <str>'`
Example (CLI)	`-pdk_version 'asap7 1.0'`
Example (API)	`chip.set('pdk', 'asap7', 'version', '1.0')`

Alphanumeric string specifying the version of the PDK. Verification of correct PDK and IP versions is a hard ASIC tapeout require in all commercial foundries. The version number can be used for design manifest tracking and tapeout checklists.

vscribe

Description	PDK: vertical scribe line width
Type	float
Unit	mm
Default Value	None
CLI Switch	• `-pdk_vscribe 'pdkname <float>'`
Example (CLI)	`-pdk_vscribe 'asap7 0.1'`
Example (API)	`chip.set('pdk', 'asap7', 'vscribe', 0.1)`

Width of the vertical scribe line used during die separation. The process is generally completed using a mechanical saw, but can be done through combinations of mechanical saws, lasers, wafer thinning, and chemical etching in more advanced technologies. The value is used to calculate effective dies per wafer and full factory cost.

wafersize

Description	PDK: wafer size
Type	float
Unit	mm
Default Value	None
CLI Switch	• `-pdk_wafersize 'pdkname <float>'`
Example (CLI)	`-pdk_wafersize 'asap7 300'`
Example (API)	`chip.set('pdk', 'asap7', 'wafersize', 300)`

Wafer diameter used in wafer based manufacturing process. The standard diameter for leading edge manufacturing is 300mm. For older process technologies and specialty fabs, smaller diameters such as 200, 100, 125, 100 are common. The value is used to calculate dies per wafer and full factory chip costs.

record

arch

Description	Record: hardware architecture
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_arch 'step index <str>'`
Example (CLI)	`-record_arch 'dfm 0 x86_64'`
Example (API)	`chip.set('record', 'arch', 'x86_64', step='dfm', index=0)`

Record tracking the hardware architecture per step and index basis. (x86_64, rv64imafdc)

distro

Description	Record: distro name
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_distro 'step index <str>'`
Example (CLI)	`-record_distro 'dfm 0 ubuntu'`

continues on next page

Table 888 – continued from previous page

Example (API)	`chip.set('record', 'distro', 'ubuntu', step='dfm', index=0)`

Record tracking the distro name per step and index basis. (ubuntu, redhat, centos)

endtime

Description	Record: end time
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_endtime 'step index <str>'`
Example (CLI)	`-record_endtime 'dfm 0 2021-09-06 12:20:20'`
Example (API)	`chip.set('record', 'endtime', '2021-09-06 12:20:20', step='dfm', index=0)`

Record tracking the end time per step and index basis. Time is reported in the ISO 8601 format YYYY-MM-DD HR:MIN:SEC

ipaddr

Description	Record: IP address
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_ipaddr 'step index <str>'`
Example (CLI)	`-record_ipaddr 'dfm 0 <addr>'`
Example (API)	`chip.set('record', 'ipaddr', '<addr>', step='dfm', index=0)`

Record tracking the IP address per step and index basis.

kernelversion

Description	Record: O/S kernel version
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_kernelversion 'step index <str>'`
Example (CLI)	`-record_kernelversion 'dfm 0 5.11.0-34-generic'`
Example (API)	`chip.set('record', 'kernelversion', '5.11.0-34-generic', step='dfm', index=0)`

Record tracking the O/S kernel version per step and index basis. Used for platforms that support a distinction between os kernels and os distributions.

macaddr

Description	Record: MAC address
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_macaddr 'step index <str>'`
Example (CLI)	`-record_macaddr 'dfm 0 <addr>'`
Example (API)	`chip.set('record', 'macaddr', '<addr>', step='dfm', index=0)`

Record tracking the MAC address per step and index basis.

machine

Description	Record: machine name
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_machine 'step index <str>'`
Example (CLI)	`-record_machine 'dfm 0 carbon'`

continues on next page

Table 893 – continued from previous page

Example (API)	`chip.set('record', 'machine', 'carbon', step='dfm', index=0)`

Record tracking the machine name per step and index basis. (myhost, localhost, ...

osversion

Description	Record: O/S version
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_osversion 'step index <str>'`
Example (CLI)	`-record_osversion 'dfm 0 20.04.1-Ubuntu'`
Example (API)	`chip.set('record', 'osversion', '20.04.1-Ubuntu', step='dfm', index=0)`

Record tracking the O/S version per step and index basis. Since there is not standard version system for operating systems, extracting information from is platform dependent. For Linux based operating systems, the 'osversion' is the version of the distro.

platform

Description	Record: platform name
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_platform 'step index <str>'`
Example (CLI)	`-record_platform 'dfm 0 linux'`
Example (API)	`chip.set('record', 'platform', 'linux', step='dfm', index=0)`

Record tracking the platform name per step and index basis. (linux, windows, freebsd)

publickey

Description	Record: public key
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_publickey 'step index <str>'`
Example (CLI)	`-record_publickey 'dfm 0 <key>'`
Example (API)	`chip.set('record', 'publickey', '<key>', step='dfm', index=0)`

Record tracking the public key per step and index basis.

region

Description	Record: cloud region
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_region 'step index <str>'`
Example (CLI)	`-record_region 'dfm 0 US Gov Boston'`
Example (API)	`chip.set('record', 'region', 'US Gov Boston', step='dfm', index=0)`

Record tracking the cloud region per step and index basis. Recommended naming methodology:

- local: node is the local machine
- onprem: node in on-premises IT infrastructure
- public: generic public cloud
- govcloud: generic US government cloud
- <region>: cloud and entity specific region string name

remoteid

Description	Record: remote job ID
Type	str
Default Value	None
CLI Switch	• `-record_remoteid 'step index <str>'`
Example (CLI)	`-record_remoteid '0123456789abcdeffedcba9876543210'`
Example (API)	`chip.set('record', 'remoteid', '0123456789abcdeffedcba9876543210')`

Record tracking the job ID for a remote run.

scversion

Description	Record: software version
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_scversion 'step index <str>'`
Example (CLI)	`-record_scversion 'dfm 0 1.0'`
Example (API)	`chip.set('record', 'scversion', '1.0', step='dfm', index=0)`

Record tracking the software version per step and index basis. Version number for the SiliconCompiler software.

starttime

Description	Record: start time
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_starttime 'step index <str>'`
Example (CLI)	`-record_starttime 'dfm 0 2021-09-06 12:20:20'`
Example (API)	`chip.set('record', 'starttime', '2021-09-06 12:20:20', step='dfm', index=0)`

Record tracking the start time per step and index basis. Time is reported in the ISO 8601 format YYYY-MM-DD HR:MIN:SEC

toolargs

Description	Record: tool CLI arguments
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_toolargs 'step index <str>'`
Example (CLI)	`-record_toolargs 'dfm 0 -I include/ foo.v'`
Example (API)	`chip.set('record', 'toolargs', '-I include/ foo.v', step='dfm', index=0)`

Record tracking the tool CLI arguments per step and index basis. Arguments passed to tool via CLI.

toolpath

Description	Record: tool path
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_toolpath 'step index <str>'`
Example (CLI)	`-record_toolpath 'dfm 0 /usr/bin/openroad'`
Example (API)	`chip.set('record', 'toolpath', '/usr/bin/openroad', step='dfm', index=0)`

Record tracking the tool path per step and index basis. Full path to tool executable used to run this task.

toolversion

Description	Record: tool version
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_toolversion 'step index <str>'`
Example (CLI)	`-record_toolversion 'dfm 0 1.0'`
Example (API)	`chip.set('record', 'toolversion', '1.0', step='dfm', index=0)`

Record tracking the tool version per step and index basis. The tool version captured corresponds to the 'tool' parameter within the 'tool' dictionary.

userid

Description	Record: userid
Type	str
Per step/index	required
Default Value	None
CLI Switch	• `-record_userid 'step index <str>'`
Example (CLI)	`-record_userid 'dfm 0 wiley'`
Example (API)	`chip.set('record', 'userid', 'wiley', step='dfm', index=0)`

Record tracking the userid per step and index basis.

schemaversion

Description	Schema version number
Type	str
Default Value	0.40.4
CLI Switch	• `-schemaversion <str>`
Example (API)	`chip.get('schemaversion')`

SiliconCompiler schema version number.

tool

exe

Description	Tool: executable name
Type	str
Default Value	None
CLI Switch	• `-tool_exe 'tool <str>'`
Example (CLI)	`-tool_exe 'openroad openroad'`
Example (API)	`chip.set('tool', 'openroad', 'exe', 'openroad')`

Tool executable name.

format

Description	Tool: file format
Type	enum
Allowed Values	• `json` • `tcl` • `yaml`
Default Value	None
CLI Switch	• `-tool_format 'tool <str>'`
Example (CLI)	`-tool_format 'yosys tcl'`
Example (API)	`chip.set('tool', 'yosys', 'format', 'tcl')`

File format for tool manifest handoff.

licenseserver

Description	Tool: license servers
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_licenseserver 'name key <str>'`

continues on next page

Table 908 – continued from previous page

Example (CLI)	`-tool_licenseserver 'atask ACME_LICENSE 1700@server'`
Example (API)	`chip.set('tool', 'acme', 'licenseserver', 'ACME_LICENSE', '1700@server')`

Defines a set of tool specific environment variables used by the executable that depend on license key servers to control access. For multiple servers, separate each server by a 'colon'. The named license variable are read at runtime (run()) and the environment variables are set.

path

Description	Tool: executable path
Type	dir
Per step/index	optional
Default Value	None
CLI Switch	• `-tool_path 'tool <dir>'`
Example (CLI)	`-tool_path 'openroad /usr/local/bin'`
Example (API)	`chip.set('tool', 'openroad', 'path', '/usr/local/bin')`

File system path to tool executable. The path is prepended to the system PATH environment variable for batch and interactive runs. The path parameter can be left blank if the 'exe' is already in the environment search path.

sbom

Description	Tool: software BOM
Type	[file]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_sbom 'tool version <file>'`
Example (CLI)	`-tool_sbom 'yosys 1.0.1 ys_sbom.json'`
Example (API)	`chip.set('tool', 'yosys', 'sbom', '1.0', 'ys_sbom.json')`

Paths to software bill of material (SBOM) document file of the tool specified on a per version basis. The SBOM includes critical package information about the tool including the list of included components, licenses, and copyright. The SBOM file is generally provided as in a a standardized open data format such as SPDX.

task

continue

Description	Task: continue option
Type	bool
Per step/index	optional
Default Value	False
CLI Switch	• `-tool_task_continue 'tool task <bool>'`
Example (CLI)	`-tool_task_continue 'verilator lint true'`
Example (API)	`chip.set('tool', 'verilator', 'task', 'lint', 'continue', True)`

Directs flow to continue even if errors are encountered during task. The default behavior is for SC to exit on error.

dir

Description	Task: setup directories
Type	[dir]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_dir 'tool task key <dir>'`
Example (CLI)	`-tool_task_dir 'verilator compile cincludes include'`
Example (API)	`chip.set('tool', 'verilator', 'task', 'compile', 'dir', 'cincludes', 'include')`

Paths to user supplied directories mapped to keys. Keys must match what's expected by the task/reference script consuming the directory.

env

Description	Task: environment variables
Type	str
Per step/index	optional
Default Value	None

continues on next page

Table 913 – continued from previous page

CLI Switch	• `-tool_task_env 'tool task env <str>'`
Example (CLI)	`-tool_task_env 'openroad cts MYVAR 42'`
Example (API)	`chip.set('tool', 'openroad', 'task', 'cts', 'env', 'MYVAR', '42')`

Environment variables to set for individual tasks. Keys and values should be set in accordance with the task's documentation. Most tasks do not require extra environment variables to function.

file

Description	Task: setup files
Type	[file]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_file 'tool task key <file>'`
Example (CLI)	`-tool_task_file 'openroad floorplan macroplace macroplace.tcl'`
Example (API)	`chip.set('tool', 'openroad', 'task', 'floorplan', 'file', 'macroplace', 'macroplace.tcl')`

Paths to user supplied files mapped to keys. Keys and filetypes must match what's expected by the task/reference script consuming the file.

input

Description	Task: inputs
Type	[file]
Per step/index	required
Default Value	[]
CLI Switch	• `-tool_task_input 'tool task <file>'`
Example (CLI)	`-tool_task_input 'openroad place place 0 oh_add.def'`
Example (API)	`chip.set('tool', 'openroad', 'task', 'place', 'input', 'oh_add.def', step='place', index='0')`

List of data files to be copied from previous flowgraph steps 'output' directory. The list of steps to copy files from is defined by the list defined by the dictionary key ['flowgraph', step, index, 'input']. All files must be available for flow to continue. If a file is missing, the program exists on an error.

keep

Description	Task: files to keep
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_keep 'tool task <str>'`
Example (CLI)	`-tool_task_keep 'surelog import slp_all'`
Example (API)	`chip.set('tool', 'surelog', 'task', 'import', 'keep', 'slpp_all')`

Names of additional files and directories in the work directory that should be kept when *['option', 'clean']* is true.

option

Description	Task: executable options
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_option 'tool task <str>'`
Example (CLI)	`-tool_task_option 'openroad cts -no_init'`
Example (API)	`chip.set('tool', 'openroad', 'task', 'cts', 'option', '-no_init')`

List of command line options for the task executable, specified on a per task and per step basis. Options must not include spaces. For multiple argument options, each option is a separate list element.

output

Description	Task: outputs
Type	[file]
Per step/index	required
Default Value	[]
CLI Switch	• `-tool_task_output 'tool task <file>'`
Example (CLI)	`-tool_task_output 'openroad place place 0 oh_add.def'`
Example (API)	`chip.set('tool', 'openroad', 'task', 'place', 'output', 'oh_add.def', step='place', index='0')`

List of data files written to the 'output' directory of the tool/task/step/index used in the keypath. All files must be available for flow to continue. If a file is missing, the program exists on an error.

postscript

Description	Task: post-step script
Type	[file]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_postscript 'tool task <file>'`
Example (CLI)	`-tool_task_postscript 'yosys syn syn_post.tcl'`
Example (API)	`chip.set('tool', 'yosys', 'task', 'syn_asic', 'postscript', 'syn_post.tcl')`

Path to a user supplied script to execute after the main execution stage of the step but before the design is saved. Exact entry point depends on the step and main script being executed. An example of a postscript entry point would be immediately after global placement.

prescript

Description	Task: pre-step script
Type	[file]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_prescript 'tool task <file>'`
Example (CLI)	`-tool_task_prescript 'yosys syn syn_pre.tcl'`
Example (API)	`chip.set('tool', 'yosys', 'task', 'syn_asic', 'prescript', 'syn_pre.tcl')`

Path to a user supplied script to execute after reading in the design but before the main execution stage of the step. Exact entry point depends on the step and main script being executed. An example of a prescript entry point would be immediately before global placement.

refdir

Description	Task: script directory
Type	[dir]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_refdir 'tool task <dir>'`
Example (CLI)	`-tool_task_refdir 'yosys syn ./myref'`
Example (API)	`chip.set('tool', 'yosys', 'task', 'syn_asic', 'refdir', './myref')`

Path to directories containing reference flow scripts, specified on a per step and index basis.

regex

Description	Task: regex filter
Type	[str]
Per step/index	optional
Default Value	[]

continues on next page

Table 922 – continued from previous page

CLI Switch	• `-tool_task_regex 'tool task suffix <str>'`
Example (CLI)	`-tool_task_regex 'openroad place errors "-v ERROR"'`
Example (API)	`chip.set('tool', 'openroad', 'task', 'place', 'regex', 'errors', '-v ERROR')`

A list of piped together grep commands. Each entry represents a set of command line arguments for grep including the regex pattern to match. Starting with the first list entry, each grep output is piped into the following grep command in the list. Supported grep options include `-v` and `-e`. Patterns starting with "-" should be directly preceded by the `-e` option. The following example illustrates the concept.

UNIX grep:

```
$ grep WARNING place.log | grep -v "bbox" > place.warnings
```

SiliconCompiler:

```
chip.set('task', 'openroad', 'regex', 'place', '0', 'warnings',
         ["WARNING", "-v bbox"])
```

The "errors" and "warnings" suffixes are special cases. When set, the number of matches found for these regexes will be added to the errors and warnings metrics for the task, respectively. This will also cause the logfile to be added to the *['tool', <tool>, 'task', <task>, 'report', ...]* parameter for those metrics, if not already present.

report

Description	Task: reports
Type	[file]
Per step/index	required
Default Value	[]
CLI Switch	• `-tool_task_report 'tool task metric <file>'`
Example (CLI)	`-tool_task_report 'openroad place holdtns place 0 place.log'`
Example (API)	`chip.set('tool', 'openroad', 'task', 'place', 'report', 'holdtns', 'place.log', step='place', index='0')`

List of report files associated with a specific 'metric'. The file path specified is relative to the run directory of the current task.

require

Description	Task: parameter requirements
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_require 'tool task <str>'`
Example (CLI)	`-tool_task_require 'openroad cts design'`
Example (API)	`chip.set('tool', 'openroad', 'task', 'cts', 'require', 'design')`

List of keypaths to required task parameters. The list is used by check_manifest() to verify that all parameters have been set up before step execution begins.

script

Description	Task: entry script
Type	[file]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_script 'tool task <file>'`
Example (CLI)	`-tool_task_script 'yosys syn syn.tcl'`
Example (API)	`chip.set('tool', 'yosys', 'task', 'syn_asic', 'script', 'syn.tcl')`

Path to the entry script called by the executable specified on a per task and per step basis.

stderr

destination

Description	Task: Destination for stderr
Type	str
Per step/index	optional
Default Value	log
CLI Switch	• `-tool_task_stderr_destination 'tool task <str>'`

continues on next page

Table 926 – continued from previous page

Example (CLI)	`-tool_task_stderr_destination 'ghdl import log'`
Example (API)	`chip.set('tool', ghdl', 'task', 'import', 'stderr', 'destination', 'log')`

Defines where to direct the output generated over stderr. Supported options are: none: the stream generated to STDERR is ignored log: the generated stream is stored in <step>.<suffix>; if not in quiet mode, it is additionally dumped to the display output: the generated stream is stored in outputs/<design>.<suffix>

suffix

Description	Task: File suffix for redirected stderr
Type	str
Per step/index	optional
Default Value	log
CLI Switch	• `-tool_task_stderr_suffix 'tool task <str>'`
Example (CLI)	`-tool_task_stderr_suffix 'ghdl import log'`
Example (API)	`chip.set('tool', 'ghdl', 'task', 'import', 'stderr', 'suffix', 'log')`

Specifies the file extension for the content redirected from stderr.

stdout

destination

Description	Task: Destination for stdout
Type	str
Per step/index	optional
Default Value	log
CLI Switch	• `-tool_task_stdout_destination 'tool task <str>'`
Example (CLI)	`-tool_task_stdout_destination 'ghdl import log'`
Example (API)	`chip.set('tool', 'ghdl', 'task', 'import', 'stdout', 'destination', 'log')`

Defines where to direct the output generated over stdout. Supported options are: none: the stream generated to STDOUT is ignored log: the generated stream is stored in <step>.<suffix>; if not in quiet mode, it is additionally dumped to the display output: the generated stream is stored in outputs/<design>.<suffix>

suffix

Description	Task: File suffix for redirected stdout
Type	str
Per step/index	optional
Default Value	log
CLI Switch	• `-tool_task_stdout_suffix 'tool task <str>'`
Example (CLI)	`-tool_task_stdout_suffix 'ghdl import log'`
Example (API)	`chip.set('tool', ghdl', 'task', 'import', 'stdout', 'suffix', 'log')`

Specifies the file extension for the content redirected from stdout.

threads

Description	Task: thread parallelism
Type	int
Per step/index	optional
Default Value	None
CLI Switch	• `-tool_task_threads 'tool task <int>'`
Example (CLI)	`-tool_task_threads 'magic drc 64'`
Example (API)	`chip.set('tool', 'magic', 'task', 'drc', 'threads', '64')`

Thread parallelism to use for execution specified on a per task and per step basis. If not specified, SC queries the operating system and sets the threads based on the maximum thread count supported by the hardware.

var

Description	Task: script variables
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_var 'tool task key <str>'`
Example (CLI)	`-tool_task_var 'openroad cts myvar 42'`
Example (API)	`chip.set('tool', 'openroad', 'task', 'cts', 'var', 'myvar', '42')`

Task script variables specified as key value pairs. Variable names and value types must match the name and type of task and reference script consuming the variable.

warningoff

Description	Task: warning filter
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_task_warningoff 'tool task <str>'`
Example (CLI)	`-tool_task_warningoff 'verilator lint COMBDLY'`
Example (API)	`chip.set('tool', 'verilator', 'task', 'lint', 'warningoff', 'COMBDLY')`

A list of tool warnings for which printing should be suppressed. Generally this is done on a per design basis after review has determined that warning can be safely ignored The code for turning off warnings can be found in the specific task reference manual.

vendor

Description	Tool: vendor
Type	str
Default Value	None
CLI Switch	• `-tool_vendor 'tool <str>'`

continues on next page

Table 933 – continued from previous page

Example (CLI)	`-tool_vendor 'yosys yosys'`
Example (API)	`chip.set('tool', 'yosys', 'vendor', 'yosys')`

Name of the tool vendor. Parameter can be used to set vendor specific technology variables in the PDK and libraries. For open source projects, the project name should be used in place of vendor.

version

Description	Tool: version
Type	[str]
Per step/index	optional
Default Value	[]
CLI Switch	• `-tool_version 'tool <str>'`
Example (CLI)	`-tool_version 'openroad >=v2.0'`
Example (API)	`chip.set('tool', 'openroad', 'version', '>=v2.0')`

List of acceptable versions of the tool executable to be used. Each entry in this list must be a version specifier as described by Python PEP-440. During task execution, the tool is called with the 'vswitch' to check the runtime executable version. If the version of the system executable is not allowed by any of the specifiers in 'version', then the job is halted pre-execution. For backwards compatibility, entries that do not conform to the standard will be interpreted as a version with an '==' specifier. This check can be disabled by setting 'novercheck' to True.

vswitch

Description	Tool: executable version switch
Type	[str]
Default Value	[]
CLI Switch	• `-tool_vswitch 'tool <str>'`
Example (CLI)	`-tool_vswitch 'openroad -version'`
Example (API)	`chip.set('tool', 'openroad', 'vswitch', '-version')`

Command line switch to use with executable used to print out the version number. Common switches include -v, -version, –version. Some tools may require extra flags to run in batch mode.

unit

capacitance

Description	Unit: capacitance
Type	str
Default Value	pf
CLI Switch	• `-unit_capacitance '<str>'`
Example (CLI)	`-unit_capacitance 'pf'`
Example (API)	`chip.set('unit', 'capacitance', pf)`

Units used for capacitance when not explicitly specified. Units are case insensitive (ie. pF == pf).

current

Description	Unit: current
Type	str
Default Value	mA
CLI Switch	• `-unit_current '<str>'`
Example (CLI)	`-unit_current 'mA'`
Example (API)	`chip.set('unit', 'current', mA)`

Units used for current when not explicitly specified. Units are case insensitive (ie. pF == pf).

energy

Description	Unit: energy
Type	str
Default Value	pj
CLI Switch	• `-unit_energy '<str>'`
Example (CLI)	`-unit_energy 'pj'`
Example (API)	`chip.set('unit', 'energy', pj)`

Units used for energy when not explicitly specified. Units are case insensitive (ie. pF == pf).

inductance

Description	Unit: inductance
Type	str
Default Value	nh
CLI Switch	• `-unit_inductance '<str>'`
Example (CLI)	`-unit_inductance 'nh'`
Example (API)	`chip.set('unit', 'inductance', nh)`

Units used for inductance when not explicitly specified. Units are case insensitive (ie. pF == pf).

length

Description	Unit: length
Type	str
Default Value	um
CLI Switch	• `-unit_length '<str>'`
Example (CLI)	`-unit_length 'um'`
Example (API)	`chip.set('unit', 'length', um)`

Units used for length when not explicitly specified. Units are case insensitive (ie. pF == pf).

mass

Description	Unit: mass
Type	str
Default Value	g
CLI Switch	• `-unit_mass '<str>'`
Example (CLI)	`-unit_mass 'g'`
Example (API)	`chip.set('unit', 'mass', g)`

Units used for mass when not explicitly specified. Units are case insensitive (ie. pF == pf).

power

Description	Unit: power
Type	str
Default Value	mw
CLI Switch	• `-unit_power '<str>'`
Example (CLI)	`-unit_power 'mw'`
Example (API)	`chip.set('unit', 'power', mw)`

Units used for power when not explicitly specified. Units are case insensitive (ie. pF == pf).

resistance

Description	Unit: resistance
Type	str
Default Value	ohm
CLI Switch	• `-unit_resistance '<str>'`
Example (CLI)	`-unit_resistance 'ohm'`
Example (API)	`chip.set('unit', 'resistance', ohm)`

Units used for resistance when not explicitly specified. Units are case insensitive (ie. pF == pf).

temperature

Description	Unit: temperature
Type	str
Default Value	C
CLI Switch	• `-unit_temperature '<str>'`
Example (CLI)	`-unit_temperature 'C'`
Example (API)	`chip.set('unit', 'temperature', C)`

Units used for temperature when not explicitly specified. Units are case insensitive (ie. pF == pf).

time

Description	Unit: time
Type	str
Default Value	ns
CLI Switch	• -unit_time '<str>'
Example (CLI)	-unit_time 'ns'
Example (API)	chip.set('unit', 'time', ns)

Units used for time when not explicitly specified. Units are case insensitive (ie. pF == pf).

voltage

Description	Unit: voltage
Type	str
Default Value	mv
CLI Switch	• -unit_voltage '<str>'
Example (CLI)	-unit_voltage 'mv'
Example (API)	chip.set('unit', 'voltage', mv)

Units used for voltage when not explicitly specified. Units are case insensitive (ie. pF == pf).

3.7.5 Nested Schemas

The SC schema has two special top-level categories that store nested subsets of the schema rather than unique parameters.

history

The "history" prefix stores configuration from past runs, indexed by jobname. Values are stored automatically at the end of `run()`, and only parameters tagged with the 'job' scope are stored. This can be used to go back and inspect the results of old runs. As a shortcut for accessing these stored values, most of the schema access functions support an optional `job` keyword arg. For example, the following line returns the number of errors from a synthesis step run as part of a job called "job0":

```
chip.get('metric', 'error', job='job0', step='syn', index='0')
```

library

The "library" prefix stores the schema parameters of library chip objects that have been imported into the current chip object, keyed by library name. These values are accessed directly using the schema access functions. For example, the following line returns the path to a LEF file associated with a library called "mylib":

```
chip.find_files('library', 'mylib', 'output', stackup, 'lef')
```

3.8 Core API

This chapter describes all public methods in the SiliconCompiler core Python API. Refer to the User Guide for architecture concepts and the *Glossary* for terminology and keyword definitions.

Schema access:

set	Sets a schema parameter field.
add	Adds item(s) to a schema parameter list.
get	Returns a schema parameter field.
getkeys	Returns a list of schema dictionary keys.
getdict	Returns a schema dictionary.
valid	Checks validity of a keypath.
help	Returns a schema parameter description.
use	Loads a SiliconCompiler module into the current chip object.

Flowgraph execution:

run	Executes tasks in a flowgraph.
node	Creates a flowgraph node.
edge	Creates a directed edge from a tail node to a head node.

Utility functions:

archive	Archive a job directory.
audit_manifest	Verifies the integrity of the post-run compilation manifest.
calc_area	Calculates the area of a rectilinear diearea.
calc_yield	Calculates raw die yield.
calc_dpw	Calculates dies per wafer.
check_checklist	Check items in a checklist.
check_manifest	Verifies the integrity of the pre-run compilation manifest.
check_logfile	Checks logfile for patterns found in the 'regex' parameter.
clock	Clock configuration helper function.
create_cmdline	Creates an SC command line interface.
find_files	Returns absolute paths to files or directories based on the keypath provided.
find_result	Returns the absolute path of a compilation result.
grep	Emulates the Unix grep command on a string.
hash_files	Generates hash values for a list of parameter files.
nodes_to_execute	Returns an ordered list of flowgraph nodes which will be executed.
load_target	Loads a target module and runs the setup() function.
read_manifest	Reads a manifest from disk and merges it with the current compilation manifest.
show	Opens a graphical viewer for the filename provided.
summary	Prints a summary of the compilation manifest.
use	Loads a SiliconCompiler module into the current chip object.
write_manifest	Writes the compilation manifest to a file.
write_flowgraph	Renders and saves the compilation flowgraph to a file.

Tool driver utility functions:

get_libraries	Returns a list of libraries included in this step/index
add_require_input	Adds input files to the require list of the task.
get_input_files	Returns a list of files from the key input and includes files from libraries if requested.
add_frontend_requires	Adds keys to the require list for the frontend task and checks if options are set, which the current frontend does not support.
get_frontend_options	Returns a dictionary of options set for the frontend and checks if options are set, which the current frontend does not support.

class siliconcompiler.**Checklist**(*chip*, *name*)

Object for configuring a checklist. This is the main object used for configuration and data for a checklist within the SiliconCompiler platform.

This inherits all methods from *Chip*.

Parameters

- **chip** (Chip) – A real only copy of the parent chip.
- **name** (*string*) – Name of the checklist.

Examples

```
>>> siliconcompiler.Checklist(chip, "tapeout")
Creates a checklist object with name "tapeout".
```

class siliconcompiler.**Chip**(*design*, *loglevel=None*)

Object for configuring and executing hardware design flows.

This is the main object used for configuration, data, and execution within the SiliconCompiler platform.

Parameters

design (*string*) – Name of the top level chip design module.

Examples

```
>>> siliconcompiler.Chip(design="top")
Creates a chip object with name "top".
```

add(**args*, *field='value'*, *step=None*, *index=None*, *package=None*)

Adds item(s) to a schema parameter list.

Adds item(s) to schema parameter list based on the keypath and value provided in the `*args`. See the *Schema Reference Manual* for documentation of all supported keypaths. New schema dictionaries are automatically created for keypaths that overlap with 'default' dictionaries.

The value provided must agree with the dictionary parameter 'type'. Accessing a non-existent keypath, providing a value that disagrees with the parameter type, or using add with a scalar parameter produces a logger error message and raises the Chip object error flag.

Parameters

- **args** (*list*) – Parameter keypath followed by a value to add.
- **field** (*str*) – Parameter field to modify.
- **step** (*str*) – Step name to modify for parameters that may be specified on a per-node basis.
- **index** (*str*) – Index name to modify for parameters that may be specified on a per-node basis.
- **package** (*str*) – Package that this file/dir depends on. Available packages are listed in the package source section of the schema.

Examples

```
>>> chip.add('input', 'rtl', 'verilog', 'hello.v')
Adds the file 'hello.v' to the list of sources.
```

allkeys(**keypath_prefix*)

Returns all keypaths in the schema as a list of lists.

Arg:

keypath_prefix (list str): Keypath prefix to search under. The
returned keypaths do not include the prefix.

archive(*jobs=None*, *step=None*, *index=None*, *include=None*, *archive_name=None*)

Archive a job directory.

Creates a single compressed archive (.tgz) based on the design, jobname, and flowgraph in the current chip manifest. Individual steps and/or indices can be archived based on arguments specified. By default, all steps and indices in the flowgraph are archived. By default, only outputs, reports, log files, and the final manifest are archived.

Parameters

- **jobs** (*list of str*) – List of jobs to archive. By default, archives only the current job.
- **step** (*str*) – Step to archive.
- **index** (*str*) – Index to archive
- **include** (*list of str*) – Override of default inclusion rules. Accepts list of glob patterns that are matched from the root of individual step/index directories. To capture all files, supply "*".
- **archive_name** (*str*) – Path to the archive

audit_manifest()

Verifies the integrity of the post-run compilation manifest.

Checks the integrity of the chip object implementation flow after the run() function has been completed. Errors, warnings, and debug messages are reported through the logger object.

Audit checks performed include:

- Time stamps
- File modifications
- Error and warning policy
- IP and design origin
- User access
- License terms
- Version checks

Returns

Returns True if the manifest has integrity, else returns False.

Example

```
>>> chip.audit_manifest()
Audits the Chip object manifest and returns 0 if successful.
```

calc_area(*step=None*, *index=None*)

Calculates the area of a rectilinear diearea.

Uses the shoelace formulate to calculate the design area using the (x,y) point tuples from the 'diearea' parameter. If only diearea parameter only contains two points, then the first and second point must be the lower left and upper right points of the rectangle. (Ref: https://en.wikipedia.org/wiki/Shoelace_formula)

Parameters

- **step** (*str*) – name of the step to calculate the area from

- **index** (*str*) – name of the step to calculate the area from

Returns
Design area (float).

Examples

```
>>> area = chip.calc_area()
```

calc_dpw(*step=None*, *index=None*)

Calculates dies per wafer.

Calculates the gross dies per wafer based on the design area, wafersize, wafer edge margin, and scribe lines. The calculation is done by starting at the center of the wafer and placing as many complete design footprints as possible within a legal placement area.

Parameters

- **step** (*str*) – name of the step use for calculation
- **index** (*str*) – name of the step use for calculation

Returns
Number of gross dies per wafer (int).

Examples

```
>>> dpw = chip.calc_dpw()
Variable dpw gets gross dies per wafer value based on the chip manifest.
```

calc_yield(*step=None*, *index=None*, *model='poisson'*)

Calculates raw die yield.

Calculates the raw yield of the design as a function of design area and d0 defect density. Calculation can be done based on the poisson model (default) or the murphy model. The die area and the d0 parameters are taken from the chip dictionary.

- Poisson model: dy = exp(-area * d0/100).
- Murphy model: dy = ((1-exp(-area * d0/100))/(area * d0/100))^2.

Parameters

- **step** (*str*) – name of the step use for calculation
- **index** (*str*) – name of the step use for calculation
- **model** (*string*) – Model to use for calculation (poisson or murphy)

Returns
Design yield percentage (float).

Examples

```
>>> yield = chip.calc_yield()
Yield variable gets yield value based on the chip manifest.
```

check_checklist(*standard*, *items=None*, *check_ok=False*)

Check items in a checklist.

Checks the status of items in a checklist for the standard provided. If a specific list of items is unspecified, all items are checked.

All items have an associated 'task' parameter, which indicates which tasks can be used to automatically validate the item. For an item to be checked, all tasks must satisfy the item's criteria, unless waivers are provided. In addition, that task must have generated EDA report files for each metric in the criteria.

For items without an associated task, the only requirement is that at least one report has been added to that item.

When 'check_ok' is True, every item must also have its 'ok' parameter set to True, indicating that a human has reviewed the item.

Parameters

- **standard** (*str*) – Standard to check.
- **items** (*list of str*) – Items to check from standard.
- **check_ok** (*bool*) – Whether to check item 'ok' parameter.

Returns

Status of item check.

Examples

```
>>> status = chip.check_checklist('iso9000', 'd000')
Returns status.
```

check_filepaths()

Verifies that paths to all files in manifest are valid.

Returns

True if all file paths are valid, otherwise False.

check_logfile(*jobname=None*, *step=None*, *index='0'*, *logfile=None*, *display=True*)

Checks logfile for patterns found in the 'regex' parameter.

Reads the content of the task's log file and compares the content found with the task's 'regex' parameter. The matches are stored in the file '<step>.<suffix>' in the current directory. The matches are logged if display is set to True.

Parameters

- **jobname** (*str*) – Job directory name. If None, *['option', 'jobname']* is used.
- **step** (*str*) – Task step name ('syn', 'place', etc). If None, *['arg', 'step']* is used.
- **index** (*str*) – Task index. Default value is 0. If None, *['arg', 'index']* is used.
- **logfile** (*str*) – Path to logfile. If None, the default task logfile is used.
- **display** (*bool*) – If True, logs matches.

Returns
Dictionary mapping suffixes to number of matches for that suffix's regex.

Examples

```
>>> chip.check_logfile(step='place')
Searches for regex matches in the place logfile.
```

check_manifest()

Verifies the integrity of the pre-run compilation manifest.

Checks the validity of the current schema manifest in memory to ensure that the design has been properly set up prior to running compilation. The function is called inside the run() function but can also be called separately. Checks performed by the check_manifest() function include:

- Has a flowgraph been defined?
- Does the manifest satisfy the schema requirement field settings?
- Are all flowgraph input names legal step/index pairs?
- Are the tool parameter setting requirements met?

Returns
Returns True if the manifest is valid, else returns False.

Examples

```
>>> manifest_ok = chip.check_manifest()
Returns True of the Chip object dictionary checks out.
```

clock(*pin*, *period*, *jitter=0*, *mode='global'*)

Clock configuration helper function.

A utility function for setting all parameters associated with a single clock definition in the schema.

The method modifies the following schema parameters:

['datasheet', 'pin', pin, 'type', mode] ['datasheet', 'pin', pin, 'tperiod', mode] ['datasheet', 'pin', pin, 'tjitter', mode]

Parameters

- **pin** (*str*) – Full hierarchical path to clk pin.
- **period** (*float*) – Clock period specified in ns.
- **jitter** (*float*) – Clock jitter specified in ns.
- **mode** (*str*) – Mode of operation (from datasheet).

Examples

```
>>> chip.clock('clk', period=1.0)
```

Create a clock named 'clk' with a 1.0ns period.

create_cmdline(*progname=None*, *description=None*, *switchlist=None*, *input_map=None*, *additional_args=None*)

Creates an SC command line interface.

Exposes parameters in the SC schema as command line switches, simplifying creation of SC apps with a restricted set of schema parameters exposed at the command line. The order of command line switch settings parsed from the command line is as follows:

1. loglevel
2. read_manifest([cfg])
3. read compiler inputs
4. all other switches
5. load_target('target')

The cmdline interface is implemented using the Python argparse package and the following use restrictions apply.

- Help is accessed with the '-h' switch.
- Arguments that include spaces must be enclosed with double quotes.
- List parameters are entered individually. (ie. -y libdir1 -y libdir2)
- For parameters with Boolean types, the switch implies "true".
- Special characters (such as '-') must be enclosed in double quotes.
- Compiler compatible switches include: -D, -I, -O{0,1,2,3}
- Verilog legacy switch formats are supported: +libext+, +incdir+

Parameters

- **progname** (*str*) – Name of program to be executed.
- **description** (*str*) – Short program description.
- **switchlist** (*list of str*) – List of SC parameter switches to expose at the command line. By default all SC schema switches are available. Parameter switches should be entered based on the parameter 'switch' field in the schema. For parameters with multiple switches, both will be accepted if any one is included in this list.
- **input_map** (*dict of str*) – Dictionary mapping file extensions to input filetypes. This is used to automatically assign positional source arguments to ['input', 'fileset', ...] keypaths based on their file extension. If None, the CLI will not accept positional source arguments.
- **additional_args** (*dict of dict*) – Dictionary of extra arguments to add to the command line parser, with the arguments matching the argparse.add_argument() call.

Returns

None if additional_args is not provided, otherwise a dictionary with the
command line options detected from the additional_args

Examples

```
>>> chip.create_cmdline(progname='sc-show',switchlist=['-input','-cfg'])
Creates a command line interface for 'sc-show' app.
```

```
>>> chip.create_cmdline(progname='sc', input_map={'v': ('rtl', 'verilog')})
All sources ending in .v will be stored in ['input', 'rtl', 'verilog']
```

```
>>> extra = chip.create_cmdline(progname='sc',
                                additional_args={'-demo': {'action': 'store_true
↪'}})
Returns extra = {'demo': False/True}
```

property design

Design name of chip object.

This is an immutable property.

edge(*flow*, *tail*, *head*, *tail_index=0*, *head_index=0*)

Creates a directed edge from a tail node to a head node.

Connects the output of a tail node with the input of a head node by setting the 'input' field of the head node in the schema flowgraph.

The method modifies the following parameters:

['flowgraph', flow, head, str(head_index), 'input']

Parameters

- **flow** (*str*) – Name of flow
- **tail** (*str*) – Name of tail node
- **head** (*str*) – Name of head node
- **tail_index** (*int*) – Index of tail node to connect
- **head_index** (*int*) – Index of head node to connect

Examples

```
>>> chip.edge('place', 'cts')
Creates a directed edge from place to cts.
```

error(*msg*, *fatal=False*)

Raises error.

If fatal is False and *['option', 'continue']* is set to True, this will log an error and set an internal error flag that will cause run() to quit. Otherwise, this will raise a SiliconCompilerError.

Parameters

- **msg** (*str*) – Message associated with error

- **fatal** (*bool*) – Whether error is always fatal

find_files(**keypath*, *missing_ok=False*, *job=None*, *step=None*, *index=None*)

Returns absolute paths to files or directories based on the keypath provided.

By default, this function first checks if the keypath provided has its *copy* parameter set to True. If so, it returns paths to the files in the build directory. Otherwise, it resolves these files based on the current working directory and SC path.

The keypath provided must point to a schema parameter of type file, dir, or lists of either. Otherwise, it will trigger an error.

Parameters

- **keypath** (*list str*) – Variable length schema key list.
- **missing_ok** (*bool*) – If True, silently return None when files aren't found. If False, print an error and set the error flag.
- **job** (*str*) – Jobname to use for dictionary access in place of the current active jobname.
- **step** (*str*) – Step name to access for parameters that may be specified on a per-node basis.
- **index** (*str*) – Index name to access for parameters that may be specified on a per-node basis.

Returns

If keys points to a scalar entry, returns an absolute path to that file/directory, or None if not found. It keys points to a list entry, returns a list of either the absolute paths or None for each entry, depending on whether it is found.

Examples

```
>>> chip.find_files('input', 'verilog')
Returns a list of absolute paths to source files, as specified in
the schema.
```

find_result(*filetype*, *step*, *jobname=None*, *index='0'*)

Returns the absolute path of a compilation result.

Utility function that returns the absolute path to a results file based on the provided arguments. The result directory structure is:

<dir>/<design>/<jobname>/<step>/<index>/outputs/<design>.filetype

Parameters

- **filetype** (*str*) – File extension (v, def, etc)
- **step** (*str*) – Task step name ('syn', 'place', etc)
- **jobname** (*str*) – Jobid directory name
- **index** (*str*) – Task index

Returns

Returns absolute path to file.

Examples

```
>>> manifest_filepath = chip.find_result('vg', 'syn')
```

Returns the absolute path to the manifest.

get(**keypath*, *field='value'*, *job=None*, *step=None*, *index=None*)

Returns a schema parameter field.

Returns a schema parameter field based on the keypath provided in the ***keypath**. See the *Schema Reference Manual* for documentation of all supported keypaths. The returned type is consistent with the type field of the parameter. Fetching parameters with empty or undefined value files returns None for scalar types and [] (empty list) for list types. Accessing a non-existent keypath produces a logger error message and raises the Chip object error flag.

Parameters

- **keypath** (*list str*) – Variable length schema key list.
- **field** (*str*) – Parameter field to fetch.
- **job** (*str*) – Jobname to use for dictionary access in place of the current active jobname.
- **step** (*str*) – Step name to access for parameters that may be specified on a per-node basis.
- **index** (*str*) – Index name to access for parameters that may be specified on a per-node basis.

Returns

Value found for the keypath and field provided.

Examples

```
>>> foundry = chip.get('pdk', 'foundry')
Returns the name of the foundry from the PDK.
```

getdict(**keypath*)

Returns a schema dictionary.

Searches the schema for the keypath provided and returns a complete dictionary. Accessing a non-existent keypath produces a logger error message and raises the Chip object error flag.

Parameters

keypath (*list str*) – Variable length ordered schema key list

Returns

A schema dictionary

Examples

```
>>> pdk = chip.getdict('pdk')
Returns the complete dictionary found for the keypath 'pdk'
```

getkeys(**keypath*, *job=None*)

Returns a list of schema dictionary keys.

Searches the schema for the keypath provided and returns a list of keys found, excluding the generic 'default' key. Accessing a non-existent keypath produces a logger error message and raises the Chip object error flag.

Parameters

- **keypath** (*list str*) – Variable length ordered schema key list
- **job** (*str*) – Jobname to use for dictionary access in place of the current active jobname.

Returns

List of keys found for the keypath provided.

Examples

```
>>> keylist = chip.getkeys('pdk')
Returns all keys for the 'pdk' keypath.
```

graph(*flow*, *subflow*, *name=None*)

Instantiates a named flow as a graph in the current flowgraph.

Parameters

- **flow** (*str*) – Name of current flow.
- **subflow** (*str*) – Name of flow to instantiate
- **name** (*str*) – Name of instance

Examples

```
>>> chip.graph('asicflow')
Instantiates a flow named 'asicflow'.
```

grep(*args*, *line*)

Emulates the Unix grep command on a string.

Emulates the behavior of the Unix grep command that is etched into our muscle memory. Partially implemented, not all features supported. The function returns None if no match is found.

Parameters

- **arg** (*string*) – Command line arguments for grep command
- **line** (*string*) – Line to process

Returns

Result of grep command (string).

hash_files(**keypath*, *update=True*, *step=None*, *index=None*)

Generates hash values for a list of parameter files.

Generates a hash value for each file found in the keypath. If existing hash values are stored, this method will compare hashes and trigger an error if there's a mismatch. If the update variable is True, the computed hash values are recorded in the 'filehash' field of the parameter, following the order dictated by the files within the 'value' parameter field.

Files are located using the find_files() function.

The file hash calculation is performed based on the 'algo' setting. Supported algorithms include SHA1, SHA224, SHA256, SHA384, SHA512, and MD5.

Parameters

- ***keypath** (*str*) – Keypath to parameter.
- **update** (*bool*) – If True, the hash values are recorded in the chip object manifest.

Returns

A list of hash values.

Examples

```
>>> hashlist = hash_files('input', 'rtl', 'verilog')
Computes, stores, and returns hashes of files in :keypath:`input, rtl, verilog`.
```

help(**keypath*)

Returns a schema parameter description.

Parameters

***keypath** (*str*) – Keypath to parameter.

Returns

A formatted multi-line help paragraph for the parameter provided.

Examples

```
>>> print(chip.help('asic','diearea'))
Displays help information about the 'asic, diearea' parameter
```

input(*filename*, *fileset=None*, *filetype=None*, *iomap=None*, *package=None*)

Adds file to a filset. The default behavior is to infer filetypes and filesets based on the suffix of the file extensions. The method is a wrapper function for set.add('input', filset, filetype,...)

Default filetype and filset based on suffix:

```
filetype  | fileset    | suffix (case insensitive)
----------|------------|-------------------------------------------
c         | hll        | c,cc,cpp,c++,cp,cxx,hpp
bsv       | hll        | bsv
scala     | hll        | scala
python    | hll        | py
chisel    | config     | sbt
verilog   | rtl        | v,sv,verilog
vhdl      | rtl        | vhd,vhdl
```

(continues on next page)

(continued from previous page)

```
liberty   | timing     | lib,ccs
def       | layout     | def
lef       | layout     | lef
gds       | layout     | gds,gds2,gdsii
oas       | layout     | oas,oasis
gerber    | layout     | gbr,gerber
odb       | layout     | odb
cdl       | netlist    | cdl
sp        | netlist    | sp,spice
verilog   | netlist    | vg
vcd       | waveform   | vcd
sdc       | constraint | sdc
upf       | constraint | upf
pcf       | constraint | pcf
xdc       | fpga       | xdc
vpr_place | fpga       | place
vpr_route | fpga       | route
```

Parameters

- **fileset** (*str*) – File grouping
- **filetype** (*str*) – File type
- **iomap** (*dict of tuple(set, type)*) – File set and type mapping based on file extension

load_target(*module*, ***kwargs*)

Loads a target module and runs the setup() function.

The function searches the installed Python packages for <name> and siliconcompiler.targets.<name> and runs the setup function in that module if found.

Parameters

- **module** (*str or module*) – Module name
- ****kwargs** (*str*) – Options to pass along to the target

Examples

```
>>> chip.load_target('freepdk45_demo', syn_np=5)
Loads the 'freepdk45_demo' target with 5 parallel synthesis tasks
```

node(*flow*, *step*, *task*, *index=0*)

Creates a flowgraph node.

Creates a flowgraph node by binding a step to a tool specific task. A tool can be an external executable or one of the built in functions in the SiliconCompiler framework). Built in functions include: minimum, maximum, join, mux, verify. The task is set to 'step' if unspecified.

The method modifies the following schema parameters:

- ['flowgraph', flow, step, index, 'tool', tool]
- ['flowgraph', flow, step, index, 'task', task]

- ['flowgraph', flow, step, index, 'task', taskmodule]
- ['flowgraph', flow, step, index, 'weight', metric]

Parameters

- **flow** (*str*) – Flow name
- **step** (*str*) – Step name
- **task** (*module/str*) – Task to associate with this node
- **index** (*int*) – Step index

Examples

```
>>> import siliconcomiler.tools.openroad.place as place
>>> chip.node('asicflow', 'apr_place', place, index=0)
Creates a 'place' task with step='apr_place' and index=0 and binds it to the
'openroad' tool.
```

nodes_to_execute(*flow=None*)

Returns an ordered list of flowgraph nodes which will be executed. This takes the from/to options into account if flow is the current flow or None.

Returns

A list of nodes that will get executed during run() (or a specific flow).

Example

```
>>> nodes = chip.nodes_to_execute()
```

output(*filename*, *fileset=None*, *filetype=None*, *iomap=None*)

Adds file to a filset. The default behavior is to infer filetypes and filesets based on the suffix of the file extensions. The method is a wrapper function for set.add('output', filset, filetype,...)

Default filetype and filset based on suffix:

```
filetype  | fileset    | suffix (case insensitive)
----------|------------|-------------------------------------------
c         | hll        | c,cc,cpp,c++,cp,cxx,hpp
bsv       | hll        | bsv
scala     | hll        | scala
python    | hll        | py
chisel    | config     | sbt
verilog   | rtl        | v,sv,verilog
vhdl      | rtl        | vhd,vhdl
liberty   | timing     | lib,ccs
def       | layout     | def
lef       | layout     | lef
gds       | layout     | gds,gds2,gdsii
oas       | layout     | oas,oasis
gerber    | layout     | gbr,gerber
odb       | layout     | odb
```

(continues on next page)

(continued from previous page)

```
cdl       | netlist    | cdl
sp        | netlist    | sp,spice
verilog   | netlist    | vg
vcd       | waveform   | vcd
sdc       | constraint | sdc
upf       | constraint | upf
pcf       | constraint | pcf
xdc       | fpga       | xdc
vpr_place | fpga       | place
vpr_route | fpga       | route
```

Parameters

- **fileset** (*str*) – File grouping
- **filetype** (*str*) – File type
- **iomap** (*dict of tuple(set, type)*) – File set and type mapping based on file extension

pipe(*flow*, *plan*)

Creates a pipeline based on an order list of key values pairs.

read_manifest(*filename*, *job=None*, *clear=True*, *clobber=True*)

Reads a manifest from disk and merges it with the current compilation manifest.

The file format read is determined by the filename suffix. Currently json (*.json) and yaml(*.yaml) formats are supported.

Parameters

- **filename** (*filepath*) – Path to a manifest file to be loaded.
- **job** (*str*) – Specifies non-default job to merge into.
- **clear** (*bool*) – If True, disables append operations for list type.
- **clobber** (*bool*) – If True, overwrites existing parameter value.

Examples

```
>>> chip.read_manifest('mychip.json')
Loads the file mychip.json into the current Chip object.
```

register_package_source(*name*, *path*, *ref=None*, *clobber=True*)

Registers a package by its name with the source path and reference

Registered package sources are stored in the package section of the schema.

Parameters

- **name** (*str*) – Package name
- **path** (*str*) – Path to the sources, can be file, git url, archive url
- **ref** (*str*) – Reference of the sources, can be commitid, branch name, tag

Examples

```
>>> chip.register_package_source('siliconcompiler_data',
        'git+https://github.com/siliconcompiler/siliconcompiler',
        'dependency-caching-rebase')
```

run()

Executes tasks in a flowgraph.

The run function sets up tools and launches runs for every node in the flowgraph starting with 'from' steps and ending at 'to' steps. From/to are taken from the schema from/to parameters if defined, otherwise from/to are defined as the entry/exit steps of the flowgraph. Before starting the process, tool modules are loaded and setup up for each step and index based on on the schema eda dictionary settings. Once the tools have been set up, the manifest is checked using the check_manifest() function and files in the manifest are hashed based on the 'hashmode' schema setting.

Once launched, each process waits for preceding steps to complete, as defined by the flowgraph 'inputs' parameter. Once a all inputs are ready, previous steps are checked for errors before the process entered a local working directory and starts to run a tool or to execute a built in Chip function.

Fatal errors within a step/index process cause all subsequent processes to exit before start, returning control to the the main program which can then exit.

Examples

```
>>> run()
Runs the execution flow defined by the flowgraph dictionary.
```

set(**args*, *field='value'*, *clobber=True*, *step=None*, *index=None*, *package=None*)

Sets a schema parameter field.

Sets a schema parameter field based on the keypath and value provided in the `*args`. See the *Schema Reference Manual* for documentation of all supported keypaths. New schema dictionaries are automatically created for keypaths that overlap with 'default' dictionaries. The write action is ignored if the parameter value is non-empty and the clobber option is set to False.

The value provided must agree with the dictionary parameter 'type'. Accessing a non-existent keypath or providing a value that disagrees with the parameter type produces a logger error message and raises the Chip object error flag.

Parameters

- **args** (*list*) – Parameter keypath followed by a value to set.
- **field** (*str*) – Parameter field to set.
- **clobber** (*bool*) – Existing value is overwritten if True.
- **step** (*str*) – Step name to set for parameters that may be specified on a per-node basis.
- **index** (*str*) – Index name to set for parameters that may be specified on a per-node basis.
- **package** (*str*) – Package that this file/dir depends on. Available packages are listed in the package source section of the schema.

Examples

```
>>> chip.set('design', 'top')
Sets the name of the design to 'top'
```

show(*filename=None*, *screenshot=False*, *extension=None*)

Opens a graphical viewer for the filename provided.

The show function opens the filename specified using a viewer tool selected based on the file suffix and the 'showtool' schema setup. The 'showtool' parameter binds tools with file suffixes, enabling the automated dynamic loading of tool setup functions. Display settings and technology settings for viewing the file are read from the in-memory chip object schema settings. All temporary render and display files are saved in the <build_dir>/_show_<jobname> directory.

Parameters

- **filename** (*path*) – Name of file to display
- **screenshot** (*bool*) – Flag to indicate if this is a screenshot or show
- **extension** (*str*) – extension of file to show

Examples

```
>>> show('build/oh_add/job0/export/0/outputs/oh_add.gds')
Displays gds file with a viewer assigned by 'showtool'
```

summary(*show_all_indices=False*, *generate_image=True*, *generate_html=True*)

Prints a summary of the compilation manifest.

Metrics from the flowgraph nodes, or from/to parameter if defined, are printed out on a per step basis. All metrics from the metric dictionary with weights set in the flowgraph dictionary are printed out.

Parameters

- **show_all_indices** (*bool*) – If True, displays metrics for all indices of each step. If False, displays metrics only for winning indices.
- **generate_image** (*bool*) – If True, generates a summary image featuring a layout screenshot and a subset of metrics. Requires that the current job has an ending node that generated a PNG file.
- **generate_html** (*bool*) – If True, generates an HTML report featuring a metrics summary table and manifest tree view. The report will include a layout screenshot if the current job has an ending node that generated a PNG file.

Examples

```
>>> chip.summary()
Prints out a summary of the run to stdout.
```

top()

Gets the name of the design's entrypoint for compilation and simulation.

This method should be used to name input and output files in tool drivers, rather than relying on chip.get('design') directly.

Returns *['option', 'entrypoint']* if it has been set, otherwise *['design']*.

unset(**keypath*, *step=None*, *index=None*)

Unsets a schema parameter.

This method effectively undoes any previous calls to `set()` made to the given keypath and step/index. For parameters with required or no per-node values, unsetting a parameter always causes it to revert to its default value, and future calls to `set()` with `clobber=False` will once again be able to modify the value.

If you unset a particular step/index for a parameter with optional per-node values, note that the newly returned value will be the global value if it has been set. To completely return the parameter to its default state, the global value has to be unset as well.

`unset()` has no effect if called on a parameter that has not been previously set.

Parameters

- **keypath** (*list*) – Parameter keypath to clear.
- **step** (*str*) – Step name to unset for parameters that may be specified on a per-node basis.
- **index** (*str*) – Index name to unset for parameters that may be specified on a per-node basis.

use(*module*, ***kwargs*)

Loads a SiliconCompiler module into the current chip object.

The behavior of this function is described in the table below

Table 947: Use behavior

Input type	Action
Module with setup function	Call *setup()* and import returned objects
Chip	Import as a library
Library	Import as a library
PDK	Import as a pdk
FPGA	Import as a fpga
Flow	Import as a flow
Checklist	Import as a checklist

valid(**keypath*, *default_valid=False*)

Checks validity of a keypath.

Checks the validity of a parameter keypath and returns True if the keypath is valid and False if invalid.

Parameters

- **default_valid** (*bool*) – Whether to consider "default" in valid
- **False.** (*keypaths as a wildcard. Defaults to*)

Returns

Boolean indicating validity of keypath.

Examples

```
>>> check = chip.valid('design')
Returns True.
>>> check = chip.valid('blah')
Returns False.
>>> check = chip.valid('metric', 'foo', '0', 'tasktime', default_valid=True)
Returns True, even if "foo" and "0" aren't in current configuration.
```

write_flowgraph(*filename*, *flow=None*, *fillcolor='#ffffff'*, *fontcolor='#000000'*, *fontsize='14'*, *border=True*, *landscape=False*)

Renders and saves the compilation flowgraph to a file.

The chip object flowgraph is traversed to create a graphviz (*.dot) file comprised of node, edges, and labels. The dot file is a graphical representation of the flowgraph useful for validating the correctness of the execution flow graph. The dot file is then converted to the appropriate picture or drawing format based on the filename suffix provided. Supported output render formats include png, svg, gif, pdf and a few others. For more information about the graphviz project, see see https://graphviz.org/

Parameters

- **filename** (*filepath*) – Output filepath
- **flow** (*str*) – Name of flowgraph to render
- **fillcolor** (*str*) – Node fill RGB color hex value
- **fontcolor** (*str*) – Node font RGB color hex value
- **fontsize** (*str*) – Node text font size
- **border** (*bool*) – Enables node border if True
- **landscape** (*bool*) – Renders graph in landscape layout if True

Examples

```
>>> chip.write_flowgraph('mydump.png')
Renders the object flowgraph and writes the result to a png file.
```

write_manifest(*filename*, *prune=True*, *abspath=False*)

Writes the compilation manifest to a file.

The write file format is determined by the filename suffix. Currently json (*.json), yaml (*.yaml), tcl (*.tcl), and (*.csv) formats are supported.

Parameters

- **filename** (*filepath*) – Output filepath
- **prune** (*bool*) – If True, only essential fields from the the Chip object schema are written to the output file.
- **abspath** (*bool*) – If set to True, then all schema filepaths are resolved to absolute filepaths.

Examples

```
>>> chip.write_manifest('mydump.json')
Prunes and dumps the current chip manifest into mydump.json
```

class siliconcompiler.**FPGA**(*chip*, *name*, *package=None*)

Object for configuring an FPGA This is the main object used for configuration and data for a FPGA within the SiliconCompiler platform.

This inherits all methods from *Chip*.

Parameters

- **chip** (Chip) – A real only copy of the parent chip.
- **name** (*string*) – Name of the FPGA.

Examples

```
>>> siliconcompiler.FPGA(chip, "lattice_ice40")
Creates a flow object with name "lattice_ice40".
```

class siliconcompiler.**Flow**(*chip*, *name*)

Object for configuring a flow. This is the main object used for configuration and data for a flow within the SiliconCompiler platform.

This inherits all methods from *Chip*.

Parameters

- **chip** (Chip) – A real only copy of the parent chip.
- **name** (*string*) – Name of the flow.

Examples

```
>>> siliconcompiler.Flow(chip, "asicflow")
Creates a flow object with name "asicflow".
```

class siliconcompiler.**Library**(*chip*, *name*, *package=None*)

Object for configuring a library. This is the main object used for configuration and data for a library within the SiliconCompiler platform.

This inherits all methods from *Chip*.

Parameters

- **chip** (Chip) – A real only copy of the parent chip.
- **name** (*string*) – Name of the library.

Examples

```
>>> siliconcompiler.Library(chip, "asap7sc7p5t")
Creates a library object with name "asap7sc7p5t".
```

class siliconcompiler.**PDK**(*chip*, *name*, *package=None*)

Object for configuring a process development kit. This is the main object used for configuration and data for a PDK within the SiliconCompiler platform.

This inherits all methods from *Chip*.

Parameters

- **chip** (Chip) – A real only copy of the parent chip.
- **name** (*string*) – Name of the PDK.

Examples

```
>>> siliconcompiler.PDK(chip, "asap7")
Creates a flow object with name "asap7".
```

class siliconcompiler.**Schema**(*cfg=None*, *manifest=None*, *logger=None*)

Object for storing and accessing configuration values corresponding to the SiliconCompiler schema.

Most user-facing interaction with the schema should occur through an instance of *Chip*, but this class is available for schema manipulation tasks that don't require the additional context of a Chip object.

The two arguments to this class are mutually exclusive. If neither are provided, the object is initialized to default values for all parameters.

Parameters

- **cfg** (*dict*) – Initial configuration dictionary. This may be a subtree of the schema.
- **manifest** (*str*) – Initial manifest.
- **logger** (*logging.Logger*) – instance of the parent logger if available

add(**args*, *field='value'*, *step=None*, *index=None*)

Adds item(s) to a schema parameter list.

See `add()` for detailed documentation.

allkeys(**keypath_prefix*)

Returns all keypaths in the schema as a list of lists.

See `allkeys()` for detailed documentation.

copy()

Returns deep copy of Schema object.

create_cmdline(*progname*, *description=None*, *switchlist=None*, *input_map=None*, *additional_args=None*, *version=None*, *print_banner=None*, *input_map_handler=None*, *preprocess_keys=None*, *post_process=None*, *logger=None*)

Creates a Schema command line interface.

Exposes parameters in the SC schema as command line switches, simplifying creation of SC apps with a restricted set of schema parameters exposed at the command line. The order of command line switch settings parsed from the command line is as follows:

1. loglevel, if available in schema
2. read_manifest([cfg]), if available in schema
3. read inputs with input_map_handler
4. all other switches
5. Run post_process

The cmdline interface is implemented using the Python argparse package and the following use restrictions apply.

- Help is accessed with the '-h' switch.
- Arguments that include spaces must be enclosed with double quotes.
- List parameters are entered individually. (ie. -y libdir1 -y libdir2)
- For parameters with Boolean types, the switch implies "true".
- Special characters (such as '-') must be enclosed in double quotes.
- Compiler compatible switches include: -D, -I, -O{0,1,2,3}
- Legacy switch formats are supported: +libext+, +incdir+

Parameters

- **progname** (*str*) – Name of program to be executed.
- **description** (*str*) – Short program description.
- **switchlist** (*list of str*) – List of SC parameter switches to expose at the command line. By default all SC schema switches are available. Parameter switches should be entered based on the parameter 'switch' field in the schema. For parameters with multiple switches, both will be accepted if any one is included in this list.
- **input_map** (*dict of str*) – Dictionary mapping file extensions to input filetypes. This is used to automatically assign positional source arguments to ['input', 'fileset', ...] keypaths based on their file extension. If None, the CLI will not accept positional source arguments.
- **additional_args** (*dict of dict*) – Dictionary of extra arguments to add to the command line parser, with the arguments matching the argparse.add_argument() call.
- **version** (*str*) – Version to report when calling with -version
- **print_banner** (*function*) – Function callback to print command line banner
- **input_map_handler** (*function*) – Function callback handle inputs to the input map
- **preprocess_keys** (*function*) – Function callback to preprocess keys that need to be corrected
- **post_process** (*function*) – Function callback to process arguments before returning

Returns

None if additional_args is not provided, otherwise a dictionary with the command line options detected from the additional_args

Examples

```
>>> schema.create_cmdline(progname='sc-show',switchlist=['-input','-cfg'])
Creates a command line interface for 'sc-show' app.
```

```
>>> schema.create_cmdline(progname='sc', input_map={'v': ('rtl', 'verilog')})
All sources ending in .v will be stored in ['input', 'rtl', 'verilog']
```

```
>>> extra = schema.create_cmdline(progname='sc',
                                  additional_args={'-demo': {'action': 'store_
↪true'}})
Returns extra = {'demo': False/True}
```

get(**keypath*, *field='value'*, *job=None*, *step=None*, *index=None*)

Returns a schema parameter field.

See `get()` for detailed documentation.

get_default(**keypath*)

Returns default value of a parameter.

Parameters

keypath (*list str*) – Variable length schema key list.

getdict(**keypath*)

Returns a schema dictionary.

See `getdict()` for detailed documentation.

getkeys(**keypath*, *job=None*)

Returns a list of schema dictionary keys.

See `getkeys()` for detailed documentation.

history(*job*)

Returns a *mutable* reference to [‘history’, job] as a Schema object.

If job doesn’t currently exist in history, create it with default values.

Parameters

job (*str*) – Name of historical job to return.

prune()

Remove all empty parameters from configuration dictionary.

Also deletes ‘help’ and ‘example’ keys.

read_manifest(*filename*, *clear=True*, *clobber=True*, *allow_missing_keys=False*)

Reads a manifest from disk and merges it with the current manifest.

The file format read is determined by the filename suffix. Currently json (*.json) and yaml(*.yaml) formats are supported.

Parameters

- **filename** (*filepath*) – Path to a manifest file to be loaded.
- **clear** (*bool*) – If True, disables append operations for list type.
- **clobber** (*bool*) – If True, overwrites existing parameter value.

- **allow_missing_keys** (*bool*) – If True, keys not present in current schema will be ignored.

Examples

```
>>> chip.read_manifest('mychip.json')
Loads the file mychip.json into the current Chip object.
```

record_history()

Copies all non-empty parameters from current job into the history dictionary.

set(**args*, *field='value'*, *clobber=True*, *step=None*, *index=None*)

Sets a schema parameter field.

See `set()` for detailed documentation.

unset(**keypath*, *step=None*, *index=None*)

Unsets a schema parameter field.

See `unset()` for detailed documentation.

valid(**args*, *default_valid=False*)

Checks validity of a keypath.

See `valid()` for detailed documentation.

write_tcl(*fout*, *prefix=''*, *step=None*, *index=None*, *template=None*)

Prints out schema as TCL dictionary

exception siliconcompiler.**SiliconCompilerError**(*message*)

Minimal Exception wrapper used to raise sc runtime errors.

class siliconcompiler.use.**Checklist**(*chip*, *name*)

Object for configuring a checklist. This is the main object used for configuration and data for a checklist within the SiliconCompiler platform.

This inherits all methods from *Chip*.

Parameters

- **chip** (Chip) – A real only copy of the parent chip.
- **name** (*string*) – Name of the checklist.

Examples

```
>>> siliconcompiler.Checklist(chip, "tapeout")
Creates a checklist object with name "tapeout".
```

class siliconcompiler.use.**FPGA**(*chip*, *name*, *package=None*)

Object for configuring an FPGA This is the main object used for configuration and data for a FPGA within the SiliconCompiler platform.

This inherits all methods from *Chip*.

Parameters

- **chip** (Chip) – A real only copy of the parent chip.

- **name** (*string*) – Name of the FPGA.

Examples

```
>>> siliconcompiler.FPGA(chip, "lattice_ice40")
Creates a flow object with name "lattice_ice40".
```

class siliconcompiler.use.**Flow**(*chip*, *name*)

Object for configuring a flow. This is the main object used for configuration and data for a flow within the SiliconCompiler platform.

This inherits all methods from *Chip*.

Parameters

- **chip** (Chip) – A real only copy of the parent chip.
- **name** (*string*) – Name of the flow.

Examples

```
>>> siliconcompiler.Flow(chip, "asicflow")
Creates a flow object with name "asicflow".
```

class siliconcompiler.use.**Library**(*chip*, *name*, *package=None*)

Object for configuring a library. This is the main object used for configuration and data for a library within the SiliconCompiler platform.

This inherits all methods from *Chip*.

Parameters

- **chip** (Chip) – A real only copy of the parent chip.
- **name** (*string*) – Name of the library.

Examples

```
>>> siliconcompiler.Library(chip, "asap7sc7p5t")
Creates a library object with name "asap7sc7p5t".
```

class siliconcompiler.use.**PDK**(*chip*, *name*, *package=None*)

Object for configuring a process development kit. This is the main object used for configuration and data for a PDK within the SiliconCompiler platform.

This inherits all methods from *Chip*.

Parameters

- **chip** (Chip) – A real only copy of the parent chip.
- **name** (*string*) – Name of the PDK.

Examples

```
>>> siliconcompiler.PDK(chip, "asap7")
Creates a flow object with name "asap7".
```

class siliconcompiler.use.**PackageChip**(*chip*, *name*, *package=None*)

add(**args*, *field='value'*, *step=None*, *index=None*, *package=None*)

Adds item(s) to a schema parameter list.

Adds item(s) to schema parameter list based on the keypath and value provided in the `*args`. See the *Schema Reference Manual* for documentation of all supported keypaths. New schema dictionaries are automatically created for keypaths that overlap with 'default' dictionaries.

The value provided must agree with the dictionary parameter 'type'. Accessing a non-existent keypath, providing a value that disagrees with the parameter type, or using add with a scalar parameter produces a logger error message and raises the Chip object error flag.

Parameters

- **args** (*list*) – Parameter keypath followed by a value to add.
- **field** (*str*) – Parameter field to modify.
- **step** (*str*) – Step name to modify for parameters that may be specified on a per-node basis.
- **index** (*str*) – Index name to modify for parameters that may be specified on a per-node basis.
- **package** (*str*) – Package that this file/dir depends on. Available packages are listed in the package source section of the schema.

Examples

```
>>> chip.add('input', 'rtl', 'verilog', 'hello.v')
Adds the file 'hello.v' to the list of sources.
```

set(**args*, *field='value'*, *clobber=True*, *step=None*, *index=None*, *package=None*)

Sets a schema parameter field.

Sets a schema parameter field based on the keypath and value provided in the `*args`. See the *Schema Reference Manual* for documentation of all supported keypaths. New schema dictionaries are automatically created for keypaths that overlap with 'default' dictionaries. The write action is ignored if the parameter value is non-empty and the clobber option is set to False.

The value provided must agree with the dictionary parameter 'type'. Accessing a non-existent keypath or providing a value that disagrees with the parameter type produces a logger error message and raises the Chip object error flag.

Parameters

- **args** (*list*) – Parameter keypath followed by a value to set.
- **field** (*str*) – Parameter field to set.
- **clobber** (*bool*) – Existing value is overwritten if True.
- **step** (*str*) – Step name to set for parameters that may be specified on a per-node basis.

- **index** (*str*) – Index name to set for parameters that may be specified on a per-node basis.
- **package** (*str*) – Package that this file/dir depends on. Available packages are listed in the package source section of the schema.

Examples

```
>>> chip.set('design', 'top')
Sets the name of the design to 'top'
```

siliconcompiler.tools._common.**add_frontend_requires**(*chip*, *supports=None*)

Adds keys to the require list for the frontend task and checks if options are set, which the current frontend does not support.

Parameters

- **chip** (Chip) – Chip object
- **supports** (*list*) – List of ['option', '*'] which the frontend supports

siliconcompiler.tools._common.**add_require_input**(*chip*, **key*, *include_library_files=True*)

Adds input files to the require list of the task.

Parameters

- **chip** (Chip) – Chip object
- **key** (*list*) – Key to check for requirements
- **add_library_files** (*bool*) – When True, files from library keys will be included

siliconcompiler.tools._common.**get_frontend_options**(*chip*, *supports=None*)

Returns a dictionary of options set for the frontend and checks if options are set, which the current frontend does not support.

Parameters

- **chip** (Chip) – Chip object
- **supports** (*list*) – List of ['option', '*'] which the frontend supports

siliconcompiler.tools._common.**get_input_files**(*chip*, **key*, *add_library_files=True*)

Returns a list of files from the key input and includes files from libraries if requested.

Parameters

- **chip** (Chip) – Chip object
- **key** (*list*) – Key to collect files from
- **add_library_files** (*bool*) – When True, files from library keys will be included

siliconcompiler.tools._common.**get_libraries**(*chip*, *include_asic=True*)

Returns a list of libraries included in this step/index

Parameters

- **chip** (Chip) – Chip object
- **include_asic** (*bool*) – When in ['option', 'mode'] == 'asic' also include the asic libraries.

3.9 CLI apps

3.9.1 sc

```
usage: sc [-h] [-schemaversion <str>] [-design <str>]
          [-input 'fileset filetype <file>']
          [-output 'fileset filetype <file>']
          [-constraint_timing_voltage 'scenario pin <float>']
          [-constraint_timing_temperature 'scenario <float>']
          [-constraint_timing_libcorner 'scenario <str>']
          [-constraint_timing_pexcorner 'scenario <str>']
          [-constraint_timing_opcond 'scenario <str>']
          [-constraint_timing_mode 'scenario <str>']
          [-constraint_timing_file 'scenario <file>']
          [-constraint_timing_check 'scenario <str>']
          [-constraint_component_placement 'inst <(float,float,float)>']
          [-constraint_component_partname 'inst <str>']
          [-constraint_component_halo 'inst <(float,float)>']
          [-constraint_component_rotation 'inst <float>']
          [-constraint_component_flip ['inst <bool>']]
          [-constraint_pin_placement 'name <(float,float,float)>']
          [-constraint_pin_layer 'name <str>']
          [-constraint_pin_side 'name <int>']
          [-constraint_pin_order 'name <int>']
          [-constraint_net_maxlength 'name <float>']
          [-constraint_net_maxresistance 'name <float>']
          [-constraint_net_ndr 'name <(float,float)>']
          [-constraint_net_minlayer 'name <str>']
          [-constraint_net_maxlayer 'name <str>']
          [-constraint_net_shield 'name <str>']
          [-constraint_net_match 'name <str>']
          [-constraint_net_diffpair 'name <str>']
          [-constraint_net_sympair 'name <str>']
          [-constraint_outline <(float,float)>]
          [-constraint_corearea <(float,float)>]
          [-constraint_coremargin <float>] [-constraint_density <float>]
          [-constraint_aspectratio <float>] [-remote [<bool>]]
          [-credentials <file>] [-cache <file>] [-nice <int>] [-mode <str>]
          [-target <str>] [-pdk <str>] [-uselambda [<bool>]] [-stackup <str>]
          [-flow <str>] [-O <str>] [-frontend <str>] [-cfg <file>]
          [-env 'key <str>'] [-var 'key <str>'] [-file 'key <file>']
          [-dir 'key <dir>'] [-loglevel <str>] [-builddir <dir>]
          [-jobname <str>] [-jobinput 'step index <str>'] [-from <str>]
          [-to <str>] [-prune 'node <(str,str)>'] [-breakpoint [<bool>]]
          [-showtool 'filetype <str>'] [-metricoff '<str>'] [-library <str>]
          [-clean [<bool>]] [-hash [<bool>]] [-nodisplay [<bool>]]
          [-quiet [<bool>]] [-jobincr [<bool>]] [-novercheck [<bool>]]
          [-relax [<bool>]] [-resume [<bool>]] [-track [<bool>]]
          [-trace [<bool>]] [-skipall [<bool>]] [-skipcheck [<bool>]]
          [-copyall [<bool>]] [-show [<bool>]] [-autoinstall [<bool>]]
          [-entrypoint <str>] [+incdir+ <dir>] [-y <dir>] [-v <file>]
          [-D <str>] [+libext+ <str>] [-param 'name <str>'] [-f <file>]
```

(continues on next page)

(continued from previous page)

```
        [-flowcontinue [<bool>]] [-continue [<bool>]] [-timeout <float>]
        [-strict [<bool>]] [-scheduler <str>] [-cores <int>] [-memory <int>]
        [-queue <str>] [-defer <str>] [-scheduler_options <str>]
        [-msgevent <str>] [-msgcontact <str>] [-arg_step <str>]
        [-arg_index <str>] [-unit_time '<str>'] [-unit_length '<str>']
        [-unit_mass '<str>'] [-unit_temperature '<str>']
        [-unit_capacitance '<str>'] [-unit_resistance '<str>']
        [-unit_inductance '<str>'] [-unit_voltage '<str>']
        [-unit_current '<str>'] [-unit_power '<str>'] [-unit_energy '<str>']
        [-fpga_partname <str>] [-fpga_vendor 'partname <str>']
        [-fpga_lutsize 'partname <int>'] [-fpga_file 'partname key <file>']
        [-fpga_var 'partname key <str>']
        [-fpga_resources_registers 'partname <str>']
        [-fpga_resources_dsps 'partname <str>']
        [-fpga_resources_brams 'partname <str>'] [-fpga_board <str>]
        [-fpga_program [<bool>]] [-fpga_flash [<bool>]]
        [-asic_logiclib <str>] [-asic_macrolib <str>]
        [-asic_delaymodel <str>] [-asic_cells_decap '<str>']
        [-asic_cells_delay '<str>'] [-asic_cells_tie '<str>']
        [-asic_cells_hold '<str>'] [-asic_cells_clkbuf '<str>']
        [-asic_cells_clkdelay '<str>'] [-asic_cells_clkinv '<str>']
        [-asic_cells_clkgate '<str>'] [-asic_cells_clkicg '<str>']
        [-asic_cells_clklogic '<str>'] [-asic_cells_dontuse '<str>']
        [-asic_cells_filler '<str>'] [-asic_cells_tap '<str>']
        [-asic_cells_endcap '<str>'] [-asic_cells_antenna '<str>']
        [-asic_libarch '<str>'] [-asic_site 'libarch <str>']
        [-pdk_foundry 'pdkname <str>'] [-pdk_node 'pdkname <float>']
        [-pdk_lambda 'pdkname <float>'] [-pdk_version 'pdkname <str>']
        [-pdk_stackup 'pdkname <str>'] [-pdk_minlayer 'pdk stackup <str>']
        [-pdk_maxlayer 'pdk stackup <str>']
        [-pdk_thickness 'pdkname stackup <float>']
        [-pdk_wafersize 'pdkname <float>']
        [-pdk_panelsize 'pdkname <(float,float)>']
        [-pdk_unitcost 'pdkname <float>'] [-pdk_d0 'pdkname <float>']
        [-pdk_hscribe 'pdkname <float>'] [-pdk_vscribe 'pdkname <float>']
        [-pdk_edgemargin 'pdkname <float>'] [-pdk_density 'pdkname <float>']
        [-pdk_devmodel 'pdkname tool simtype stackup <file>']
        [-pdk_pexmodel 'pdkname tool stackup corner <file>']
        [-pdk_layermap 'pdkname tool src dst stackup <file>']
        [-pdk_display 'pdkname tool stackup <file>']
        [-pdk_aprtech 'pdkname tool stackup libarch filetype <file>']
        [-pdk_lvs_runset 'pdkname tool stackup name <file>']
        [-pdk_lvs_waiver 'pdkname tool stackup name <file>']
        [-pdk_drc_runset 'pdkname tool stackup name <file>']
        [-pdk_drc_waiver 'pdkname tool stackup name <file>']
        [-pdk_erc_runset 'pdkname tool stackup name <file>']
        [-pdk_erc_waiver 'pdkname tool stackup name <file>']
        [-pdk_fill_runset 'pdkname tool stackup name <file>']
        [-pdk_fill_waiver 'pdkname tool stackup name <file>']
        [-pdk_file 'pdkname tool key stackup <file>']
        [-pdk_directory 'pdkname tool key stackup <dir>']
        [-pdk_var 'pdkname tool stackup key <str>']
```

(continues on next page)

(continued from previous page)

```
          [-pdk_doc_homepage 'pdkname <file>']
          [-pdk_doc_datasheet 'pdkname <file>']
          [-pdk_doc_reference 'pdkname <file>']
          [-pdk_doc_userguide 'pdkname <file>']
          [-pdk_doc_install 'pdkname <file>']
          [-pdk_doc_quickstart 'pdkname <file>']
          [-pdk_doc_releasenotes 'pdkname <file>']
          [-pdk_doc_tutorial 'pdkname <file>'] [-tool_exe 'tool <str>']
          [-tool_sbom 'tool version <file>'] [-tool_path 'tool <dir>']
          [-tool_vswitch 'tool <str>'] [-tool_vendor 'tool <str>']
          [-tool_version 'tool <str>'] [-tool_format 'tool <str>']
          [-tool_licenseserver 'name key <str>']
          [-tool_task_warningoff 'tool task <str>']
          [-tool_task_continue ['tool task <bool>']]
          [-tool_task_regex 'tool task suffix <str>']
          [-tool_task_option 'tool task <str>']
          [-tool_task_var 'tool task key <str>']
          [-tool_task_env 'tool task env <str>']
          [-tool_task_file 'tool task key <file>']
          [-tool_task_dir 'tool task key <dir>']
          [-tool_task_input 'tool task <file>']
          [-tool_task_output 'tool task <file>']
          [-tool_task_stdout_destination 'tool task <str>']
          [-tool_task_stdout_suffix 'tool task <str>']
          [-tool_task_stderr_destination 'tool task <str>']
          [-tool_task_stderr_suffix 'tool task <str>']
          [-tool_task_require 'tool task <str>']
          [-tool_task_report 'tool task metric <file>']
          [-tool_task_refdir 'tool task <dir>']
          [-tool_task_script 'tool task <file>']
          [-tool_task_prescript 'tool task <file>']
          [-tool_task_postscript 'tool task <file>']
          [-tool_task_keep 'tool task <str>']
          [-tool_task_threads 'tool task <int>']
          [-flowgraph_input 'flow step index <(str,str)>']
          [-flowgraph_weight 'flow step index metric <float>']
          [-flowgraph_goal 'flow step index metric <float>']
          [-flowgraph_tool 'flow step index <str>']
          [-flowgraph_task 'flow step index <str>']
          [-flowgraph_taskmodule 'flow step index <str>']
          [-flowgraph_args 'flow step index <str>']
          [-flowgraph_timeout 'flow step index <float>']
          [-flowgraph_status 'flow step index <str>']
          [-flowgraph_select 'flow step index <(str,str)>']
          [-checklist_description 'standard item <str>']
          [-checklist_requirement 'standard item <str>']
          [-checklist_dataformat 'standard item <str>']
          [-checklist_rationale 'standard item <str>']
          [-checklist_criteria 'standard item <str>']
          [-checklist_task 'standard item <(str,str,str)>']
          [-checklist_report 'standard item <file>']
          [-checklist_waiver 'standard item metric <file>']
```

(continues on next page)

(continued from previous page)

```
          [-checklist_ok ['standard item <bool>']]
          [-metric_errors 'step index <int>']
          [-metric_warnings 'step index <int>']
          [-metric_drvs 'step index <int>']
          [-metric_unconstrained 'step index <int>']
          [-metric_coverage 'step index <float>']
          [-metric_security 'step index <float>']
          [-metric_luts 'step index <int>'] [-metric_dsps 'step index <int>']
          [-metric_brams 'step index <int>']
          [-metric_cellarea 'step index <float>']
          [-metric_totalarea 'step index <float>']
          [-metric_utilization step index <float>]
          [-metric_logicdepth step index <int>]
          [-metric_peakpower 'step index <float>']
          [-metric_averagepower 'step index <float>']
          [-metric_dozepower 'step index <float>']
          [-metric_idlepower 'step index <float>']
          [-metric_leakagepower 'step index <float>']
          [-metric_sleeppower 'step index <float>']
          [-metric_irdrop 'step index <float>']
          [-metric_holdpaths 'step index <int>']
          [-metric_setuppaths 'step index <int>']
          [-metric_holdslack 'step index <float>']
          [-metric_holdwns 'step index <float>']
          [-metric_holdtns 'step index <float>']
          [-metric_setupslack 'step index <float>']
          [-metric_setupwns 'step index <float>']
          [-metric_setuptns 'step index <float>']
          [-metric_fmax 'step index <float>']
          [-metric_macros 'step index <int>']
          [-metric_cells 'step index <int>']
          [-metric_registers 'step index <int>']
          [-metric_buffers 'step index <int>']
          [-metric_transistors 'step index <int>']
          [-metric_pins 'step index <int>'] [-metric_nets 'step index <int>']
          [-metric_vias 'step index <int>']
          [-metric_wirelength 'step index <float>']
          [-metric_overflow 'step index <int>']
          [-metric_memory 'step index <float>']
          [-metric_exetime 'step index <float>']
          [-metric_tasktime 'step index <float>']
          [-metric_totaltime 'step index <float>']
          [-record_userid 'step index <str>']
          [-record_publickey 'step index <str>']
          [-record_machine 'step index <str>']
          [-record_macaddr 'step index <str>']
          [-record_ipaddr 'step index <str>']
          [-record_platform 'step index <str>']
          [-record_distro 'step index <str>']
          [-record_arch 'step index <str>']
          [-record_starttime 'step index <str>']
          [-record_endtime 'step index <str>']
```

(continues on next page)

(continued from previous page)

```
          [-record_region 'step index <str>']
          [-record_scversion 'step index <str>']
          [-record_toolversion 'step index <str>']
          [-record_toolpath 'step index <str>']
          [-record_toolargs 'step index <str>']
          [-record_osversion 'step index <str>']
          [-record_kernelversion 'step index <str>']
          [-record_remoteid 'step index <str>']
          [-datasheet_partnumber '<str>'] [-datasheet_type '<str>']
          [-datasheet_doc '<file>'] [-datasheet_abstraction '<str>']
          [-datasheet_series '<str>'] [-datasheet_manufacturer '<str>']
          [-datasheet_description '<str>'] [-datasheet_features '<str>']
          [-datasheet_grade '<str>'] [-datasheet_qual '<str>']
          [-datasheet_trl '<int>'] [-datasheet_status '<str>']
          [-datasheet_fmax '<float>'] [-datasheet_ops '<float>']
          [-datasheet_iobw '<float>'] [-datasheet_iocount '<int>']
          [-datasheet_ram '<float>'] [-datasheet_peakpower '<float>']
          [-datasheet_io_arch 'name <str>'] [-datasheet_io_gen 'name <str>']
          [-datasheet_io_fmax 'name <float>']
          [-datasheet_io_width 'name <int>']
          [-datasheet_io_channels 'name <int>']
          [-datasheet_proc_arch 'name <str>']
          [-datasheet_proc_features 'name <str>']
          [-datasheet_proc_datatypes 'name <str>']
          [-datasheet_proc_archsize 'name <int>']
          [-datasheet_proc_cores 'name <int>']
          [-datasheet_proc_fmax 'name <int>']
          [-datasheet_proc_ops 'name <int>']
          [-datasheet_proc_mults 'name <int>']
          [-datasheet_proc_icache 'name <int>']
          [-datasheet_proc_dcache 'name <int>']
          [-datasheet_proc_l2cache 'name <int>']
          [-datasheet_proc_l3cache 'name <int>']
          [-datasheet_proc_sram 'name <int>']
          [-datasheet_proc_nvm 'name <int>']
          [-datasheet_memory_bits 'name <int>']
          [-datasheet_memory_width 'name <int>']
          [-datasheet_memory_depth 'name <int>']
          [-datasheet_memory_banks 'name <int>']
          [-datasheet_memory_fmax 'name <(float,float,float)>']
          [-datasheet_memory_tcycle 'name <(float,float,float)>']
          [-datasheet_memory_twr 'name <(float,float,float)>']
          [-datasheet_memory_trd 'name <(float,float,float)>']
          [-datasheet_memory_trefresh 'name <(float,float,float)>']
          [-datasheet_memory_terase 'name <(float,float,float)>']
          [-datasheet_memory_bwrd 'name <(float,float,float)>']
          [-datasheet_memory_bwwr 'name <(float,float,float)>']
          [-datasheet_memory_erd 'name <(float,float,float)>']
          [-datasheet_memory_ewr 'name <(float,float,float)>']
          [-datasheet_memory_twearout 'name <(float,float,float)>']
          [-datasheet_memory_tcl 'name <(int,int,int)>']
          [-datasheet_memory_trcd 'name <(int,int,int)>']
```

(continues on next page)

(continued from previous page)

```
          [-datasheet_memory_trp 'name <(int,int,int)>']
          [-datasheet_memory_tras 'name <(int,int,int)>']
          [-datasheet_fpga_arch 'name <str>']
          [-datasheet_fpga_luts 'name <int>']
          [-datasheet_fpga_registers 'name <int>']
          [-datasheet_fpga_plls 'name <int>']
          [-datasheet_fpga_mults 'name <int>']
          [-datasheet_fpga_totalram 'name <int>']
          [-datasheet_fpga_distram 'name <int>']
          [-datasheet_fpga_blockram 'name <int>']
          [-datasheet_analog_arch 'name <str>']
          [-datasheet_analog_features 'name <str>']
          [-datasheet_analog_resolution 'name <int>']
          [-datasheet_analog_channels 'name <int>']
          [-datasheet_analog_samplerate 'name <(float,float,float)>']
          [-datasheet_analog_enob 'name <(float,float,float)>']
          [-datasheet_analog_inl 'name <(float,float,float)>']
          [-datasheet_analog_dnl 'name <(float,float,float)>']
          [-datasheet_analog_snr 'name <(float,float,float)>']
          [-datasheet_analog_sinad 'name <(float,float,float)>']
          [-datasheet_analog_sfdr 'name <(float,float,float)>']
          [-datasheet_analog_thd 'name <(float,float,float)>']
          [-datasheet_analog_imd3 'name <(float,float,float)>']
          [-datasheet_analog_hd2 'name <(float,float,float)>']
          [-datasheet_analog_hd3 'name <(float,float,float)>']
          [-datasheet_analog_hd4 'name <(float,float,float)>']
          [-datasheet_analog_nsd 'name <(float,float,float)>']
          [-datasheet_analog_phasenoise 'name <(float,float,float)>']
          [-datasheet_analog_gain 'name <(float,float,float)>']
          [-datasheet_analog_pout 'name <(float,float,float)>']
          [-datasheet_analog_pout2 'name <(float,float,float)>']
          [-datasheet_analog_pout3 'name <(float,float,float)>']
          [-datasheet_analog_vofferror 'name <(float,float,float)>']
          [-datasheet_analog_vgainerror 'name <(float,float,float)>']
          [-datasheet_analog_cmrr 'name <(float,float,float)>']
          [-datasheet_analog_psnr 'name <(float,float,float)>']
          [-datasheet_analog_s21 'name <(float,float,float)>']
          [-datasheet_analog_s11 'name <(float,float,float)>']
          [-datasheet_analog_s22 'name <(float,float,float)>']
          [-datasheet_analog_s12 'name <(float,float,float)>']
          [-datasheet_analog_noisefigure 'name <(float,float,float)>']
          [-datasheet_analog_ib1db 'name <(float,float,float)>']
          [-datasheet_analog_oob1db 'name <(float,float,float)>']
          [-datasheet_analog_iip3 'name <(float,float,float)>']
          [-datasheet_limit_tstorage '<(float,float)>']
          [-datasheet_limit_tsolder '<(float,float)>']
          [-datasheet_limit_tj '<(float,float)>']
          [-datasheet_limit_ta '<(float,float)>']
          [-datasheet_limit_tid '<(float,float)>']
          [-datasheet_limit_sel '<(float,float)>']
          [-datasheet_limit_seb '<(float,float)>']
          [-datasheet_limit_segr '<(float,float)>']
```

(continues on next page)

(continued from previous page)

```
          [-datasheet_limit_set '<(float,float)>']
          [-datasheet_limit_seu '<(float,float)>']
          [-datasheet_limit_vhbm '<(float,float)>']
          [-datasheet_limit_vcdm '<(float,float)>']
          [-datasheet_limit_vmm '<(float,float)>']
          [-datasheet_thermal_rja '<float>']
          [-datasheet_thermal_rjct '<float>']
          [-datasheet_thermal_rjcb '<float>']
          [-datasheet_thermal_rjb '<float>']
          [-datasheet_thermal_tjt '<float>']
          [-datasheet_thermal_tjb '<float>'] [-datasheet_package_name '<str>']
          [-datasheet_package_drawing '<file>']
          [-datasheet_package_pincount '<int>']
          [-datasheet_package_length '<(float,float,float)>']
          [-datasheet_package_width '<(float,float,float)>']
          [-datasheet_package_thickness '<(float,float,float)>']
          [-datasheet_package_pinpitch '<(float,float,float)>']
          [-datasheet_pin_map 'name bump <(float,float)>']
          [-datasheet_pin_type 'name mode <str>']
          [-datasheet_pin_dir 'name mode <str>']
          [-datasheet_pin_complement 'name mode <str>']
          [-datasheet_pin_standard 'name mode <str>']
          [-datasheet_pin_interface 'name mode <str>']
          [-datasheet_pin_resetvalue 'name mode <str>']
          [-datasheet_pin_vmax 'pin mode <(float,float,float)>']
          [-datasheet_pin_vnominal 'pin mode <(float,float,float)>']
          [-datasheet_pin_vol 'pin mode <(float,float,float)>']
          [-datasheet_pin_voh 'pin mode <(float,float,float)>']
          [-datasheet_pin_vil 'pin mode <(float,float,float)>']
          [-datasheet_pin_vih 'pin mode <(float,float,float)>']
          [-datasheet_pin_vcm 'pin mode <(float,float,float)>']
          [-datasheet_pin_vdiff 'pin mode <(float,float,float)>']
          [-datasheet_pin_voffset 'pin mode <(float,float,float)>']
          [-datasheet_pin_vnoise 'pin mode <(float,float,float)>']
          [-datasheet_pin_vslew 'pin mode <(float,float,float)>']
          [-datasheet_pin_vhbm 'pin mode <(float,float,float)>']
          [-datasheet_pin_vcdm 'pin mode <(float,float,float)>']
          [-datasheet_pin_vmm 'pin mode <(float,float,float)>']
          [-datasheet_pin_cap 'pin mode <(float,float,float)>']
          [-datasheet_pin_rdiff 'pin mode <(float,float,float)>']
          [-datasheet_pin_rin 'pin mode <(float,float,float)>']
          [-datasheet_pin_rup 'pin mode <(float,float,float)>']
          [-datasheet_pin_rdown 'pin mode <(float,float,float)>']
          [-datasheet_pin_rweakup 'pin mode <(float,float,float)>']
          [-datasheet_pin_rweakdown 'pin mode <(float,float,float)>']
          [-datasheet_pin_power 'pin mode <(float,float,float)>']
          [-datasheet_pin_isupply 'pin mode <(float,float,float)>']
          [-datasheet_pin_ioh 'pin mode <(float,float,float)>']
          [-datasheet_pin_iol 'pin mode <(float,float,float)>']
          [-datasheet_pin_iinject 'pin mode <(float,float,float)>']
          [-datasheet_pin_ishort 'pin mode <(float,float,float)>']
          [-datasheet_pin_ioffset 'pin mode <(float,float,float)>']
```

(continues on next page)

(continued from previous page)

```
          [-datasheet_pin_ibias 'pin mode <(float,float,float)>']
          [-datasheet_pin_ileakage 'pin mode <(float,float,float)>']
          [-datasheet_pin_tperiod 'pin mode <(float,float,float)>']
          [-datasheet_pin_tpulse 'pin mode <(float,float,float)>']
          [-datasheet_pin_tjitter 'pin mode <(float,float,float)>']
          [-datasheet_pin_thigh 'pin mode <(float,float,float)>']
          [-datasheet_pin_tlow 'pin mode <(float,float,float)>']
          [-datasheet_pin_tduty 'pin mode <(float,float,float)>']
          [-datasheet_pin_tsetup 'pin mode relpin <(float,float,float)>']
          [-datasheet_pin_thold 'pin mode relpin <(float,float,float)>']
          [-datasheet_pin_tskew 'pin mode relpin <(float,float,float)>']
          [-datasheet_pin_tdelayr 'pin mode relpin <(float,float,float)>']
          [-datasheet_pin_tdelayf 'pin mode relpin <(float,float,float)>']
          [-datasheet_pin_trise 'pin mode relpin <(float,float,float)>']
          [-datasheet_pin_tfall 'pin mode relpin <(float,float,float)>']
          [-package_version <str>] [-package_description <str>]
          [-package_keyword <str>] [-package_doc_homepage <str>]
          [-package_doc_datasheet <file>] [-package_doc_reference <file>]
          [-package_doc_userguide <file>] [-package_doc_quickstart <file>]
          [-package_doc_releasenotes <file>] [-package_doc_testplan <file>]
          [-package_doc_signoff <file>] [-package_doc_tutorial <file>]
          [-package_license <str>] [-package_licensefile <file>]
          [-package_organization <str>] [-package_author_name 'userid <str>']
          [-package_author_email 'userid <str>']
          [-package_author_username 'userid <str>']
          [-package_author_location 'userid <str>']
          [-package_author_organization 'userid <str>']
          [-package_author_publickey 'userid <str>']
          [-package_source_path 'source <str>']
          [-package_source_ref 'source <str>'] [-version]
          [source ...]

    ------------------------------------------------------------
    SiliconCompiler is an open source compiler framework that
    aims to enable automated translation from source code to
    silicon.

    The sc program includes the followins steps.

    1. Read command line arguments
    2. If not set, 'design' is set to base of first source file.
    3. If not set, 'target' is set to 'skywater130_demo'.
    4. Run compilation
    5. Display summary

    Sources: https://github.com/siliconcompiler/siliconcompiler
    ------------------------------------------------------------


positional arguments:
  source                Input files with filetype inferred by extension
```

(continues on next page)

(continued from previous page)

```
options:
  -h, --help            show this help message and exit
  -schemaversion <str>  Schema version number
  -design <str>         Design top module name
  -input 'fileset filetype <file>'
                        Input: files
  -output 'fileset filetype <file>'
                        Output: files
  -constraint_timing_voltage 'scenario pin <float>'
                        Constraint: pin voltage level
  -constraint_timing_temperature 'scenario <float>'
                        Constraint: temperature
  -constraint_timing_libcorner 'scenario <str>'
                        Constraint: library corner
  -constraint_timing_pexcorner 'scenario <str>'
                        Constraint: pex corner
  -constraint_timing_opcond 'scenario <str>'
                        Constraint: operating condition
  -constraint_timing_mode 'scenario <str>'
                        Constraint: operating mode
  -constraint_timing_file 'scenario <file>'
                        Constraint: SDC files
  -constraint_timing_check 'scenario <str>'
                        Constraint: timing checks
  -constraint_component_placement 'inst <(float,float,float)>'
                        Constraint: Component placement
  -constraint_component_partname 'inst <str>'
                        Constraint: Component part name
  -constraint_component_halo 'inst <(float,float)>'
                        Constraint: Component halo
  -constraint_component_rotation 'inst <float>'
                        Constraint: Component rotation
  -constraint_component_flip ['inst <bool>']
                        Constraint: Component flip option
  -constraint_pin_placement 'name <(float,float,float)>'
                        Constraint: Pin placement
  -constraint_pin_layer 'name <str>'
                        Constraint: Pin layer
  -constraint_pin_side 'name <int>'
                        Constraint: Pin side
  -constraint_pin_order 'name <int>'
                        Constraint: Pin order
  -constraint_net_maxlength 'name <float>'
                        Constraint: Net max length
  -constraint_net_maxresistance 'name <float>'
                        Constraint: Net max resistance
  -constraint_net_ndr 'name <(float,float)>'
                        Constraint: Net routing rule
  -constraint_net_minlayer 'name <str>'
                        Constraint: Net minimum routing layer
  -constraint_net_maxlayer 'name <str>'
                        Constraint: Net maximum routing layer
```

(continues on next page)

(continued from previous page)

```
-constraint_net_shield 'name <str>'
                      Constraint: Net shielding
-constraint_net_match 'name <str>'
                      Constraint: Net matched routing
-constraint_net_diffpair 'name <str>'
                      Constraint: Net diffpair
-constraint_net_sympair 'name <str>'
                      Constraint: Net sympair
-constraint_outline <(float,float)>
                      Constraint: Layout outline
-constraint_corearea <(float,float)>
                      Constraint: Layout core area
-constraint_coremargin <float>
                      Constraint: Layout core margin
-constraint_density <float>
                      Constraint: Layout density
-constraint_aspectratio <float>
                      Constraint: Layout aspect ratio
-remote [<bool>]      Enable remote processing
-credentials <file>   User credentials file
-cache <file>         User cache directory
-nice <int>           Tool execution scheduling priority
-mode <str>           Compilation mode
-target <str>         Compilation target
-pdk <str>            PDK target
-uselambda [<bool>]   Use lambda scaling
-stackup <str>        Stackup target
-flow <str>           Flow target
-O <str>, -optmode <str>
                      Optimization mode
-frontend <str>       Compilation frontend
-cfg <file>           Configuration manifest
-env 'key <str>'      Environment variables
-var 'key <str>'      Custom variables
-file 'key <file>'    Custom files
-dir 'key <dir>'      Custom directories
-loglevel <str>       Logging level
-builddir <dir>       Build directory
-jobname <str>        Job name
-jobinput 'step index <str>'
                      Input job name
-from <str>           Start flowgraph execution from
-to <str>             End flowgraph execution with
-prune 'node <(str,str)>'
                      Prune flowgraph branches starting with
-breakpoint [<bool>]  Breakpoint list
-showtool 'filetype <str>'
                      Select data display tool
-metricoff '<str>'    Metric summary filter
-library <str>        Soft libraries
-clean [<bool>]       Clean up after run
-hash [<bool>]        Enable file hashing
```

(continues on next page)

(continued from previous page)

```
-nodisplay [<bool>]   Headless execution
-quiet [<bool>]       Quiet execution
-jobincr [<bool>]     Autoincrement jobname
-novercheck [<bool>]  Disable version checking
-relax [<bool>]       Relax design checking
-resume [<bool>]      Resume build
-track [<bool>]       Enable provenance tracking
-trace [<bool>]       Enable debug traces
-skipall [<bool>]     Skip all tasks
-skipcheck [<bool>]   Skip manifest check
-copyall [<bool>]     Copy all inputs to build directory
-show [<bool>]        Show layout
-autoinstall [<bool>]
                      Option: auto install packages
-entrypoint <str>     Program entry point
+incdir+ <dir>, -I <dir>, -idir <dir>
                      Design search paths
-y <dir>, -ydir <dir>
                      Design module search paths
-v <file>, -vlib <file>
                      Design libraries
-D <str>, -define <str>
                      Design pre-processor symbol
+libext+ <str>, -libext <str>
                      Design file extensions
-param 'name <str>'   Design parameter
-f <file>, -cmdfile <file>
                      Design compilation command file
-flowcontinue [<bool>]
                      Flow continue-on-error
-continue [<bool>]    Implementation continue-on-error
-timeout <float>      Option: Timeout value
-strict [<bool>]      Option: Strict checking
-scheduler <str>      Option: Scheduler platform
-cores <int>          Option: Scheduler core constraint
-memory <int>         Option: Scheduler memory constraint
-queue <str>          Option: Scheduler queue
-defer <str>          Option: Scheduler start time
-scheduler_options <str>
                      Option: Scheduler arguments
-msgevent <str>       Option: Message event trigger
-msgcontact <str>     Option: Message contact
-arg_step <str>       ARG: Step argument
-arg_index <str>      ARG: Index argument
-unit_time '<str>'    Unit: time
-unit_length '<str>'  Unit: length
-unit_mass '<str>'    Unit: mass
-unit_temperature '<str>'
                      Unit: temperature
-unit_capacitance '<str>'
                      Unit: capacitance
-unit_resistance '<str>'
```

(continues on next page)

(continued from previous page)

```
                      Unit: resistance
-unit_inductance '<str>'
                      Unit: inductance
-unit_voltage '<str>'
                      Unit: voltage
-unit_current '<str>'
                      Unit: current
-unit_power '<str>'   Unit: power
-unit_energy '<str>'  Unit: energy
-fpga_partname <str>  FPGA: part name
-fpga_vendor 'partname <str>'
                      FPGA: vendor name
-fpga_lutsize 'partname <int>'
                      FPGA: lutsize
-fpga_file 'partname key <file>'
                      FPGA: file
-fpga_var 'partname key <str>'
                      FPGA: var
-fpga_resources_registers 'partname <str>'
                      FPGA: list of registers names
-fpga_resources_dsps 'partname <str>'
                      FPGA: list of dsps names
-fpga_resources_brams 'partname <str>'
                      FPGA: list of brams names
-fpga_board <str>     FPGA: board name
-fpga_program [<bool>]
                      FPGA: program enable
-fpga_flash [<bool>]  FPGA: flash enable
-asic_logiclib <str>  ASIC: logic libraries
-asic_macrolib <str>  ASIC: macro libraries
-asic_delaymodel <str>
                      ASIC: delay model
-asic_cells_decap '<str>'
                      ASIC: decap cell list
-asic_cells_delay '<str>'
                      ASIC: delay cell list
-asic_cells_tie '<str>'
                      ASIC: tie cell list
-asic_cells_hold '<str>'
                      ASIC: hold cell list
-asic_cells_clkbuf '<str>'
                      ASIC: clkbuf cell list
-asic_cells_clkdelay '<str>'
                      ASIC: clkdelay cell list
-asic_cells_clkinv '<str>'
                      ASIC: clkinv cell list
-asic_cells_clkgate '<str>'
                      ASIC: clkgate cell list
-asic_cells_clkicg '<str>'
                      ASIC: clkicg cell list
-asic_cells_clklogic '<str>'
                      ASIC: clklogic cell list
```

(continues on next page)

(continued from previous page)

```
-asic_cells_dontuse '<str>'
                      ASIC: dontuse cell list
-asic_cells_filler '<str>'
                      ASIC: filler cell list
-asic_cells_tap '<str>'
                      ASIC: tap cell list
-asic_cells_endcap '<str>'
                      ASIC: endcap cell list
-asic_cells_antenna '<str>'
                      ASIC: antenna cell list
-asic_libarch '<str>'
                      ASIC: library architecture
-asic_site 'libarch <str>'
                      ASIC: Library sites
-pdk_foundry 'pdkname <str>'
                      PDK: foundry name
-pdk_node 'pdkname <float>'
                      PDK: process node
-pdk_lambda 'pdkname <float>'
                      PDK: Lambda value
-pdk_version 'pdkname <str>'
                      PDK: version
-pdk_stackup 'pdkname <str>'
                      PDK: metal stackups
-pdk_minlayer 'pdk stackup <str>'
                      PDK: minimum routing layer
-pdk_maxlayer 'pdk stackup <str>'
                      PDK: maximum routing layer
-pdk_thickness 'pdkname stackup <float>'
                      PDK: unit thickness
-pdk_wafersize 'pdkname <float>'
                      PDK: wafer size
-pdk_panelsize 'pdkname <(float,float)>'
                      PDK: panel size
-pdk_unitcost 'pdkname <float>'
                      PDK: unit cost
-pdk_d0 'pdkname <float>'
                      PDK: process defect density
-pdk_hscribe 'pdkname <float>'
                      PDK: horizontal scribe line width
-pdk_vscribe 'pdkname <float>'
                      PDK: vertical scribe line width
-pdk_edgemargin 'pdkname <float>'
                      PDK: wafer edge keep-out margin
-pdk_density 'pdkname <float>'
                      PDK: transistor density
-pdk_devmodel 'pdkname tool simtype stackup <file>'
                      PDK: device models
-pdk_pexmodel 'pdkname tool stackup corner <file>'
                      PDK: parasitic TCAD models
-pdk_layermap 'pdkname tool src dst stackup <file>'
                      PDK: layer map file
```

(continues on next page)

(continued from previous page)

```
-pdk_display 'pdkname tool stackup <file>'
                      PDK: display file
-pdk_aprtech 'pdkname tool stackup libarch filetype <file>'
                      PDK: APR technology files
-pdk_lvs_runset 'pdkname tool stackup name <file>'
                      PDK: LVS runset files
-pdk_lvs_waiver 'pdkname tool stackup name <file>'
                      PDK: LVS waiver files
-pdk_drc_runset 'pdkname tool stackup name <file>'
                      PDK: DRC runset files
-pdk_drc_waiver 'pdkname tool stackup name <file>'
                      PDK: DRC waiver files
-pdk_erc_runset 'pdkname tool stackup name <file>'
                      PDK: ERC runset files
-pdk_erc_waiver 'pdkname tool stackup name <file>'
                      PDK: ERC waiver files
-pdk_fill_runset 'pdkname tool stackup name <file>'
                      PDK: FILL runset files
-pdk_fill_waiver 'pdkname tool stackup name <file>'
                      PDK: FILL waiver files
-pdk_file 'pdkname tool key stackup <file>'
                      PDK: special file
-pdk_directory 'pdkname tool key stackup <dir>'
                      PDK: special directory
-pdk_var 'pdkname tool stackup key <str>'
                      PDK: special variable
-pdk_doc_homepage 'pdkname <file>'
                      PDK: documentation homepage
-pdk_doc_datasheet 'pdkname <file>'
                      PDK: datasheet
-pdk_doc_reference 'pdkname <file>'
                      PDK: reference
-pdk_doc_userguide 'pdkname <file>'
                      PDK: userguide
-pdk_doc_install 'pdkname <file>'
                      PDK: install
-pdk_doc_quickstart 'pdkname <file>'
                      PDK: quickstart
-pdk_doc_releasenotes 'pdkname <file>'
                      PDK: releasenotes
-pdk_doc_tutorial 'pdkname <file>'
                      PDK: tutorial
-tool_exe 'tool <str>'
                      Tool: executable name
-tool_sbom 'tool version <file>'
                      Tool: software BOM
-tool_path 'tool <dir>'
                      Tool: executable path
-tool_vswitch 'tool <str>'
                      Tool: executable version switch
-tool_vendor 'tool <str>'
                      Tool: vendor
```

(continues on next page)

(continued from previous page)

```
-tool_version 'tool <str>'
                      Tool: version
-tool_format 'tool <str>'
                      Tool: file format
-tool_licenseserver 'name key <str>'
                      Tool: license servers
-tool_task_warningoff 'tool task <str>'
                      Task: warning filter
-tool_task_continue ['tool task <bool>']
                      Task: continue option
-tool_task_regex 'tool task suffix <str>'
                      Task: regex filter
-tool_task_option 'tool task <str>'
                      Task: executable options
-tool_task_var 'tool task key <str>'
                      Task: script variables
-tool_task_env 'tool task env <str>'
                      Task: environment variables
-tool_task_file 'tool task key <file>'
                      Task: setup files
-tool_task_dir 'tool task key <dir>'
                      Task: setup directories
-tool_task_input 'tool task <file>'
                      Task: inputs
-tool_task_output 'tool task <file>'
                      Task: outputs
-tool_task_stdout_destination 'tool task <str>'
                      Task: Destination for stdout
-tool_task_stdout_suffix 'tool task <str>'
                      Task: File suffix for redirected stdout
-tool_task_stderr_destination 'tool task <str>'
                      Task: Destination for stderr
-tool_task_stderr_suffix 'tool task <str>'
                      Task: File suffix for redirected stderr
-tool_task_require 'tool task <str>'
                      Task: parameter requirements
-tool_task_report 'tool task metric <file>'
                      Task: reports
-tool_task_refdir 'tool task <dir>'
                      Task: script directory
-tool_task_script 'tool task <file>'
                      Task: entry script
-tool_task_prescript 'tool task <file>'
                      Task: pre-step script
-tool_task_postscript 'tool task <file>'
                      Task: post-step script
-tool_task_keep 'tool task <str>'
                      Task: files to keep
-tool_task_threads 'tool task <int>'
                      Task: thread parallelism
-flowgraph_input 'flow step index <(str,str)>'
                      Flowgraph: step input
```

(continues on next page)

(continued from previous page)

```
-flowgraph_weight 'flow step index metric <float>'
                      Flowgraph: metric weights
-flowgraph_goal 'flow step index metric <float>'
                      Flowgraph: metric goals
-flowgraph_tool 'flow step index <str>'
                      Flowgraph: tool selection
-flowgraph_task 'flow step index <str>'
                      Flowgraph: task selection
-flowgraph_taskmodule 'flow step index <str>'
                      Flowgraph: task module
-flowgraph_args 'flow step index <str>'
                      Flowgraph: setup arguments
-flowgraph_timeout 'flow step index <float>'
                      Flowgraph: task timeout value
-flowgraph_status 'flow step index <str>'
                      Flowgraph: task status
-flowgraph_select 'flow step index <(str,str)>'
                      Flowgraph: task select record
-checklist_description 'standard item <str>'
                      Checklist: item description
-checklist_requirement 'standard item <str>'
                      Checklist: item requirement
-checklist_dataformat 'standard item <str>'
                      Checklist: item data format
-checklist_rationale 'standard item <str>'
                      Checklist: item rational
-checklist_criteria 'standard item <str>'
                      Checklist: item criteria
-checklist_task 'standard item <(str,str,str)>'
                      Checklist: item task
-checklist_report 'standard item <file>'
                      Checklist: item report
-checklist_waiver 'standard item metric <file>'
                      Checklist: item metric waivers
-checklist_ok ['standard item <bool>']
                      Checklist: item ok
-metric_errors 'step index <int>'
                      Metric: total errors
-metric_warnings 'step index <int>'
                      Metric: total warnings
-metric_drvs 'step index <int>'
                      Metric: total drvs
-metric_unconstrained 'step index <int>'
                      Metric: total unconstrained
-metric_coverage 'step index <float>'
                      Metric: coverage
-metric_security 'step index <float>'
                      Metric: security
-metric_luts 'step index <int>'
                      Metric: FPGA LUTs used
-metric_dsps 'step index <int>'
                      Metric: FPGA DSP slices used
```

(continues on next page)

(continued from previous page)

```
-metric_brams 'step index <int>'
                      Metric: FPGA BRAM tiles used
-metric_cellarea 'step index <float>'
                      Metric: cellarea
-metric_totalarea 'step index <float>'
                      Metric: totalarea
-metric_utilization step index <float>
                      Metric: area utilization
-metric_logicdepth step index <int>
                      Metric: logic depth
-metric_peakpower 'step index <float>'
                      Metric: peakpower
-metric_averagepower 'step index <float>'
                      Metric: averagepower
-metric_dozepower 'step index <float>'
                      Metric: dozepower
-metric_idlepower 'step index <float>'
                      Metric: idlepower
-metric_leakagepower 'step index <float>'
                      Metric: leakagepower
-metric_sleeppower 'step index <float>'
                      Metric: sleeppower
-metric_irdrop 'step index <float>'
                      Metric: peak IR drop
-metric_holdpaths 'step index <int>'
                      Metric: holdpaths
-metric_setuppaths 'step index <int>'
                      Metric: setuppaths
-metric_holdslack 'step index <float>'
                      Metric: holdslack
-metric_holdwns 'step index <float>'
                      Metric: holdwns
-metric_holdtns 'step index <float>'
                      Metric: holdtns
-metric_setupslack 'step index <float>'
                      Metric: setupslack
-metric_setupwns 'step index <float>'
                      Metric: setupwns
-metric_setuptns 'step index <float>'
                      Metric: setuptns
-metric_fmax 'step index <float>'
                      Metric: fmax
-metric_macros 'step index <int>'
                      Metric: macros
-metric_cells 'step index <int>'
                      Metric: cells
-metric_registers 'step index <int>'
                      Metric: registers
-metric_buffers 'step index <int>'
                      Metric: buffers
-metric_transistors 'step index <int>'
                      Metric: transistors
```

(continues on next page)

(continued from previous page)

```
-metric_pins 'step index <int>'
                      Metric: pins
-metric_nets 'step index <int>'
                      Metric: nets
-metric_vias 'step index <int>'
                      Metric: vias
-metric_wirelength 'step index <float>'
                      Metric: wirelength
-metric_overflow 'step index <int>'
                      Metric: overflow
-metric_memory 'step index <float>'
                      Metric: memory
-metric_exetime 'step index <float>'
                      Metric: exetime
-metric_tasktime 'step index <float>'
                      Metric: tasktime
-metric_totaltime 'step index <float>'
                      Metric: totaltime
-record_userid 'step index <str>'
                      Record: userid
-record_publickey 'step index <str>'
                      Record: public key
-record_machine 'step index <str>'
                      Record: machine name
-record_macaddr 'step index <str>'
                      Record: MAC address
-record_ipaddr 'step index <str>'
                      Record: IP address
-record_platform 'step index <str>'
                      Record: platform name
-record_distro 'step index <str>'
                      Record: distro name
-record_arch 'step index <str>'
                      Record: hardware architecture
-record_starttime 'step index <str>'
                      Record: start time
-record_endtime 'step index <str>'
                      Record: end time
-record_region 'step index <str>'
                      Record: cloud region
-record_scversion 'step index <str>'
                      Record: software version
-record_toolversion 'step index <str>'
                      Record: tool version
-record_toolpath 'step index <str>'
                      Record: tool path
-record_toolargs 'step index <str>'
                      Record: tool CLI arguments
-record_osversion 'step index <str>'
                      Record: O/S version
-record_kernelversion 'step index <str>'
                      Record: O/S kernel version
```

(continues on next page)

(continued from previous page)

```
-record_remoteid 'step index <str>'
                      Record: remote job ID
-datasheet_partnumber '<str>'
                      Datasheet: part number
-datasheet_type '<str>'
                      Datasheet: part type
-datasheet_doc '<file>'
                      Datasheet: part documentation
-datasheet_abstraction '<str>'
                      Datasheet: abstraction level
-datasheet_series '<str>'
                      Datasheet: device series
-datasheet_manufacturer '<str>'
                      Datasheet: part manufacturer
-datasheet_description '<str>'
                      Datasheet: description
-datasheet_features '<str>'
                      Datasheet: part features
-datasheet_grade '<str>'
                      Datasheet: part manufacturing grade
-datasheet_qual '<str>'
                      Datasheet: qualification
-datasheet_trl '<int>'
                      Datasheet: technology readiness level
-datasheet_status '<str>'
                      Datasheet: product status
-datasheet_fmax '<float>'
                      Datasheet: device maximum frequency
-datasheet_ops '<float>'
                      Datasheet: total device operations per second
-datasheet_iobw '<float>'
                      Datasheet: total I/O bandwidth
-datasheet_iocount '<int>'
                      Datasheet: total number of I/Os
-datasheet_ram '<float>'
                      Datasheet: total device RAM
-datasheet_peakpower '<float>'
                      Datasheet: peak power
-datasheet_io_arch 'name <str>'
                      Datasheet: io standard
-datasheet_io_gen 'name <str>'
                      Datasheet: io generation
-datasheet_io_fmax 'name <float>'
                      Datasheet: io maximum frequency
-datasheet_io_width 'name <int>'
                      Datasheet: io width
-datasheet_io_channels 'name <int>'
                      Datasheet: io channels
-datasheet_proc_arch 'name <str>'
                      Datasheet: processor architecture
-datasheet_proc_features 'name <str>'
                      Datasheet: processor features
```

(continues on next page)

(continued from previous page)

```
-datasheet_proc_datatypes 'name <str>'
                      Datasheet: processor datatypes
-datasheet_proc_archsize 'name <int>'
                      Datasheet: processor architecture size
-datasheet_proc_cores 'name <int>'
                      Datasheet: processor number of cores
-datasheet_proc_fmax 'name <int>'
                      Datasheet: processor maximum frequency
-datasheet_proc_ops 'name <int>'
                      Datasheet: processor operations per cycle per core
-datasheet_proc_mults 'name <int>'
                      Datasheet: processor hard multiplier units per core
-datasheet_proc_icache 'name <int>'
                      Datasheet: processor l1 icache size
-datasheet_proc_dcache 'name <int>'
                      Datasheet: processor l1 dcache size
-datasheet_proc_l2cache 'name <int>'
                      Datasheet: processor l2 cache size
-datasheet_proc_l3cache 'name <int>'
                      Datasheet: processor l3 cache size
-datasheet_proc_sram 'name <int>'
                      Datasheet: processor local sram
-datasheet_proc_nvm 'name <int>'
                      Datasheet: processor local non-volatile memory
-datasheet_memory_bits 'name <int>'
                      Datasheet: memory total bits
-datasheet_memory_width 'name <int>'
                      Datasheet: memory width
-datasheet_memory_depth 'name <int>'
                      Datasheet: memory depth
-datasheet_memory_banks 'name <int>'
                      Datasheet: memory banks
-datasheet_memory_fmax 'name <(float,float,float)>'
                      Datasheet: memory max frequency
-datasheet_memory_tcycle 'name <(float,float,float)>'
                      Datasheet: memory access clock cycle
-datasheet_memory_twr 'name <(float,float,float)>'
                      Datasheet: memory write clock cycle
-datasheet_memory_trd 'name <(float,float,float)>'
                      Datasheet: memory read clock cycle
-datasheet_memory_trefresh 'name <(float,float,float)>'
                      Datasheet: memory refresh time
-datasheet_memory_terase 'name <(float,float,float)>'
                      Datasheet: memory erase time
-datasheet_memory_bwrd 'name <(float,float,float)>'
                      Datasheet: memory maximum read bandwidth
-datasheet_memory_bwwr 'name <(float,float,float)>'
                      Datasheet: memory maximum write bandwidth
-datasheet_memory_erd 'name <(float,float,float)>'
                      Datasheet: memory read energy
-datasheet_memory_ewr 'name <(float,float,float)>'
                      Datasheet: memory write energy
```

(continues on next page)

(continued from previous page)

```
-datasheet_memory_twearout 'name <(float,float,float)>'
                      Datasheet: memory write/erase wear-out
-datasheet_memory_tcl 'name <(int,int,int)>'
                      Datasheet: memory column address latency
-datasheet_memory_trcd 'name <(int,int,int)>'
                      Datasheet: memory row address latency
-datasheet_memory_trp 'name <(int,int,int)>'
                      Datasheet: memory row precharge time latency
-datasheet_memory_tras 'name <(int,int,int)>'
                      Datasheet: memory row active time latency
-datasheet_fpga_arch 'name <str>'
                      Datasheet: fpga architecture
-datasheet_fpga_luts 'name <int>'
                      Datasheet: fpga LUTs (4 input)
-datasheet_fpga_registers 'name <int>'
                      Datasheet: fpga registers
-datasheet_fpga_plls 'name <int>'
                      Datasheet: fpga pll blocks
-datasheet_fpga_mults 'name <int>'
                      Datasheet: fpga multiplier/dsp elements
-datasheet_fpga_totalram 'name <int>'
                      Datasheet: fpga total ram
-datasheet_fpga_distram 'name <int>'
                      Datasheet: fpga distributed ram
-datasheet_fpga_blockram 'name <int>'
                      Datasheet: fpga block ram
-datasheet_analog_arch 'name <str>'
                      Datasheet: analog architecture
-datasheet_analog_features 'name <str>'
                      Datasheet: analog features
-datasheet_analog_resolution 'name <int>'
                      Datasheet: Analog architecture resolution
-datasheet_analog_channels 'name <int>'
                      Datasheet: Analog parallel channels
-datasheet_analog_samplerate 'name <(float,float,float)>'
                      Datasheet: Analog sample rate
-datasheet_analog_enob 'name <(float,float,float)>'
                      Datasheet: Analog effective number of bits
-datasheet_analog_inl 'name <(float,float,float)>'
                      Datasheet: Analog integral nonlinearity
-datasheet_analog_dnl 'name <(float,float,float)>'
                      Datasheet: Analog differential nonlinearity
-datasheet_analog_snr 'name <(float,float,float)>'
                      Datasheet: Analog signal to noise ratio
-datasheet_analog_sinad 'name <(float,float,float)>'
                      Datasheet: Analog signal to noise and distortion ratio
-datasheet_analog_sfdr 'name <(float,float,float)>'
                      Datasheet: Analog spurious-free dynamic range
-datasheet_analog_thd 'name <(float,float,float)>'
                      Datasheet: Analog total harmonic distortion
-datasheet_analog_imd3 'name <(float,float,float)>'
                      Datasheet: Analog 3rd order intermodulation distortion
```

(continues on next page)

(continued from previous page)

```
-datasheet_analog_hd2 'name <(float,float,float)>'
                      Datasheet: Analog 2nd order harmonic distortion
-datasheet_analog_hd3 'name <(float,float,float)>'
                      Datasheet: Analog 3rd order harmonic distortion
-datasheet_analog_hd4 'name <(float,float,float)>'
                      Datasheet: Analog 4th order harmonic distortion
-datasheet_analog_nsd 'name <(float,float,float)>'
                      Datasheet: Analog noise spectral density
-datasheet_analog_phasenoise 'name <(float,float,float)>'
                      Datasheet: Analog phase noise
-datasheet_analog_gain 'name <(float,float,float)>'
                      Datasheet: Analog gain
-datasheet_analog_pout 'name <(float,float,float)>'
                      Datasheet: Analog output power
-datasheet_analog_pout2 'name <(float,float,float)>'
                      Datasheet: Analog 2nd harmonic power
-datasheet_analog_pout3 'name <(float,float,float)>'
                      Datasheet: Analog 3rd harmonic power
-datasheet_analog_vofferror 'name <(float,float,float)>'
                      Datasheet: Analog offset error
-datasheet_analog_vgainerror 'name <(float,float,float)>'
                      Datasheet: Analog gain error
-datasheet_analog_cmrr 'name <(float,float,float)>'
                      Datasheet: Analog common mode rejection ratio
-datasheet_analog_psnr 'name <(float,float,float)>'
                      Datasheet: Analog power supply noise rejection
-datasheet_analog_s21 'name <(float,float,float)>'
                      Datasheet: Analog rf gain
-datasheet_analog_s11 'name <(float,float,float)>'
                      Datasheet: Analog rf input return loss
-datasheet_analog_s22 'name <(float,float,float)>'
                      Datasheet: Analog rf output return loss
-datasheet_analog_s12 'name <(float,float,float)>'
                      Datasheet: Analog rf reverse isolation
-datasheet_analog_noisefigure 'name <(float,float,float)>'
                      Datasheet: Analog rf noise figure
-datasheet_analog_ib1db 'name <(float,float,float)>'
                      Datasheet: Analog rf in band 1 dB compression point
-datasheet_analog_oob1db 'name <(float,float,float)>'
                      Datasheet: Analog rf out of band 1 dB compression
                      point
-datasheet_analog_iip3 'name <(float,float,float)>'
                      Datasheet: Analog rf 3rd order input intercept point
-datasheet_limit_tstorage '<(float,float)>'
                      Datasheet: limit storage temperature limits
-datasheet_limit_tsolder '<(float,float)>'
                      Datasheet: limit solder temperature limits
-datasheet_limit_tj '<(float,float)>'
                      Datasheet: limit junction temperature limits
-datasheet_limit_ta '<(float,float)>'
                      Datasheet: limit ambient temperature limits
-datasheet_limit_tid '<(float,float)>'
```

(continues on next page)

(continued from previous page)

```
                      Datasheet: limit total ionizing dose threshold
-datasheet_limit_sel '<(float,float)>'
                      Datasheet: limit single event latchup threshold
-datasheet_limit_seb '<(float,float)>'
                      Datasheet: limit single event burnout threshold
-datasheet_limit_segr '<(float,float)>'
                      Datasheet: limit single event gate rupture threshold
-datasheet_limit_set '<(float,float)>'
                      Datasheet: limit single event transient threshold
-datasheet_limit_seu '<(float,float)>'
                      Datasheet: limit single event upset threshold
-datasheet_limit_vhbm '<(float,float)>'
                      Datasheet: limit ESD human body model voltage level
-datasheet_limit_vcdm '<(float,float)>'
                      Datasheet: limit ESD charge device model voltage level
-datasheet_limit_vmm '<(float,float)>'
                      Datasheet: limit ESD machine model voltage level
-datasheet_thermal_rja '<float>'
                      Datasheet: thermal junction to ambient resistance
-datasheet_thermal_rjct '<float>'
                      Datasheet: thermal junction to case (top) resistance
-datasheet_thermal_rjcb '<float>'
                      Datasheet: thermal junction to case (bottom)
                      resistance
-datasheet_thermal_rjb '<float>'
                      Datasheet: thermal junction to board resistance
-datasheet_thermal_tjt '<float>'
                      Datasheet: thermal junction to top model
-datasheet_thermal_tjb '<float>'
                      Datasheet: thermal junction to bottom model
-datasheet_package_name '<str>'
                      Datasheet: package name
-datasheet_package_drawing '<file>'
                      Datasheet: package drawing
-datasheet_package_pincount '<int>'
                      Datasheet: package pincount
-datasheet_package_length '<(float,float,float)>'
                      Datasheet: package length
-datasheet_package_width '<(float,float,float)>'
                      Datasheet: package width
-datasheet_package_thickness '<(float,float,float)>'
                      Datasheet: package thickness
-datasheet_package_pinpitch '<(float,float,float)>'
                      Datasheet: package pitch
-datasheet_pin_map 'name bump <(float,float)>'
                      Datasheet: pin map
-datasheet_pin_type 'name mode <str>'
                      Datasheet: pin type
-datasheet_pin_dir 'name mode <str>'
                      Datasheet: pin direction
-datasheet_pin_complement 'name mode <str>'
                      Datasheet: pin complement
```

(continues on next page)

(continued from previous page)

```
-datasheet_pin_standard 'name mode <str>'
                      Datasheet: pin standard
-datasheet_pin_interface 'name mode <str>'
                      Datasheet: pin interface map
-datasheet_pin_resetvalue 'name mode <str>'
                      Datasheet: pin reset value
-datasheet_pin_vmax 'pin mode <(float,float,float)>'
                      Datasheet: pin absolute maximum voltage
-datasheet_pin_vnominal 'pin mode <(float,float,float)>'
                      Datasheet: pin nominal operating voltage
-datasheet_pin_vol 'pin mode <(float,float,float)>'
                      Datasheet: pin low output voltage level
-datasheet_pin_voh 'pin mode <(float,float,float)>'
                      Datasheet: pin high output voltage level
-datasheet_pin_vil 'pin mode <(float,float,float)>'
                      Datasheet: pin low input voltage level
-datasheet_pin_vih 'pin mode <(float,float,float)>'
                      Datasheet: pin high input voltage level
-datasheet_pin_vcm 'pin mode <(float,float,float)>'
                      Datasheet: pin common mode voltage
-datasheet_pin_vdiff 'pin mode <(float,float,float)>'
                      Datasheet: pin differential voltage
-datasheet_pin_voffset 'pin mode <(float,float,float)>'
                      Datasheet: pin offset voltage
-datasheet_pin_vnoise 'pin mode <(float,float,float)>'
                      Datasheet: pin random voltage noise
-datasheet_pin_vslew 'pin mode <(float,float,float)>'
                      Datasheet: pin slew rate
-datasheet_pin_vhbm 'pin mode <(float,float,float)>'
                      Datasheet: pin ESD human body model voltage level
-datasheet_pin_vcdm 'pin mode <(float,float,float)>'
                      Datasheet: pin ESD charge device model voltage level
-datasheet_pin_vmm 'pin mode <(float,float,float)>'
                      Datasheet: pin ESD machine model voltage level
-datasheet_pin_cap 'pin mode <(float,float,float)>'
                      Datasheet: pin capacitance
-datasheet_pin_rdiff 'pin mode <(float,float,float)>'
                      Datasheet: pin differential pair resistance
-datasheet_pin_rin 'pin mode <(float,float,float)>'
                      Datasheet: pin input resistance
-datasheet_pin_rup 'pin mode <(float,float,float)>'
                      Datasheet: pin output pullup resistance
-datasheet_pin_rdown 'pin mode <(float,float,float)>'
                      Datasheet: pin output pulldown resistance
-datasheet_pin_rweakup 'pin mode <(float,float,float)>'
                      Datasheet: pin weak pullup resistance
-datasheet_pin_rweakdown 'pin mode <(float,float,float)>'
                      Datasheet: pin weak pulldown resistance
-datasheet_pin_power 'pin mode <(float,float,float)>'
                      Datasheet: pin power consumption
-datasheet_pin_isupply 'pin mode <(float,float,float)>'
                      Datasheet: pin supply current
```

(continues on next page)

(continued from previous page)

```
-datasheet_pin_ioh 'pin mode <(float,float,float)>'
                      Datasheet: pin output high current
-datasheet_pin_iol 'pin mode <(float,float,float)>'
                      Datasheet: pin output low current
-datasheet_pin_iinject 'pin mode <(float,float,float)>'
                      Datasheet: pin injection current
-datasheet_pin_ishort 'pin mode <(float,float,float)>'
                      Datasheet: pin short circuit current
-datasheet_pin_ioffset 'pin mode <(float,float,float)>'
                      Datasheet: pin offset current
-datasheet_pin_ibias 'pin mode <(float,float,float)>'
                      Datasheet: pin bias current
-datasheet_pin_ileakage 'pin mode <(float,float,float)>'
                      Datasheet: pin leakage current
-datasheet_pin_tperiod 'pin mode <(float,float,float)>'
                      Datasheet: pin minimum period
-datasheet_pin_tpulse 'pin mode <(float,float,float)>'
                      Datasheet: pin pulse width
-datasheet_pin_tjitter 'pin mode <(float,float,float)>'
                      Datasheet: pin rms jitter
-datasheet_pin_thigh 'pin mode <(float,float,float)>'
                      Datasheet: pin pulse width high
-datasheet_pin_tlow 'pin mode <(float,float,float)>'
                      Datasheet: pin pulse width low
-datasheet_pin_tduty 'pin mode <(float,float,float)>'
                      Datasheet: pin duty cycle
-datasheet_pin_tsetup 'pin mode relpin <(float,float,float)>'
                      Datasheet: pin setup time
-datasheet_pin_thold 'pin mode relpin <(float,float,float)>'
                      Datasheet: pin hold time
-datasheet_pin_tskew 'pin mode relpin <(float,float,float)>'
                      Datasheet: pin timing skew
-datasheet_pin_tdelayr 'pin mode relpin <(float,float,float)>'
                      Datasheet: pin propagation delay (rise)
-datasheet_pin_tdelayf 'pin mode relpin <(float,float,float)>'
                      Datasheet: pin propagation delay (fall)
-datasheet_pin_trise 'pin mode relpin <(float,float,float)>'
                      Datasheet: pin rise transition
-datasheet_pin_tfall 'pin mode relpin <(float,float,float)>'
                      Datasheet: pin fall transition
-package_version <str>
                      Package: version
-package_description <str>
                      Package: description
-package_keyword <str>
                      Package: keyword
-package_doc_homepage <str>
                      Package: documentation homepage
-package_doc_datasheet <file>
                      Package: datasheet document
-package_doc_reference <file>
                      Package: reference document
```

(continues on next page)

(continued from previous page)

```
  -package_doc_userguide <file>
                        Package: userguide document
  -package_doc_quickstart <file>
                        Package: quickstart document
  -package_doc_releasenotes <file>
                        Package: releasenotes document
  -package_doc_testplan <file>
                        Package: testplan document
  -package_doc_signoff <file>
                        Package: signoff document
  -package_doc_tutorial <file>
                        Package: tutorial document
  -package_license <str>
                        Package: license identifiers
  -package_licensefile <file>
                        Package: license files
  -package_organization <str>
                        Package: sponsoring organization
  -package_author_name 'userid <str>'
                        Package: author name
  -package_author_email 'userid <str>'
                        Package: author email
  -package_author_username 'userid <str>'
                        Package: author username
  -package_author_location 'userid <str>'
                        Package: author location
  -package_author_organization 'userid <str>'
                        Package: author organization
  -package_author_publickey 'userid <str>'
                        Package: author publickey
  -package_source_path 'source <str>'
                        Package data source path
  -package_source_ref 'source <str>'
                        Package data source reference
  -version              show program's version number and exit
```

3.9.2 sc-dashboard

```
usage: sc-dashboard [-h] [-design <str>] [-cfg <file>] [-loglevel <str>]
                    [-jobname <str>] [-arg_step <str>] [-arg_index <str>]
                    [-version] [-port <port>]
                    [-graph_cfg <[manifest name, manifest path> [<[manifest name,␣
↪manifest path> ...]]

------------------------------------------------------------
SC app to open a dashboard for a given manifest.

To open:
    sc-dashboard -cfg <path to manifest>
```

(continues on next page)

(continued from previous page)

```
To specify a different port than the default:
    sc-dashboard -cfg <path to manifest> -port 10000

To include another chip object to compare to:
    sc-dashboard -cfg <path to manifest> -graph_cfg <name of manifest> <path to other␣
↪manifest>
        -graph_cfg <path to other manifest> ...
------------------------------------------------------------

options:
  -h, --help            show this help message and exit
  -design <str>         Design top module name
  -cfg <file>           Configuration manifest
  -loglevel <str>       Logging level
  -jobname <str>        Job name
  -arg_step <str>       ARG: Step argument
  -arg_index <str>      ARG: Index argument
  -version              show program's version number and exit
  -port <port>          port to open the dashboard app on
  -graph_cfg <[manifest name, manifest path> [<[manifest name, manifest path> ...]
                        chip name - optional, path to chip manifest (json)
```

3.9.3 sc-issue

```
usage: sc-issue [-h] [-cfg <file>] [-loglevel <str>] [-arg_step <str>]
                [-arg_index <str>] [-tool_task_option 'tool task <str>']
                [-tool_task_var 'tool task key <str>'] [-version] [-generate]
                [-exclude_libraries] [-exclude_pdks] [-hash_files] [-run]
                [-use <module>] [-add_library <library>] [-add_pdk <pdk>]
                [-file <file>]

------------------------------------------------------------
Restricted SC app that generates a sharable testcase from a
failed flow or runs an issue generated with this program.

To generate a testcase, use:
    sc-issue -generate -cfg <stepdir>/outputs/<design>.pkg.json

    or include a different step/index than what the cfg_file is pointing to:
    sc-issue -generate -cfg <otherdir>/outputs/<design>.pkg.json -arg_step <step> -arg_
↪index <index>

    or include specific libraries while excluding others:
    sc-issue -generate -cfg <stepdir>/outputs/<design>.pkg.json -exclude_libraries -add_
↪library sram -add_library gpio

To run a testcase, use:
    sc-issue -run -file sc_issue_<...>.tar.gz
------------------------------------------------------------
```

(continues on next page)

(continued from previous page)

```
options:
  -h, --help            show this help message and exit
  -cfg <file>           Configuration manifest
  -loglevel <str>       Logging level
  -arg_step <str>       ARG: Step argument
  -arg_index <str>      ARG: Index argument
  -tool_task_option 'tool task <str>'
                        Task: executable options
  -tool_task_var 'tool task key <str>'
                        Task: script variables
  -version              show program's version number and exit
  -generate             generate a testcase
  -exclude_libraries    flag to ensure libraries are excluded in the testcase
  -exclude_pdks         flag to ensure pdks are excluded in the testcase
  -hash_files           flag to hash the files in the schema before generating
                        the manifest
  -run                  run a provided testcase
  -use <module>         modules to load into test run
  -add_library <library>
                        library to include in the testcase, if not provided
                        all libraries will be added according to the
                        -exclude_libraries flag
  -add_pdk <pdk>        pdk to include in the testcase, if not provided all
                        libraries will be added according to the -exclude_pdks
                        flag
  -file <file>          filename for the generated testcase
```

3.9.4 sc-remote

```
usage: sc-remote [-h] [-credentials <file>] [-cfg <file>] [-version]
                 [-configure] [-server <server>] [-reconnect] [-cancel]
                 [-delete]

-------------------------------------------------------------
SC app that provides an entry point to common remote / server
interactions.

To generate a configuration file, use:
    sc-remote -configure

    or to specify a specific server and/or port:
    sc-remote -configure -server https://example.com
    sc-remote -configure -server https://example.com:1234

To check an ongoing job's progress, use:
    sc-remote -cfg <stepdir>/outputs/<design>.pkg.json

To cancel an ongoing job, use:
    sc-remote -cancel -cfg <stepdir>/outputs/<design>.pkg.json
```

(continues on next page)

(continued from previous page)

```
To reconnect an ongoing job, use:
    sc-remote -reconnect -cfg <stepdir>/outputs/<design>.pkg.json

To delete a job, use:
    sc-remote -delete -cfg <stepdir>/outputs/<design>.pkg.json
------------------------------------------------------------

options:
  -h, --help           show this help message and exit
  -credentials <file>  User credentials file
  -cfg <file>          Configuration manifest
  -version             show program's version number and exit
  -configure           create configuration file for the remote
  -server <server>     address of server for configure
  -reconnect           reconnect to a running job on the remote
  -cancel              cancel a running job on the remote
  -delete              delete a job on the remote
```

3.9.5 sc-run

```
usage: sc-run [-h] [-cfg <file>] [-loglevel <str>] [-quiet [<bool>]]
              [-relax [<bool>]] [-version]

    ------------------------------------------------------------
    Restricted SC app that accepts one or more json based cfg files
    as inputs and executes the SC run() method.
    ------------------------------------------------------------


options:
  -h, --help       show this help message and exit
  -cfg <file>      Configuration manifest
  -loglevel <str>  Logging level
  -quiet [<bool>]  Quiet execution
  -relax [<bool>]  Relax design checking
  -version         show program's version number and exit
```

3.9.6 sc-server

```
usage: sc-server [-h] [-schemaversion <str>] [-port <int>] [-cluster <str>]
                 [-nfsmount <dir>] [-auth [<bool>]] [-cfg <file>]
                 [-loglevel <str>] [-version]

------------------------------------------------------------
Silicon Compiler Collection Remote Job Server (sc-server)
------------------------------------------------------------

options:
  -h, --help            show this help message and exit
```

(continues on next page)

(continued from previous page)

```
  -schemaversion <str>  Schema version number
  -port <int>           Port number to run the server on.
  -cluster <str>        Type of compute cluster to use.
  -nfsmount <dir>       Directory of mounted shared NFS storage.
  -auth [<bool>]        Flag determining whether to enable authenticated and
                        encrypted jobs.
  -cfg <file>           Configuration manifest
  -loglevel <str>       Logging level
  -version              show program's version number and exit
```

3.9.7 sc-show

```
usage: sc-show [-h] [-design <str>] [-input 'fileset filetype <file>']
               [-cfg <file>] [-loglevel <str>] [-jobname <str>]
               [-arg_step <str>] [-arg_index <str>] [-version] [-ext <ext>]
               [-screenshot]
               [source ...]

    ------------------------------------------------------------
    Restricted SC app that displays the layout of a design
    based on a file provided or tries to display the final
    layout based on loading the json manifest from:
    build/<design>/job0/<design>.pkg.json

    Examples:

    sc-show -design adder
    (displays build/adder/job0/export/0/outputs/adder.gds)

    sc-show -design adder -arg_step floorplan
    (displays build/adder/job0/floorplan/0/outputs/adder.def)

    sc-show -design adder -arg_step place -arg_index 1
    (displays build/adder/job0/place/1/outputs/adder.def)

    sc-show -design adder -jobname rtl2gds
    (displays build/adder/rtl2gds/export/0/outputs/adder.gds)

    sc-show -cfg build/adder/rtl2gds/adder.pkg.json
    (displays build/adder/rtl2gds/export/0/outputs/adder.gds)

    sc-show -design adder -ext odb
    (displays build/adder/job0/export/1/outputs/adder.odb)

    sc-show build/adder/job0/route/1/outputs/adder.def
    (displays build/adder/job0/route/1/outputs/adder.def)


positional arguments:
  source                Input files with filetype inferred by extension
```

(continues on next page)

(continued from previous page)

```
options:
  -h, --help            show this help message and exit
  -design <str>         Design top module name
  -input 'fileset filetype <file>'
                        Input: files
  -cfg <file>           Configuration manifest
  -loglevel <str>       Logging level
  -jobname <str>        Job name
  -arg_step <str>       ARG: Step argument
  -arg_index <str>      ARG: Index argument
  -version              show program's version number and exit
  -ext <ext>            (optional) Specify the extension of the file to show.
  -screenshot           (optional) Will generate a screenshot and exit.
```

3.10 Server API

3.10.1 delete_job

Request

Schema describing parameters for deleting project data from remote storage.

Username		
	description	User account ID. Required for authentication if the data was originally imported by a valid user.
	type	string
	regex match	^[^\s;]*$
Authentication Key		
	description	Password or Base64-encoded decryption key for the user account, depending on the server's authentication scheme.
	type	string
Job Hash		
	description	UUID associated with the data that should be deleted.
	type	string
	regex match	^[0-9a-f]{32}$

Response

Reason	Status Code	Response Format
Job does not exist	**404**	`{ "message": "String", "success": "Boolean" }`
Job was deleted	**200**	`{ "message": "String", "success": "Boolean" }`

3.10.2 cancel_job

Request

Schema describing parameters for canceling an ongoing job run.

Username	
description	User account ID. Required for authentication if the data was originally imported by a valid user.
type	string
regex match	^[^\s;]*$
Authentication Key	
description	Base64-encoded decryption key for the user account's public key. Required if 'username' is provided.
type	string
Job Hash	
description	UUID associated with the data that should be cancelled.
type	string
regex match	^[0-9a-f]{32}$

Response

Reason	Status Code	Response Format
Job does not exist	**404**	`{ "message": "String", "success": "Boolean" }`
Job was canceled	**200**	`{ "message": "String", "success": "Boolean" }`

3.10.3 check_server

Request

Schema describing parameters for checking on server-side software versions, and optionally a given user account's information.

Username	
description	User account identifier.
type	string
regex match	^[^\s;]*$
Authentication Key	
description	Key/password for the user's account.
type	string

Response

Reason	Status Code	Response Format
Unauthenticated call	**200**	`{ "status": "String", "terms": "String", "versions": { "sc": "String", "sc_schema": "String", "sc_server": " String" } }`
Authenticated call	**200**	`{ "status": "String", "terms": "String", "versions": { "sc": "String", "sc_schema": "String", "sc_server": " String" }, "user_info": { "compute_time": ↪"Integer", "bandwidth_kb": ↪"Integer" } }`

3.10.4 get_results/{job_hash}.tar.gz

Request

Schema describing parameters for checking the progress of an ongoing job.

Username	
description	User account ID. Required for authentication if the job was originally created by a valid user.
type	string
regex match	^[^\s;]*$
Authentication Key	
description	Base64-encoded decryption key for the user account's public key. Required if 'username' is provided.
type	string
Flowgraph node	
description	Retrieve results only for a specific node in the flowgraph. Used to retrieve logs/etc for in-progress jobs.
type	string

Response

Reason	Status Code	Response Format
Job does not exist	**404**	`{ "message": "String" }`
Result size is over the download limit	**503**	`{ "message": "String" }`
Results are available	**200**	`[ "File Response" ]`

3.10.5 remote_run

Request

Schema for metadata required to remotely run a siliconcompiler job stage.

Username	
description	User account ID. Required if data is encrypted.
type	string
regex match	^[^\s;]*$
Authentication Key	
description	Password or Base64-encoded decryption key for the user account, depending on the server's authentication scheme.
type	string

Response

Reason	Status Code	Response Format
Too many nodes in flowgraph	**403**	`{ "message": "String" }`
User account is out of bandwidth or time	**400**	`{ "message": "String" }`
Job started successfully	**200**	`{ "message": "String", "interval": "Integer", "job_hash": "String" }`

3.10.6 check_progress

Request

Schema describing parameters for checking the progress of an ongoing job.

Username	
description	User account ID. Required for authentication if the job was originally created by a valid user.
type	string
regex match	^[^\s;]*$
Authentication Key	
description	Base64-encoded decryption key for the user account's public key. Required if 'username' is provided.
type	string
Job Hash	
description	UUID associated with the data that the job is operating on.
type	string
regex match	^[0-9a-f]{32}$
Job ID	
description	ID associated with the 'job_hash' and the individual job that is being checked on.
type	string

Response

Reason	Status Code	Response Format
Job is not running	**200**	`{ "message": "String", "status": "String" }`
Job cannot be found	**404**	`{ "message": "String", "status": "String" }`
Job is running	**200**	`{ "message": "String", "status": "String", "[nodename]": { "status": "String", "elapsed_time": "String (Optional)" } }`

3.11 Leflib API

sc_leflib.**parse**(*path*)

Parses LEF file.

Given a path to a LEF file, this function parses the file and returns a dictionary representing the contents of the LEF file. If there's an error while reading or parsing the file, this function returns None instead.

Note that this function does not return all information contained in the LEF. The subset of information returned includes:

- LEF version
- Bus bit characters
- Divider characters
- Units
- Manufacturing grid
- Use min spacing
- Clearance measure
- Fixed mask
- Layer information
  - Type
  - Width
  - Direction
  - Offset
  - Pitch
- Max stack via
- Viarules
- Sites
- Macro information
  - Size
  - Pins
  - Obstructions

The dictionary returned by this function is designed to mimic the structure of the LEF file as closely as possible, and this function does minimal legality checking. The order all top-level objects appear in the dictionary is guaranteed to match the LEF file. It looks like follows:

```
{
    'version': 5.8,
    'busbitchars': '<>',
    'dividerchar': ':',
    'units': {
        'capacitance': 10.0,
        'current': 10000.0,
```

(continues on next page)

(continued from previous page)

```
        'database': 20000.0,
        'frequency': 10.0,
        'power': 10000.0,
        'resistance': 10000.0,
        'time': 100.0,
        'voltage': 1000.0
    },
    'manufacturinggrid': 0.05,
    'useminspacing': {'OBS': 'OFF'},
    'clearancemeasure': 'MAXXY',
    'fixedmask': True,
    'layers': {
        'M1': {
            'type': 'ROUTING',
            'direction': 'HORIZONTAL',
            'offset': (0.1, 0.2),
            'pitch': 1.8,
            'width': 1.0
        },
        'V1': {
            'type': 'CUT',
        },
        ...
    },
    'maxviastack': {'range': {'bottom': 'm1', 'top': 'm7'}, 'value': 4},
    'viarules': {
        '<name>': {
            'generate': True,
            'layers': [
                {'enclosure': {'overhang1': 1.4,
                               'overhang2': 1.5},
                 'name': 'M1',
                 'width': {'max': 19.0, 'min': 0.1}},
                {'enclosure': {'overhang1': 1.4,
                               'overhang2': 1.5},
                 'name': 'M2',
                 'width': {'max': 1.9, 'min': 0.2}},
                {'name': 'M3',
                 'rect': (-0.3, -0.3, -0.3, 0.3),
                 'resistance': 0.5,
                 'spacing': {'x': 5.6, 'y': 7.0}}
            ]
        },
        '<name>': {
            {'layers': [
                {'direction': 'VERTICAL',
                 'name': 'M1',
                 'width': {'max': 9.6, 'min': 9.0}},
                 {'direction': 'HORZIONTAL',
                 'name': 'M1',
                 'width': {'max': 3.0, 'min': 3.0}}
            ]}
```

(continues on next page)

(continued from previous page)

```
        },
        ...
    }
    'macros': {
        '<name>': {
            'size': {
                'width': 5,
                'height': 8
            },
            'pins': {
                '<name>': {
                    'ports': [{
                        'class': 'CORE',
                        'layer_geometries': [{
                            'layer': 'M1',
                            'exceptpgnet': True,
                            'spacing': 0.01,
                            'designrulewidth': 0.05,
                            'width': 1.5,
                            'shapes': [
                                {
                                    'rect': (0, 0, 5, 5),
                                    'mask': 1,
                                    'iterate': {
                                        'num_x': 2,
                                        'num_y': 3,
                                        'space_x': 1,
                                        'space_y': 4
                                    }
                                },
                                {
                                    'path': [(0, 0), (5, 0), (0, 5)],
                                    'iterate': ...
                                },
                                {
                                    'polygon': [(0, 0), (5, 0), (0, 5)],
                                    'iterate': ...
                                }
                            ],
                            'via': {
                                'pt': (2, 3),
                                'name': 'via1',
                                'iterate': ...
                            }
                        }]
                    }]
                },
                ...
            }
        },
        ...
    }
```

(continues on next page)

(continued from previous page)

```
}
```

If some entry is not specified in the LEF, the corresponding key will not be present in the dictionary.

Parameters
path (*str*) – Path to LEF file to parse.

3.12 Design Glossary

3.12.1 Architecture

- ADC: Analog to Digital Converter
- AES: Advanced encryption standard
- Adder: Circuit to add two numbers
- ALU: Arithmetic logic unit
- Amdahl's Law: Amdahl's law of diminishing returns for speeding up fixed workloads
- Arbiter: Arbitrates between competing requesters
- ASIC: Application specific integrated circuit.
- Audio codec: Device/program that compresses/decompresses digital audio
- Boolean algebra: Algebra in which variables are either true or false
- BTB: Branch target buffer
- Cache: Local storage of program and/or data for future use.
- Cache coherence: Consistency of shared data that is stored in multiple local caches.
- CAM: Content addressable memory
- CISC: Complex instruction set computing
- Coprocessor: A processor used to supplement operations of a primary (host) processor.
- CPI: Cycles per instruction
- CPU: Central processing unit
- CRC: Cyclic redundancy check
- CSA: Carry save adder
- DAC: Digital to Analog Converter
- Distributed Computing: Computer with components working towards common goal with without strict coupling.
- DLL: Delay locked loop
- DMA: Direct memory access
- DDR:Double data rate
- DDS: Direct digital synthesis
- DSM: Distributed shared memory
- DSP: Digital signal processor

- ECC: Error correcting code
- Ethernet: Family of standard network technologies
- Fault Tolerance: The ability of a system to keep operating in the event of failure of one of its components.
- FRAM: Non-volatile RAM based on ferroelectric layer.
- FPGA: Field-programmable gate array is a chip that can be reprogrammed “in the field”.
- FIFO: First in first out buffer
- GPU: Integrated circuit for accelerating the creation of graphics on a display.
- DRAM: Dynamic random-access semiconductor memory
- Flash: Non-volatile semiconductor memory
- FFT: Fast Fourier transform
- FPU: Floating point unit
- GPIO: General purpose input output, controllable at run time
- Gray code: Binary system where successive values differ by one bit
- HBM: High bandwidth memory
- I2C: Multi-master 2 wire bus
- LAN: Local area network
- LFSR: Linear feedback shift register
- LSB: Least significant bit
- LUT: An array that replaces runtime computation with a simpler array indexing operation
- LVDS: Low-voltage differential signaling (also TIA/EIA-644)
- MII: Media independent interface for PHY chips
- MIMD: Multiple instructions multiple data architecture
- MMU: Memory management unit
- MSB: Most significant bit
- MUX: Multiplexer
- Multiplier: Binary multiplier
- NCO: Numerically controlled oscillator
- NOC: Network on a chip
- Parallel Computing: A type of computation where many operations are carried out simultaneously.
- PCM: Phase change memory
- PCIe: High Speed serial computer expansion bus
- PIC: Programmable interrupt controller
- Priority Encoder: A circuit or algorithm that compresses multiple binary inputs into a smaller number of outputs
- PLL: Phase locked loop
- PWM: Pulse width modulation
- Q: Q fixed point number format

- RAID: Redundant array of disks
- Reconfigurable Computing: Collection of customizable datapaths connected together by a fabric
- RISC: Reduced instruction set computing
- ROM: Read only memory (denser than RAM)
- SBC: Single board computers
- SDR: Software defined radio
- SERDES: Serializer/deserializer
- Shift Register: Set of registers that shifts bits one position at a time
- SIMD: Single instruction multiple data
- Schmitt Trigger: Comparator circuit with hysteresis
- SPI: Synchronous 4 wire master/slave interface
- SRAM: Static random access semiconductor memory
- TLB: Translation lookaside buffer
- UART: Asynchronous 2 wire point to point interface
- USB: 2 wire point to point 5 V interface
- Video codec: Device/program that compresses/decompresses digital video
- Virtual Memory: The automatic mapping of virtual program addresses to physical addresses
- VLIW: Very long instruction level parallelism
- WAN: Wide area network
- WIFI: Wireless local area network
- 8b10b: Code that maps 8-bits to 10bit DC balanced symbols

3.12.2 Design

- Antenna effect: Plasma induced gate oxide damage that can occur during semiconductor processing.
- Asynchronous logic: Logic not governed by a clock circuit or global clock.
- ATPG: Automatic test pattern generation
- BIST: Built in Self Test
- Chip: A set of electronic circuits on one small plate ("chip") of semiconductor material, normally silicon.
- Clock domain crossing: Traversal of signal in synchronous digital ssytem from one clock domain to another.
- Clock gating: Technique whereby clock in synchronous logic is shut off when idle.
- CMOS: Complimentary metal-oxide semiconductor
- Cross talk: The coupling of nearby signals on a chip, usually through capacitive coupling.
- CTS: Clock tree synthesis
- Domino logic: Fast clocked logic with reduced capacitive load
- DEF: Design Exchange Format for layout
- DFM: Extended DRC rules specifying how to make a high yielding design.

- DFT: Design for test
- Die: Small block of semiconductor material that can be cut ("diced") from a silicon wafer.
- DRC: Design Rule Constraints specifying manufacturing constraints.
- DV: Design verification is the process of verifying that the logic design conforms to specification.
- ECO: Engineering change order
- EDA: Electronic Design Automation tools used to enhance chip design productivity.
- EDA companies: List of EDA companies
- Electromigration: Transport of material caused by the gradual movement of the ions in a conductor.
- EMI: Electromagnetic interference.
- ESD: Electrostatic discharge is the sudden flow of electricity between two electrically charged objects.
- Fabless: The design and sale of semiconductor devices while outsourcing the manufacturing to 3rd party.
- FEOL: Front end of line processing. Includes all chip processing up to but not including metal interconnect layers.
- Flip-flop:: A clocked circuit that has two stable states and can be used to store state information.
- Foundry: Semiconductor company offering manufacturing services.
- Full custom design: Design methodology involving layout and interconnection of individual transistors.
- GDSII: Binary format of design database sent to foundry.
- Hardware Emulation: Process of imitating the behavior a system under design with another piece of hardware.
- HDL: Specialized hardware description language for describing electronic circuits.
- Hold time: Minimum time synchronous input should hold steady after clock event.
- IP: Semiconductor reusable design blocks containing author's Intellectual Property.
- IP Vendors: List of commercial semiconductor IP vendors.
- ISI: Intersymbol interference
- Jitter: Deviation from perfect periodicity.
- Latchup: Short circuit due to creation of a low-impedance path between the power supply rails of a circuit.
- Layout: Physical representation of an integrated circuit.
- LEF: Standard Cell Library Exchange Format layout.
- Logical Effort: Technique used to normalize (and optimize) digital circuits speed paths.
- LVS: Layout Versus Schematic software checks that the layout is identical to the netlist.
- Mask Works: Copyright law dedicated to 2D and 3D integrated circuit "layouts".
- Mealy machine: A finite state machine whose outputs depend on current state and the current inputs.
- Metastability: Ability of a digital electronic system to persist for an unbounded time in an unstable equilibrium.
- MLS: Packaging and handling precautions for some semiconductors.
- Moore Machine: Finite state machine whose outputs depend only on its current state.
- Moore's Law: Observation by Moore that the number of transistors in an IC doubles approximately every two years.
- MOSFET: Metal oxide field effect transistor.

- MOSIS: Foundry service project offering MPWs and low volume manufacturing.
- MPW: Multi-project wafer service that integrates multiple designs on one reticle (aka "shuttle").
- MTBF: Mean time between failures.
- Multi-threshold CMOS: CMOS technology with multiple transistor types with different threshold voltages.
- Optical proximity correction: Technique used to compensate for semiconductor diffraction/process effects.
- Pass Transistor Logic: Logic that connects input to non-gate terminal of mosfet transistor.
- Physical design: Physical design flow ("layout").
- PDK: Process design kits consisting of a minimum set of files needed to design in a specific process.
- Power gating: Technique used to reduce leakage/standby power by shutting of the supply to the circuit.
- P&R: Automated Place and Route of a circuit using an EDA tool.
- PVT Corners: Represents the extreme process, voltage, temperature that could occur in a given semiconductor process.
- Radiation Hardening: Act of making devices resistant to damage caused by ionizing radiation.
- RTL: Design abstraction for digital circuit design.
- Setup time: Minimum time synchronous input should be ready before clock event.
- SEU: Change of state caused by one single ionizing particle (ions, electrons, photons...).
- Signoff: The final approval that the design is ready to be sent to foundry for manufacturing.
- SOC: System On Chip
- Spice: Open source analog electronic circuit simulator.
- STA: Method of computing the expected timing of a digital circuit without requiring full circuit simulation.
- Standard Cell Design: Design process relying on a fixed set of standard cells.
- Subthreshold Leakage: Current between source and drain in MOSFET when transistor is "off".
- Synchronous logic: Logic whose state is controlled by a synchronous clock.
- Synthesis: Translation of high level design description (e.g. Verilog) to a netlist format (e.g. standard cell gate level).
- SystemC: Set of C++ classes and macros for simulation. Commonly used for high level modeling and testing.
- Tape-out: Act of sending photomask chip database ("layout") to the manufacturer.
- TCL: Scripting language used by most of the leading EDA chip design tools.
- Transistor: A semiconductor device used to amplify/switch electronic signals.
- Verilog: The dominant hardware description language (HDL) for chip design.
- VLSI: Very large Integrated Circuit (somewhat outdated term, everything is VLSI today).
- Von Neumann architecture: Computer architecture in which instructions and data are stored in the same memory.
- UVM: Universal Verification Methodology

3.12.3 Manufacturing

- BEOL: Back end of line processing for connecting together devices using metal interconnects.
- Dicing: Act of cutting up wafer into individual dies.
- FinFet: Non planar, double-gate transistor.
- Photo-lithography: Process used in micro-fabrication to pattern parts of a thin film or the bulk of a substrate.
- Photomasks: Opaque plates with holes or transparencies that allow light to shine through in a defined pattern.
- Reticle: A set of photomasks used by a stepper to step and print patterns onto a silicon wafer.
- Semiconductor Fabrication: Process used to create the integrated circuits.
- Silicon: Element (Si), forms the basis of the electronic revolution.
- Silicon on insulator: Layered silicon–insulator–silicon with reduced parasitic capacitance.
- Stepper: Machine that passes light through reticle onto the silicon wafer being processed.
- TSV: Vertical electrical connection (via) passing completely through a silicon wafer or die.
- Wafer:: Thin slice of semiconductor material used in electronics for the fabrication of integrated circuits.
- Wafer thinning: Wafer thickness reduction to allow for stacking and high density packaging.

3.12.4 Packaging

- 3D IC's: The process of stacking integrated circuits and connecting them through TSVs.
- BGA: Ball grid array is a type of surface-mount packaging (a chip carrier) used for integrated circuits.
- BGA substrate: A miniaturized PCB that mates the silicon die to BGA pins.
- Bumping: Placing of bumps on wafer/dies in preparation for package assembly.
- DIMM: Dual in line memory module.
- Flip-chip: Method of bonding a silicon die to package using solder bumps.
- IC Assembly: Semiconductor die is encased in a supporting case "package".
- Interposer: Electrical interface used to spread a connection to a wider pitch.
- Heat sink: A passive heat exchanger.
- Heat pipe: Device for efficiently transferring heat between two solid interfaces .
- KGD: Known Good Die. Dies that have been completely tested at wafer probe.
- Leadframe: Metal structure inside a chip package that carry signals from the die to the outside.
- POP: Package on Package
- SIP: System In Package
- SMT: Technique whereby packaged chips are mounted directly onto the PCB surface.
- Through-hole: TPackage pins inserted in drilled holes and soldered on opposite side of the board.
- Wirebond: Method of bonding a silicon die to a package using wires.
- WSI: Wafer scale integration

3.12.5 Test

- Arbitrary Waveform Generator: Electronic instrument used to generate arbitrary signal waveforms.
- ATE: Automatic Test Equipment for testing integrated circuits.
- Burn-in: Process of screening parts for potential premature life time failures.
- DIB: Device Interface Board for interfacing DUT to ATE. Also called DUT board, probe card, load board, PIB.
- DMM: Electronic instrument for measuring voltage, current, and resistance.
- DUT: Device under test
- FIB: Focused ion beam
- JTAG: Industry standard for verifying and testing/debugging printed circuit boards after manufacturing.
- Logic Analyzer: Electronic instrument for capturing multiple digital signal from a system.
- MCM: Multi-chip Module
- Oscilloscope: Electronic instrument for tracking the change of an electrical signal over time.
- Probe Card: A direct interface between electronic test systems and a semiconductor wafer.
- SEM: Scanning electron microscope
- Shmoo Plot: An ASCII plot of a component response over a range of conditions.
- Spectrum Analyzer: Electronic instrument for measuring the power of the spectrum of an unknown signal.

3.13 Slurm setup

The SiliconCompiler project is capable of deferring individual job steps to hosts in a high-performance computing cluster. Currently, only the Slurm job scheduler is supported.

In order to use this functionality, you will need access to a Slurm cluster. Configuring a performant HPC cluster is outside the scope of this document, but this chapter will walk you through the process of configuring your local machine as a "cluster" consisting of a single host. It will also describe how to test the Slurm functionality, and how to add new hosts to a cluster.

This chapter will use Ubuntu 20.04 as a target OS, but the process should be very similar for other UNIX-like systems.

3.13.1 Initial Slurm Configuration

User and Group Configuration

Before installing the Slurm software, you should create users and groups named *slurm* and *munge*:

```
groupadd slurm
groupadd munge
useradd slurm -g slurm
useradd munge -g munge
```

It is **VERY IMPORTANT** that these users and groups have the same UIDs and GIDs on every host in the cluster. Our test "cluster" will only have a single host, but if you want to set up a real cluster later, you can create these groups and users with specific IDs:

```
groupadd slurm -g <slurm_gid>
groupadd munge -g <munge_gid>
useradd slurm -u <slurm_uid> -g <slurm_gid>
useradd munge -u <munge_uid> -g <munge_gid>
```

You can find existing user and group IDs in the */etc/passwd* file, or by using the *id* and *getent* commands:

```
id -u <username>
getent group <groupname>
```

Note that changing a user or group's ID after it already exists can cause issues, because filesystem permissions are usually not updated to match the new IDs.

Slurm Daemon Installation

The core Slurm scheduling logic is contained in two daemons:

- *slurmctld*: The "Slurm control daemon" runs on a "control node". This daemon manages the "compute nodes" in a cluster and delegates jobs to them.
- *slurmd*: The "Slurm daemon" runs on "compute nodes". This daemon listens for new commands from the "control node", and executes jobs which are sent to the host that it is running on.

Because we are setting up a test cluster on a single host, we will run both daemons on the same machine.

The slurm daemons, and a supporting *slurm-client* package, should be available in the package managers of common Linux distributions. In Ubuntu, you can install them with:

```
apt-get install slurmctld slurmd slurm-client
```

Slurm also relies on *munge* to encrypt communications between hosts. It should be installed as a dependency of the Slurm packages, but if not:

```
apt-get install munge
```

Munge will create a default key when it is installed at */etc/munge/munge.key*. This key must be identical on every host in the cluster, so if you set up a cluster with multiple hosts, you will need to copy this file onto every host in the cluster. You can re-generate a new munge key with:

```
sudo create-munge-key
```

Slurm Configuration Files

A minimal Slurm cluster requires three basic configuration files:

- */etc/slurm-llnl/cgroup.conf*: A "control group" configuration file. This describes which resources Slurm is allowed to access, and how Slurm should manage system resources.
- */etc/slurm-llnl/cgroup_allowed_devices.conf*: A list of filesystem paths to devices which Slurm should be able to access. The name and location of this file is arbitrary; it will be referenced in the *cgroup.conf* file.
- */etc/slurm-llnl/slurm.conf*: The core cluster configuration file. This describes which hosts should be included in a cluster, what those hosts' capabilities and roles are. It can also include config variables for various cluster behaviors.

Most of the contents of the example files presented here come from the Slurm project's documentation. You may need to extend or modify them if you create a more complex cluster with more hosts or system resources.

cgroup.conf

A minimal */etc/slurm-llnl/cgroup.conf* file is fairly brief:

```
CgroupMountpoint="/sys/fs/cgroup"
CgroupAutomount=yes
CgroupReleaseAgentDir="/etc/slurm-llnl/cgroup"
AllowedDevicesFile="/etc/slurm-llnl/cgroup_allowed_devices.conf"
ConstrainCores=no
TaskAffinity=no
ConstrainRAMSpace=yes
ConstrainSwapSpace=no
ConstrainDevices=no
AllowedRamSpace=100
AllowedSwapSpace=0
MaxRAMPercent=100
MaxSwapPercent=100
MinRAMSpace=30
```

If your Linux installation places its control group devices in a different directory from */sys/fs/cgroup*, you may need to modify that parameter.

The *AllowedDevicesFile* parameter refers to the *cgroup_allowed_devices.conf* file which we will create in the next section.

You can read more about *cgroup.conf* parameters in the Slurm documentation.

cgroup_allowed_devices.conf

The list of cgroup devices which Slurm should be allowed to access is typically quite short. This file path should match the value of *AllowedDevicesFile* in your *etc/slurm-llnl/cgroup.conf* file:

```
/dev/null
/dev/urandom
/dev/zero
/dev/sda*
/dev/cpu/*/*
/dev/pts/*
```

If your system has other core resources that the cluster may need to access, you can add those device paths here.

slurm.conf

The most reliable way of setting up a comprehensive *slurm.conf* file is by using Slurm's "configurator" web tool, but the large number of fields can be confusing if you are new to Slurm.

If you want to use the "configurator" tool, it is available online on the Slurm website in a "normal" and "easy" version. It is a simple HTML page, though, so you can also run it locally by installing the Slurm documentation package:

```
apt-get install slurm-wlm-doc
```

The "configurator" pages will then be installed on your local machine, and you will be able to open them in a web browser. The default file paths in Ubuntu are:

```
/usr/share/doc/slurm-wlm/html/configurator.html
/usr/share/doc/slurm-wlm/html/configurator.easy.html
```

For our minimal single-host test cluster, you can skip the "configurator" and use this as a template:

```
# slurm.conf file.
# Put this file on all nodes of your cluster.
# See the slurm.conf man page for more information.
SlurmctldHost=<your_hostname>

MpiDefault=none
ProctrackType=proctrack/cgroup
ReturnToService=1
SlurmctldPidFile=/run/slurmctld.pid
SlurmdPidFile=/run/slurmd.pid
SlurmdSpoolDir=/var/spool/slurmd
SlurmUser=slurm
StateSaveLocation=/var/spool/slurm
SwitchType=switch/none
TaskPlugin=task/cgroup
#SrunPortRange=<start_port>-<end_port>

# SCHEDULING
SchedulerType=sched/backfill
SelectType=select/linear

# LOGGING AND ACCOUNTING
AccountingStorageType=accounting_storage/none
ClusterName=<cluster_name>

JobAcctGatherType=jobacct_gather/none
SlurmctldLogFile=/var/log/slurm-llnl/log.log
SlurmdLogFile=/var/log/slurm-llnl/dlog.log

# COMPUTE NODES
NodeName=<your_hostname> NodeAddr=<your_dns> Features=SHARED <host_info>
PartitionName=debug Nodes=<your_hostname> Default=YES MaxTime=INFINITE State=U
```

Note the *Features=SHARED* parameter in the compute node definition. Slurm allows you to define attributes to restrict which compute nodes are allowed to run which jobs. The SiliconCompiler project uses *SHARED* as a catch-all feature when more complex job delegation is not required.

The <*your_hostname*> values should be set to the output of the *hostname* command, and the <*your_dns*> value should be set to a value which DNS services will resolve to your host. Cloud providers will typically provide a public DNS for virtual hosts, but you can use *localhost* for a test cluster on your local machine.

The <*host_info*> values define the capabilities of a compute node; number of CPUs, available RAM, etc. You can get these values for a host by running *slurmd -C*, and copying the output from *CPUs=[...]* through *RealMemory=[...]*. You may want to reduce the *RealMemory* value a bit, because Slurm will take nodes out of service if their available RAM falls below that threshold.

The *SrunPortRange* value is commented out here, but you can use it to limit the range of ports which Slurm is allowed to use for "phoning home" from compute nodes to the control node. If you want to avoid using reserved ports or set up a firewall rule with more restrictive port ranges than 0-65535, you can set your desired port range with this parameter.

The *ClusterName* parameter is arbitrary. I like to name clusters after roads, but things like mountains, forests, mythical

figures, etc. are also good choices. Slurm's documentation recommends using lowercase characters in the name. It looks like this parameter is mostly used for accounting in Slurm's optional database extension, so it should not be too important in a minimal test cluster.

Important note: Every host in your cluster should use an identical *slurm.conf* file. If you eventually set up a cluster with shared networked storage, you can easily propagate changes to *slurm.conf* by placing the file in shared storage, and making each host's */etc/slurm-llnl/slurm.conf* file a symbolic link to <*shared_storage*>*/slurm.conf*.

Slurm Startup

To start your slurm cluster, all you need to do is restart the *slurmctld* and *slurmd* daemons. These daemons should be restarted on all hosts in the cluster whenever you make changes to configuration files like *slurm.conf*:

```
sudo service restart slurmctld
sudo service restart slurmd
```

Once the daemons are running with the correct config files loaded, you should be able to issue commands to the cluster using *srun*:

```
srun hostname
```

Running SiliconCompiler on a Cluster

To run a SiliconCompiler job on your cluster, all you need to do is set the *['option', 'scheduler', 'name']* schema parameter to *slurm*. If you are using the Python API:

```
chip.set('option', 'scheduler', 'name', 'slurm')
```

If you are running a job from the command-line, simply add *-scheduler slurm* to the command.

There are a few restrictions to be aware of if you decide to set up a more complex cluster:

- Clustered jobs must be run from a host which is acting as the "control node" for a Slurm cluster.
- The build directory must be placed in a location which is accessible to all hosts in the cluster. If you have multiple hosts in your cluster, they will need to share a networked storage drive using a protocol such as NFS.

3.14 Licenses

3.14.1 Software License

Original SiliconCompiler software is licensed under an Apache 2.0 license.

3.14.2 Documentation License

This document is released under a Creative Commons Attribution 4.0 International License.

3.14.3 Dependency Licenses

SC relies on the following Python dependencies, which are licensed as shown. Note this table includes dependencies of the dependencies directly declared by SiliconCompiler.

Name	Version	License
Deprecated	1.2.14	MIT License
GitPython	3.1.43	BSD License
Jinja2	3.1.3	BSD License
MarkupSafe	2.1.5	BSD License
PyGithub	2.3.0	GNU Library or Lesser General Public License (LGPL)
PyJWT	2.8.0	MIT License
PyNaCl	1.5.0	Apache License 2.0
PyYAML	6.0.1	MIT License
aiohttp	3.9.5	Apache Software License
aiosignal	1.3.1	Apache Software License
altair	5.3.0	BSD License
attrs	23.2.0	MIT License
blinker	1.8.1	MIT License
cachetools	5.3.3	MIT License
certifi	2024.2.2	Mozilla Public License 2.0 (MPL 2.0)
cffi	1.16.0	MIT License
charset-normalizer	3.3.2	MIT License
click	8.1.7	BSD License
cryptography	42.0.5	Apache Software License; BSD License
defusedxml	0.7.1	Python Software Foundation License
distro	1.9.0	Apache Software License
fasteners	0.19	Apache Software License
fastjsonschema	2.19.1	BSD License
frozenlist	1.4.1	Apache Software License
gitdb	4.0.11	BSD License
graphviz	0.20.3	MIT License
idna	3.7	BSD License
isodate	0.6.1	BSD License
jsonschema	4.22.0	MIT License
jsonschema-specifications	2023.12.1	MIT License
lambdapdk	0.1.19	Apache License
markdown-it-py	3.0.0	MIT License
mdurl	0.1.2	MIT License
multidict	6.0.5	Apache Software License
netifaces	0.11.0	MIT License
networkx	3.3	BSD License
numpy	1.26.4	BSD License
packaging	23.2	Apache Software License; BSD License
pandas	2.2.2	BSD License
pillow	10.3.0	Historical Permission Notice and Disclaimer (HPND)
protobuf	4.25.3	3-Clause BSD License
psutil	5.9.8	BSD License

continues on next page

Table 948 – continued from previous page

pyarrow	16.0.0	Apache Software License
pycparser	2.22	BSD License
pydeck	0.9.0	Apache License 2.0
pyparsing	3.1.2	MIT License
python-dateutil	2.9.0.post0	Apache Software License; BSD License
pytz	2024.1	MIT License
rdflib	7.0.0	BSD License
referencing	0.35.0	MIT License
requests	2.31.0	Apache Software License
rich	13.7.1	MIT License
rpds-py	0.18.0	MIT License
sc-leflib	0.2.0	Apache License 2.0
siliconcompiler	0.21.11	Apache License 2.0
six	1.16.0	MIT License
smmap	5.0.1	BSD License
streamlit	1.33.0	Apache Software License
streamlit-agraph	0.0.45	UNKNOWN
streamlit-javascript	0.1.5	MIT License
streamlit-tree-select	0.0.5	UNKNOWN
tenacity	8.2.3	Apache Software License
toml	0.10.2	MIT License
toolz	0.12.1	BSD License
tornado	6.4	Apache Software License
tzdata	2024.1	Apache Software License
urllib3	2.2.1	MIT License
watchdog	4.0.0	Apache Software License
wrapt	1.16.0	BSD License
yarl	1.9.4	Apache Software License

The precompiled SiliconCompiler wheels distributed on PyPI include the following bundled dependencies:

Name	License
Si2 LEF parser (distributed by OpenROAD)	Apache Software License
Surelog	Apache Software License

3.15 Revision History and Change Log

The changes in each SiliconCompiler release version are described below. Commit version shown in (). Where applicable, the contributors that suggested a given feature are shown in [].

3.15.1 SiliconCompiler 0.21.11 (2024-04-26)

Minor:

- Added helper functions to tcl to make accessing schema information easier.
- Added helper functions to allow tools to have consistent behavior around picking from pdk/library/option variables during setup.
- Ensure the remote client downloads the final results from the remote to get any logs and error messages.
- Tools:
  - openroad: fixed default routing via settings.
  - klayout: made hide_layer variable more consistent with klayouts layer information, and added support for .gz file handling in show and screenshotting tasks.

3.15.2 SiliconCompiler 0.21.10 (2024-04-22)

Minor:

- Switch to use templates for generating tool files instead of python print statements to make it easier to maintain.
- Added logic depth to schema metrics.
- Tools:
  - yosys: added support for handling hierarchy selection with globbing, added better handling of library merging to ensure liberty templates are properly copied in.
  - openroad: added support for extracting logic depth of the design and added irdrop extraction to tool driver.

3.15.3 SiliconCompiler 0.21.9 (2024-04-12)

Minor:

- General cleanup of documentation building.
- Tools:
  - yosys: added support for specifying blackbox models via [‘tool’, ‘yosys’, ‘task’, ‘syn_asic’, ‘var’, ‘blackbox_modules’]

3.15.4 SiliconCompiler 0.21.8 (2024-04-11)

Minor:

- Fixed packaging extraction when downloading from github.
- Fixed remote run file retrieval to avoid errors when extracting and moving files.
- Tools:
  - vpr: added support for show and screenshoting tasks.

3.15.5 SiliconCompiler 0.21.7 (2024-04-02)

Minor:

- Update remote code to honor constraints transmitted by the remote runner.
- Add fall back to package locking when file system locking is not available.
- Fixed handling of github artifact packages.
- Added build scripts for ubuntu22.
- Tools:
  - vpr: add reporting images to tool driver, these can be disabled via ['tool', 'vpr', 'task', 'place' or 'route', 'var', 'enable_images'] = 'false'.

3.15.6 SiliconCompiler 0.21.6 (2024-03-28)

Minor:

- Fixed handling of files with codec errors.
- Update package lock file handling to avoid race conditions in parallel flows.
- Tools:
  - yosys: minor code cleanup in FPGA flow.
  - vpr: add support for clock routing.
  - surelog: added wrapper comments in output file for parsing to indicate where segments of code came from.

3.15.7 SiliconCompiler 0.21.5 (2024-03-21)

Minor:

- Added a github data source to handle data from private repositories.
- Tools:
  - vpr: added pin metrics collection for FPGA flows.

3.15.8 SiliconCompiler 0.21.4 (2024-03-15)

Minor:

- Added keys to schema to track FPGA resources.
- Tools:
  - yosys: add metrics collection for FPGA flows.
  - vpr: added metrics collection for FPGA flows.

3.15.9 SiliconCompiler 0.21.3 (2024-03-13)

Minor:

- Tools:
  - openroad: fixed handling of pin constraints
  - yosys: updated to take advantange of native dont_use interfaces

3.15.10 SiliconCompiler 0.21.2 (2024-03-08)

Minor:

- Fix importing of package data from non libraries
- Fix unlocking behavior in schema keys

3.15.11 SiliconCompiler 0.21.1 (2024-03-07)

Minor:

- Fix handling of python module packages on older versions of python.

3.15.12 SiliconCompiler 0.21.0 (2024-03-07)

Major:

- Added support for handling soft libraries in the schema via ['option', 'library'] and updated all frontend task drivers to support these.

Minor:

- Ensure that libraries, which import a library, are handled correctly.

3.15.13 SiliconCompiler 0.20.3 (2024-03-06)

Minor:

- Added testing to ensure command line arguments are all checked for functionality and follow the schema pattern.
- Fixed python package lookup when the installed package does not match the name of the distribution.
- Added print controls to command line applications to allow custom arguments to be displayed along with schema values via the key *sc_print:*.
- Updated the conda environment and added testing to ensure functionality.

3.15.14 SiliconCompiler 0.20.2 (2024-02-20)

Minor:

- Fixed some issued identified by tclint.
- Added helper function to better detect locally installed and editable python packages for package registration.

3.15.15 SiliconCompiler 0.20.1 (2024-02-12)

Major:

- Moved from hosting third party pdks in SiliconCompiler to fully using lamdapdk.
- Added a demonstration target for gf180.

Minor:

- Tools:
  - vpr: added support for pin constraints.
  - openroad: added estimated routing congestion to the task image writing and updated fmax metric collection to use OpenROAD directly.

3.15.16 SiliconCompiler 0.20.0 (2024-02-01)

Major:

- Update file path hashing to include package information.

Minor:

- Tools:
  - yosys: update FPGA synthesis to better map LUTs and memories.
  - openroad: added option to control CTS with obstruction awareness via via ['tool', 'openroad', 'task', 'cts', 'var', 'cts_obstruction_aware'].
  - klayout: added operation to be able to rename cells in layout via ['tool', 'klayout', 'task', 'operations', 'var', 'operations', 'rename_cell'].

3.15.17 SiliconCompiler 0.19.1 (2024-01-24)

Major:

- Tools:
  - yosys: updated implementation for FPGA flow to better support flip-flop mapping and hard macro extraction and mapping.

Minor:

- Package downloading to ensure GitHub release artifacts can be downloaded from private repositories.

3.15.18 SiliconCompiler 0.19.0 (2024-01-05)

Major:

- Expanded and clarified the datasheet category of the schema.

3.15.19 SiliconCompiler 0.18.2 (2023-12-18)

Minor:

- Fixed resolving paths in package sources when paths contain environmental variables.

3.15.20 SiliconCompiler 0.18.1 (2023-12-13)

Minor:

- Fixed importing of libraries to collect package sources during import.
- Tools:
  - klayout: fixed loading of schema in the presence of other modules named schema.

3.15.21 SiliconCompiler 0.18.0 (2023-12-04)

Major:

- Added ['option', 'cache'] keypath to control the location of the cached data.
- Added support for Python 3.12

3.15.22 SiliconCompiler 0.17.0 (2023-11-16)

Major:

- Added ['package', 'source', '<name>', 'path'] and ['package', 'source', '<name>', 'ref'] to support directly downloading of required data for a design, pdk, library, etc.
- Removed support for $SCPATH and ['option', 'scpath'] in favor of using package sources.

Minor:

- Tools:
  - yosys: fixed handling of blackboxes during verilog reading, improved the sdc parsing to better estimate the clock for yosys-abc
  - OpenROAD: fixed PDN file handling to only read files once to avoid errors from redefining the same grids.

3.15.23 SiliconCompiler 0.16.3 (2023-11-02)

Minor:

- Removed unused ['flowgraph', '<graph>', '<step>', '<index>', 'valid'] field from schema.
- Tools:
  - klayout: fixed DEF to GDS generation by ensuring the correct units are used during DEF read in and fixes the stream writing to honor OASIS if requested.
  - OpenROAD: fixed handling of unidirectional layers in routing task and added tasks to support generating OpenRCX parasitic extraction decks.

3.15.24 SiliconCompiler 0.16.2 (2023-10-13)

Minor:

- Added line numbers to the error and warning log files to aid in tracking down the message in the main log.

3.15.25 SiliconCompiler 0.16.1 (2023-10-11)

Minor:

- Fixed incorrect settings when submitting remote jobs.

3.15.26 SiliconCompiler 0.16.0 (2023-10-09)

Major:

- Deprecated support for Python 3.6 and 3.7.
- Added ['option', 'from'], ['option', 'to'], and ['option', 'prune'] to better control the execution of the flowgraph.
- Removed leflib from SilconCompiler and use the standalone implementation provided via *pip install sc-leflib*.

Minor:

- Moved the built in server to use the Schema class for handling its settings.
- Fixed handling of Windows paths when submitting jobs from a Windows machine to a linux runner.
- Corrected the return behavior from the schema *.get()* to ensure lists are copied instead of being returned by reference.
- Fixed behavior where *sc-show* would require both step and index to be specified and allowed for directories to be specified as the show target.
- Tools:
  - yosys: Updated to support new ABC interface for passing along dont_use cells.
  - verilator: added support for assertions via ['tool', 'verilator', 'task', 'compile'/'lint', 'var', 'enable_assert']

3.15.27 SiliconCompiler 0.15.4 (2023-09-25)

Minor:

- Added *-design* to *sc-dashboard* and ensured common behavior between *sc-show* and *sc-dashboard*.
- Disabled sorting of keys in writing of the json schema to preserve the order in the original dictionary.
- Added an alias (*siliconcompiler*) to the commandline application *sc* to allow Windows users to be better able to use SiliconCompiler.
- Tools:
  - surelog: Ensure the bundled version is built using a static zlib library for Windows distribution.
  - OpenROAD: Added more controls over the heatmap image generation and to only write heatmaps with data available.

3.15.28 SiliconCompiler 0.15.3 (2023-09-21)

Minor:

- Fixes to *sc-show* for viewing remote files to ensure it does not attempt to use the file paths from the remote, but instead the local file paths.
- Tools:
  - klayout: Use build technology files and layer display files in show and ensure these are placed into the outputs directory when exporting to a GDS.

3.15.29 SiliconCompiler 0.15.2 (2023-09-18)

Minor:

- Fixes minor bug in the dashboard which prevented the launching the dashboard.
- Ensures all tool output goes through the logger.
- Improved implementation of *.valid()* to provide faster checking.
- Tools:
  - OpenROAD: fixed bugs in instance creation and pin access function calls. Implemented support for multiple library corners assigned to a single scenario in timing constraints.
  - yosys: Implemented support for multiple library corners assigned to a single scenario in timing constraints.

3.15.30 SiliconCompiler 0.15.1 (2023-09-08)

Major:

- Merged *sc-configure* into *sc-remote* to unify the remote interface

Minor:

- Fixed missing information in documentation build for flowgraphs.
- Removed call to *delete_job* on remote jobs, relying on remote to handle cleanup instead.
- Refactored core run functions for better code management.

3.15.31 SiliconCompiler 0.15.0 (2023-08-31)

Major:

- Updated schema to remove unused ['option', 'skipstep'] key and add record to store the remote job id in ['record', 'remoteid'].

Minor:

- Added additional error checking and file cleanup to remote run to ensure empty files are not left behind.
- Fixed handling of setting list of tuples in the schema to corrected parse the values.
- Tools:
  - OpenROAD - disabled timing optimizations by default, added support for generating report images in the screenshot task via ['tool', 'openroad', 'task', 'screenshot', 'var', 'include_report_images'].
  - Chisel - added support for handling *build.sbt* via the input fileset ['input', 'config', 'chisel'].

3.15.32 SiliconCompiler 0.14.0 (2023-08-21)

Major:

- Reworked FPGA to allow for better handling of FPGA parameters in the schema.
- Added missing mechanical parameters to the schema and updating the signal interface naming for clarity.
- Stop execution upon node failures, instead of continuing with repeated failures.

Minor:

- Ensure ['option', 'nodisplay'] is set when the environment does not support graphics.
- Added error checking for ['option', 'steplist'] to match current flowgraph.
- Removed physyn step from asicflow.
- Fixed issue when running remote and the submitted files are not updated.
- Tools:
  - OpenROAD - update default value for [..., 'var', 'grt_macro_extension'] to '0' to allow for better routing.
  - verilator - reworked options selection to allow for better control of user selected tool options.
  - yosys - added support for blackboxes in libraries via ['library', <lib>, 'output', 'blackbox', 'verilog'].

3.15.33 SiliconCompiler 0.13.2 (2023-08-10)

Major:

- Added graphs view to dashboard for comparing metrics across runs.
- Added `sc-remote` app for checking server status and interacting with running remote jobs (replaces `sc-ping`).

Minor:

- Added checkbox to dashboard for enabling "raw" view of manifest.
- Tools:
  - OpenROAD - made repair_design more verbose (requires updated version), added option to generate design images at end of task via ['tool', 'openroad', 'task', <task>, 'var', 'ord_enable_images'], tweaked pin placement behavior, added SDF file output to export task.

  - surelog - fixed driver to sanitize escape characters in Verilog output, added option to disable write cache via ['tool', 'surelog', 'task', 'parse', 'var', 'disable_write_cache'].
  - klayout - added option to allow missing cells in stream files via ['library', <lib>, 'option', 'var', 'klayout_allow_missing_cell'], added option to set DB units via ['pdk', <pdk>, 'var', 'klayout', 'units', <stackup>].

3.15.34 SiliconCompiler 0.13.1 (2023-07-21)

Minor:

- Improved remote run reliability, including graceful time-outs when server not responding and thread-safety fixes.
- Added ['option', 'libext'] support for Surelog, Verilator, and Icarus.
- Removed dashboard support for Python 3.6.
- Made aesthetic tweaks to dashboard.
- Fixed bundled Surelog on pre-macOS 12.
- Tools:
  - yosys - fixed synthesis strategies.
  - verilator - added multithreading, added FST trace format support.
  - OpenROAD - bumped minimum version to better support IR drop analysis, added additional controls for abstract LEF generation, updated scripts to improve ORFS correspondence, fixed to use correct layers for parasitic estimation.

3.15.35 SiliconCompiler 0.13.0 (2023-07-07)

Major:

- Added dashboard to SiliconCompiler to allow better inspection of the run information and added *sc-dashboard* app to open and display the dashboard.

Minor:

- Added ['input', 'constraint', 'upf'] as a recognized format
- Tools:
  - surelog - added support for lowmem option via ['tool', 'surelog', 'task', 'parse', 'var', 'enable_lowmem']

3.15.36 SiliconCompiler 0.12.3 (2023-06-23)

Major:

- Added new flow `screenshotflow` to enable generating high quality stream images via klayout and imagemagick/montage.
- Added new tool `execute` to enable executing the output of a previous task, such as in the case of compiling a binary in one step and executing it in the next.

Minor:

- `sc-show` fixed error handling when attempting to how a file without a manifest.
- Added support for *pathlib.Path* objects when setting file and dir type parameters in the manifest.
- Tools:

- yosys - fixed marking liberty files dont_use and ensure each library is merged together for ABC. Updated driver to use lower case true/false for ['tool', 'openroad', 'task', '*', 'var'] to be consistent with other tools.
- verilator - added support for pins-bv via ['tool', 'verilator', 'task', 'compile', 'var', 'pins_bv'] and compile modes via ['tool', 'verilator', 'task', 'compile', 'var', 'mode'].

3.15.37 SiliconCompiler 0.12.2 (2023-06-14)

Major:

- Added additional arguments to `sc-show` to provide access to specific steps and indices in the run.

Minor:

- Tools:
  - OpenROAD - added generation of separate timing and power reports in the reports/ directory to provide better insights into the design, added additional parameters to timing repair to enable minimizing total negative slack ['tool', 'openroad', 'task', 'place', 'var', 'rsz_repair_tns'], and added support for RTL-MP for macro placement.
  - yosys - added initial support for hierarchical synthesis via ['tool', 'yosys', 'task', 'syn_asic', 'var', 'hier_threshold'], this is disabled by default while it is still in development.

3.15.38 SiliconCompiler 0.12.1 (2023-06-07)

Major:

- Fixed writing of manifest to preserve values that were previously removed incorrectly.
- Updated recording of ['record', ...] to track tool versions, tool options, task start and end times, and SiliconCompiler version, while preserving control of sensitive records like ['record', 'ipaddr'] with ['option', 'track'].

Minor:

- Fixed handling of `sc-issue` to avoid clobbering the user set options and only bundle the required files for a testcase to minimize the size of the file.
- Added error checking for `create_cmdline` to check for invalid arguments.
- Tools:
  - OpenROAD - added support for disallowing one site gaps in detail placement ['tool', 'openroad', 'task', 'place', 'var', 'dpl_disallow_one_site']. Added support for ['option', 'warningoff']
  - surelog - Added support for ['option', 'warningoff']
  - Verilator - Added support for ['option', 'warningoff']

3.15.39 SiliconCompiler 0.12.0 (2023-05-24)

Major:

- Added/updated parameters in ['datasheet'] section of the schema to allow for better capturing of design datasheet.
- Updated Verilator tool driver to support CFLAGS/LDFLAGS and fix linting task.
- Added operations task to klayout to allow for unit operations on GDSs like merging, adding outlines, rotating, etc.
- Added options to `archive()` to support archiving multiple jobs and filtering files to include.

Minor:

- Added enforcement of ['option', 'mode'] to ensure it is set for better manifest checking.
- Added per-pin voltage constraints for better support of multiple power/voltage domains, ['constraint', 'timing', scenario, 'voltage', pin].
- Fixed tool drivers to ensure proper use of `find_files()` is not done in `setup()`.
- Added check for permissions error while collecting child process memory statistics.

3.15.40 SiliconCompiler 0.11.2 (2023-05-15)

Major:

- Updated `sc-issue` to generate self-contained testcases to allow for better sharing of testcases.
- Updated klayout tool driver to support map-file option for DEF-GDS export step, and remove need for hard coded options in `.lyt` file in favor of getting values from the schema.

Minor:

- Updated loading order to target in commandline interface to ensure schema parameers are set before loading target.
- Error checking for flowgraphs with duplicated edges.
- Added `-ext` to `sc-show` command to control what file is opened when multiple files are available.
- Tools:
  - OpenROAD - added flags to control antenna repairs: ['tool', 'openroad', 'task', 'route', 'var', 'ant_check'] and ['tool', 'openroad', 'task', 'route', 'var', 'ant_repair']; added clock buffer selection option from the library with ['library', lib, 'option', 'openroad_cts_clock_buffer']

3.15.41 SiliconCompiler 0.11.1 (2023-05-03)

Major:

- Revamped documentation for better readability and navigability.

Minor:

- Fixed handling when loading schemas from JSON to ensure values are normalized correctly.
- When a tool fails, the last 10 lines of the log is printed when ['option', 'quiet'] is set to aid in debugging.
- Updated server/client to use python tarfile module instead of spawning subprocesses.
- Implemented python linting.
- Tools:
  - OpenROAD - implemented snapping to sites or manufacturing grid for component placement via ['tool', 'openroad', 'task', 'floorplan', 'var', 'ifp_snap_strategy'] and added ability to select SDC IO buffer for automatic constraint generation via ['tool', 'openroad', 'task', 'floorplan', 'var', 'sdc_buffer'].
  - yosys - fixed invalid keypath access when logiclib is incorrectly specified.

3.15.42 SiliconCompiler 0.11.0 (2023-04-17)

Major:

- `sc-issue` added as command line application to support sharing runnable testcases.
- Removed the requirement that the initial task of the flowgraph be called 'import' and the final task be called 'export'.
- Fully implemented `.node()` to take in a task module, simplifying the construction of a flowgraph and removing its dependence on ['option', 'scpath'].

Minor:

- ['tool', tool, version, 'sbom'] added to be able to track a tools SBOM.
- `.hash_files()` updated to honor the 'hash' field for the file parameters.
- `.calc_yield()`, `.calc_area()`, `.calc_dpw()` updated to new schema parameters.

3.15.43 SiliconCompiler 0.10.2 (2023-04-04)

Major:

- Support for Python 3.11
- Building arm64 wheels for surelog

Minor:

- Tools:
  - general - deployed docker based CI with automatic tool building and testing
  - klayout - support for OAS stream files via ['tool', 'klayout', 'task', 'export', 'var', 'stream'] = 'oas', better detection of missing layout cells
  - yosys - support for controlling buffer insertion via ['tool', 'yosys', 'task', 'syn_asic', 'var', 'add_buffers'] = 'True'/'False'
  - openroad - correct handling of INF timing in metrics
- Improve memory recording to account for child processes
- General documentation cleanup

3.15.44 SiliconCompiler 0.10.1 (2023-03-11)

Major:

- Adding `._record_metric()` for tool drivers to use when recording metrics to ensure they honor the schema units and record the source of the metric at the same time.

Minor:

- Improved error handling and messaging for remote jobs.
- Fixed HTML summary report and PNG summary image not getting rebuilt when calling `.summary()`
- Updated `.summary()` table to display units and format numbers accordingly.

3.15.45 SiliconCompiler 0.10.0 (2023-03-08)

Major:

- Schema overhauled (see schema documentation for details):
  - Added ability to override certain schema parameters on a per-step/index basis.
  - Added step and index to schema access methods.
  - Expanded constraints category to include component and pin placement.
  - Cleaned up parameters which were duplicated in other categories.
  - Implemented ['input', ...] and ['output', ...] filesets along with `.input()` and `.output()` helper functions
- Added tasks to tools prevent step name from getting used for task identification.
- Implemented `.use()` in favor of `.load_pdk()`, `.load_flow()`, etc. to take in Python modules instead of strings.
- Changed libraries, flows, checklists, and pdks to explicitly return a particular class object.
- Generate a summary PNG.
- Removed Floorplanning API.
- Added support for custom macros and scripts in the remote workflow.

Minor:

- Updated OpenROAD scripts to support a hierarchical flow.
- Updated Yosys scripts to better support hierarchy.
- Improved auto documentation generation.
- Updated pdk and library settings to provide additional corners where available.
- Updated documentation, including Installation, Quickstart Guide and Tutorials.
- Added Fmax as a first-order metric to the Schema

3.15.46 SiliconCompiler 0.9.6 (2022-10-03)

Major:

- Fixed bug that causes tool setup information to be lost when running a flow in multiple chunks using a steplist.

Minor:

- Fixed old schema references in Yosys synthesis strategy scripts.
- Updated error message for missing file requirements.
- Updated OpenROAD scripts to handle multiple LEF files.
- Updated KLayout driver to use batch mode flag and capture more warnings.
- Updated Verilator driver to implement ['option', 'trace], ['option', 'warningoff'], and provide passthroughs for CFLAGS and LDFLAGS.
- Removed support for 'extraopts' passthrough in Verilator driver.
- Updated version of Surelog bundled with wheels distribution.

3.15.47 SiliconCompiler 0.9.5 (2022-09-12)

Minor:

- Schema: Added ['tool', <tool>, 'prescript'/'postscript', <step>, <index>] to support user-supplied pre- and post-scripts for script-based tools.
- Schema: Added ['tool', <tool>, 'file', <step>, <index>] passthrough parameter.
- Added runtime logic to terminate tools that do not exit on their own after a job is interrupted with ctrl-c.
- Fixed KLayout show bugs.
- Fixed issue building SC in editable mode using newer versions of Pip/setuptools.

3.15.48 SiliconCompiler 0.9.4 (2022-08-25)

Major:

- Changed `run()` logic to not reset metrics to zero.
  - `summary()` will only display metrics that have been explicitly set.

Minor:

- Schema: Changed ['constraint', <scenario>, 'libcorner'] from scalar to list.
- Added support for –latches option in GHDL driver.
- Added `:keypath:` directive to distributed Sphinx extensions.
- Added reports and final manifest to `archive()` outputs.
- Fixed bug where `job` argument to `find_files()` was not handled properly.
- Fixed pin sizes and PDN vias in Caravel wrapper example.
- Updated flow scripts to support newer version of OpenROAD.
- Updated version of Surelog bundled with wheels distribution.

3.15.49 SiliconCompiler 0.9.3 (2022-08-01)

Major:

- Added basic editing functionality for signoff checklists in HTML report.
- Changed file collection behavior:
  - For local runs, inputs are not copied into import/ at all.
  - For remote runs, inputs are copied into import/0/inputs/ only, not outputs/.
- Implemented ['option', 'entrypoint'], allowing users to specify an alternative top-level.
- Implemented support for "pure Python" tools.
  - A `run()` method inside a tool setup file will be run instead of an executable.

Minor:

- Changed `run()` behavior to read metrics from all leaf tasks.
- Fixed implementation of ['option', 'jobincr'].
- Fixed bug causing exception on `summary()` for machines with a default encoding other than UTF-8.

- Fixed logfile reading logic to gracefully handle invalid characters.
- Improved error messages for some common issues.

3.15.50 SiliconCompiler 0.9.2 (2022-07-08)

Major:

- Schema: Added ['option', 'flowcontinue'] to control whether flow continues when a tool reports errors.
  - This used to be controlled by ['tool', <tool>, 'continue'], but this parameter is meant to feed directly into tools (rather than controlling the SC runtime).
- Schema: Added ['option', 'continue'] parameter to control whether errors in the Python API are fatal.
  - The default value makes errors fatal, setting this parameter to True reverts to the old behavior.
- Added VPR-based FPGA bitstream generation flow.
- Added logic to set errors and warnings metrics based on ['tool', <tool>, 'regex', ...] matches. This reduces tool driver boilerplate and makes the metrics consistent with the generated regex match files.

Minor:

- Changed default technology target for `sc` app.
- Changed KLayout show script to always use a dark background.
- Changed `check_manifest()` to allow tool tasks to have multiple inputs (behaving as if they were merged with a "join" builtin).
- Changed `check_manifest()` to return True on success rather than 0 (the previous behavior didn't match the documentation).
- Changed Yosys and OpenROAD tool drivers to make them easier to use in flows with alternate step names.
- Changed GHDL tool driver to allow additional CLI options via ['tool', <tool>, 'var', ..., 'extraopts'].
- Removed return codes from `post_process()`.

3.15.51 SiliconCompiler 0.9.1 (2022-06-21)

Major:

- Added input filetype inference based on file extension (restores functionality lost in 0.9.0).
- Added manifest tree viewer to HTML report.
- Added simulator exe compilation support to Verilator.
- Improved TCL manifest generation:
  - Fixed escaping of special characters and whitespace.
  - Fixed insertion of "$env" in filepaths.
  - Changed tuple printing to be TCL list instead of tuple-like string.

Minor:

- Schema: Added tool CLI arguments to ['record', ...] schema.
- Changed create_cmdline() switchlist parameter to accept switch names as specified on command line.
- Changed setup module docs generator to be packaged with SC.

- Changed HTML report to be self-contained.
- Fixed CSV manifest generation.

3.15.52 SiliconCompiler 0.9.0 (2022-05-19)

Major:

- Schema: Reorganized entire schema! Changes summarized below:
  - Cleaned and consolidated top-level organization, most parameters are now nested.
  - Moved build configuration options underneath ['option', ...].
  - Added ['output', ...] to store pointers to flow outputs.
  - Added ['model', ...] to store pointers to design abstractions (timing libraries, layouts, etc).
  - Added ['datasheet', ...] to store information about design's interface.
  - Added ['unit', ...] to store user driven SI units specification (temp,voltage, etc)
  - Renamed ['eda', ...] to ['tool', ...]
  - Renamed ['mcmm', ...] to ['constraint', ...]
  - Replaced ['source'], ['constraint'], and ['read', ...] with more flexible ['input', <filetype>] to supply input files.
  - Added support for storing multiple PDKs in schema and selecting which one to use for run (analogous to flows).
  - Change ['flowgraph'] to support modular flow composition
  - Added support for package management ['depgrah'].
  - Added checklist support.
  - Removed special ['library', ...] keypaths. All libraries are now created as Chip objects, and have their full config imported into a parent chip's schema.
- Added 'sup' packaging utility
- Added ability to configure stdout and stderr redirection on a per-tool basis (thanks to @suppamax for implementing).
- Added flexible tool version checking based on PEP-440 standard, now enabled by default.
- Added 'clean' feature for cleaning up intermediate tool outputs.
- Added 'resume' feature for restarting failed flows (for debugging).
- Added automatic capture of peak memory usage (adds dependency on psutil).
- Changed ['design'] to be a required parameter for instantiating a Chip.
- Changed error behavior to consistently raise exceptions rather than exit.
- Removed tool script copy feature, so now all EDA scripts are run from the reference directory.

Minor:

- Schema: Added ability to store per-parameter designer notes.
- Added offline wheels distribution.
- Added `read_lef()` function to help with PDK bring-up.

- Added environment variables to replay scripts.
- Added LVS/DRC signoff flow and top-level GDS stream out flow.
- Added native support for Sky130 I/O library, along with Heartbeat + padring example.
- Changed internals to minimize SC performance overhead with large flowgraphs.
- Changed task runtime tracking to distinguish between time spent in tool and total time.
- Fixed breakpoints to work more consistently across tools.

Note: Since there was no public release of version 0.8.0, this list summarizes all changes since 0.7.0.

3.15.53 SiliconCompiler 0.7.0 (2022-03-02)

Major:

- Schema: Added ability to specify environment variables on a per-tool, per-task basis.
- Schema: Added per-tool 'techmap' parameter to library schema.
- Added browser-viewable report generation to core.summary().

Minor:

- Schema: Added filetypes to library schema: 'def', 'gerber', 'netlist', 'model' category.
- Schema: Added 'stackup' key to library lef/gds parameters.
- Schema: Changed 'pdk' and 'stackup' library parameters to lists.
- Schema: Added 'dir' passthrough to library schema.
- Schema: Added 'nodisplay' option to schema to better support headless jobs.
- Schema: Added 'licensefile' to package parameters to support non-standard licenses.
- Schema: Added 'gerber' to read schema.
- Schema: Added several cell categories to library schema.
- Changed how PDK-specific Yosys and OpenROAD parameters are driven to avoid hardcoding process info in tool drivers.
- Fixed step ordering bug in core.summary().
- Fixed bug with how 'arg', 'index' is handled.
- Fixed small bugs in automatic documentation generation.
- Added core.check_filepaths() helper.

3.15.54 SiliconCompiler 0.6.0 (2022-02-11)

Major:

- Schema: Added 'flow' key to flowgraph to enable multi-flow targets.
- Schema: Added 'flow' parameter to enable selection between flows in flowgraph.
- Schema: Changed '_' separated tuple target to a single 'module' load target.
- Schema: Added 'regex' for grep like functionality.
- Schema: Changed metal grid to use PDK metal name as the major key.

- Schema: Added 'tool' key to PDK settings to avoid tool file conflicts.
- Schema: Added 'units' parameter to enable tech agnostic SDC.
- Schema: Added ability to specify tricky apr setup files on a per tool basis (tracks, taps, vias, antenna, etc).
- Schema: Added checklist functionality
- Added core.grep function
- Added core.check_logfile function to core API to emulate grep behavior
- Added core.load_{target, flow, lib, pdk} functions to core API in place of target()
- Added asap7 target
- Added docker support for basic RTL2GDS tool chain
- Removed core.target() function

Minor:

- Schema: Changed lib 'driver' to move into cells (consistency)
- Schema: Added site symmetry to avoid full lef parser.
- Schema: Changed tool version switch to a list
- Schema: Changed 'asic', 'targetlib' to 'asic', 'logiclib' for clarity.
- Schema: Changed 'eda, 'report' parameter guideline to always use 'metric' as keyword
- Schema: Added -skip_check option to speed up new target bringup
- Schema: Added -skip_step option to enable skipping specific steps
- Schema: Added 'pdk', ['file', 'directory', 'variable'] parameters to enable tool-specific PDK setups.
- Schema: Changed cell types to be hardcoded (tapcell, buf, clkbuf, etc) to avoid fragmentation.

3.15.55 SiliconCompiler 0.4.1 (2022-01-06)

Minor:

- Fix bug in Yosys parameter requirements spec that made check_manifest() too strict

3.15.56 SiliconCompiler 0.4.0 (2022-01-05)

Major:

- Schema: Add 'tool' key to PDK fields
- Schema: Remove unneeded 'record' keys
- Implement automatic record-keeping
- Implement checks that flow make sense in terms of file I/O and that required files resolve
- Allow importing multiple files with the same basename

Minor:

- Automatically configure KLayout path on macOS
- Allow importing multiple files with the same basename
- Implement -I CLI switch for include directory

3.15.57 SiliconCompiler 0.3.1 (2021-12-21)

Minor:

- Fix sc-show on Windows.

3.15.58 SiliconCompiler 0.3.0 (2021-12-21)

Major:

- Schema: add 'read' section.
- Schema: Add alternate frontend support.

Minor:

- Fix old version of Surelog bundled with wheels

3.15.59 SiliconCompiler 0.1.1 (2021-12-08)

Minor:

- Fix: Prevent sc-show crash when PDK files are not found.
- Fix: Ensure sc-show can find KLayout executable on Windows

3.15.60 SiliconCompiler 0.1.0 (2021-12-03)

Major:

- First public release!!!

3.16 Dashboard

To start, run the command:

```
sc-dashboard -cfg <path to manifest>
```

You can specify the port by adding a port flag. If you don't provide one, the port will default to 8501:

```
sc-dashboard -cfg <path to manifest> -port <port number>
```

And/or you can include extra chips by adding one or multiple `-graph_cfg` flags. The name of the manifest is optional. If you don't provide one, the name will default to the path to manifest:

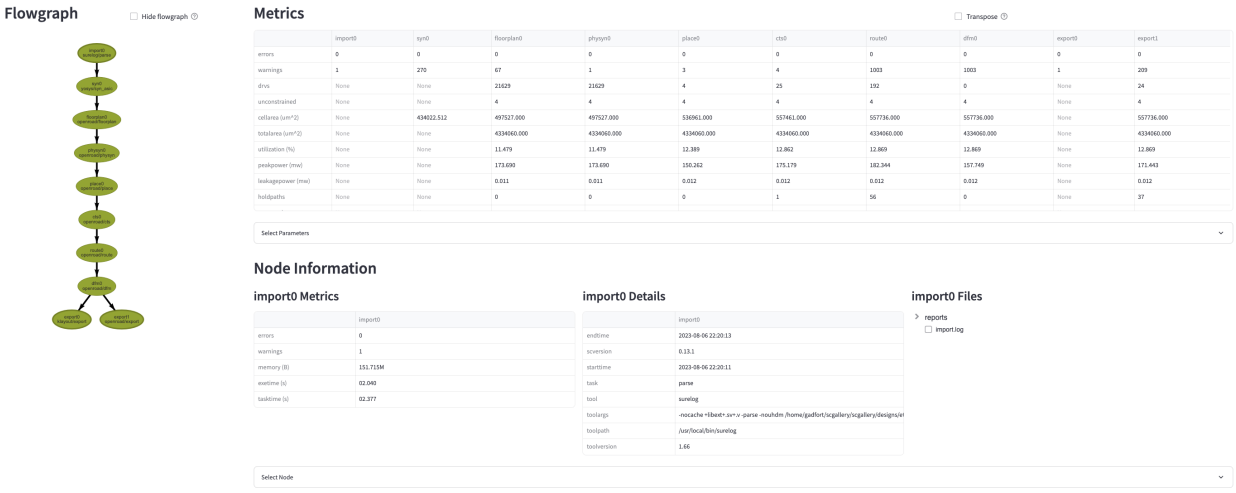
```
sc-dashboard -cfg <path to manifest> -graph_cfg <manifest name> <path to manifest> -
↪graph_cfg <manifest name> <path to manifest>
```

3.16.1 Header

In the header, you can select any of the chips that are part of the history of the chip loaded in and any of the other chips loaded in through the `-graph_cfg` flag (see arrow below).

3.16.2 Metrics Tab

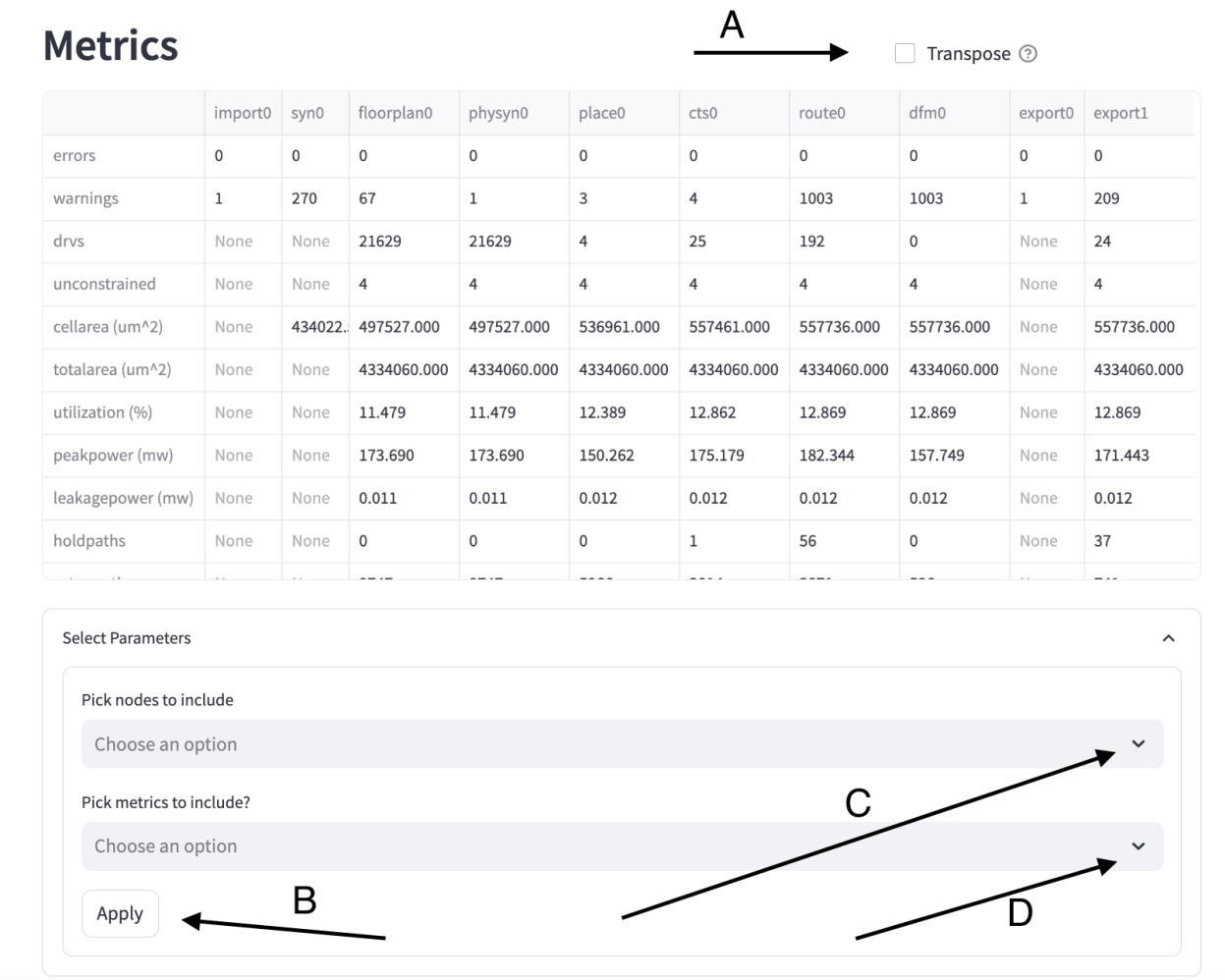
You will load into the metrics tab similar to the one below upon running *sc-dashboard*.



Metrics Section

The metrics section displays an overview of each value for each metric tracked for each node. Click “Transpose” to switch the axes (arrow A in image below).

You may use the “Select Parameter” expander to specify certain nodes and/or metrics (arrow C and D in image below). Click “Apply” to make those changes (arrow B in image below). If you don’t specify any nodes and/or metrics, all of the nodes and/or metrics will be shown.

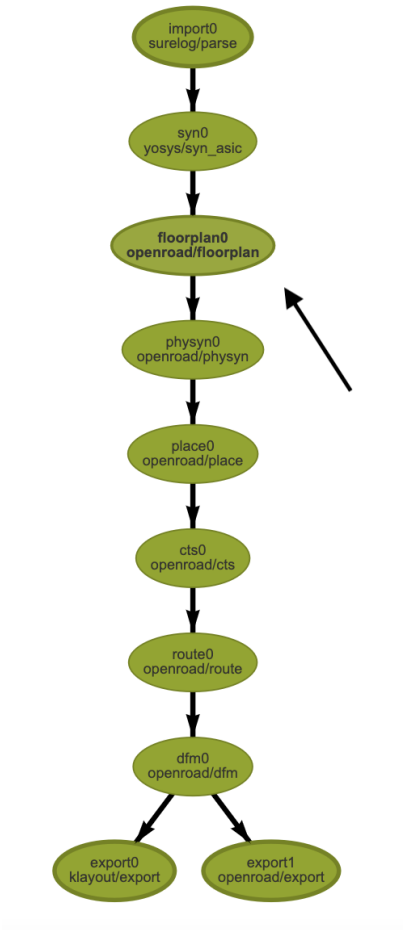


	import0	syn0	floorplan0	physyn0	place0	cts0	route0	dfm0	export0	export1
errors	0	0	0	0	0	0	0	0	0	0
warnings	1	270	67	1	3	4	1003	1003	1	209
drvs	None	None	21629	21629	4	25	192	0	None	24
unconstrained	None	None	4	4	4	4	4	4	None	4
cellarea (um^2)	None	434022..	497527.000	497527.000	536961.000	557461.000	557736.000	557736.000	None	557736.000
totalarea (um^2)	None	None	4334060.000	4334060.000	4334060.000	4334060.000	4334060.000	4334060.000	None	4334060.000
utilization (%)	None	None	11.479	11.479	12.389	12.862	12.869	12.869	None	12.869
peakpower (mw)	None	None	173.690	173.690	150.262	175.179	182.344	157.749	None	171.443
leakagepower (mw)	None	None	0.011	0.011	0.012	0.012	0.012	0.012	None	0.012
holdpaths	None	None	0	0	0	1	56	0	None	37

Flowgraph Section

The flowgraph section displays the data dependencies for each node. Nodes are color-coded based on their task status. Green means task status is a success, red means task status is a failure, and yellow means task status is pending. Currently, task status should never be yellow because you cannot view the dashboard while the build is not done. This is functionality we hope to add. Paths that are part of the 'winning path' will have bolded edges.

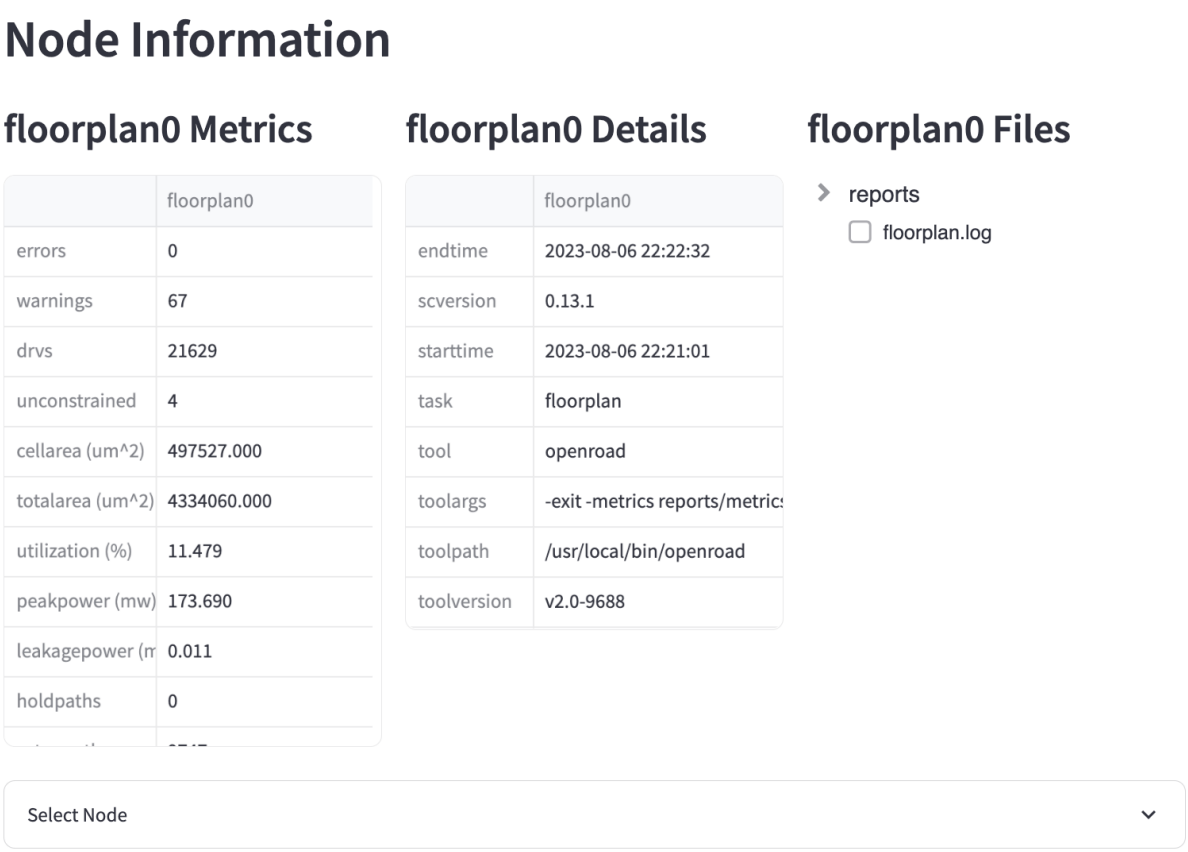
To activate the flowgraph, click on it once. This allows you to interact with the flowgraph. You can drag nodes around, pan the view, and zoom in/out. You can also click nodes to select them for the *Node Information Section*. You can see a node is selected by it becoming bolded (in the image below, that's floorplan0). Double clicking nodes will send you to a blank html page. We are aware of this bug.



Node Information Section

Below is the node information section. It consists of three subsections - node metrics, node details, and node files.

You can select a node using the "Select Node" expander as seen with arrow's A and B below. Click "Apply" to make the change.



Node Metrics Subsection

The node metrics subsection consists of all of not `None` values recorded for each of the metrics recorded for the selected node.

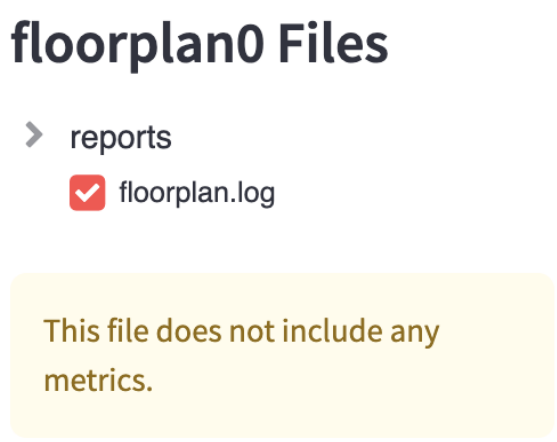
Node Details Subsection

The node details subsection consists of all of the characteristics about this node that are not reflected in the metrics section.

Node Files Subsection

The node files subsection consists of all of the files for a given node that are in the build directory.

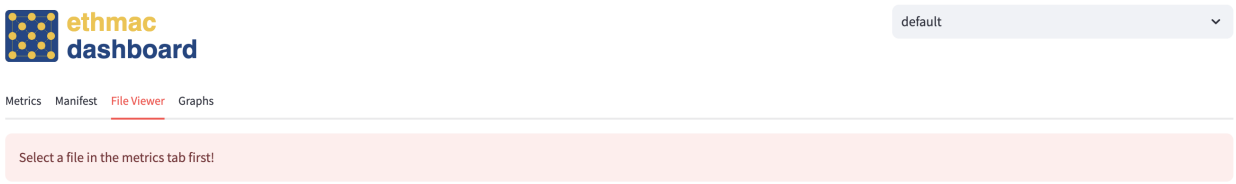
Selecting a node will display a list of the metrics that the file informs below the file tree. In the picture below, no metrics can be found in the floorplan.log file.



3.16.3 File Viewer Tab

The selected node you clicked in the *Node Files Subsection* will appear here. The header is the name of the file selected. You can download the file by clicking the download button (as by the arrow in the image below).

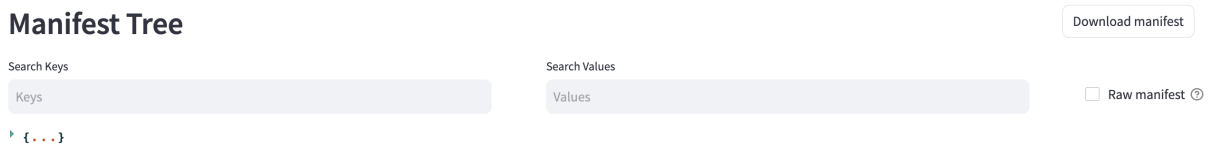
If no file is selected, the error message below will be displayed telling you to select a file first.



3.16.4 Manifest Tab

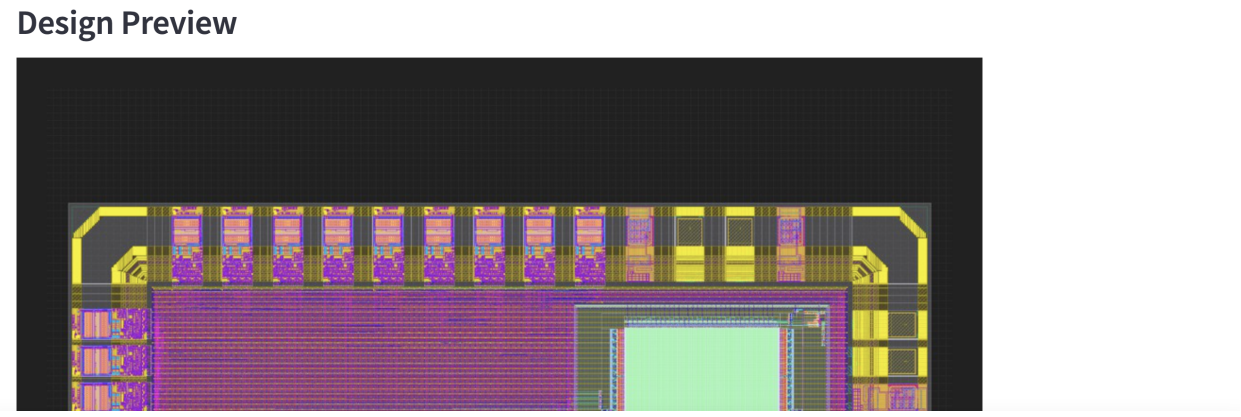
The next tab you can select is the manifest tab. This displays the manifest after it has been filtered through to make it more readable. More specifically, if the *pernode* value of the leaf of the Schema is *pernode* is "never", the value of the leaf is the value of the leaf['node']['global']['global']['value']. If there is no value for that, then it is the value of the leaf['node']['default']['default']['value']. Outside of that, the nodes will be concatenated, or if the step and index is *default* and *default* or "global" and "global", the node will be *default* or "global", respectively.

To view the manifest, click the arrow on the dictionary (arrow A). The search bars will return partial matches for either the keys (arrow B in image below) or the values (arrow C in image below). Press enter to search. If you do not want to search, delete any text in the search bars and press enter. You may download the JSON as you view it at any point (arrow D in image below). The name of the file generated is "manifest.json". You can view the raw manifest by clicking the checkbox to the right of the search bar (arrow E in image below).



3.16.5 Display Preview Tab

This displays the preview image of the chip if there is one in the directory (example given below). If not, this tab will not be included.

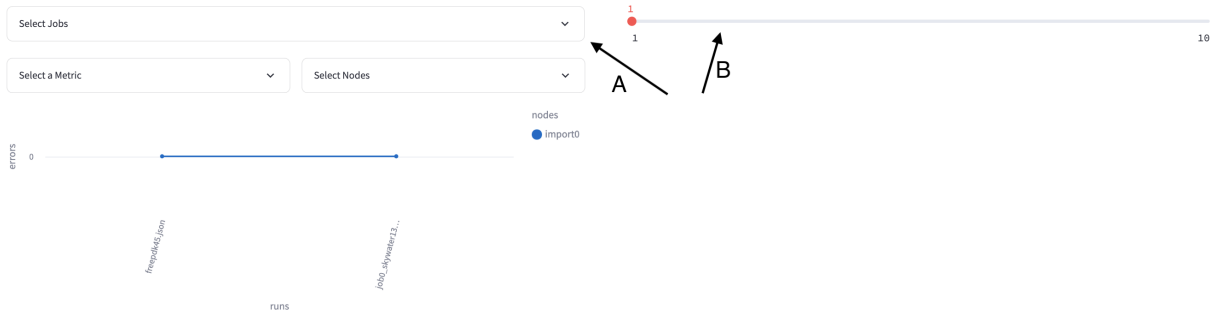


3.16.6 Graphs Tab

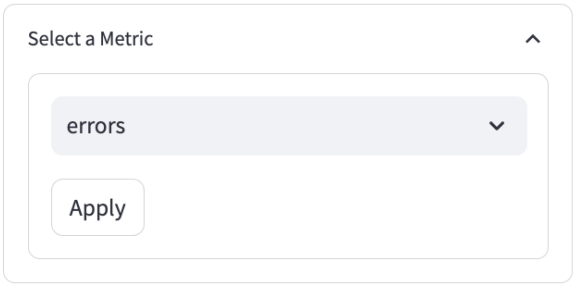
This tab is meant to make comparisons between nodes for a given metric over many chip objects.

At the top of the panel, select which runs/jobs to include for all the graphs (arrow A in image below). These are the runs from the chip's history and the runs included with the `-graph_cfg` flag.

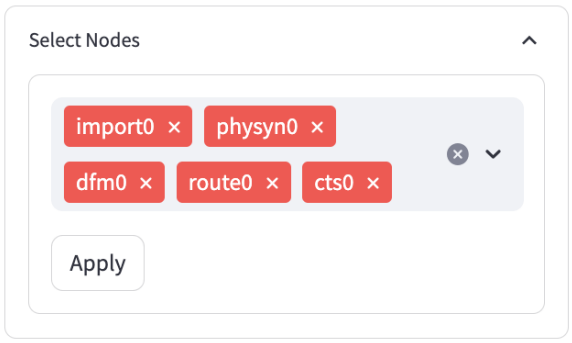
Move the slider to add more graphs or remove old ones (arrow B in image below). Removing old graphs will remove them in the reverse order in which they were added.



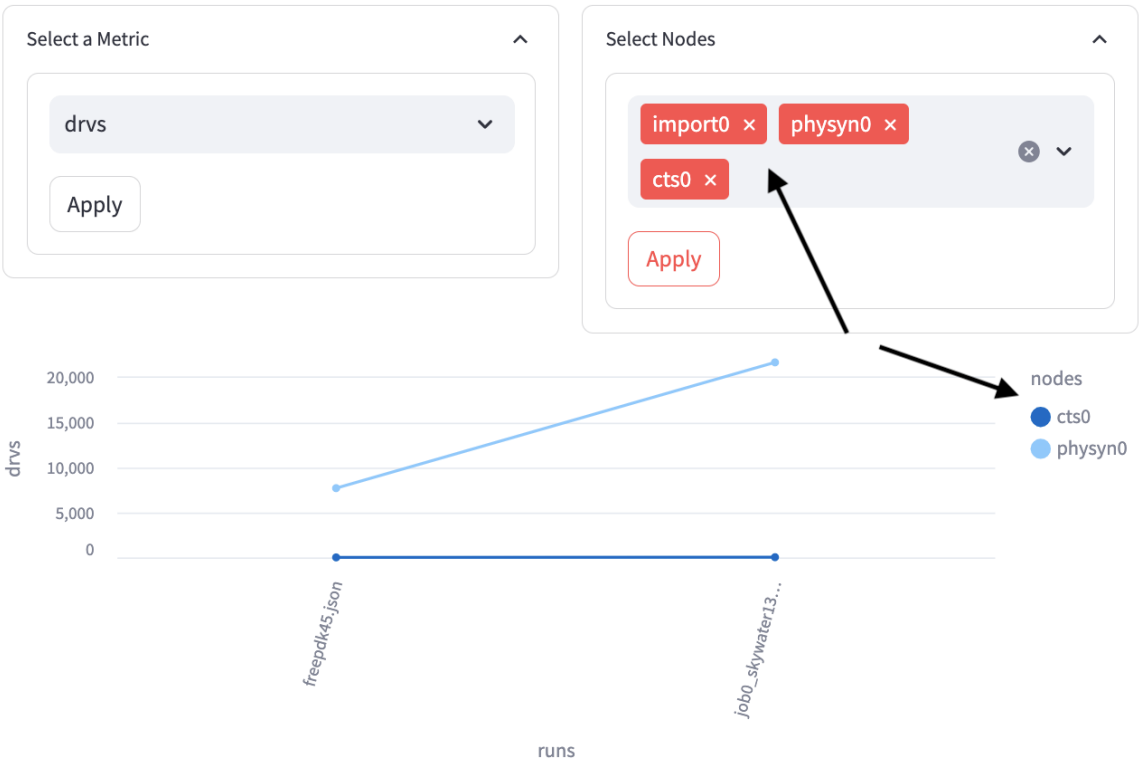
For each graph, you must select one metric (show in image below). A random metric will be pre-selected. Click 'Apply' to see the changes.



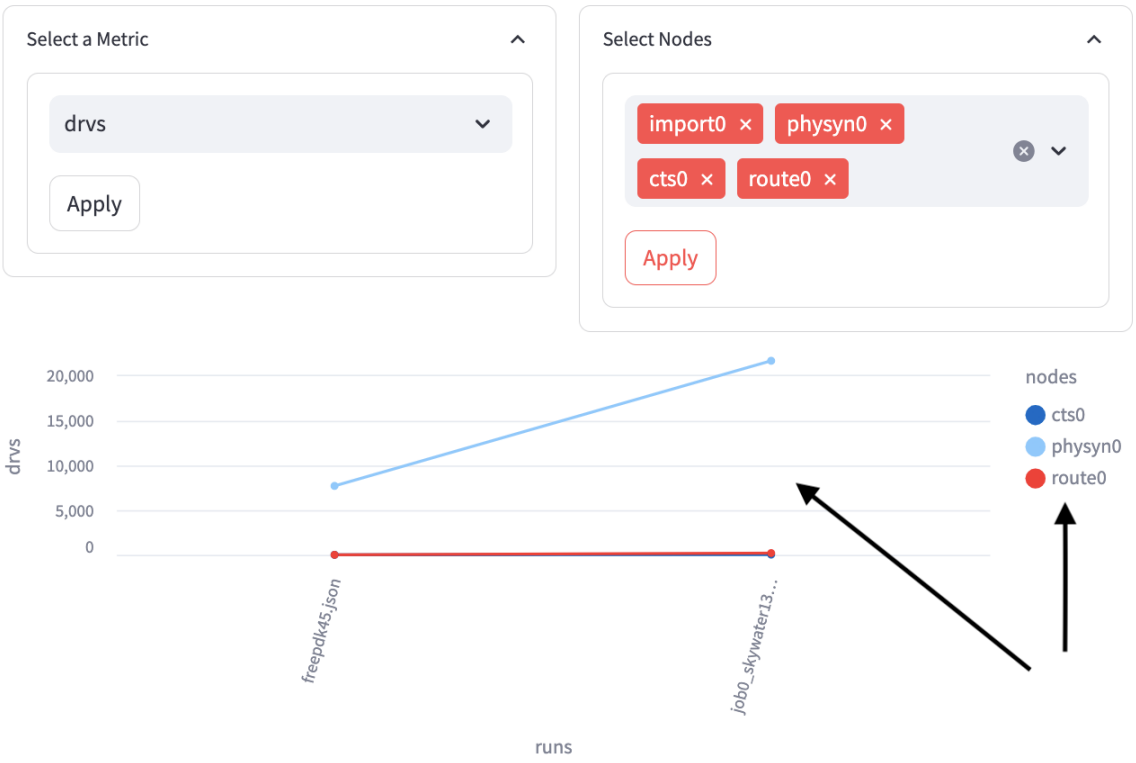
You may select any amount of nodes (show in image below). A random node will be pre-selected. If you select 0 nodes, a blank graph will appear. Click 'Apply' to see the changes.



Sometimes nodes may not have values for a metric, in which case they will not be included in the graph. In the image below, import0 is not in the legend.



Sometimes nodes that are in the legend are not visible on the graph. What has happened is that they have the exact same values as some other node. Consider deselecting other nodes in this case. In the image below, cts0 is barely visible on the graph.


Select a Metric
drvs
Apply
Select Nodes
import0 ×
physyn0 ×
cts0 ×
route0 ×
Apply
20,000
15,000
10,000
5,000
0
drvs
freepdk45.json
job0_skywater13...
runs
nodes
cts0
physyn0
route0

PYTHON MODULE INDEX

S

INDEX