OpenROAD

OpenROAD Team

Apr 08, 2024

CONTENTS

The OpenROAD ("Foundations and Realization of Open, Accessible Design") project was launched in June 2018 within the DARPA IDEA program. OpenROAD aims to bring down the barriers of cost, expertise and unpredictability that currently block designers' access to hardware implementation in advanced technologies. The project team (Qualcomm, Arm and multiple universities and partners, led by UC San Diego) is developing a fully autonomous, open-source tool chain for digital SoC layout generation, focusing on the RTL-to-GDSII phase of system-on-chip design. Thus, OpenROAD holistically attacks the multiple facets of today's design cost crisis: engineering resources, design tool licenses, project schedule, and risk.

The IDEA program targets no-human-in-loop (NHIL) design, with 24-hour turnaround time and zero loss of power-performance-area (PPA) design quality.

The NHIL target requires tools to adapt and auto-tune successfully to flow completion, without (or, with minimal) human intervention. Machine intelligence augments human expertise through efficient modeling and prediction of flow and optimization outcomes throughout the synthesis, placement and routing process. This is complemented by development of metrics and machine learning infrastructure.

The 24-hour runtime target implies that problems must be strategically decomposed throughout the design process, with clustered and partitioned subproblems being solved and recomposed through intelligent distribution and management of computational resources. This ensures that the NHIL design optimization is performed within its available `[threads * hours]` "box" of resources. Decomposition that enables parallel and distributed search over cloud resources incurs a quality-of-results loss, but this is subsequently recovered through improved flow predictability and enhanced optimization.

For a technical description of the OpenROAD flow, please refer to our DAC-2019 paper: Toward an Open-Source Digital Flow: First Learnings from the OpenROAD Project. The paper is also available from ACM Digital Library. Other publications and presentations are linked here.

CHAPTER ONE

DOCUMENTATION

The OpenROAD Project has two releases:

- Application (github) (*docs*): The application is a standalone binary for digital place and route that can be used by any other RTL-GDSII flow controller.
- Flow (github) (docs): This is the native OpenROAD flow that consists of a set of integrated scripts for an autonomous RTL-GDSII flow using OpenROAD and other open-source tools.

CHAPTER TWO

CODE OF CONDUCT

Please read our code of conduct *here*.

CHAPTER

THREE

HOW TO CONTRIBUTE

If you are willing to **contribute**, see the *Getting Involved* section.

If you are a **developer** with EDA background, learn more about how you can use OpenROAD as the infrastructure for your tools in the *Developer Guide* section.

OpenROAD uses Git for version control and contributions. Get familiarised with a quickstart tutorial to contribution *here*.

CHAPTER FOUR

HOW TO GET IN TOUCH

We maintain the following channels for communication:

- Project homepage and news: https://theopenroadproject.org
- Twitter: https://twitter.com/OpenROAD_EDA
- Issues and bugs:
  - OpenROAD: https://github.com/The-OpenROAD-Project/OpenROAD/issues
- Discussions:
  - OpenROAD: https://github.com/The-OpenROAD-Project/OpenROAD/discussions
- Inquiries: openroad@ucsd.edu

See also our *FAQs*.

CHAPTER

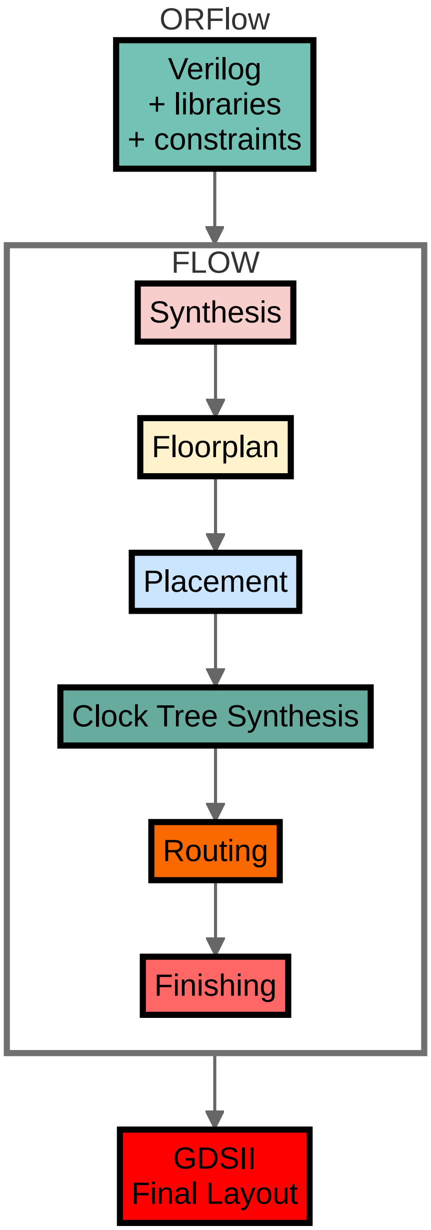
FIVE

SITE MAP

5.1 OpenROAD

5.1.1 About OpenROAD

OpenROAD is the leading open-source, foundational application for semiconductor digital design. The OpenROAD flow delivers an Autonomous, No-Human-In-Loop (NHIL) flow, 24 hour turnaround from RTL-GDSII for rapid design exploration and physical design implementation.


ORFlow
Verilog
+ libraries
+ constraints
FLOW
Synthesis
Floorplan
Placement
Clock Tree Synthesis
Routing
Finishing
GDSII
Final Layout

5.1.2 OpenROAD Mission

OpenROAD eliminates the barriers of cost, schedule risk and uncertainty in hardware design to promote open access to rapid, low-cost IC design software and expertise and system innovation. The OpenROAD application enables flexible flow control through an API with bindings in Tcl and Python.

OpenROAD is used in research and commercial applications such as,

- OpenROAD-flow-scripts from OpenROAD
- OpenLane from Efabless
- Silicon Compiler from Zero ASIC
- Hammer from UC Berkeley
- OpenFASoC from IDEA-FASoC for mixed-signal design flows

OpenROAD fosters a vibrant ecosystem of users through active collaboration and partnership through software development and key alliances. Our growing user community includes hardware designers, software engineers, industry collaborators, VLSI enthusiasts, students and researchers.

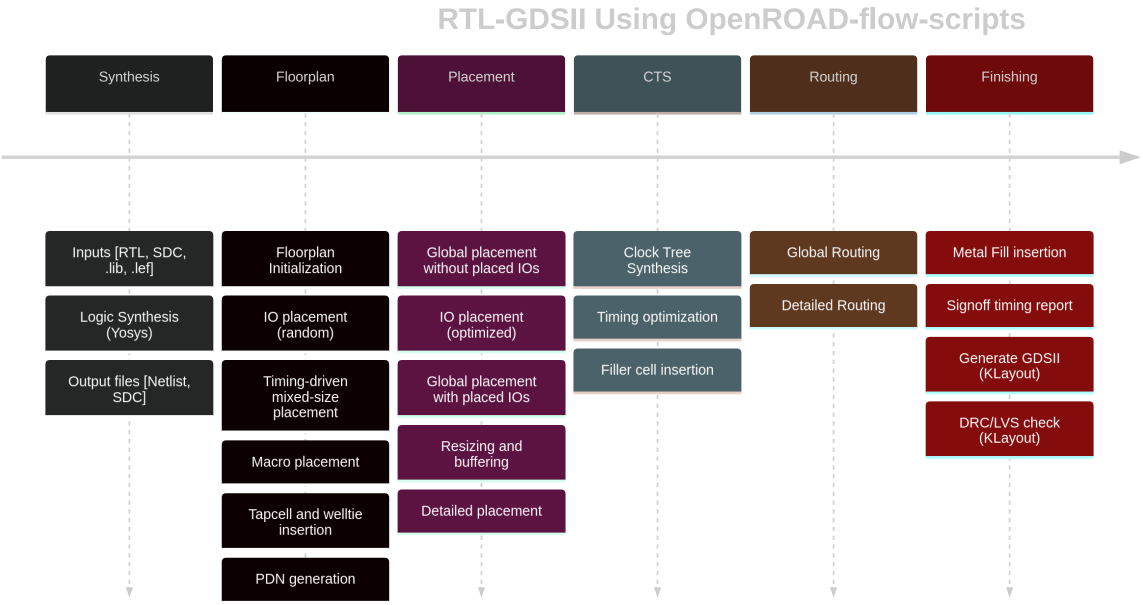
OpenROAD strongly advocates and enables IC design-based education and workforce development initiatives through training content and courses across several global universities, the Google-SkyWater shuttles also includes GlobalFoundries shuttles, design contests and IC design workshops. The OpenROAD flow has been successfully used to date in over 600 silicon-ready tapeouts for technologies up to 12nm.

5.1.3 Getting Started with OpenROAD-flow-scripts

OpenROAD provides OpenROAD-flow-scripts as a native, ready-to-use prototyping and tapeout flow. However, it also enables the creation of any custom flow controllers based on the underlying tools, database and analysis engines. Please refer to the flow documentation here.

OpenROAD-flow-scripts (ORFS) is a fully autonomous, RTL-GDSII flow for rapid architecture and design space exploration, early prediction of QoR and detailed physical design implementation. However, ORFS also enables manual intervention for finer user control of individual flow stages through Tcl commands and Python APIs.

Figure below shows the main stages of the OpenROAD-flow-scripts:



Here are the main steps for a physical design implementation using OpenROAD;

- `Floorplanning`

  - Floorplan initialization - define the chip area, utilization
  - IO pin placement (for designs without pads)
  - Tap cell and well tie insertion
  - PDN- power distribution network creation
- `Global Placement`
  - Macro placement (RAMs, embedded macros)
  - Standard cell placement
  - Automatic placement optimization and repair for max slew, max capacitance, and max fanout violations and long wires
- `Detailed Placement`
  - Legalize placement - align to grid, adhere to design rules
  - Incremental timing analysis for early estimates
- `Clock Tree Synthesis`
  - Insert buffers and resize for high fanout nets
- `Optimize setup/hold timing`
- `Global Routing`
  - Antenna repair
  - Create routing guides
- `Detailed Routing`
  - Legalize routes, DRC-correct routing to meet timing, power constraints
- `Chip Finishing`
  - Parasitic extraction using OpenRCX
  - Final timing verification
  - Final physical verification
  - Dummy metal fill for manufacturability
  - Use KLayout or Magic using generated GDS for DRC signoff

GUI

The OpenROAD GUI is a powerful visualization, analysis, and debugging tool with a customizable Tcl interface. The below figures show GUI views for various flow stages including floorplanning, placement congestion, CTS and post-routed design.

Floorplan

Automatic Hierarchical Macro Placement

Placement Congestion Visualization

CTS

Routing

PDK Support

The OpenROAD application is PDK independent. However, it has been tested and validated with specific PDKs in the context of various flow controllers.

OpenLane supports SkyWater 130nm and GlobalFoundries 180nm.

OpenROAD-flow-scripts supports several public and private PDKs including:

Open-Source PDKs

- `GF180` - 180nm
- `SKY130` - 130nm
- `Nangate45` - 45nm
- `ASAP7` - Predictive FinFET 7nm

Proprietary PDKs

These PDKS are supported in OpenROAD-flow-scripts only. They are used to test and calibrate OpenROAD against commercial platforms and ensure good QoR. The PDKs and platform-specific files for these kits cannot be provided due to NDA restrictions. However, if you are able to access these platforms independently, you can create the necessary platform-specific files yourself.

- `GF55` - 55nm
- `GF12` - 12nm
- `Intel22` - 22nm

- `Intel16` - 16nm
- `TSMC65` - 65nm

5.1.4 Tapeouts

OpenROAD has been used for full physical implementation in over 600 tapeouts in SKY130 and GF180 through the Google-sponsored, Efabless MPW shuttle and ChipIgnite programs.

OpenTitan SoC on GF12LP - Physical design and optimization using OpenROAD

Continuous Tapeout Integration into CI

The OpenROAD project actively adds successfully taped out MPW shuttle designs to the CI regression testing. Examples of designs include Open processor cores, RISC-V based SoCs, cryptocurrency miners, robotic app processors, amateur satellite radio transceivers, OpenPower-based Microwatt etc.

5.1.5 Build OpenROAD

To build OpenROAD tools locally in your machine, follow steps from *here*.

5.1.6 Regression Tests

There are a set of executable regression test scripts in `./test/`.

```
# run tests for all tools
./test/regression

# run all flow tests
./test/regression flow

# run <tool> tests
./test/regression <tool>

# run all <tool>-specific unit tests
cd src/<tool>
./test/regression

# run only <TEST_NAME> for <tool>
cd src/<tool>
./test/regression <TEST_NAME>
```

The flow tests check results such as worst slack against reference values. Use `report_flow_metrics [test]...` to see all of the metrics.

```
% report_flow_metrics gcd_nangate45
                       insts    area util slack_min slack_max  tns_max clk_skew max_slew␣
↪max_cap max_fanout DPL ANT drv
gcd_nangate45            368     564  8.8     0.112    -0.015     -0.1    0.004        0␣
↪       0          0   0   0   0
```

To update a failing regression, follow the instructions below:

```
# update log files (i.e. *ok)
save_ok <TEST_NAME>

# update "*.metrics" for tests that use flow test
save_flow_metrics <TEST_NAME>

# update "*.metrics_limits" files
save_flow_metrics_limits <TEST_NAME>
```

5.1.7 Run

```
openroad [-help] [-version] [-no_init] [-exit] [-gui]
         [-threads count|max] [-log file_name] cmd_file
  -help              show help and exit
  -version           show version and exit
  -no_init           do not read .openroad init file
  -threads count|max use count threads
  -no_splash         do not show the license splash at startup
  -exit              exit after reading cmd_file
  -gui               start in gui mode
  -python            start with python interpreter [limited to db operations]
  -log <file_name>   write a log in <file_name>
  cmd_file           source cmd_file
```

OpenROAD sources the Tcl command file `~/.openroad` unless the command line option `-no_init` is specified.

OpenROAD then sources the command file `cmd_file` if it is specified on the command line. Unless the `-exit` command line flag is specified, it enters an interactive Tcl command interpreter.

A list of the available tools/modules included in the OpenROAD app and their descriptions are available *here*.

5.1.8 Git Quickstart

OpenROAD uses Git for version control and contributions. Get familiarised with a quickstart tutorial to contribution *here*.

5.1.9 Understanding Warning and Error Messages

Seeing OpenROAD warnings or errors you do not understand? We have compiled a table of all messages and you may potentially find your answer here.

5.1.10 License

BSD 3-Clause License. See LICENSE file.

5.1.11 Installing OpenROAD

Build

The first step, independent of the build method, is to download the repository:

```
git clone --recursive https://github.com/The-OpenROAD-Project/OpenROAD.git
cd OpenROAD
```

OpenROAD git submodules (cloned by the `--recursive` flag) are located in `src/`.

The default build type is RELEASE to compile optimized code. The resulting executable is in `build/src/openroad`.

Optional CMake variables passed as `-D<var>=<value>` arguments to CMake are show below.

Argument	Value
`CMAKE_BUILD_TYPE`	DEBUG, RELEASE
`CMAKE_CXX_FLAGS`	Additional compiler flags
`TCL_LIBRARY`	Path to Tcl library
`TCL_HEADER`	Path to `tcl.h`
`ZLIB_ROOT`	Path to `zlib`
`CMAKE_INSTALL_PREFIX`	Path to install binary

Note: There is a `openroad_build.log` file that is generated with every build in the build directory. In case of filing issues, it can be uploaded in the "Relevant log output" section of OpenROAD issue forms.

Install dependencies

You may follow our helper script to install dependencies as follows:

```
sudo ./etc/DependencyInstaller.sh
```

WARNING

`etc/DependencyInstaller.sh` defaults to installing system packages and requires sudo access. These packages can affect your environment. We recommend users install dependencies locally using setup.sh from OpenROAD-flow-scripts.

Build Manually

```
mkdir build && cd build
cmake ..
make
make install
```

The default install directory is `/usr/local`. To install in a different directory with CMake use:

```
cmake .. -DCMAKE_INSTALL_PREFIX=<prefix_path>
```

Alternatively, you can use the `DESTDIR` variable with make.

```
make DESTDIR=<prefix_path> install
```

Build using support script

```
./etc/Build.sh
# To build with debug option enabled and if the Tcl library is not on the default path
./etc/Build.sh -cmake="-DCMAKE_BUILD_TYPE=DEBUG -DTCL_LIB=/path/to/tcl/lib"
```

The default install directory is `/usr/local`. To install in a different directory use:

```
./etc/Build.sh -cmake="-DCMAKE_INSTALL_PREFIX=<prefix_path>"
```

LTO Options

By default, OpenROAD is built with link time optimizations enabled. This adds about 1 minute to compile times and improves the runtime by about 11%. If you would like to disable LTO pass `-DLINK_TIME_OPTIMIZATION=OFF` when generating a build.

Build with Address Sanitizer

To enable building with Address Sanitizer, use the argument `-DASAN=ON`. Setting the `ASAN` variable to `ON` adds necessary compile and link options for using Address Sanitizer.

> **Note:** Address Sanitizer adds instrumentation for detecting memory errors. Enabling this option will cause OpenROAD to run slower and consume more RAM.

Build with Prebuilt Binaries

Courtesy of Precision Innovations, there are pre-built binaries of OpenROAD with self-contained dependencies released on a regular basis. Refer to this link here.

5.1.12 Tutorials

OpenROAD Flow Scripts Tutorial

Flow tutorial can be accessed from OpenROAD Flow Scripts documentation here.

5.1.13 Git Quickstart

This tutorial serves as a quickstart to Git and contributing to our repository. If you have not already set up OpenROAD, please follow the instructions *here*.

Tip: This basic tutorial gives instruction for basic password Git authentication. If you would like to setup SSH authentication, please follow this guide.

Forking

You will need your own fork to work on the code. Go to the `OpenROAD` project page and hit the `Fork` button. You will want to clone your fork to your machine:

```
git clone https://github.com/your-user-name/OpenROAD.git
cd OpenROAD
git remote add upstream https://github.com/The-OpenROAD-Project/OpenROAD.git
git fetch upstream
```

This creates the directory `OpenROAD` and connects your repository to the upstream (master project) *OpenROAD* repository.

Creating a branch

You want your master branch to reflect only production-ready code, so create a feature branch for making your changes. For example:

```
git checkout master && git branch shiny-new-feature
git checkout shiny-new-feature
# Or equivalently,
git checkout master && checkout -b shiny-new-feature
```

This changes your working directory to the shiny-new-feature branch. Keep any changes in this branch specific to one bug or feature so it is clear what the branch brings to OpenROAD. You can have many shiny-new-features and switch in between them using the git checkout command.

When creating this branch, make sure your master branch is up to date with the latest upstream master version. To update your local master branch, you can do:

```
git checkout master
git pull upstream master
```

When you want to update the feature branch with changes in master after you created the branch, check the section on *updating a PR*.

Committing your code

Keep style fixes to a separate commit to make your pull request more readable. Once you've made changes, you can see them by typing:

```
git status
```

If you have created a new file, it is not being tracked by git. Add it by typing:

```
git add path/to/file-to-be-added.py
```

Doing `git status` again should give something like:

```
# On branch shiny-new-feature
#
#       modified:   /relative/path/to/file-you-added.py
#
```

Finally, commit your changes to your local repository with an explanatory commit message. Do note the `-s` option is needed for developer signoff.

```
git commit -s -m "your commit message goes here"
```

Pushing your changes

When you want your changes to appear publicly on your GitHub page, push your forked feature branch's commits:

```
git push origin shiny-new-feature
```

Here `origin` is the default name given to your remote repository on GitHub. You can see the remote repositories:

```
git remote -v
```

If you added the upstream repository as described above you will see something like:

```
origin  https://github.com/your-user-name/OpenROAD.git (fetch)
origin  https://github.com/your-user-name/OpenROAD.git (push)
upstream        https://github.com/The-OpenROAD-Project/OpenROAD.git (fetch)
upstream        https://github.com/The-OpenROAD-Project/OpenROAD.git (push)
```

Now your code is on GitHub, but it is not yet a part of the OpenROAD project. For that to happen, a pull request needs to be submitted on GitHub.

Review your code

When you're ready to ask for a code review, file a pull request. Before you do, once again make sure that you have followed all the guidelines outlined in the *Developer's Guide* regarding code style, tests, performance tests, and documentation. You should also double check your branch changes against the branch it was based on:

1. Navigate to your repository on GitHub – https://github.com/your-user-name/OpenROAD
2. Click on `Branches`
3. Click on the `Compare` button for your feature branch

4. Select the `base` and `compare` branches, if necessary. This will be `master` and `shiny-new-feature`, respectively.

Submitting the pull request

If everything looks good, you are ready to make a pull request. A pull request is how code from a local repository becomes available to the GitHub community and can be looked at and eventually merged into the master version. This pull request and its associated changes will eventually be committed to the master branch and available in the next release. To submit a pull request:

1. Navigate to your repository on GitHub
2. Click on the `Compare & pull request` button
3. You can then click on `Commits` and `Files Changed` to make sure everything looks okay one last time
4. Write a description of your changes in the `Preview Discussion` tab
5. Click `Send Pull Request`.

This request then goes to the repository maintainers, and they will review the code.

Updating your pull request

Based on the review you get on your pull request, you will probably need to make some changes to the code. In that case, you can make them in your branch, add a new commit to that branch, push it to GitHub, and the pull request will be automatically updated. Pushing them to GitHub again is done by:

```
git push origin shiny-new-feature
```

This will automatically update your pull request with the latest code and restart the *Continuous Integration* tests.

Another reason you might need to update your pull request is to solve conflicts with changes that have been merged into the master branch since you opened your pull request.

To do this, you need to `merge upstream master` in your branch:

```
git checkout shiny-new-feature
git fetch upstream
git merge upstream/master
```

If there are no conflicts (or they could be fixed automatically), a file with a default commit message will open, and you can simply save and quit this file.

If there are merge conflicts, you need to solve those conflicts. See this article for an explanation on how to do this. Once the conflicts are merged and the files where the conflicts were solved are added, you can run `git commit` to save those fixes.

If you have uncommitted changes at the moment you want to update the branch with master, you will need to `stash` them prior to updating.

See also:

See the stash docs.

This will effectively store your changes and they can be reapplied after updating.

After the feature branch has been updated locally, you can now update your pull request by pushing to the branch on GitHub:

```
git push origin shiny-new-feature
```

Tips for a successful pull request

If you have made it to the `Review your code` phase, one of the core contributors may take a look. Please note however that a handful of people are responsible for reviewing all of the contributions, which can often lead to bottlenecks.

To improve the chances of your pull request being reviewed, you should:

- **Reference an open issue** for non-trivial changes to clarify the PR's purpose
- **Ensure you have appropriate tests**. These should be the first part of any PR
- **Keep your pull requests as simple as possible**. Larger PRs take longer to review
- **Ensure that CI is in a green state**. Reviewers may not even look otherwise
- **Keep updating your pull request**, either by request or every few days

Acknowledgements

This page has been adapted from pandas Developer Guide.

5.2 OpenROAD

OpenROAD is run using Tcl scripts. The following commands are used to read and write design data.

```
read_lef [-tech] [-library] filename
read_def filename
write_def [-version 5.8|5.7|5.6|5.5|5.4|5.3] filename
read_verilog filename
write_verilog filename
read_db filename
write_db filename
write_abstract_lef filename
```

Use the Tcl `source` command to read commands from a file.

```
source [-echo] file
```

If an error is encountered in a command while reading the command file, then the error is printed and no more commands are read from the file. If `file_continue_on_error` is 1 then OpenROAD will continue reading commands after the error.

If `exit_on_error` is 1 then OpenROAD will exit when it encounters an error.

OpenROAD can be used to make a OpenDB database from LEF/DEF, or Verilog (flat or hierarchical). Once the database is made it can be saved as a file with the `write_db` command. OpenROAD can then read the database with the `read_db` command without reading LEF/DEF or Verilog.

The `read_lef` and `read_def` commands can be used to build an OpenDB database as shown below. The `read_lef -tech` flag reads the technology portion of a LEF file. The `read_lef -library` flag reads the MACROs in the LEF file. If neither of the `-tech` and `-library` flags are specified they default to `-tech -library` if no technology has been read and `-library` if a technology exists in the database.

```
read_lef liberty1.lef
read_def reg1.def
# Write the db for future runs.
write_db reg1.db
```

The `read_verilog` command is used to build an OpenDB database as shown below. Multiple Verilog files for a hierarchical design can be read. The `link_design` command is used to flatten the design and make a database.

```
read_lef liberty1.lef
read_verilog reg1.v
link_design top
# Write the db for future runs.
write_db reg1.db
```

5.2.1 Example scripts

Example scripts demonstrating how to run OpenROAD on sample designs can be found in /test. Flow tests taking sample designs from synthesizable RTL Verilog to detail-routed final layout in the open-source technologies Nangate45 and Sky130HD are shown below.

```
gcd_nangate45.tcl
aes_nangate45.tcl
tinyRocket_nangate45.tcl
gcd_sky130hd.tcl
aes_sky130hd.tcl
ibex_sky130hd.tcl
```

Each of these designs use the common script `flow.tcl`.

5.2.2 Abstract LEF Support

OpenROAD contains an abstract LEF writer that can take your current design and emit an abstract LEF representing the external pins of your design and metal obstructions.

```
write_abstract_lef (-bloat_factor bloat_factor|-bloat_occupied_layers) \
                   filename
```

Options

Switch Name	Description
`-bloat_factor`	Specifies the bloat factor used when bloating then merging shapes into LEF obstructions. The factor is measured in # of default metal pitches for the respective layer. A factor of `0` will result in detailed LEF obstructions
`-bloat_occupied_layers`	Generates cover obstructions (obstructions over the entire layer) for each layer where shapes are present

Examples

```
read reg1.db

# Bloat metal shapes by 3 pitches (respectively for every layer) and then merge
write_abstract_lef -bloat_factor 3 reg1_abstract.lef

# Produce cover obstructions for each layer with shapes present
write_abstract_lef -bloat_occupied_layers reg1_abstract.lef
```

Global Connections

Add global connections

The add_global_connection command is used to specify how to connect power and ground pins on design instances to the appropriate supplies.

```
add_global_connection -net net_name \
                      [-inst_pattern inst_regular_expression] \
                      -pin_pattern pin_regular_expression \
                      (-power|-ground) \
                      [-region region_name]
```

Options

Switch Name	Description
-net	Specifies the name of the net in the design to which connections are to be added
-inst_pattern	Optional specifies a regular expression to select a set of instances from the design. (Default: .*)
-pin_pattern	Species a regular expression to select pins on the selected instances to connect to the specified net
-power	Specifies that the net it a power net
-ground	Specifies that the net is a ground net
-region	Specifies the name of the region for this rule

Examples

```
# Stdcell power/ground pins
add_global_connection -net VDD -pin_pattern {^VDD$} -power
add_global_connection -net VSS -pin_pattern {^VSS$} -ground

# SRAM power ground pins
add_global_connection -net VDD -pin_pattern {^VDDPE$}
add_global_connection -net VDD -pin_pattern {^VDDCE$}
add_global_connection -net VSS -pin_pattern {^VSSE$}
```

Perform global connections

The `global_connect` command is used to connect power and ground pins on design instances to the appropriate supplies.

```
global_connect
```

Clear global connection rules

The `clear_global_connect` command is used remove all defined global connection rules.

```
clear_global_connect
```

Report global connection rules

The `report_global_connect` command is used print out the currently defined global connection rules.

```
report_global_connect
```

Report cell type usage

The `report_cell_usage` command is used to print out the usage of cells for each type of cell.

```
report_cell_usage
```

5.2.3 TCL functions

Get the die and core areas as a list in microns: `llx lly urx ury`

```
ord::get_die_area
ord::get_core_area
```

5.2.4 FAQs

Check out GitHub discussion about this tool.

5.2.5 License

BSD 3-Clause License.

5.2.6 OpenDB

The OpenDB (odb) module in OpenROAD is a design database to support tools for physical chip design. It was originally developed by Athena Design Systems. Nefelus, Inc. acquired the rights to the code and open-sourced it with BSD-3 license in 2019 to support the DARPA OpenROAD project.

The structure of OpenDB is based on the text file formats LEF (library) and DEF (design) formats version 5.6. OpenDB supports a binary file format to save and load the design much faster than using LEF and DEF.

OpenDB is written in C++ 98 with standard library style iterators. The classes are designed to be fast enough to base an application on without having to copy them into application-specific structures.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Directory structure

```
include/odb/db.h - public header for all database classes
src/db - private/internal database representations
src/lefin - LEF reader
src/lefout - LEF writer
src/defin - DEF reader
src/defout - DEF writer
```

Database API

We are still working on documenting the APIs. We have over 1,800 objects and functions that we are still documenting (for both TCL and Python). **Contributions are very welcome in this effort**. Find starting points below.

Python

After building successfully, run `openroad -python` to enable the Python interpreter. You can find examples on using the API from Python under `test/python/` directory.

To list the full set of the Python classes exposed run `openroad -python` then:

```
import openroad
import odb
print(', '.join(dir(openroad)))
print(', '.join(dir(odb)))
```

C++

All public database classes are defined in `db.h`. These class definitions provide all functions for examining and modifying the database objects. The database is an object itself, so multiple database objects can exist simultaneously (no global state).

`dbTypes.h` defines types returned by database class member functions.

All database objects are in the `odb` namespace.

- `dbChip`
- `dbBlock`
- `dbTech`
- `dbLib`

All database objects have a 32bit object identifier accessed with the `dbObject::getOID` base class member function that returns a `uint`. This identifier is preserved across save/restores of the database so it should be used to reference database object by data structures instead of pointers if the reference lifetime is across database save/restores. OIDs allow the database to have exactly the same layout across save/restores.

The database distance units are **nanometers** and use the type `uint`.

Create Physical Cluster

Description TBC.

```
create_physical_cluster cluster_name
```

Options

Switch Name	Description
`cluster_name`	Name of cluster.

Create Child Physical Clusters

Description TBC.

```
create_child_physical_clusters
    [-top_module]
or
create_child_physical_clusters
    [-modinst path]
```

Options

Switch Name	Description
top_module	TBC.
-modinst	TBC.

Set NDR Layer Rule

Description TBC.

```
set_ndr_layer_rule
    tech
    ndr
    layerName
    input
    isSpacing
```

Options

Switch Name	Description
tech	TBC.
ndr	TBC.
values	TBC.
isSpacing	TBC.

Set NDR Rules

Description TBC.

```
set_ndr_rules
    tech
    ndr
    values
    isSpacing
```

Options

Switch Name	Description
tech	TBC.
ndr	TBC.
layerName	TBC.
input	TBC.

Create NDR

Description TBC.

```
create_ndr
    -name name
    [-spacing val]
    [-width val]
    [-via val]
```

Options

Switch Name	Description
-name	TBC.
-spacing	TBC.
-width	TBC.
-via	TBC.

Create Voltage Domain

Description TBC.

```
create_voltage_domain
    domain_name
    -area {llx lly urx ury}
```

Options

Switch Name	Description
-domain_name	TBC.
-area	TBC.

Delete Physical Cluster

Description TBC.

```
delete_physical_cluster cluster_name
```

Options

Switch Name	Description
cluster_name	TBC.

Delete Voltage Domain

Description TBC.

```
delete_voltage_domain domain_name
```

Options

Switch Name	Description
domain_name	TBC.

Assign Power Net

Description TBC.

```
assign_power_net
    -domain domain_name
    -net snet_name
```

Options

Switch Name	Description
-domain_name	TBC.
-net	TBC.

Assign Ground Net

Description TBC.

```
assign_ground_net
    -domain domain_name
    -net snet_name
```

Options

Switch Name	Description
`-domain_name`	TBC.
`-net`	TBC.

Add to Physical Cluster

Description TBC.

```
add_to_physical_cluster
    [-modinst path]
    cluster_name
or
add_to_physical_cluster
    [-inst inst_name]
    cluster_name
or
add_to_physical_cluster
    [-physical_cluster cluster_name]
    cluster_name
```

Options

Switch Name	Description
`-modinst`	TBC.
`-inst`	TBC.
`-physical_cluster`	TBC.
`cluster_name`	TBC.

Remove From Physical Cluster

Description TBC.

```
remove_from_physical_cluster
    [-parent_module module_name]
    [-modinst modinst_name]
    cluster_name
or
remove_from_physical_cluster
    [-inst inst_name]
    cluster_name
or
remove_from_physical_cluster
    [-physical_cluster cluster_name]
    cluster_name
```

Options

Switch Name	Description
-parent_module	TBC.
-modinst	TBC.
-inst	TBC.
-physical_cluster	TBC.
-cluster_name	TBC.

Report Physical Clusters

Description TBC.

```
report_physical_clusters
```

Report Voltage Domains

Description TBC.

```
report_voltage_domains
```

Report Group

Description TBC.

```
report_group group
```

Options

Switch Name	Description
group	TBC.

Write Guides

This command writes global routing guides, which can be used as input for global routing.

Example: `write_guides route.guide`.

```
write_guides file_name
```

Options

Switch Name	Description
`file_name`	Guide file name.

Write Macro Placement

This command writes macro placement.

```
write_macro_placement file_name
```

Options

Switch Name	Description
`file_name`	Macro placement file name.

Example scripts

After building successfully, run OpenDB Tcl shell using `../../build/src/odb/src/swig/tcl/odbtcl`. An example usage:

```
set db [dbDatabase_create]
set lef_parser [new_lefin $db true]
set tech [lefin_createTech $lef_parser ./src/odb/test/data/gscl45nm.lef]
```

You can find examples on using the API from Tcl under `test/tcl/` directory.

The full set of the Tcl commands exposed can be found under `./build/src/swig/tcl/opendb_wrapper.cpp`. Search for `SWIG_prefix`.

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Database Internals

The internal description included here is paraphrased from Lukas van Ginneken by James Cherry.

The database separates the implementation from the interface, and as a result, each class becomes two classes, a public one and a private one. For instance, `dbInst` has the public API functions, while class `_dbInst` has the private data fields.

The objects are allocated in dynamically resizable tables, the implementation of which is in `dbTable.hpp`. Each table consists of a number of pages, each containing 128 objects. The table contains the body of the `struct`, not a set of pointers. This eliminates most of the pointer overhead while iteration is accomplished by stepping through the table.

Thus, grouping these objects does not require a doubly-linked list and saves 16 bytes per object (at the cost of some table overhead). Each object has an id, which is the index into the table. The lowest 7 bits are the index in the page, while the higher bits are the page number. Object id's are persistent when saving and reading the data model to disk, even as pointer addresses may change.

Everything in the data model can be stored on disk and restored from disk exactly the way it was. An extensive set of equality tests and diff functions make it possible to check for even the smallest deviation. The capability to save an exact copy of the state of the system makes it possible to create a checkpoint. This is a necessary capability for debugging complex systems.

The code follows the definition of LEF and DEF closely and reflects many of the idiosyncrasies of LEF and DEF. The code defines many types of objects to reflect LEF and DEF constructs although it sometimes uses different terminology, for instance, the object to represent a library cell is called `dbMaster` while the LEF keyword is MACRO.

The data model supports the EEQ and LEQ keywords (i.e., electrically equivalent and logically equivalent Masters), which could be useful for sizing. However, it does not support any logic function representation. In general, there is very limited support for synthesis-specific information: no way to represent busses, no way to represent logic function, very limited understanding of signal flow, limited support of timing information, and no support for high level synthesis or test insertion.

The db represents routing as in DEF, representing a trace from point to point with a given width. The layout for a net is stored in a class named `dbWire` and it requires a special `dbWireDecoder` (which works like an iterator) to unpack the data and another `dbWireEncoder` to pack it. The data model does not support a region query and objects that are in the same layer are scattered about the data model and are of different classes.

This means that whatever tool is using the layout information will have to build its own data structures that are suitable to the layout operations of that tool. For instance, the router, the extractor, and the DRC engine would each have to build their unique data structures. This encourages batch mode operation (route the whole chip, extract the whole chip, run DRC on the whole chip).

Limitations

FAQs

Check out GitHub discussion about this tool.

LICENSE

BSD 3-Clause License. See LICENSE file.

Automatic Code Generator

This is an automatic code generation tool for OpenDB objects and Iterators. To test the tool you can use the following command

```
python3 gen.py --json schema.json --src_dir ../db --include_dir ../../include/odb --
↪templates templates
```

Where schema.json is the json file that includes the requirements, src is the source files directory, include is the include directory, and templates is the directory including the jinja templates for the classes.

Empty sections are removed by default from the output. If you need to add someting to a section that is currently empty, you can run the generator with –keep_empty to preserve them. Once the section is filled in, the flag can be dropped and the code regnerated to remove the remaining empty sections.

Python Unit Tests

Running tests

For running the python unit tests you will need to install first *testtools* and *unittest-parallel* which enables concurrent unit testing

```
pip3 install testtools
pip3 install unittest-parallel
```

Then, you can run the unit tests in sequence by running

```
../unitTests.sh
```

or in parallel by running

```
../unitTests.sh parallel
```

Note: The test cases within each Unit Test run in parallel in both situations

---

Tests Structure

The directory unitTestsPython includes unit tests for OpenDB Python APIs. Any test file starts with 'Test' followed by the test target.

odbUnitTest.py:

This includes `TestCase` class which inherits from `unittest.TestCase` with additional functionalities:

- `changeAndTest(self,obj,SetterName,GetterName,expectedVal,*args)`which is a function for changing a value and testing for the effect of that change where:
  - `obj` is the object to be tested
  - `SetterName` is the name of the function to be called for changing a value
  - `GetterName` is the name of the function to be called for testing the effect
  - `expectedVal` is the expected value for the testing
  - `*args` are the arguments passed to the `SetterName` function

  So, in the end, the expected behavior is:

```
obj.SetterName(*args)

assert(obj.GetterName()==expectedVal)
```

- `check(self,obj,GetterName,expectedVal,*args)` which tests against expected value
- `change(self,obj,SetterName,*args)` which changes a value in the object
- `main()` runs the `TestCase` in sequential order
- `mainParallel(Test)` runs the passed `Test` class in parallel

helper.py:

A set of functions for creating simple db instances to be used for testing. You can find the description of each function in the comments

TestNet.py:

Unit test class for testing dbNet. It inherits from `odbUnitTest.TestCase` . it consists of

- `setUp(self)` function to be called before each test case. Here, we create the database with the desired chip, block, masters, instances and nets.
- `tearDown(self)` function to be called after each test case. Here, we destroy our db.
- `test_*(self)` the test cases functions. Their names should start with `test` for the unittest suite to recognize.

TestDestroy.py:

Integration test class for testing the `destroy(*args)` function on OpenDB.

- `test_destroy_net` destroying net and testing for the effect on the *block,inst, iterms and bterms*
- `test_destroy_inst` destroying instance and testing for the effect on *block, iterms, net, bterms*
- `test_destroy_bterm` destroying bterm and testing for the effect on *block and net*
- `test_destroy_block` destroying block and testing for the effect on *block(parent and child relation), and chip*
- `test_destroy_bpin` destroying bpin and testing for the effect on *bterm*
- `test_create_destroy_wire` destroying wire and test for the effect on *net*
- `test_destroy_capnode` destroying capnode and test for the effect on *net(node and connected ccsegs)*
- `test_destroy_ccseg` destroying ccseg and test for the effect on *node,block and net*
- `test_destroy_lib` destroying lib and test for the effect on *db*
- `test_destroy_obstruction` destroying obstruction and test for the effect on *block*
- `test_create_regions` creating regions and test for the effect on *block and region(parent and child relation)*
- `test_destroy_region_child` destroying _ and test for the effect on *block and region(parent)*
- `test_destroy_region_parent` destroying _ and test for the effect on *block*

TestBlock.py:

Unit Test for dbBlock

- `test_find` testing the find function with *BTerm, Child, Inst, Net, ITerm, ExtCornerBlock, nonDefaultRule, Region*
- Testing the ComputeBBox() function through the first call of getBBox:
  - `test_bbox0` testing empty block box
  - `test_bbox1` testing block box with Inst placed
  - `test_bbox2` testing block box with Inst and BPin placed
  - `test_bbox3` testing block box with Inst, BPin and Obstruction placed

  - `test_bbox3` testing block box with Inst, BPin, Obstruction and SWire placed

TestBTerm.py:

Unit Test for dbBTerm

- `test_idle` testing for idle disconnected `BTerm` behavior
- `test_connect` testing connect function of `BTerm` on `BTerm` and `Net`
- `test_disconnect` testing disconnect function of `BTerm` on `BTerm` and `Net`

TestInst.py:

Unit Test for dbInst

- `test_swap_master` testing swap master function

TestITerm.py:

Unit Test for dbITerm

- `test_idle` testing for disconnected ITerm without a net
- `test_connection_from_iterm` testing the connect(ITerm,…) and disconnect functions of ITerm and their effect on ITerm and Net
- `test_connection_from_inst` testing the connect(Inst,…) and disconnect functions of ITerm and their effect on ITerm and Net
- Testing for getAvgXY() function
  - `test_avgxy_R0` testing with default orientation R0
  - `test_avgxy_R90` testing with different orientation R90 for transformation

---

Problems Found In Testing

- multiple core dumps that leads to aborting the process:
  - dbNet.get1st*() (when nothing on top of the list)
  - childRegion.getParent() (after destroying the parent region)
- Implementation of ComputeBBox() is flawed and needs to be reconsidered

Adding new fields in DB Object

For example `add_pitchDiag` in object `DbTechLayer`.

	Action	File	Source Code
1	Add Fields at the .h file	dbTechLayer.h	
2	Define a keyword for db rev number	dbDatabase.h	`#define ADS_DB_DF58 52`
3	Set the current rev number same as	dbDatabase.h	`#define ADS_DB_SCHEMA_MINOR 52`
4	Stream in new fields Conditionally upon Schema number	dbTechLayer.cpp	`if ( stream.getDatabase()->isSchema(ADS_DB_DF58) ) { stream >> layer._pitchDiag;`
5	Stream out new fields Conditionally upon Schema number	dbTechLayer.cpp	`if ( stream.getDatabase()->isSchema(ADS_DB_DF58) ) { stream << layer._pitchDiag;`
6	Conditionally Diff new fields	dbTechLayer.cpp	`if ( stream.getDatabase()->isSchema(ADS_DB_DF58) ) { DIFF_FIELD(_pitchDiag);`
7	Conditionally Diff Out new fields	dbTechLayer.cpp	`if ( stream.getDatabase()->isSchema(ADS_DB_DF58) ) { DIFF_OUT_FIELD(_pitchDiag);`
8	Created access APIs to the fields	dbTechLayer.cpp	`"dbTechLayer::getPitchDiag(), dbTechLayer::setPitchDiag( int pitch )"`
9	Add new APIs in include/db.h	db.h	`class dbTechLayer`

5.2.7 Graphical User Interface

The graphical user interface can be access by launching OpenROAD with `-gui` or by opening it from the command-line with `gui::show`.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Add Buttons to the Toolbar

This command creates toolbar button with name set using the `-text` flag and accompanying logic in the `-script` flag.

Returns: name of the new button, either `name` or `buttonX`.

```
create_toolbar_button
    [-name name]
    -text button_text
    -script tcl_script
    [-echo]
```

Options

Switch Name	Description
`-name`	The name of the button, used when deleting the button.
`-text`	The text to put on the button.
`-script`	The tcl script to evaluate when the button is pressed.
`-echo`	This indicate that the commands in the `tcl_script` should be echoed in the log.

Remove Toolbar Button

To remove toolbar button:

```
gui::remove_toolbar_button
    name
```

Options

Switch Name	Description
`-name`	The name of the button, used when deleting the button.

Add items to the Menubar

This command add items to the menubar. Returns: name of the new item, either `name` or `actionX`.

```
create_menu_item
    [-name name]
    [-path menu_path]
    -text item_text
    -script tcl_script
    [-shortcut key_shortcut]
    [-echo]
```

Options

Switch Name	Description
`-name`	(optional) name of the item, used when deleting the item.
`-path`	(optional) Menu path to place the new item in (hierarchy is separated by /), defaults to "Custom Scripts", but this can also be "Tools" or "New menu/New submenu".
`-text`	The text to put on the item.
`-script`	The tcl script to evaluate when the button is pressed.
`-shortcut`	(optional) key shortcut to trigger this item.
`-echo`	(optional) indicate that the commands in the `tcl_script` should be echoed in the log.

Remove items from the Menubar

To remove menu item:

```
gui::remove_menu_item
    name
```

Options

Switch Name	Description
`-name`	name of the item, used when deleting the item.

Save Image

This command can be both be used when the GUI is active and not active to save a screenshot with various options.

```
save_image
    [-resolution microns_per_pixel]
    [-area {x0 y0 x1 y1}]
    [-width width]
    [-display_option {option value}]
    filename
```

Options

Switch Name	Description
`filename`	path to save the image to.
`-area`	x0, y0 - first corner of the layout area (in microns) to be saved, default is to save what is visible on the screen unless called when gui is not active and then it selected the whole block. x1, y1 - second corner of the layout area (in microns) to be saved, default is to save what is visible on the screen unless called when gui is not active and then it selected the whole block.
`-resolution`	resolution in microns per pixel to use when saving the image, default will match what the GUI has selected.
`-width`	width of the output image in pixels, default will be computed from the resolution. Cannot be used with `-resolution`.
`-display_option`	specific setting for a display option to show or hide specific elements. For example, to hide metal1 `-display_option {Layers/metal1 false}`, to show routing tracks `-display_option {Tracks/Pref true}`, or to show everthing `-display_option {* true}`.

Save Clocktree Image

This command saves the screenshot of clocktree given options to `filename`.

```
save_clocktree_image
    filename
    -clock clock_name
    [-width width]
    [-height height]
    [-corner corner]
```

Options

Switch Name	Description
`filename`	path to save the image to.
`-clock`	name of the clock to save the clocktree for.
`-corner`	name of the timing corner to save the clocktree for, default to the first corner defined.
`-height`	height of the image in pixels, defaults to the height of the GUI widget.
`-width`	width of the image in pixels, defaults to the width of the GUI widget.

Select Objects

This command selects object based on options. Returns: number of objects selected.

```
select
    -type object_type
    [-name glob_pattern]
    [-filter attribute=value]
    [-case_insensitive]
    [-highlight group]
```

Options

Switch Name	Description
`-type`	name of the object type. For example, `Inst` for instances, `Net` for nets, and `DRC` for DRC violations.
`-name`	(optional) filter selection by the specified name. For example, to only select clk nets `*clk*`. Use `-case_insensitive` to filter based on case insensitive instead of case sensitive.
`-filter`	(optional) filter selection based on the objects' properties. `attribute` represents the property's name and `value` the property's value. In case the property holds a collection (e. g. BTerms in a Net) or a table (e. g. Layers in a Generate Via Rule) `value` can be any element within those. A special case exists for checking whether a collection is empty or not by using the value `CONNECTED`. This can be useful to select a specific group of elements (e. g. BTerms=CONNECTED will select only Nets connected to Input/Output Pins).
`-highlight`	(optional) add the selection to the specific highlighting group. Values can be 0 to 7.

Display Timing Cones

This command displays timing cones for a pin given options.

```
display_timing_cone
    pin
    [-fanin]
    [-fanout]
    [-off]
```

Options

Switch Name	Description
`pin`	name of the instance or block pin.
`-fanin`	(optional) display the fanin timing cone.
`-fanout`	(optional) display the fanout timing cone.
`-off`	(optional) remove the timing cone.

Focus Net

This command limits the drawing to specified net.

```
focus_net
    net
    [-remove]
    [-clear]
```

Options

Switch Name	Description
`pin`	name of the net.
`-remove`	(optional) removes the net from from the focus.
`-clear`	(optional) clears all nets from focus.

TCL functions

Is GUI Enabled

Determine is the GUI is active:

```
gui::enabled
```

Trigger GUI to Load Design

Announce to the GUI that a design was loaded (note: this is only needed when the design was loaded through the odb API and not via `read_def` or `read_db`):

```
gui::design_created
```

Load DRC Result

To load the results of a DRC report:

```
gui::load_drc
    filename
```

Options

Switch Name	Description
`filename`	Filename for DRC report.

Show GUI

To open the GUI from the command-line (this command does not return until the GUI is closed):

```
gui::show
    script
    interactive
```

Options

Switch Name	Description
`script`	TCL script to evaluate in the GUI.
`interactive`	Boolean if true, the GUI should open in an interactive session (default), or if false that the GUI would execute the script and return to the terminal.

Hide GUI

To close the GUI and return to the command-line:

```
gui::hide
```

Layout Fit

To fit the whole layout in the window:

```
gui::fit
```

Zoom to a specific region

To zoom in our out to a specific region:

```
gui::zoom_to
    x0 y0 x1 y1
```

Options

Switch Name	Description
`x0, y0, x1, y1`	first and second corner of the layout area in microns.

Zoom In

To zoom in the layout:

```
gui::zoom_in
    x y
```

Options

Switch Name	Description
`x, y`	new center of layout in microns.

Zoom Out

To zoom out the layout:

```
gui::zoom_out
    x y
```

Options

Switch Name	Description
`x, y`	new center of layout in microns.

Center At

To move the layout to new area:

```
gui::center_at
    x y
```

Options

Switch Name	Description
`x, y`	new center of layout in microns.

Set Resolution

To change the resolution to a specific value:

```
gui::set_resolution
    resolution
```

Options

Switch Name	Description
`resolution`	database units per pixel.

Add a single net to selection

To add a single net to the selected items:

```
gui::selection_add_net
    name
```

Options

Switch Name	Description
`name`	name of the net to add.

Add multiple nets to selection

To add several nets to the selected items using a regex:

```
gui::selection_add_nets
    name_regex
```

Options

Switch Name	Description
`name_regex`	regular expression of the net names to add.

Add a single inst to selection

To add a single instance to the selected items:

```
gui::selection_add_inst
    name
```

Options

Switch Name	Description
`name`	name of the instance to add.

Add multiple insts to selection

To add several instances to the selected items using a regex:

```
gui::selection_add_insts
    name_regex
```

Options

Switch Name	Description
`name_regex`	regular expression of the instance names to add.

Select at point or area

To add items at a specific point or in an area:

Example usage:

```
gui::select_at x y
gui::select_at x y append
gui::select_at x0 y0 x1 y1
gui::select_at x0 y0 x1 y1 append
```

```
gui::select_at
    x0 y0 x1 y1
    [append]

Or

gui::select_at
    x y
    [append]
```

Options

Switch Name	Description
`x, y`	point in the layout area in microns.
`x0, y0, x1, y1`	first and second corner of the layout area in microns.
`append`	if `true` (the default value) append the new selections to the current selection list, else replace the selection list with the new selections.

Select next item from selection

To navigate through multiple selected items: Returns: current index of the selected item.

```
gui::select_next
```

Select previous item from selection

To navigate through multiple selected items: Returns: current index of the selected item.

```
gui::select_previous
```

Clear Selection

To clear the current set of selected items:

```
gui::clear_selections
```

Get Selection Property

To get the properties for the current selection in the Inspector:

```
gui::get_selection_property
    name
```

Options

Switch Name	Description
`name`	name of the property. For example, `Type` for object type or `bbox` for the bounding box of the object.

Animate Selection

To animate the current selection in the Inspector:

```
gui::selection_animate
    [repeat]
```

Options

Switch Name	Description
`repeat`	indicate how many times the animation should repeat, default value is 0 repeats. If the value is 0, the animation will repeat indefinitely.

Highlight Net

To highlight a net:

```
gui::highlight_net
    name
    [highlight_group]
```

Options

Switch Name	Description
`name`	name of the net to highlight.
`highlight_group`	group to add the highlighted net to, defaults to `0`, valid groups are `0 - 7`.

Highlight Instance

To highlight an instance:

```
gui::highlight_inst
    name
    [highlight_group]
```

Options

Switch Name	Description
`name`	name of the instance to highlight.
`highlight_group`	group to add the highlighted instance to, defaults to `0`, valid groups are `0 - 7`.

Clear Highlight Groups

To clear the highlight groups:

```
gui::clear_highlights
    [highlight_group]
```

Options

Switch Name	Description
`highlight_group`	group to clear, defaults to `0`, valid groups are `-1 - 7`. Use `-1` to clear all groups.

Add Ruler to Layout

To add a ruler to the layout:

1. either press `k` and use the mouse to place it visually. To disable snapping for the ruler when adding, hold the `Ctrl` key, and to allow non-horizontal or vertical snapping when completing the ruler hold the `Shift` key.
2. or use the command:

Returns: name of the newly created ruler.

```
gui::add_ruler
    x0 y0 x1 y1
    [label]
    [name]
    [euclidian]
```

Options

Switch Name	Description
`x0, y0, x1, y1`	first and second end point of the ruler in microns.
`label`	text label for the ruler.
`name`	name of the ruler.
`euclidian`	`1` for euclidian ruler, and `0` for regular ruler.

Delete a single ruler

To remove a single ruler:

```
gui::delete_ruler
    name
```

Options

Switch Name	Description
`name`	name of the ruler.

Clear All Rulers

To remove all the rulers:

```
gui::clear_rulers
```

Set Heatmap

To control the settings in the heat maps:

The currently availble heat maps are:

- `Power`
- `Routing`
- `Placement`
- `IRDrop`
- `RUDY`[1]

These options can also be modified in the GUI by double-clicking the underlined display control for the heat map.

```
gui::set_heatmap
    name
    [option]
    [value]
```

Options

Switch Name	Description
`name`	is the name of the heatmap.
`option`	is the name of the option to modify. If option is `rebuild` the map will be destroyed and rebuilt.
`value`	is the new value for the specified option. This is not used when rebuilding map.

Dump Heatmap to file

To save the raw data from the heat maps ins a comma separated value (CSV) format:

```
gui::dump_heatmap
    name
    filename
```

Options

Switch Name	Description
`name`	is the name of the heatmap.
`filename`	path to the file to write the data to.

[1] RUDY means Rectangular Uniform wire DensitY, which can predict the routing density very rough and quickly. You can see this notion in this paper

GUI Display Controls

Control the visible and selected elements in the layout:

```
gui::set_display_controls
    name
    [display_type]
    [value]
```

Options

Switch Name	Description
`name`	is the name of the control. For example, for the power nets option this would be `Signals/Power` or could be `Layers/*` to set the option for all the layers.
`display_type`	is either `visible` or `selectable`
`value`	is either `true` or `false`

Check Display Controls

To check the visibility or selectability of elements in the layout:

```
gui::check_display_controls
    name
    display_type
```

Options

Switch Name	Description
`name`	is the name of the control. For example, for the power nets option this would be `Signals/Power` or could be `Layers/*` to set the option for all the layers.
`display_type`	is either `visible` or `selectable`

Save Display Controls

When performing a batch operation changing the display controls settings, the following command can be used to save the current state of the display controls.

```
gui::save_display_controls
```

Restore Display Controls

This command restores display controls.

```
gui::restore_display_controls
```

Input Dialog

To request user input via the GUI: Returns: a string with the input, or empty string if canceled.

```
gui::input_dialog
    title
    question
```

Options

Switch Name	Description
`title`	is the title of the input message box.
`question`	is the text for the message box.

Pause script execution

Pause the execution of the script:

```
gui::pause
    [timeout]
```

Options

Switch Name	Description
`timeout`	is specified in milliseconds, if it is not provided the pause will last until the user presses the Continue button.

Show widget

To open a specific layout widget:

```
gui::show_widget
      name
```

Options

Switch Name	Description
`name`	of the widget. For example, the display controls would be “Display Control”.

Hide widget

To close a specific layout widget:

```
gui::hide_widget
    name
```

Options

Switch Name	Description
`name`	of the widget. For example, the display controls would be “Display Control”.

License

BSD 3-Clause License. See LICENSE file.

5.2.8 Partition Manager

The partitioning module (`par`) is based on TritonPart, an open-source constraints-driven partitioner. `par` can be used to partition a hypergraph or a gate-level netlist.

Highlights

- Start of the art multiple-constraints driven partitioning “multi-tool”
- Optimizes cost function based on user requirement
- Permissive open-source license
- Solves multi-way partitioning with following features:
  - Multidimensional real-value weights on vertices and hyperedges
  - Multilevel coarsening and refinement framework
  - Fixed vertices constraint
  - Timing-driven partitioning framework
  - Group constraint: Groups of vertices need to be in same block
  - Embedding-aware partitioning

Dependency

We use Google OR-Tools as our ILP solver.

Our recommendation is to follow the OpenROAD *DependencyInstaller* for installation of this requirement.

Alternatively, you may also install Google OR-Tools following these instructions.

> **Warning:** Due to a build issue, TritonPart is not supported for macOS. Stay tuned to this page for updates!

Main Algorithm

An overview of the TritonPart algorithm is shown below. It takes as inputs

- Hypergraph $H(V, E)$ in `.hgr` format.
- Vertex weight $w_v \in \mathcal{R}_{+}^{m}$
- Hyperedge weight $w_e \in \mathcal{R}_{+}^{n}$
- Number of blocks K.
- Imbalance factor ϵ.
- User-specified cost function ϕ.

There are five main steps in the main algorithm, mainly 1) constraints-driven coarsening, 2) initial partitioning, 3) refinement, 4) cut-overlay clustering and partitioning (COCP), and 5) V-cycle refinement. The steps for the timing-aware algorithm may be found in the next *section*.

1. Constraints-Driven Coarsening

The first step involves multilevel coarsening. Specifically, at each level, clusters of vertices are identified, and the merged and represented as a single vertex in the resulting coarser hypergraph. In this algorithm, the First-Choice scheme is used, which traverses the vertices in the hypergraph according to a given ordering and merges pairs of vertices with high connectivity. The connectivity between a pair of vertices (u, v) is measured as follows:

$$r(u,v) = \sum_{e \in \{I(v) \cap I(u)\}} \frac{\langle \alpha, w_e \rangle}{|e| - 1}$$

To efficiently manage multiple constraints, the following enhancements are made to the coarsening scheme above:

- **Fixed Vertex Constraint**: Fixed vertices that belong to the same partitioning block are merged into a single vertex.
- **Grouping Constraint**: Vertices that belong to the same group are merged into a single vertex.
- **Embedding Constraint**: The embedding information is incorporated into the heavy-edge rating function. The new connectivity is updated as follows:

$$\hat{r}(u,v) = r(u,v) + \rho \frac{1}{||X_u - X_v||_2}$$

where ρ is a normalization factor set to the average distance between two vertex embeddings. When vertices $v_1, ..., v_t$ are merged into a single vertex v_{coarse}, the corresponding vertex embedding $X_{v_{coarse}}$ is defined as the *center of gravity* of t vertices:

$$X_{v_{coarse}} = \sum_{j=1}^{t} \frac{||w_{v_j}||}{M} X_{v_j},\ where\ M = \sum_{j=1}^{t} ||w_{v_j}||$$

- **Community Guidance**: Only vertices within the same community are considered for merging.
- **Tie-breaking mechanism**: If multiple neighbor pairs have the same rating score, combine the lexicographically first unmatched vertex to break ties.

2. Initial Partitioning

After completing the coarsening process, an initial partitioning solution for the coarsest hypergraph H_c is derived. Two sub-steps are involved in this: the best partitioning solution from random and VILE partitioning is chosen from $\eta = 50$ runs as a warm-start to the ILP-based partitioner. The optimization is based on only the cut size rather than the cost function ϕ to keep the runtime reasonable.

3. Refinement

After a feasible solution H_{c_ξ} is obtained by initial partitioning, uncoarsening and move-based refinement is performed to improve the partitioning solution. Three refinement heuristics are applied in sequence:

- **K-way pairwise FM (PM)**: This addresses multi-way partitioning as concurrent bi-partitioning problems in a restricted version of K-way Fiduccia–Mattheyses (FM) algorithm. First, $\lfloor K/2 \rfloor$ pairs of blocks are obtained, with refinement-specific vertex movements restricted to associated paired blocks. Next, two-way FM is concurrently performed on all the block pairs. finally, a new configuration of block pairs is computed at the end of the PM.
- **Direct K-way FM**: Using K priority queues, for each block V_i, establish a priority queue that stores the vertices that can be potentially moved from the current block to block V_i. This queue is ordered according to the gain of the vertices. Gain is defined as the reduction in cost function from the movement of the vertex from the current block to V_i. Next, after a vertex move, each priority queue is updated independently, thus enabling parallel updates via multi-threading. Next, early-stop is implemented by limiting the maximum number of vertices moved to 100 per pass. Finally, the *corking effect* is mitigated by traversing the priority queue belonging to the vertex with the highest gain and identifying a feasible vertex move.
- **Greedy Hyperedge Refinement (HER)**: First, randomly visit all hyperedges. For each hyperedge e that crosses the partition boundary, determine whether a subset of vertices in e can be moved without violating the multi-dimensional balance constraints. The objective is to make e entirely constrained in a block.

4. Cut-Overlay Clustering and Partitioning (COCP)

Cut-overlay Clustering and Partitioning (COCP) is a mechanism to combine multiple good-quality partitioning solutions to generate an improved solution. To begin, the sets of hyperedges cut in the θ candidate solutions are denoted as $E_1, ..., E_\theta \subset E$. First, $\cup_{i=1}^{\theta} E_i$ is removed from the hypergraph $H(V, E)$, resulting in a number of connected components. Next, all vertices within each connected component are merged to form a coarser hypergraph $H_{overlay}$. If the number of vertices in $H_{overlay}$ is less than thr_{ilp}, ILP-based partitioning is performed. If not, a single round of constraints-driven coarsening is conducted to further reduce the size of $H_{overlay}$ and generate a coarser hypergraph $H^{'}_{overlay}$. Finally, multilevel refinement is performed to further improve the partitioning solution at each level of the hierarchy and return the improved solution $S^{'}$.

5. V-Cycle Refinement

Cut-overlay clustering and partitioning produces a high-quality partitioning solution $S^{'}$. To improve it, there are three phases similar to *hMETIS*, namely multilevel coarsening, ILP-based partitioning, and refinement. Firstly, in multilevel partitioning, $S^{'}$ is used as a community guidance for the constraints-driven coarsening. Only vertices within the same block are permitted to be merged to ensure that the current solution $S^{'}$ is preserved in the coarsest hypergraph H_{c_ξ}. In the ILP-based partitioning phase, if the number of vertices in H_{c_ξ} does not exceed thr_{ilp}, run ILP-based partitioning to improve $S^{'}$. Otherwise, continue with $S^{'}$ in successive iterations of these two steps (default set to 2). The refinement phase is conducted as per step 3.

TritonPart algorithm at a glance

Timing Aware Algorithm

`par` can also be used as a timing-aware partitioning framework. A slack propagation methodology is used that optimizes cuts for both timing-critical and timing-noncritical paths.

1. Extraction of Timing Paths and Slack Information

First, the top P timing-critical paths and the slack information of each hyperedge using the wireload model (WLM) is obtained from *OpenSTA*. The timing cost of each path is then calculated:

$$t_p = (1 - \frac{slack_p - \Delta}{clock\_period})^{\mu}$$

where a fixed extra delay Δ is introduced for timing guardband, and μ (default 2) is the exponent.

The snaking factor of a path $SF(p)$ is defined as the maximum number of block reentries along the path p. The timing cost of a hyperedge is computed using the timing weight corresponding to hyperedge slack $slack_e$ and the accumulated timing cost of all paths traversing the hyperedge.

$$t_e = (1 - \frac{slack_e - \Delta}{clock\_period})^{\mu} + \sum_{\{p|e \in p\}} t_p$$

2. Timing-aware Coarsening

The timing-aware feature is achieved by adding a timing cost of hyperedge t_e to the connectivity score earlier mentioned. If vertices (u, v) are associated with multiple critical paths, then they are more likely to be merged. This is reflected in the update score function:

$$r_t(u, v) = \hat{r}(u, v) + \sum_{e \in \{I(v) \cap I(u)\}} \frac{\beta t_e}{|e| - 1}$$

3. Timing-aware Refinement

Timing-aware refinement is based on a similar cost function as the main algorithm. Instead, an additional slack propagation step is performed at the end of each PM/FM/HER pass.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Partition Hypergraph Netlist

This command performs hypergraph netlist partitioning.

```
triton_part_hypergraph
    -hypergraph_file hypergraph_file
    -num_parts num_parts
    -balance_constraint balance_constraint
    [-base_balance base_balance]
    [-scale_factor scale_factor]
```

(continues on next page)

(continued from previous page)

```
    [-seed seed]
    [-vertex_dimension vertex_dimension]
    [-hyperedge_dimension hyperedge_dimension]
    [-placement_dimension placement_dimension]
    [-fixed_file fixed_file]
    [-community_file community_file]
    [-group_file group_file]
    [-placement_file placement_file]
    [-e_wt_factors e_wt_factors]
    [-v_wt_factors <v_wt_factors>]
    [-placement_wt_factors <placement_wt_factors>]
    [-thr_coarsen_hyperedge_size_skip thr_coarsen_hyperedge_size_skip]
    [-thr_coarsen_vertices thr_coarsen_vertices]
    [-thr_coarsen_hyperedges thr_coarsen_hyperedges]
    [-coarsening_ratio coarsening_ratio]
    [-max_coarsen_iters max_coarsen_iters]
    [-adj_diff_ratio adj_diff_ratio]
    [-min_num_vertices_each_part min_num_vertices_each_part]
    [-num_initial_solutions num_initial_solutions]
    [-num_best_initial_solutions num_best_initial_solutions]
    [-refiner_iters refiner_iters]
    [-max_moves max_moves]
    [-early_stop_ratio early_stop_ratio]
    [-total_corking_passes total_corking_passes]
    [-v_cycle_flag v_cycle_flag ]
    [-max_num_vcycle max_num_vcycle]
    [-num_coarsen_solutions num_coarsen_solutions]
    [-num_vertices_threshold_ilp num_vertices_threshold_ilp]
    [-global_net_threshold global_net_threshold]
```

Options

Switch Name	Description
`-num_parts`	Number of partitions. The default value is `2`, and the allowed values are integers `[0, MA`
`-balance_constraint`	Allowed imbalance between blocks. The default value is `1.0`, and the allowed values are
`-base_balance`	Tcl list of baseline imbalance between partitions. The default value is `{1.0}`, and the allo
`-scale_factor`	KIV. The default value is `{1.0}`, and the allowed values are floats that sum up to `1.0`.
`-seed`	Random seed. The default value is `0`, and the allowed values are integers `[-MAX_INT, M`
`-vertex_dimension`	Number of vertices in the hypergraph. The default value is `1`, and the allowed values are
`-hyperedge_dimension`	Number of hyperedges in hypergraph. The default value is `1`, and the allowed values are
`-placement_dimension`	Number of dimensions for canvas if placement information is provided. The default value
`-hypergraph_file`	Path to hypergraph file.
`-fixed_file`	Path to fixed vertices constraint file.
`-community_file`	Path to `community` attributes file to guide the partitioning process.
`-group_file`	Path to `stay together` attributes file.
`-placement_file`	Placement information file, each line corresponds to a group fixed vertices, community, a
`-e_wt_factors`	Hyperedge weight factor.
`-v_wt_factors`	Vertex weight factors.
`-placement_wt_factors`	Placement weight factors.

Table 1 – continued from

Switch Name	Description
`-thr_coarsen_hyperedge_size_skip`	Threshold for ignoring large hyperedge (default 200, integer).
`-thr_coarsen_vertices`	Number of vertices of coarsest hypergraph (default 10, integer).
`-thr_coarsen_hyperedges`	Number of vertices of coarsest hypergraph (default 50, integer).
`-coarsening_ratio`	Coarsening ratio of two adjacent hypergraphs (default 1.6, float).
`-max_coarsen_iters`	Number of iterations (default 30, integer).
`-adj_diff_ratio`	Minimum difference of two adjacent hypergraphs (default 0.0001, float).
`-min_num_vertices_each_part`	Minimum number of vertices in each partition (default 4, integer).
`-num_initial_solutions`	Number of initial solutions (default 50, integer).
`-num_best_initial_solutions`	Number of top initial solutions to filter out (default 10, integer).
`-refiner_iters`	Refinement iterations (default 10, integer).
`-max_moves`	The allowed moves for each Fiduccia-Mattheyes (FM) algorithm pass or greedy refineme
`-early_stop_ratio`	Describes the ratio e where if the $n_{movedvertices} > n_{vertices} * e$, the tool exits the curren
`-total_corking_passes`	Maximum level of traversing the buckets to solve the “corking effect” (default 25, integer
`-v_cycle_flag`	Disables v-cycle is used to refine partitions (default true, bool).
`-max_num_vcycle`	Maximum number of `vcycles` (default 1, integer).
`-num_coarsen_solutions`	Number of coarsening solutions with different randoms seed (default 3, integer).
`-num_vertices_threshold_ilp`	Describes threshold t, the number of vertices used for integer linear programming (ILP) p
`-global_net_threshold`	If the net is larger than this, it will be ignored by TritonPart (default 1000, integer).

Evaluate Hypergraph Partition

This command evaluates hypergraph partition.

```
evaluate_hypergraph_solution
  -num_parts num_parts
  -balance_constraint balance_constraint
  -hypergraph_file hypergraph_file
  -solution_file solution_file
  [-base_balance base_balance]
  [-scale_factor scale_factor]
  [-vertex_dimension vertex_dimension]
  [-hyperedge_dimension hyperedge_dimension]
  [-fixed_file fixed_file]
  [-group_file group_file]
  [-e_wt_factors e_wt_factors]
  [-v_wt_factors v_wt_factors]
```

Options

Switch Name	Description
`-num_parts`	Number of partitions. The default value is `2`, and the allowed values are integers `[0, MAX_INT]`.
`-balance_constraint`	Allowed imbalance between blocks. The default value is `1.0`, and the allowed values are floats.
`-vertex_dimension`	Number of vertices in the hypergraph. The default value is `1`, and the allowed values are integers `[0, MAX_INT]`.
`-hyperedge_dimension`	Number of hyperedges in hypergraph. The default value is `1`, and the allowed values are integers `[0, MAX_INT]`.
`-hypergraph_file`	Path to hypergraph file.
`-solution_file`	Path to solution file.
`-base_balance`	Tcl list of baseline imbalance between partitions. The default value is `{1.0}`, and the allowed values are floats that sum up to `1.0`.
`-scale_factor`	KIV. The default value is `{1.0}`, and the allowed values are floats that sum up to `1.0`.
`-fixed_file`	Path to fixed vertices constraint file.
`-group_file`	Path to `stay together` attributes file.
`-e_wt_factors`	Hyperedge weight factor.
`-v_wt_factors`	Vertex weight factor.

Partition Netlist

This command partitions the design netlist. Note that design must be loaded in memory.

```
triton_part_design
    [-num_parts num_parts]
    [-balance_constraint balance_constraint]
    [-base_balance base_balance]
    [-scale_factor scale_factor]
    [-seed seed]
    [-timing_aware_flag timing_aware_flag]
    [-top_n top_n]
    [-placement_flag placement_flag]
    [-fence_flag fence_flag]
    [-fence_lx fence_lx]
    [-fence_ly fence_ly]
    [-fence_ux fence_ux]
    [-fence_uy fence_uy]
    [-fixed_file fixed_file]
    [-community_file community_file]
    [-group_file group_file]
    [-solution_file solution_file]
    [-net_timing_factor net_timing_factor]
    [-path_timing_factor path_timing_factor]
    [-path_snaking_factor path_snaking_factor]
    [-timing_exp_factor timing_exp_factor]
    [-extra_delay extra_delay]
    [-guardband_flag guardband_flag]
    [-e_wt_factors e_wt_factors]
    [-v_wt_factors v_wt_factors]
```

(continues on next page)

(continued from previous page)

```
    [-placement_wt_factors placement_wt_factors]
    [-thr_coarsen_hyperedge_size_skip thr_coarsen_hyperedge_size_skip]
    [-thr_coarsen_vertices thr_coarsen_vertices]
    [-thr_coarsen_hyperedges thr_coarsen_hyperedges]
    [-coarsening_ratio coarsening_ratio]
    [-max_coarsen_iters max_coarsen_iters]
    [-adj_diff_ratio adj_diff_ratio]
    [-min_num_vertices_each_part min_num_vertices_each_part]
    [-num_initial_solutions num_initial_solutions]
    [-num_best_initial_solutions num_best_initial_solutions]
    [-refiner_iters refiner_iters]
    [-max_moves max_moves]
    [-early_stop_ratio early_stop_ratio]
    [-total_corking_passes total_corking_passes]
    [-v_cycle_flag v_cycle_flag ]
    [-max_num_vcycle max_num_vcycle]
    [-num_coarsen_solutions num_coarsen_solutions]
    [-num_vertices_threshold_ilp num_vertices_threshold_ilp]
    [-global_net_threshold global_net_threshold]
```

Options

Switch Name	Description
`-num_parts`	Number of partitions. The default value is `2`, and the allowed values are integers `[0, MA`
`-balance_constraint`	Allowed imbalance between blocks. The default value is `1.0`, and the allowed values are
`-base_balance`	Tcl list of baseline imbalance between partitions. The default value is `{1.0}`, and the allo
`-scale_factor`	KIV. The default value is `{1.0}`, and the allowed values are floats that sum up to `1.0`.
`-seed`	Random seed. The default value is `1`, and the allowed values are integers `[-MAX_INT, M`
`-timing_aware_flag`	Enable timing-driven mode. The default value is `true`, and the allowed values are boolea
`-top_n`	Extract the top n critical timing paths. The default value is `1000`, and the allowed values
`-placement_flag`	Enable placement driven partitioning. The default value is `false`, and the allowed values
`-fence_flag`	Consider fences in the partitioning. The default value is `false`, and the allowed values a
`-fence_lx`	Fence lower left x in microns. The default value is `0.0`, and the allowed values are floats.
`-fence_ly`	Fence lower left y in microns. The default value is `0.0`, and the allowed values are floats.
`-fence_ux`	Fence upper right x in microns. The default value is `0.0`, and the allowed values are float
`-fence_uy`	Fence upper right y in microns. The default value is `0.0`, and the allowed values are float
`-fixed_file`	Path to fixed vertices constraint file
`-community_file`	Path to `community` attributes file to guide the partitioning process.
`-group_file`	Path to `stay together` attributes file.
`-solution_file`	Path to solution file.
`-net_timing_factor`	Hyperedge timing weight factor (default 1.0, float).
`-path_timing_factor`	Cutting critical timing path weight factor (default 1.0, float).
`-path_snaking_factor`	Snaking a critical path weight factor (default 1.0, float).
`-timing_exp_factor`	Timing exponential factor for normalized slack (default 1.0, float).
`-extra_delay`	Extra delay introduced by a cut (default 1e-9, float).
`-guardband_flag`	Enable timing guardband option (default false, bool).
`-e_wt_factors`	Hyperedge weight factor.
`-v_wt_factors`	Vertex weight factor.

Table 2

Switch Name	Description
`-placement_wt_factors`	Placement weight factor.
`-thr_coarsen_hyperedge_size_skip`	Threshold for ignoring large hyperedge. The default value is `1000`, and the allowed value
`-thr_coarsen_vertices`	Number of vertices of coarsest hypergraph. The default value is `10`, and the allowed valu
`-thr_coarsen_hyperedges`	Number of vertices of the coarsest hypergraph. The default value is `50`, and the allowed v
`-coarsening_ratio`	Coarsening ratio of two adjacent hypergraphs. The default value is `1.5`, and the allowed
`-max_coarsen_iters`	Number of iterations. The default value is `30`, and the allowed values are integers `[0, MA`
`-adj_diff_ratio`	Minimum ratio difference of two adjacent hypergraphs. The default value is `0.0001`, and
`-min_num_vertices_each_part`	Minimum number of vertices in each partition. The default value is `4`, and the allowed va
`-num_initial_solutions`	Number of initial solutions. The default value is `100`, and the allowed values are integers
`-num_best_initial_solutions`	Number of top initial solutions to filter out. The default value is `10`, and the allowed valu
`-refiner_iters`	Refinement iterations. The default value is `10`, and the allowed values are integers `[0, M`
`-max_moves`	The allowed moves for each Fiduccia-Mattheyes (FM) algorithm pass or greedy refineme
`-early_stop_ratio`	Describes the ratio e where if the $n_{movedvertices} > n_{vertices} * e$, the tool exists the curre
`-total_corking_passes`	Maximum level of traversing the buckets to solve the "corking effect". The default value
`-v_cycle_flag`	Disables v-cycle is used to refine partitions. The default value is `true`, and the allowed v
`-max_num_vcycle`	Maximum number of vcycles. The default value is `1`, and the allowed values are integers
`-num_coarsen_solutions`	Number of coarsening solutions with different randoms seed. The default value is `4`, and
`-num_vertices_threshold_ilp`	Describes threshold t, the number of vertices used for integer linear programming (ILP) p
`-global_net_threshold`	If the net is larger than this, it will be ignored by TritonPart. The default value is `1000`, an

Evaluate Netlist Partition

This command evaluates partition design solution.

```
evaluate_part_design_solution
    [-num_parts num_parts]
    [-balance_constraint balance_constraint]
    [-base_balance base_balance]
    [-scale_factor scale_factor]
    [-timing_aware_flag timing_aware_flag]
    [-top_n top_n]
    [-fence_flag fence_flag]
    [-fence_lx fence_lx]
    [-fence_ly fence_ly]
    [-fence_ux fence_ux]
    [-fence_uy fence_uy]
    [-fixed_file fixed_file]
    [-community_file community_file]
    [-group_file group_file]
    [-hypergraph_file hypergraph_file]
    [-hypergraph_int_weight_file hypergraph_int_weight_file]
    [-solution_file solution_file]
    [-net_timing_factor net_timing_factor]
    [-path_timing_factor path_timing_factor]
    [-path_snaking_factor path_snaking_factor]
    [-timing_exp_factor timing_exp_factor]
    [-extra_delay extra_delay]
    [-guardband_flag guardband_flag]
    [-e_wt_factors e_wt_factors]
    [-v_wt_factors v_wt_factors]
```

Options

Switch Name	Description
-num_parts	Number of partitions. The default value is 2, and the allowed values are integers [0, MAX_INT].
-balance_constraint	Allowed imbalance between blocks. The default value is 1.0, and the allowed values are floats.
-base_balance	Tcl list of baseline imbalance between partitions. The default value is {1.0}, and the allowed values are floats that sum up to 1.0.
-scale_factor	KIV. The default value is {1.0}, and the allowed values are floats that sum up to 1.0.
-timing_aware_flag	Enable timing-driven mode. The default value is true, and the allowed values are booleans.
-top_n	Extract the top n critical timing paths. The default value is 1000, and the allowed values are integers [0, MAX_INT].
-fence_flag	Consider fences in the partitioning. The default value is false, and the allowed values are booleans.
-fence_lx	Fence lower left x in microns. The default value is 0.0, and the allowed values are floats.
-fence_ly	Fence lower left y in microns. The default value is 0.0, and the allowed values are floats.
-fence_ux	Fence upper right x in microns. The default value is 0.0, and the allowed values are floats.
-fence_uy	Fence upper right y in microns. The default value is 0.0, and the allowed values are floats.
-fixed_file	Path to fixed vertices constraint file.
-community_file	Path to community attributes file to guide the partitioning process.
-group_file	Path to stay together attributes file.
-hypergraph_file	Path to hypergraph file.
-hypergraph_int_weight_file	Path to hMETIS format integer weight file.
-solution_file	Path to solution file.
-net_timing_factor	Hyperedge timing weight factor. The default value is 1.0, and the allowed values are floats.
-path_timing_factor	Cutting critical timing path weight factor. The default value is 1.0, and the allowed values are floats.
-path_snaking_factor	Snaking a critical path weight factor. The default value is 1.0, and the allowed values are floats.
-timing_exp_factor	Timing exponential factor for normalized slack. The default value is 1.0, and the allowed values are floats.
-extra_delay	Extra delay introduced by a cut. The default value is 1e-9, and the allowed values are floats.
-guardband_flag	Enable timing guardband option. The default value is 1false, and the allowed values are booleans.
-e_wt_factors	Hyperedge weight factors.
-v_wt_factors	Vertex weight factors.

Write Partition to Verilog

This command writes the partition result to verilog.

```
write_partition_verilog
    [-port_prefix prefix]
    [-module_suffix suffix]
    [-partitioning_id part_id]
    [file]
```

Options

Switch Name	Description
-port_prefix	Port name prefix.
-module_suffix	Module name suffix.
file	Filename to write partition verilog to.

Read the Partition file

This command reads the partition file into design.

```
read_partitioning
    -read_file name
    [-instance_map_file file_path]
```

Switch Name	Description
-read_file	Read partitioning file (usually with the extension .part). The file format must match the same format as the output of write_partition_verilog.
-instance_map_file	Instance mapping file.

Example Scripts

How to partition a hypergraph in the way you would using hMETIS (min-cut partitioning)

```
triton_part_hypergraph -hypergraph_file des90.hgr -num_parts 5 -balance_constraint 2 -
↪seed 2
```

You can also check the provided example here.

How to perform the embedding-aware partitioning

```
set num_parts 2
set balance_constraint 2
set seed 0
set design sparcT1_chip2
set hypergraph_file "${design}.hgr"
set placement_file "${design}.hgr.ubfactor.2.numparts.2.embedding.dat"
set solution_file "${design}.hgr.part.${num_parts}"

triton_part_hypergraph  -hypergraph_file $hypergraph_file -num_parts $num_parts \
                        -balance_constraint $balance_constraint \
                        -seed $seed  \
                        -placement_file ${placement_file} -placement_wt_factors { 0.
↪00005 0.00005 } \
                        -placement_dimension 2

```

You can find the provided example here.

How to partition a netlist

```
# set technology information
set ALL_LEFS “list_of_lefs”
set ALL_LIBS “list_of_libs”
# set design information
set design “design_name”
set top_design “top_design”
set netlist “netlist.v”
set sdc “timing.sdc”
foreach lef_file ${ALL_LEFS} {
  read_lef $lef_file
}
foreach lib_file ${ALL_LIBS} {
  read_lib $lib_file
}
read_verilog $netlist
link_design $top_design
read_sdc $sdc

set num_parts 5
set balance_constraint 2
set seed 0
set top_n 100000
# set the extra_delay_cut to 20% of the clock period
# the extra_delay_cut is introduced for each cut hyperedge
set extra_delay_cut 9.2e-10
set timing_aware_flag true
set timing_guardband true
set part_design_solution_file "${design}_part_design.hgr.part.${num_parts}"

##########################################################################################
```

(continues on next page)

(continued from previous page)

```
↪#####
## TritonPart with slack progagation
#########################################################################################
↪#####
puts "Start TritonPart with slack propagation"
# call triton_part to partition the netlist
triton_part_design -num_parts $num_parts -balance_constraint $balance_constraint \
                   -seed $seed -top_n $top_n \
                   -timing_aware_flag $timing_aware_flag -extra_delay $extra_delay_cut \
                   -guardband_flag $timing_guardband \
                   -solution_file $part_design_solution_file
```

You can find the provided example here.

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

References

1. Bustany, I., Kahng, A. B., Koutis, I., Pramanik, B., & Wang, Z. (2023). K-SpecPart: A Supervised Spectral Framework for Multi-Way Hypergraph Partitioning Solution Improvement. arXiv preprint arXiv:2305.06167. (.pdf)
2. Bustany, I., Gasparyan, G., Kahng, A. B., Koutis, I., Pramanik, B., & Wang, Z. (2023). "An Open-Source Constraints-Driven General Partitioning Multi-Tool for VLSI Physical Design", Proc. ACM/IEEE International Conference of Computer-Aided Design 2023,(.pdf).

License

BSD 3-Clause License. See LICENSE file.

5.2.9 Restructure

The restructure module in OpenROAD (`rmp`) is based on an interface to ABC for local resynthesis. The package allows logic restructuring that targets area or timing. It extracts a cloud of logic using the OpenSTA timing engine, and passes it to ABC through `blif` interface. Multiple recipes for area or timing are run to obtain multiple structures from ABC; the most desirable among these is used to improve the netlist. The ABC output is read back by a `blif` reader which is integrated to OpenDB. `blif` writer and reader also support constants from and to OpenDB. Reading back of constants requires insertion of tie cells which should be provided by the user as per the interface described below.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Restructure

Restructuring can be done in two modes: area or delay.

- Method 1: Area Mode Example: `restructure -liberty_file ckt.lib -target area -tielo_pin ABC -tiehi_pin DEF`
- Method 2: Timing Mode Example: `restructure -liberty_file ckt.lib -target delay -tielo_pin ABC -tiehi_pin DEF -slack_threshold 1 -depth_threshold 2`

```
restructure
    [-slack_threshold slack_val]
    [-depth_threshold depth_threshold]
    [-target area|delay]
    [-abc_logfile logfile]
    [-liberty_file liberty_file]
    [-tielo_port  tielo_pin_name]
    [-tiehi_port tiehi_pin_name]
    [-work_dir work_dir]
```

Options

Switch Name	Description
`-liberty_file`	Liberty file with description of cells used in design. This is passed to ABC.
`-target`	Either `area` or `delay`. In area mode, the focus is area reduction, and timing may degrade. In delay mode, delay is likely reduced, but the area may increase. The default value is `area`.
`-slack_threshold`	Specifies a (setup) timing slack value below which timing paths need to be analyzed for restructuring. The default value is `0`, and the allowed values are floats `[0, MAX_FLOAT]`.
`-depth_threshold`	Specifies the path depth above which a timing path would be considered for restructuring. The default value is `16`, and the allowed values are `[0, MAX_INT]`.
`-tielo_pin`	Tie cell pin that can drive constant zero. The format is `<cell>/<port>`.
`-tiehi_pin`	Tie cell pin that can drive constant one. The format is `<cell>/<port>`.
`-abc_logfile`	Output file to save abc logs to.
`-work_dir`	Name of the working directory for temporary files. If not provided, `run` directory would be used.

Example scripts

Example scripts on running `rmp` for a sample design of `gcd` as follows:

```
./test/gcd_restructure.tcl
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

Authors

- Sanjiv Mathur
- Ahmad El Rouby

License

BSD 3-Clause License. See *LICENSE* file.

5.2.10 Initialize Floorplan

This tool initializes floorplan constraints, die/core area, and makes tracks.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Initialize Floorplan

Do note that there are two ways of setting the floorplan dimensions. The user can either specify manually die/core area, or specify the utilization/aspect ratio. If you set both, unexpected behaviour might occur.

- Method 1: Automatic die size calculation Example: `initialize_floorplan -utilization 70 -aspect_ratio 1.0 -core_space 0.0 -sites FreePDK45_38x28_10R_NP_162NW_34O`
- Method 2: Set die/core area Example: `initialize_floorplan -die_area 0 0 2000 2000 -core_area 100 100 1900 1900` -sites FreePDK45_38x28_10R_NP_162NW_34O`

The die area and core area used to write ROWs can be specified explicitly with the `-die_area` and `-core_area` arguments. Alternatively, the die and core areas can be computed from the design size and utilization as shown below:

The -site argument determines the basic single-height rows to make. For a hybrid row design the site should have a row pattern. Any sites referenced by an instance in the netlist will also have rows constructed for them.

Additional sites are specified if you wish to have rows for a site that is not used in the nelist but may be needed later. For example, you might not have any double height cells in the incoming netlist but you expect some to be generated by flop clustering later in the flow.

Example computation:

```
core_area = design_area / (utilization / 100)
core_width = sqrt(core_area / aspect_ratio)
core_height = core_width * aspect_ratio
core = ( core_space_left, core_space_bottom )
      ( core_space_left + core_width, core_space_bottom + core_height )
die =  ( 0, 0 )
      ( core_width + core_space_left + core_space_right,
        core_height + core_space_bottom + core_space_top )
```

```
initialize_floorplan
  [-utilization util]
  [-aspect_ratio ratio]
  [-core_space space | {bottom top left right}]
  [-die_area {llx lly urx ury}]
  [-core_area {llx lly urx ury}]
  [-additional_sites site_names]
  [-site site_name]
```

Options

Switch Name	Description
`-utilization`	Percentage utilization. Allowed values are `double` in the range `(0-100]`.
`-aspect_ratio`	Ratio $\frac{height}{width}$. The default value is `1.0` and the allowed values are floats `[0, 1.0]`.
`-core_space`	Space around the core, default `0.0` microns. Allowed values are either one value for all margins or a set of four values, one for each margin. The order of the four values are: `{bottom top left right}`.
`-additional_sites`	List of sites to make rows for (e.g. `{SITEXX, SITEYY}`)
`-site`	Site name.
`-die_area`	Die area coordinates in microns (lower left x/y and upper right x/y coordinates).
`-core_area`	Core area coordinates in microns (lower left x/y and upper right x/y coordinates).

Make Tracks

The initialize_floorplan command removes existing tracks. Use the make_tracks command to add routing tracks to a floorplan.

```
make_tracks
    [layer]
    [-x_pitch x_pitch]
    [-y_pitch y_pitch]
    [-x_offset x_offset]
    [-y_offset y_offset]
```

Options

Switch Name	Description
layer	Select layer name to make tracks for. Defaults to all layers.
-x_pitch, -y_pitch	If set, overrides the LEF technology x-/y- pitch. Use the same unit as in the LEF file.
-x_offset, -y_offset	If set, overrides the LEF technology x-/y- offset. Use the same unit as in the LEFT file.

Insert tieoff cells

This comamnd inserts tiecells.

```
insert_tiecells
    tie_pin
    [-prefix inst_prefix]
```

Options

Switch Name	Description
tie_pin	Indicates the master and port to use to tie off nets. For example, LOGIC0_X1/Z for the Nangate45 library, where LOGIC0_X1 is the master and Z is the output port on the master.
-prefix	Used to control the prefix of the new tiecell names. This will default to TIEOFF_.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the *swig file*.

Command Name	Description
microns_to_mfg_grid	Convert microns to manufacturing grid DBU.

Example scripts

Example scripts on running `ifp` for a sample design of `mpd_top` are as follows:

```
./test/upf_test.tcl
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

License

BSD 3-Clause License. See *LICENSE* file.

5.2.11 Pin Placer

Place pins on the boundary of the die on the track grid to minimize net wirelengths. Pin placement also creates a metal shape for each pin using min-area rules.

For designs with unplaced cells, the net wirelength is computed considering the center of the die area as the unplaced cells position.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Define Pin Shape Pattern

The `define_pin_shape_pattern` command defines a pin placement grid on the specified layer. This grid has positions inside the die area, not only at the edges of the die boundary.

```
define_pin_shape_pattern
    [-layer layer]
    [-x_step x_step]
    [-y_step y_step]
    [-region {llx lly urx ury} | *]
    [-size {width height}]
    [-pin_keepout dist]
```

Options

Switch Name	Description
`-layer`	The single top-most routing layer of the placement grid.
`-x_step`, `-y_step`	The distance (in microns) between each valid position on the grid in the x- and y-directions, respectively.
`-region`	The `{llx, lly, urx, ury}` region of the placement grid (in microns). If the `*` is specified, the region will be the entire die area.
`-size`	The width and height (in microns) of the pins assigned to this grid. The centers of the pins are placed on the grid positions. Pins may have half of their shapes outside the defined region.
`-pin_keepout`	The boundary (in microns) around existing routing obstructions that the pins should avoid; this defaults to the `layer` minimum spacing.

Face-to-Face direct-bonding IOs

The `define_pin_shape_pattern` command can be used to place pins in any metal layer with the minimum allowed spacing to facilitate 3DIC integration of chips using face-to-face packaging technologies. These technologies include micro bumps and hybrid bonding for high density face-to-face interconnect.

Set IO Pin Constraints

The `set_io_pin_constraint` command sets region constraints for pins according to the pin direction or the pin name. This command can be called multiple times with different constraints.

You can use the `set_io_pin_constraint` command to restrict pins to the pin placement grid created with the `define_pin_shape_pattern` command.

It is possible to use the `-region`, `-group` and `-order` arguments together per `set_io_pin_constraint` call, but the `-mirrored_pins` argument should be called alone.

```
set_io_pin_constraint
    [-direction direction]
    [-pin_names names]
    [-region edge:interval]
    [-mirrored_pins names]
```

(continues on next page)

(continued from previous page)

```
    [-group]
    [-order]
```

Options

Switch Name	Description
-direction	Pin direction (input, output, inout, or feedthrough).
-pin_names	List of names. Only one of (-direction, -pin_names) should be used in a single call for the set_io_pin_constraint command.
-region	Syntax is -region edge:interval. The edge values are (top|bottom|left|right). The interval can be the whole edge with the wildcard * value or a range of values.
-mirrored_pins	List of pins that sets pairs of pins that will be symmetrically placed in the vertical or the horizontal edges. The number of pins in this list **must be even**. For example, in set_io_pin_constraint -mirrored_pins {pin1 pin2 pin3 pin4 pin5 pin6}, the pins pin1 and pin2 will be placed symmetrically to each other. Same for pin3 and pin4, and for pin5 and pin6.
-group	Flag places together on the die boundary the pin list defined in -pin_names, similar to the -group_pins option on the place_pins command.
-order	Flag places the pins ordered in ascending x/y position and must be used only when -group is also used.

The edge values are (up, top, bottom, left, right), where up is the grid created by define_pin_shape_pattern. To restrict pins to the pin placement grid defined with define_pin_shape_pattern use:

- -region up:{llx lly urx ury} to restrict the pins into a specific region in the grid. The region is defined in microns.
- -region up:* to restrict the pins into the entire region of the grid.

The up option is only available when the pin placement grid is created with the define_pin_shape_pattern command.

Clear IO Pin Constraints

The clear_io_pin_constraints command clears all the previously-defined constraints and pin shape patterns created with set_io_pin_constraint or define_pin_shape_pattern.

```
clear_io_pin_constraints
```

Set Pin Length

The set_pin_length command defines the length of all vertical and horizontal pins.

```
set_pin_length
    [-hor_length h_length]
    [-ver_length v_length]
```

Options

Switch Name	Description
-hor_length	The length (in microns) of the horizontal pins.
-ver_length	The length (in microns) of the vertical pins.

The default length of the pins is the minimum length necessary to respect the minimum area defined in the routing layer they were placed. The width of the pins is the minimum width defined in the routing layer.

Set Pin Length Extension

The set_pin_length_extension command defines the an extension of the length of all vertical and horizontal pins. Note that this command may generate pins partially outside the die area.

```
set_pin_length_extension
    [-hor_extension h_extension]
    [-ver_extension v_extension]
```

Options

Switch Name	Description
-hor_extension	The length (in microns) for the horizontal pins.
-ver_extension	The length (in microns) for the vertical pins.

Set Pin Thickness Multiplier

The set_pin_thick_multiplier command defines a multiplier for the thickness of all vertical and horizontal pins.

```
set_pin_thick_multiplier
    [-hor_multiplier h_mult]
    [-ver_multiplier v_mult]
```

Options

Switch Name	Description
-hor_multiplier	The thickness multiplier for the horizontal pins.
-ver_multiplier	The thickness multiplier for the vertical pins.

Set Simulated Annealing

The set_simulated_annealing command defines the parameters for simulated annealing pin placement.

```
set_simulated_annealing
    [-temperature temperature]
    [-max_iterations iter]
    [-perturb_per_iter perturbs]
    [-alpha alpha]
```

Options

Switch Name	Description
-temperature	Temperature parameter. The default value is 1.0, and the allowed values are floats [0, MAX_FLOAT].
-max_iterations	The maximum number of iterations. The default value is 2000, and the allowed values are integers [0, MAX_INT].
-perturb_per_iter	The number of perturbations per iteration. The default value is 0, and the allowed values are integers [0, MAX_INT].
-alpha	The temperature decay factor. The default value is 0.985, and the allowed values are floats (0, 1].

Simulated Annealing Debug Mode

The simulated_annealing_debug command allows you to debug the simulated annealing pin placement with a pause mode.

```
simulated_annealing_debug
    [-iters_between_paintings iters]
    [-no_pause_mode no_pause_mode]
```

Options

Switch Name	Description
-iters_between_paintings	Determines the number of iterations between updates.
-no_pause_mode	Print solver state every second based on iters_between_paintings.

Place specific Pin

The place_pin command places a specific pin in the specified location with the specified size. It is recommended that individual pins be placed before the place_pins command, as the routing tracks occupied by these individual pins will be blocked, preventing overlaps.

To place an individual pin:

```
place_pin
    -pin_name pin_name
    -layer layer
    -location {x y}
    [-pin_size {width height}]
    [-force_to_die_boundary]
    [-placed_status]
```

Options

Switch Name	Description
`-pin_name`	The name of a pin of the design.
`-layer`	The routing layer where the pin is placed.
`-location`	The center of the pin (in microns).
`-pin_size`	The width and height of the pin (in microns).
`-force_to_die_boundary`	When this flag is enabled, the pin will be snapped to the nearest routing track, next to the die boundary.

Place all Pins

The `place_pins` command places all pins together. Use the following command to perform pin placement:

Developer arguments:

- `-random`, `-random_seed`

```
place_pins
    -hor_layers h_layers
    -ver_layers v_layers
    [-random_seed seed]
    [-random]
    [-corner_avoidance length]
    [-min_distance distance]
    [-min_distance_in_tracks]
    [-exclude region]
    [-group_pins pin_list]
    [-annealing]
    [-write_pin_placement file_name]
```

Options

Switch Name	Description
`-hor_layers`	The layers to create the metal shapes of pins placed in horizontal tracks. It can be a single layer or a list of layer names.
`-ver_layers`	The layers to create the metal shapes of pins placed in vertical tracks. It can be a single layer or a list of layer names.
`-corner_avoidance`	The distance (in microns) from each corner within which pin placement should be avoided.
`-min_distance`	The minimum distance between pins on the die boundary. This distance can be in microns (default) or in number of tracks between each pin. The default value is the length of two routing tracks between each pin.
`-min_distance_in_tracks`	Flag that allows setting the min distance in number of tracks instead of microns.
`-exclude`	A region where pins cannot be placed. Either `top
`-group_pins`	A list of pins to be placed together on the die boundary.
`-annealing`	Flag to enable simulated annealing pin placement.
`-write_pin_placement`	A file with the pin placement generated in the format of multiple calls for the `place_pin` command.

The `exclude` option syntax is `-exclude edge:interval`. The `edge` values are (top|bottom|left|right). The `interval` can be the whole edge, with the `*` value, or a range of values. For example, in `place_pins -hor_layers metal2 -ver_layers metal3 -exclude top:* -exclude right:15-60.5 -exclude left:*-50` three intervals are excluded: the whole top edge, the right edge from 15 microns to 60.5 microns, and the left edge from its beginning to 50 microns.

Developer Arguments

Switch Name	Description
`-random_seed`	Specify the seed for random operations.
`-random`	When this flag is enabled, the pin placement is random.

Write Pin Placement

The `write_pin_placement` command writes a file with the pin placement in the format of multiple calls for the `place_pin` command:

```
write_pin_placement
    file_name
```

Options

Switch Name	Description
`file_name`	The name of the file with the pin placement.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

Command Name	Description
`parse_edge`	Parse edge (top/bottom/left/right).
`parse_direction`	Parse direction.
`parse_excludes_arg`	Parse excluded arguments.
`parse_group_pins_arg`	Parse group pins arguments.
`parse_layer_name`	Parse layer name.
`parse_pin_names`	Parse pin names.
`get_edge_extreme`	Get extremes of edge.
`exclude_intervals`	Set exclude interval.
`add_pins_to_constraint`	Add pins to constrained region.
`add_pins_to_top_layer`	Add pins to top layer.

Example scripts

Example scripts of `ppl` running on a sample design of `gcd` as follows:

```
./test/gcd.tcl
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

References

- This code depends on Munkres.

FAQs

Check out GitHub discussion about this tool.

License

BSD 3-Clause License. See LICENSE file.

5.2.12 Chip-level Connections

The chip-level connections module in OpenROAD (`pad`) is based on the open-source tool ICeWall. In this utility, either place an IO ring around the boundary of the chip and connect with either wirebond pads or a bump array.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Place IO Terminals

In the case where the bond pads are integrated into the padcell, the IO terminals need to be placed. This command place terminals on the padring.

Example usage:

```
place_io_terminals u_*/PAD
place_io_terminals u_*/VDD
```

```
place_io_terminals
    -allow_non_top_layer
    inst_pins
```

Options

Switch Name	Description
`-allow_non_top_layer`	Allow the terminal to be placed below the top layer.
`inst_pins`	Instance pins to place the terminals on.

Defining a Bump Array

This command defines a bump array.

Example usage:

```
make_io_bump_array -bump BUMP -origin "200 200" -rows 14 -columns 14 -pitch "200 200"
```

```
make_io_bump_array
    -bump master
    -origin {x y}
    -rows rows
    -columns columns
    -pitch {x y}
    [-prefix prefix]
```

Options

Switch Name	Description
`-bump`	Name of the bump master.
`-origin`	Origin of the array.
`-rows`	Number of rows to create.
`-columns`	Number of columns to create.
`-pitch`	Pitch of the array.
`-prefix`	Name prefix for the bump array. The default value is `BUMP_`.

Remove Entire Bump Array

This command removes the entire bump array.

Example usage:

```
remove_io_bump_array -bump BUMP
```

```
remove_io_bump_array
    -bump master
```

Options

Switch Name	Description
`-bump`	Name of the bump master.

Remove a single Bump Instance

This command removes a single bump instance.

```
remove_io_bump
    instance_name
```

Options

Switch Name	Description
`instance_name`	Name of the bump.

Assign a net to IO Bump

This command assigns a net to a bump instance.

Example usage:

```
assign_io_bump -net p_ddr_addr_9_o BUMP_6_0
assign_io_bump -net p_ddr_addr_8_o BUMP_6_2
assign_io_bump -net DVSS BUMP_6_4
assign_io_bump -net DVDD BUMP_7_3
assign_io_bump -net DVDD -terminal u_dvdd/DVDD BUMP_8_3
assign_io_bump -net p_ddr_addr_7_o BUMP_7_1
assign_io_bump -net p_ddr_addr_6_o BUMP_7_0
```

```
assign_io_bump
    -net net
    [-terminal iterm]
    [-dont_route]
    instance
```

Options

Switch Name	Description
`-net`	Net to connect to.
`-terminal`	Instance terminal to route to.
`-dont_route`	Flag to indicate that this bump should not be routed, only perform assignment.
`instance`	Name of the bump.

Make IO Sites

This command defines an IO site for the pads to be placed into.

Example usage:

```
make_io_sites -horizontal_site IOSITE_H -vertical_site IOSITE_V -corner_site IOSITE_C -
↪offset 35
make_io_sites -horizontal_site IOSITE_H -vertical_site IOSITE_V -corner_site IOSITE_C -
↪offset 35 -rotation_horizontal R180
```

```
make_io_sites
    -horizontal_site site
    -vertical_site site
    -corner_site site
```

(continues on next page)

(continued from previous page)

```
    -offset offset
    [-rotation_horizontal rotation]
    [-rotation_vertical rotation]
    [-rotation_corner rotation]
    [-ring_index index]
```

Options

Switch Name	Description
`-horizontal_site`	Name of the site for the horizontal pads (east and west).
`-vertical_site`	Name of the site for the vertical pads (north and south).
`-corner_site`	Name of the site for the corner cells.
`-offset`	Offset from the die edge to place the rows.
`-rotation_horizontal`	Rotation to apply to the horizontal sites to ensure pads are placed correctly. The default value is `R0`.
`-rotation_vertical`	Rotation to apply to the vertical sites to ensure pads are placed correctly. The default value is `R0`.
`-rotation_corner`	Rotation to apply to the corner sites to ensure pads are placed correctly. The default value is `R0`.
`-ring_index`	Used to specify the index of the ring in case of multiple rings.

Remove IO Rows

When the padring is complete, the following command can remove the IO rows to avoid causing confusion with the other tools.

```
remove_io_rows
```

Placing Corner Cells

This command places the corner cells.

```
place_corners
    master
    [-ring_index index]
```

Options

Switch Name	Description
`master`	Name of the master for the corners.
`-ring_index`	Used to specify the index of the ring in case of multiple rings.

Example usage:

```
place_corners sky130_fd_io__corner_bus_overlay
```

Placing Pads

To place a pad into the pad ring.

Example usage:

```
place_pad -row IO_SOUTH -location 280.0 {u_clk.u_in}
place_pad -row IO_SOUTH -location 360.0 -mirror {u_reset.u_in}
place_pad -master sky130_fd_io__top_ground_hvc_wpad -row IO_SOUTH -location 439.5 {u_vzz_
↪0}
place_pad -master sky130_fd_io__top_power_hvc_wpad -row IO_SOUTH -location 517.5 {u_v18_
↪0}
```

```
place_pad
    -row row_name
    -location offset
    -mirror
    [-master master]
    name
```

Options

Switch Name	Description
`-row`	Name of the row to place the pad into, examples include: `IO_NORTH`, `IO_SOUTH`, `IO_WEST`, `IO_EAST`, `IO_NORTH_0`, `IO_NORTH_1`.
`-location`	Offset from the bottom left chip edge to place the pad at.
`-mirror`	Specifies if the pad should be mirrored.
`-master`	Name of the instance master if the instance needs to be created.
`name`	Name of the instance.

Placing IO Filler Cells

To place the IO filler cells.

Example usage:

```
place_io_fill -row IO_NORTH s8iom0s8_com_bus_slice_10um s8iom0s8_com_bus_slice_5um␣
↪s8iom0s8_com_bus_slice_1um
place_io_fill -row IO_SOUTH s8iom0s8_com_bus_slice_10um s8iom0s8_com_bus_slice_5um␣
↪s8iom0s8_com_bus_slice_1um
place_io_fill -row IO_WEST s8iom0s8_com_bus_slice_10um s8iom0s8_com_bus_slice_5um␣
↪s8iom0s8_com_bus_slice_1um
place_io_fill -row IO_EAST s8iom0s8_com_bus_slice_10um s8iom0s8_com_bus_slice_5um␣
↪s8iom0s8_com_bus_slice_1um
```

```
place_io_fill
    -row row_name
    [-permit_overlaps masters]
    masters
```

Options

Switch Name	Description
-row	Name of the row to place the pad into, examples include: IO_NORTH, IO_SOUTH, IO_WEST, IO_EAST, IO_NORTH_0, IO_NORTH_1.
-permit_overlaps	Names of the masters for the IO filler cells that allow for overlapping.
masters	Names of the masters for the IO filler cells.

Connecting Ring Signals

Once the ring is complete, use the following command to connect the ring signals.

```
connect_by_abutment
```

Place Wirebond Pads

To place the wirebond pads over the IO cells.

Example usage:

```
place_bondpad -bond PAD IO_*
```

```
place_bondpad
    -bond master
    [-offset {x y}]
    [-rotation rotation]
    io_instances
```

Options

Switch Name	Description
-bond	Name of the bondpad master.
-offset	Offset to place the bondpad at with respect to the io instance.
-rotation	Rotation of the bondpad.
io_instances	Names of the instances to add bond pads to.

Make False IO Site

If the library does not contain sites for the IO cells, the following command can be used to add them. This should not be used unless the sites are not in the library.

Example usage:

```
make_fake_io_site -name IO_HSITE -width 1 -height 204
make_fake_io_site -name IO_VSITE -width 1 -height 200
make_fake_io_site -name IO_CSITE -width 200 -height 204
```

```
make_fake_io_site
    -name name
    -width width
    -height height
```

Options

Switch Name	Description
-name	Name of the site.
-width	Width of the site (in microns).
-height	Height of the site (in microns).

Redistribution Layer Routing

To route the Redistribution Layer (RDL) for the bump arrays.

```
rdl_route
    -layer layer
    [-bump_via access_via]
    [-pad_via access_via]
    [-width width]
    [-spacing spacing]
    [-turn_penalty penalty]
    [-allow45]
    nets
```

Options

Switch Name	Description
`-layer`	Layer to route on.
`-bump_via`	Via to use to to connect the bump to the routing layer.
`-pad_via`	Via to use to to connect the pad cell to the routing layer.
`-width`	Width of the routing. Defaults to minimum width for each respective layer.
`-spacing`	Spacing of the routing. Defaults to minimum spacing for each respective layer.
`-turn_penalty`	Scaling factor to apply to discurage turning to allow for straighter routes. The default value is `2.0`, and the allowed values are floats.
`-allow45`	Specifies that 45 degree routing is permitted.
`nets`	Nets to route.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

Command Name	Description
`find_site`	Find site given site name.
`find_master`	Find master given master name.
`find_instance`	Find instance given instance name.
`find_net`	Find net given net name.
`assert_required`	Assert argument that is required for `cmd`
`connect_iterm`	Connect instance terminals. Required inputs are: `inst_name`, `iterm_name`, `net_name`.
`convert_tcl`	These functions read from $ICeWall::library parameters to generate a standalone Tcl script.

Example Scripts

Example scripts for running ICeWall functions can be found in `./test`.

```
./test/assign_bumps.tcl
./test/bump_array_make.tcl
./test/bump_array_remove.tcl
./test/bump_array_remove_single.tcl
./test/connect_by_abutment.tcl
./test/make_io_sites.tcl
./test/place_bondpad.tcl
./test/place_bondpad_stagger.tcl
./test/place_pad.tcl
./test/rdl_route.tcl
./test/rdl_route_45.tcl
./test/rdl_route_assignments.tcl
```

Regression Tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

License

BSD 3-Clause License. See LICENSE file.

5.2.13 Macro Placement

The macro placement module in OpenROAD (`mpl`) is based on TritonMacroPlacer, an open-source ParquetFP-based macro cell placer. The macro placer places macros/blocks honoring halos, channels and cell row "snapping". Run `global_placement` before macro placement.

Approximately $\left\lceil [\frac{numMacros}{3}]^{1.5} \right\rceil$ quadrisections of the initial placed mixed-size layout are explored and packed using ParquetFP-based annealing. The best resulting floorplan according to a heuristic evaluation function is kept.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Macro Placement

This command performs macro placement. For placement style, `corner_max_wl` means that choosing the partitions that maximise the wirelength of connections between the macros to force them to the corners. Vice versa for `corner_min_wl`.

Macros will be placed with $max(halo * 2, channel)$ spacing between macros, and between macros and the fence/die boundary. If no solutions are found, try reducing the channel/halo.

```
macro_placement
    [-halo {halo_x halo_y}]
    [-channel {channel_x channel_y}]
    [-fence_region {lx ly ux uy}]
    [-snap_layer snap_layer_number]
    [-style corner_wax_wl|corner_min_wl]
```

Options

Switch Name	Description
`-halo`	Horizontal and vertical halo around macros (microns).
`-channel`	Horizontal and vertical channel width between macros (microns).
`-fence_region`	Restrict macro placements to a region (microns). Defaults to the core area.
`-snap_layer`	Snap macro origins to this routing layer track.
`-style`	Placement style, to choose either `corner_max_wl` or `corner_min_wl`. The default value is `corner_max_wl`.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

Command Name	Description
`macro_placement_debug`	Macro placement debugging. Note that GUI must be present for this command, otherwise a segfault will occur.

Example scripts

Example scripts demonstrating how to run TritonMacroPlace on a sample design of `east_west` as follows:

```
./test/east_west.tcl
./test/east_west1.tcl
./test/east_west2.tcl
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

License

BSD 3-Clause License. See LICENSE file.

5.2.14 Hierarchical Macro Placement

A hierarchical automatic macro placer for large-scale complex IP blocks, “Hier-RTLMP”. This tool builds on the existing RTLMP (`mpl`) framework, adopting a multilevel physical planning approach that exploits the hierarchy and data flow inherent in the design RTL.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

RTL Macro Placer

This command executes the Hier-RTLMP algorithm for macro placement.

```
rtl_macro_placer
    [-max_num_macro max_num_macro]
    [-min_num_macro min_num_macro]
    [-max_num_macro max_num_macro]
    [-max_num_inst max_num_inst]
    [-min_num_inst min_num_inst]
    [-tolerance tolerance]
    [-max_num_level max_num_level]
    [-coarsening_ratio coarsening_ratio]
    [-num_bundled_ios num_bundled_ios]
    [-large_net_threshold large_net_threshold]
    [-signature_net_threshold signature_net_threshold]
    [-halo_width halo_width]
    [-halo_height halo_height]
    [-fence_lx fence_lx]
    [-fence_ly fence_ly]
    [-fence_ux fence_ux]
    [-fence_uy fence_uy]
    [-area_weight area_weight]
    [-outline_weight outline_weight]
    [-wirelength_weight wirelength_weight]
    [-guidance_weight guidance_weight]
    [-fence_weight fence_weight]
    [-boundary_weight boundary_weight]
    [-notch_weight notch_weight]
    [-macro_blockage_weight macro_blockage_weight]
    [-pin_access_th pin_access_th]
    [-target_util target_util]
```

(continues on next page)

(continued from previous page)

```
    [-target_dead_space target_dead_space]
    [-min_ar min_ar]
    [-snap_layer snap_layer]
    [-bus_planning]
    [-report_directory report_directory]
    [-write_macro_placement file_name]
```

Options

Switch Name	Description
`-max_num_macro`, `-min_num_macro`	Maximum/minimum number of macros in a cluster. The default value is `0` for both, and the allowed values are integers `[0, MAX_INT]`.
`-max_num_inst`, `-min_num_inst`	Maximum/minimum number of standard cells in a cluster. The default value is `0` for both, and the allowed values are integers `[0, MAX_INT]`.
`-tolerance`	Add a margin to the minimum and maximum number of macros/std cells in a cluster. For min, we multiply by (1 - `tol`), and for the max (1 + `tol`). This is to improve the robustness of hierarchical clustering. The allowed values are floats `[0, 1)`, and the default value is `0.1`.
`-max_num_level`	Maximum depth of physical hierarchical tree. The default value is `2`, and the allowed values are integers `[0, MAX_INT]`.
`-coarsening_ratio`	The larger the coarsening_ratio, the faster the convergence process. The allowed values are floats, and the default value is `10.0`.
`-num_bundled_ios`	Specifies the number of bundled pins for the left, right, top, and bottom boundaries. The default value is `3`, and the allowed values are integers `[0, MAX_INT]`.
`-large_net_threshold`	Ignore nets with many connections during clustering, such as global nets. The default value is `50`, and the allowed values are integers `[0, MAX_INT]`.
`-signature_net_threshold`	Minimum number of connections between two clusters to be identified as connected. The default value is `50`, and the allowed values are integers `[0, MAX_INT]`.
`-halo_width`	Horizontal/vertical halo around macros (microns). The allowed values are floats, and the default value is `0.0`.
`-fence_lx`, `-fence_ly`, `-fence_ux`, `-fence_uy`	Defines the global fence bounding box coordinates. The default values are the core area coordinates).
`-pin_access_th`	Specifies the pin access threshold value of macros. The default value is `0.0`, and the allowed values are floats [0,1].
`-target_util`	Specifies the target utilization of `MixedCluster` and has higher priority than target_dead_space. The allowed values are floats, and the default value is `0.25`.
`-target_dead_space`	Specifies the target dead space percentage, which influences the utilization of `StandardCellCluster`. The allowed values are floats, and the default value is `0.05`.
`-min_ar`	Specifies the minimum aspect ratio a, or the ratio of its width to height of a `StandardCellCluster` from $[a, \frac{1}{a}]$. The allowed values are floats, and the default value is `0.33`.
`-snap_layer`	Snap macro origins to this routing layer track. The default value is 4, and the allowed values are integers `[1, MAX_LAYER]`).
`-bus_planning`	Flag to enable bus planning. We recommend to enable bus planning for technologies with very limited routing layers such as SKY130 and GF180. As for technologies such as NanGate45 and ASAP7, we recommend to keep it disabled.
`-report_directory`	Save reports to this directory.
`-write_macro_placement`	Generates a file with the placement of the macros placed by HierRTLMP flow in the format of multiple calls for the `place_macro` command.

Simulated Annealing Weight parameters

Do note that while action probabilities are normalized to 1.0, the weights are not necessarily normalized.

Switch Name	Description
`-area_weight`	Weight for the area of current floorplan. The allowed values are floats, and the default value is `0.1`.
`-outline_weight`	Weight for violating the fixed outline constraint, meaning that all clusters should be placed within the shape of their parent cluster. The allowed values are floats, and the default value is `100.0`.
`-wirelength_weight`	Weight for half-perimeter wirelength. The allowed values are floats, and the default value is `100.0`.
`-guidance_weight`	Weight for guidance cost or clusters being placed near specified regions if users provide such constraints. The allowed values are floats, and the default value is `10.0`.
`-fence_weight`	Weight for fence cost, or how far the macro is from zero fence violation. The allowed values are floats, and the default value is `10.0`.
`-boundary_weight`	Weight for the boundary, or how far the hard macro clusters are from boundaries. Note that mixed macro clusters are not pushed, thus not considered in this cost. The allowed values are floats, and the default value is `50.0`.
`-notch_weight`	Weight for the notch, or the existence of dead space that cannot be used for placement & routing. Note that this cost applies only to hard macro clusters. The allowed values are floats, and the default value is `10.0`.
`-macro_blockage_weight`	Weight for macro blockage, or the overlapping instances of the macro. The allowed values are floats, and the default value is `10.0`.

Place Macro

Command for placement of one specific macro.

```
place_macro
    -macro_name macro_name
    -location {x y}
    [-orientation orientation]
```

Options

Switch Name	Description
`-macro_name`	The name of a macro of the design.
`-location`	The lower left corner of the macro in microns.
`-orientation`	The orientation according to odb. If nothing is specified, defaults to `R0`. We only allow `R0`, `MY`, `MX` and `R180`.

Example scripts

Example of a script demonstrating how to run `mpl2` on a sample design of `bp_fe_top` as follows:

```
./test/bp_fe_top.tcl
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

References

1. A. B. Kahng, R. Varadarajan and Z. Wang, “RTL-MP: Toward Practical, Human-Quality Chip Planning and Macro Placement”, (.pdf), Proc. ACM/IEEE Intl. Symp. on Physical Design, 2022, pp. 3-11.
2. A. B. Kahng, R. Varadarajan and Z. Wang, “Hier-RTLMP: A hierarchical automatic macro placer for large-scale complex IP blocks.”, (.pdf), arXiv preprint arXiv:2304.11761, 2023.

FAQs

Check out GitHub discussion about this tool.

License

BSD 3-Clause License. See LICENSE file.

5.2.15 Tapcell

Tapcell and endcap insertion.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Add Tapcell/Endcap

This command inserts tapcells or endcaps.

The figures below show two examples of tapcell insertion. When only the `-tapcell_master` and `-endcap_master` masters are given, the tapcell placement is similar to Figure 1. When the remaining masters are give, the tapcell placement is similar to Figure 2.

Figure 1: Tapcell insertion representation	Figure 2: Tapcell insertion around macro representation

```
tapcell
    [-tapcell_master tapcell_master]
    [-tap_prefix tap_prefix]
    [-endcap_master endcap_master]
    [-endcap_prefix endcap_prefix]
    [-distance dist]
    [-disallow_one_site_gaps]
    [-halo_width_x halo_x]
    [-halo_width_y halo_y]
    [-tap_nwin2_master tap_nwin2_master]
    [-tap_nwin3_master tap_nwin3_master]
    [-tap_nwout2_master tap_nwout2_master]
    [-tap_nwout3_master tap_nwout3_master]
    [-tap_nwintie_master tap_nwintie_master]
    [-tap_nwouttie_master tap_nwouttie_master]
    [-cnrcap_nwin_master cnrcap_nwin_master]
    [-cnrcap_nwout_master cnrcap_nwout_master]
    [-incnrcap_nwin_master incnrcap_nwin_master]
    [-incnrcap_nwout_master incnrcap_nwout_master]
    [-tbtie_cpp tbtie_cpp]
    [-endcap_cpp endcap_cpp]
    [-no_cell_at_top_bottom]
```

Options

Switch Name	Description
`-tapcell_master`	Master used as a tapcell.
`-tap_prefix`	Prefix for the tapcell instances. The default value is `TAP_`.
`-endcap_master`	Master used as an endcap.
`-endcap_prefix`	Prefix for the endcaps instances. The default value is `PHY_`.
`-distance`	Distance (in microns) between each tapcell in the checkerboard.
`-disallow_one_site_gaps`	KEV.
`-halo_width_x`	Horizontal halo size (in microns) around macros during cut rows.
`-halo_width_y`	Vertical halo size (in microns) around macros during cut rows.
`-tap_nwintie_master`	Master cell placed at the top and bottom of
`-tap_nwin2_master`	Master cell placed at the top and bottom of macros and the core area according the row orientation. This master should be smaller than `tap_nwintie_master`
`-tap_nwin3_master`	Master cell placed at the top and bottom of macros and the core area according the row orientation. This master should be smaller than `tap_nwin2_master`.
`-tap_nwouttie_master`	Master cell placed at the top and bottom of macros and the core area according the row orientation.
`-tap_nwout2_master`	Master cell placed at the top and bottom of macros and the core area according the row orientation. This master should be smaller than `tap_nwouttie_master`.
`-tap_nwout3_master`	Master cell placed at the top and bottom of macros and the core area according the row orientation
`-incnrcap_nwin_master`	Master cell placed at the corners of macros, according the row orientation.
`-incnrcap_nwout_master`	Master cell placed at the corners of macros, according the row orientation.
`-cnrcap_nwin_master`	Macro cell placed at the corners the core area according the row orientation.
`-cnrcap_nwout_master`	Macro cell placed at the corners the core area according the row orientation.
`-tbtie_cpp`	Option is deprecated.
`-endcap_cpp`	Option is deprecated.
`-no_cell_at_top_bottom`	Option is deprecated.

Cut Rows

This command cuts rows.

```
cut_rows
    [-endcap_master endcap_master]
    [-halo_width_x halo_x]
    [-halo_width_y halo_y]
```

Options

Switch Name	Description
`-endcap_master`	Master used as an endcap.
`-halo_width_x`	Horizontal halo size (in microns) around macros during cut rows.
`-halo_width_y`	Vertical halo size (in microns) around macros during cut rows.

Only adding boundary/endcap cells

Place endcaps into the design, the naming for the arguments to `place_endcaps` is based on the LEF58 `CLASS` specification foe endcaps.

```
place_endcaps
    [-corner master]
    [-edge_corner master]
    [-endcap masters]
    [-endcap_horizontal masters]
    [-endcap_vertical master]
    [-prefix prefix]
    [-left_top_corner master]
    [-right_top_corner master]
    [-left_bottom_corner master]
    [-right_bottom_corner master]
    [-left_top_edge master]
    [-right_top_edge master]
    [-left_bottom_edge master]
    [-right_bottom_edge master]
    [-left_edge master]
    [-right_edge master]
    [-top_edge masters]
    [-bottom_edge masters]
```

Options

Switch Name	Description
`-prefix`	Prefix to use for the boundary cells. Defaults to “PHY_”.
`-corner`	Master for the corner cells on the outer corners.
`-edge_corner`	Master for the corner cells on the inner corners.
`-endcap`	Master used as an endcap.
`-endcap_horizontal`	List of masters for the top and bottom row endcaps. (overrides `-endcap`).
`-endcap_vertical`	Master for the left and right row endcaps. (overrides `-endcap`).
`-left_top_corner`	Master for the corner cells on the outer top left corner. (overrides `-corner`).
`-right_top_corner`	Master for the corner cells on the outer top right corner. (overrides `-corner`).
`-left_bottom_corner`	Master for the corner cells on the outer bottom left corner. (overrides `-corner`).
`-right_bottom_corner`	Master for the corner cells on the outer bottom right corner. (overrides `-corner`).
`-left_top_edge`	Master for the corner cells on the inner top left corner. (overrides `-edge_corner`).
`-right_top_edge`	Master for the corner cells on the inner top right corner. (overrides `-edge_corner`).
`-left_bottom_edge`	Master for the corner cells on the inner bottom left corner. (overrides `-edge_corner`).
`-right_bottom_edge`	Master for the corner cells on the inner bottom right corner. (overrides `-edge_corner`).
`-left_edge`	Master for the left row endcaps. (overrides `-endcap_vertical`).
`-right_edge`	Master for the right row endcaps. (overrides `-endcap_vertical`).
`-top_edge`	List of masters for the top row endcaps. (overrides `-endcap_horizontal`).
`-bottom_edge`	List of masters for the bottom row endcaps. (overrides `-endcap_horizontal`).

Only adding Tapcells

This command is used for tapcell placement only.

```
place_tapcells
    -master tapcell_master
    -distance dist
```

Options

Switch Name	Description
-master	Master to use for the tapcells.
-distance	Distance between tapcells.

Remove Tapcells/Endcaps

This command is used for removing tapcells or endcaps based on their prefix.

```
tapcell_ripup
    -tap_prefix tap_prefix
    -endcap_prefix endcap_prefix
```

Options

Switch Name	Description
-tap_prefix	Remove tapcells with said prefix.
-endcap_prefix	Remove endcaps with said prefix.

Example scripts

You can find script examples for both 45nm and 14nm in `./etc/scripts`

```
./etc/scripts/example_14nm.tcl
./etc/scripts/example_45nm.tcl
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

License

BSD 3-Clause License. See *LICENSE* file.

5.2.16 Power Distribution Network Generator

The power distribution network (PDN) generator module in OpenROAD (`pdn`) is based on the PDNGEN tool. This utility aims to simplify the process of adding a power grid into a floorplan. The aim is to specify a small set of power grid policies to be applied to the design, such as layers to use, stripe width and spacing, then have the utility generate the actual metal straps. Grid policies can be defined over the stdcell area, and over areas occupied by macros.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Build Power Grid

Build a power grid in accordance with the information specified.

```
pdngen
    [-skip_trim]
    [-dont_add_pins]
    [-reset]
    [-ripup]
    [-report_only]
    [-failed_via_report file]
    [-verbose]
```

Options

Switch Name	Description
`-skip_trim`	Skip the metal trim step, which attempts to remove metal stubs.
`-dont_add_pins`	Prevent the creation of block pins.
`-reset`	Reset the grid and domain specifications.
`-ripup`	Ripup the existing power grid, as specified by the voltage domains.
`-report_only`	Print the current specifications.
`-failed_via_report`	Generate a report file which can be viewed in the DRC viewer for all the failed vias (ie. those that did not get built or were removed).

Define Voltage Domain

Defines a named voltage domain with the names of the power and ground nets for a region.

This region must already exist in the floorplan before referencing it with the `set_voltage_domain` command. If the `-region` argument is not supplied then region is the entire core area of the design.

Example usage:

```
set_voltage_domain -power VDD -ground VSS
set_voltage_domain -name TEMP_ANALOG -region TEMP_ANALOG -power VIN -ground VSS
set_voltage_domain -region test_domain -power VDD -ground VSS -secondary_power VREG
```

```
set_voltage_domain
    -name domain_name
    -power power_net_name
    -ground ground_net_name
    [-region region_name]
    [-secondary_power secondary_power_net]
    [-switched_power switched_power_net]
```

Options

Switch Name	Description
`-name`	Defines the name of the voltage domain. The default is “Core” or region name if provided.
`-power`	Specifies the name of the power net for this voltage domain.
`-ground`	Specifies the name of the ground net for this voltage domain.
`-region`	Specifies a region of the design occupied by this voltage domain.
`-secondary_power`	Specifies the name of the secondary power net for this voltage domain.
`-switched_power`	Specifies the name of the switched power net for switched power domains.

Define Power Grids

> **Warning:** `define_pdn_grid` is overloaded with two different signatures. Take note of the arguments when using this function!

- Method 1: General Usage Define the rules to describe a power grid pattern to be placed in the design.

Example usage:

```
define_pdn_grid -name main_grid -pins {metal7} -voltage_domain {CORE TEMP_ANALOG}
```

- Method 2: Macros Define the rules for one or more macros.

Example usage:

```
define_pdn_grid -macro -name ram          -orient {R0 R180 MX MY} -grid_over_pg_pins  -
↪starts_with POWER -pin_direction vertical
define_pdn_grid -macro -name rotated_rams -orient {E FE W FW}     -grid_over_boundary -
↪starts_with POWER -pin_direction horizontal
```

- Method 3: Modify existing power domain Modify pre-existing power domain.

Example usage:

```
define_pdn_grid -name main_grid -existing
```

```
define_pdn_grid
    [-name name]
    [-macro]
    [-existing]
    [-voltage_domains list_of_domain_names]
    [-grid_over_pg_pins|-grid_over_boundary]
    [-orient list_of_valid_orientations]
    [-instances list_of_instances]
    [-cells list_of_cells]
    [-default]
    [-halo list_of_halo_values]
    [-pins list_of_pin_layers]
    [-starts_with POWER|GROUND]
    [-obstructions list_of_layers]
    [-power_switch_cell name]
    [-power_control signal_name]
    [-power_control_network STAR|DAISY]
```

Options

Switch Name	Description
-name	Defines a name to use when referring to this grid definition.
-voltage_domains	Defines the name of the voltage domain for this grid (Default: Last domain created).
-pins	Defines a list of layers which where the power straps will be promoted to block pins.
-starts_with	Specifies whether the first strap placed will be POWER or GROUND (Default: GROUND).
-obstructions	Specify the layers to add routing blockages, in order to avoid DRC violations.
-macro	Defines the type of grid being added as a macro.
-grid_over_pg_pins, -grid_over_boundary	Place the power grid over the power ground pins of the macro. (Default True), or Place the power grid over the entire macro.
-orient	For a macro, defines a set of valid orientations. LEF orientations (N, FN, S, FS, E, FE, W and FW) can be used as well as standard geometry orientations (R0, R90, R180, R270, MX, MY, MXR90 and MYR90). Macros with one of the valid orientations will use this grid specification.
-instances	For a macro, defines a set of valid instances. Macros with a matching instance name will use this grid specification.
-cells	For a macro, defines a set of valid cells. Macros which are instances of one of these cells will use this grid specification.
-default	For a macro, specifies this is a default grid that can be overwritten.
-halo	Specifies the default minimum separation of selected macros from other cells in the design. This is only used if the macro does not define halo values in the LEF description. If 1 value is specified it will be used on all 4 sides, if two values are specified, the first will be applied to left/right sides and the second will be applied to top/bottom sides, if 4 values are specified, then they are applied to left, bottom, right and top sides respectively (Default: 0).
-obstructions	Specify the layers to add routing blockages, in order to avoid DRC violations.
-power_switch_cell	Defines the name of the coarse grain power switch cell to be used wherever the stdcell rail connects to the rest of the power grid. The mesh layers are associated with the unswitched power net of the voltage domain, whereas the stdcell rail is associated with the switched power net of the voltage domain. The placement of a power switch cell connects the unswitched power mesh to the switched power rail through a power switch defined by the `define_power_switch_cell` command.
-power_control	Defines the name of the power control signal used to control the switching of the inserted power switches.
-power_control_network	Defines the structure of the power control signal network. Choose from STAR, or DAISY. If STAR is specified, then the network is wired as a high-fanout net with the power control signal driving the power control pin on every power switch. If DAISY is specified then the power switches are connected in a daisy-chain configuration - note, this requires that the power swich defined by the `define_power_switch_cell` command defines an acknowledge pin for the switch.
-existing	Flag to enable defining for existing routing solution.

Power Switch Cell insertion

Define a power switch cell that will be inserted into a power grid

Example usage:

```
define_power_switch_cell -name POWER_SWITCH -control SLEEP -switched_power VDD -power␣
↪VDDG -ground VSS
```

```
define_power_switch_cell
    -name name
```

(continues on next page)

(continued from previous page)

```
    -control control_pin
    -power_switchable power_switchable_pin
    -power unswitched_power_pin
    -ground ground_pin
    [-acknowledge acknowledge_pin_name]
```

Options

Switch Name	Description
-name	The name of the power switch cell.
-control	The name of the power control port of the power switch cell.
-switched_power	Defines the name of the pin that outputs the switched power net.
-power	Defines the name of the pin that connects to the unswitched power net.
-ground	Defines the name of the pin that connects to the ground net.
-acknowledge	Defines the name of the output control signal of the power control switch if it has one.

Add PDN Straps/Stripes

Defines a pattern of power and ground stripes in a single layer to be added to a power grid.

Example usage:

```
add_pdn_stripe -grid main_grid -layer metal1 -followpins
add_pdn_stripe -grid main_grid -layer metal2 -width 0.17 -followpins
add_pdn_stripe -grid main_grid -layer metal4 -width 0.48 -pitch 56.0 -offset 2 -starts_
↪with GROUND
```

```
add_pdn_stripe
    -layer layer_name
    [-grid grid_name]
    [-width width_value]
    [-followpins]
    [-extend_to_core_ring]
    [-pitch pitch_value]
    [-spacing spacing_value]
    [-offset offset_value]
    [-starts_with POWER|GROUND]
    [-extend_to_boundary]
    [-snap_to_grid]
    [-number_of_straps count]
    [-nets list_of_nets]
```

Options

Switch Name	Description
-layer	Specifies the name of the layer for these stripes.
-grid	Specifies the grid to which this stripe definition will be added. (Default: Last grid defined by define_pdn_grid).
-width	Value for the width of stripe.
-followpins	Indicates that the stripe forms part of the stdcell rails, pitch and spacing are dictated by the stdcell rows, the -width is not needed if it can be determined from the cells.
-extend_to_core_ring	Extend the stripes to the core PG ring.
-pitch	Value for the distance between each power/ground pair.
-spacing	Optional specification of the spacing between power/ground pairs within a single pitch (Default: pitch / 2).
-offset	Value for the offset of the stripe from the lower left corner of the design core area.
-starts_with	Specifies whether the first strap placed will be POWER or GROUND (Default: grid setting).
-extend_to_boundary	Extend the stripes to the boundary of the grid.
-snap_to_grid	Snap the stripes to the defined routing grid.
-number_of_straps	Number of power/ground pairs to add.
-nets	Limit straps to just this list of nets.

Add Sroute Connect

The add_sroute_connect command is employed for connecting pins located outside of a specific power domain to the power ring, especially in cases where multiple power domains are present. During sroute, multi-cut vias will be added for new connections. The use of fixed vias from the technology file should be specified for the connection using the add_sroute_connect command. The use of max_rows and max_columns defines the row and column limit for the via stack.

Example:

```
add_sroute_connect  -net "VIN" -outerNet "VDD" -layers {met1 met4} -cut_pitch {200 200} -fixed_vias {M3M4_PR_M} -metalwidths {1000 1000} -metalspaces {800} -ongrid {met3 met4} -insts "temp_analog_1.a_header_0"
```

```
add_sroute_connect
    -layers list_of_2_layers
    -cut_pitch pitch_value
    [-net net]
    [-outerNet outerNet]
    [-fixed_vias list_of_vias]
    [-max_rows rows]
    [-max_columns columns]
    [-metalwidths metalwidths]
    [-metalspaces metalspaces]
    [-ongrid ongrid_layers]
    [-insts inst]
```

Options

Switch Name	Description
`-net`	The inner net where the power ring exists.
`-outerNet`	The outer net where instances/pins that need to get connected exist.
`-layers`	The metal layers for vertical stripes within inner power ring.
`-cut_pitch`	Distance between via cuts when the two layers are parallel, e.g., overlapping stdcell rails. (Default:200 200)
`-fixed_vias`	List of fixed vias to be used to form the via stack.
`-max_rows`	Maximum number of rows when adding arrays of vias. (Default:10)
`-max_columns`	Maximum number of columns when adding arrays of vias. (Default:10)
`-metalwidths`	Width for each metal layer.
`-metalspaces`	Spacing of each metal layer.
`-ongrid`	List of intermediate layers in a via stack to snap onto a routing grid.
`-insts`	List of all the instances that contain the pin that needs to get connected with power ring. (Default:nothing)

Add PDN Ring

The `add_pdn_ring` command is used to define power/ground rings around a grid region. The ring structure is built using two layers that are orthogonal to each other. A power/ground pair will be added above and below the grid using the horizontal layer, with another power/ground pair to the left and right using the vertical layer. Together these 4 pairs of power/ground stripes form a ring around the specified grid. Power straps on these layers that are inside the enclosed region are extend to connect to the ring.

Example usage:

```
add_pdn_ring -grid main_grid -layer {metal6 metal7} -widths 5.0 -spacings  3.0 -core_
↪offset 5
```

```
add_pdn_ring
    -layers layer_name
    -widths width_value|list_of_2_values
    -spacings spacing_value|list_of_2_values
    [-grid grid_name]
    [-core_offsets offset_value]
    [-pad_offsets offset_value]
    [-add_connect]
    [-extend_to_boundary]
    [-connect_to_pads]
    [-connect_to_pad_layers layers]
    [-starts_with POWER|GROUND]
    [-nets list_of_nets]
    [-ground_pads pads]
    [-power_pads pads]
```

Options

Switch Name	Description
`-layers`	Specifies the name of the layer for these stripes.
`-widths`	Value for the width of the stdcell rail.
`-spacings`	Optional specification of the spacing between power/ground pairs within a single pitch. (Default: pitch / 2).
`-grid`	Specifies the name of the grid to which this ring defintion will be added. (Default: Last grid created by `define_pdn_grid`).
`-core_offsets`	Value for the offset of the ring from the grid region.
`-pad_offsets`	When defining a power grid for the top level of an SoC, can be used to define the offset of ring from the pad cells.
`-add_connect`	Automatically add a connection between the two layers.
`-extend_to_boundary`	Extend the rings to the grid boundary.
`-connect_to_pads`	The core side of the pad pins will be connected to the ring.
`-connect_to_pad_layers`	Restrict the pad pins layers to this list.
`-starts_with`	Specifies whether the first strap placed will be POWER or GROUND (Default: grid setting).
`-nets`	Limit straps to just this list of nets.

Add PDN Connect

The `add_pdn_connect` command is used to define which layers in the power grid are to be connected together. During power grid generation, vias will be added for overlapping power nets and overlapping ground nets. The use of fixed vias from the technology file can be specified or else via stacks will be constructed using VIARULEs. If VIARULEs are not available in the technology, then fixed vias must be used.

Example usage:

```
add_pdn_connect -grid main_grid -layers {metal1 metal2} -cut_pitch 0.16
add_pdn_connect -grid main_grid -layers {metal2 metal4}
add_pdn_connect -grid main_grid -layers {metal4 metal7}

add_pdn_connect -grid ram -layers {metal4 metal5}
add_pdn_connect -grid ram -layers {metal5 metal6}
add_pdn_connect -grid ram -layers {metal6 metal7}

add_pdn_connect -grid rotated_rams -layers {metal4 metal6}
add_pdn_connect -grid rotated_rams -layers {metal6 metal7}
```

```
add_pdn_connect
    -layers list_of_two_layers
    [-grid grid_name]
    [-cut_pitch pitch_value]
    [-fixed_vias list_of_fixed_vias]
    [-dont_use_vias list_of_vias]
    [-max_rows rows]
    [-max_columns columns]
    [-ongrid ongrid_layers]
    [-split_cuts split_cuts_mapping]
```

Options

Switch Name	Description
`-layers`	Layers to be connected where there are overlapping power or overlapping ground nets.
`-grid`	Specifies the name of the grid definition to which this connection will be added (Default: Last grid created by `define_pdn_grid`).
`-cut_pitch`	When the two layers are parallel e.g. overlapping stdcell rails, specify the distance between via cuts.
`-fixed_vias`	List of fixed vias to be used to form the via stack.
`-dont_use_vias`	List or pattern of vias to not use to form the via stack.
`-max_rows`	Maximum number of rows when adding arrays of vias.
`-max_columns`	Maximum number of columns when adding arrays of vias.
`-ongrid`	List of intermediate layers in a via stack to snap onto a routing grid.
`-split_cuts`	Specifies layers to use split cuts on with an associated pitch, for example `{metal3 0.380 metal5 0.500}`.

Repairing power grid vias after detailed routing

To remove vias which generate DRC violations after detailed placement and routing use `repair_pdn_vias`.

```
repair_pdn_vias
    [-all]
    [-net net_name]
```

Options

Switch Name	Description
`-all`	Repair vias on all supply nets.
`-net`	Repair only vias on the specified net.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the *source file* or the *swig file*.

Command Name	Description
`name_cmp`	Compare 2 input strings `obj1` and `obj2` if they are equal.
`check_design_state`	Check if design is loaded.
`get_layer`	Get the layer reference of layer name.
`get_voltage_domains`	Get Tcl list of power domains in design.
`match_orientation`	Checks if a given orientation `orient` is within a list of orientations `orients`.
`get_insts`	Get Tcl list of instances.
`get_masters`	Get Tcl list of masters.
`get_one_to_two`	If a Tcl list has one element `{x}`, Tcl list `{x x}` is returned. If a Tcl list of two elements `{y y}`, list as is returned. Otherwise, for any other list lengths, error is triggered.
`get_one_to_four`	Similar logic for above function, except the logic only works for lists of length one, two and four respectively. All other list lengths triggers error.
`get_obstructions`	Get Tcl list of layers.
`get_starts_with`	If value starts with `POWER`, return 1; else if value starts with `GROUND` return 0; else return error.
`get_mterm`	Find master terminal.
`get_orientations`	Get list of valid orientations.

Example scripts

Defining a SoC power grid with pads

```
add_global_connection -net VDD -pin_pattern {^VDD$} -power
add_global_connection -net VDD -pin_pattern {^VDDPE$}
add_global_connection -net VDD -pin_pattern {^VDDCE$}
add_global_connection -net VSS -pin_pattern {^VSS$} -ground
add_global_connection -net VSS -pin_pattern {^VSSE$}

set_voltage_domain -power VDD -ground VSS

define_pdn_grid -name "Core"
add_pdn_ring -grid "Core" -layers {metal8 metal9} -widths 5.0 -spacings 2.0 -core_
↪offsets 4.5 -connect_to_pads

add_pdn_stripe -followpins -layer metal1 -extend_to_core_ring

add_pdn_stripe -layer metal4 -width 0.48 -pitch 56.0 -offset 2.0 -extend_to_core_ring
add_pdn_stripe -layer metal7 -width 1.40 -pitch 40.0 -offset 2.0 -extend_to_core_ring
add_pdn_stripe -layer metal8 -width 1.40 -pitch 40.0 -offset 2.0 -extend_to_core_ring
add_pdn_stripe -layer metal9 -width 1.40 -pitch 40.0 -offset 2.0 -extend_to_core_ring

add_pdn_connect -layers {metal1 metal4}
add_pdn_connect -layers {metal4 metal7}
add_pdn_connect -layers {metal7 metal8}
add_pdn_connect -layers {metal8 metal9}
add_pdn_connect -layers {metal9 metal10}
```

(continues on next page)

(continued from previous page)

```

pdngen
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

Currently the following assumptions are made:

1. The design is rectangular
2. The input floorplan includes the stdcell rows, placement of all macro blocks and IO pins.
3. The stdcells rows will be cut around macro placements

FAQs

Check out GitHub discussion about this tool.

License

BSD 3-Clause License. See LICENSE file.

Read UPF Utility

This module contains functionality to read, and modify information from Unified Power Format (UPF) files.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Read UPF

Sources the UPF file.

```
read_upf
    -file file
```

Options

Switch Name	Description
`-file`	Path to `.upf` file.

Write UPF

Writes the UPF file.

```
write_upf file_name
```

Options

Switch Name	Description
`file_name`	Filename argument.

Create Power Domain

This command creates power domain for a group of modules.

```
create_power_domain
    [-elements elements]
    name
```

Options

Switch Name	Description
`-elements`	List of module paths that belong this this domain OR * for top domain.
`name`	Domain name.

Create Logic Port

This command creates logic port. Direction must be specified from: in, out, inout.

```
create_logic_port
    [-direction direction]
    port_name
```

Options

Switch Name	Description
-direction	Direction of the port (in, out, inout).
port_name	Port name.

Create Power Switch

This command creates power switch.

```
create_power_switch
    [-domain domain]
    [-output_supply_port output_supply_port]
    [-input_supply_port input_supply_port]
    [-control_port control_port]
    [-on_state on_state]
    name
```

Options

Switch Name	Description
-domain	Power domain name.
-output_supply_port	Output supply port of the switch.
-input_supply_port	Input supply port of the switch.
-control_port	Control port on the switch.
-on_state	One of {state_name, input_supply_port, boolean_expression}.
name	Power switch name.

Create or Update Isolation Strategy

This command creates or update isolation strategy.

```
set_isolation
    [-domain domain]
    [-applies_to applies_to]
    [-clamp_value clamp_value]
    [-isolation_signal isolation_signal]
    [-isolation_sense isolation_sense]
```

(continues on next page)

(continued from previous page)

```
    [-location location]
    [-update]
    name
```

Options

Switch Name	Description
-domain	Power domain
-applies_to	Restricts the strategy to apply one of these (inputs, outputs, both).
-clamp_value	Value the isolation can drive (0, 1).
-isolation_signal	The control signal for this strategy.
-isolation_sense	The active level of isolation control signal.
-location	Domain in which isolation cells are placed (parent, self, fanout).
-update	Only available if using existing strategy, will error if the strategy doesn't exist.
name	Isolation strategy name.

Set Interface cell

This command sets the interface cell.

```
use_interface_cell
    [-domain domain]
    [-strategy strategy]
    [-lib_cells lib_cells]
```

Options

Switch Name	Description
-domain	Power domain name.
-strategy	Isolation strategy name.
-lib_cells	List of lib cells that could be used.

Set Domain Area

This command sets the power domain area.

```
set_domain_area
    domain_name
    -area {llx lly urx ury}
```

Options

Switch Name	Description
domain_name	Power domain name.
-area	x-/y- coordinates in microns for the lower left and upper right corners of the power domain area.

Map existing power switch

This command maps existing power switch.

```
map_power_switch
    [-switch_name_list switch_name_list]
    [-lib_cells lib_cells]
    [-port_map port_map]
```

Options

Switch Name	Description
-switch_name_list	A list of switches (as defined by create_power_switch) to map.
-lib_cells	A list of library cells that could be mapped to the power switch
-port_map	A map that associates model ports defined by create_power_switch to logical ports

Set Level Shifter

This command sets level shifter. Options coming soon.

```
set_level_shifter
    [-domain domain] \
    [-elements elements] \
    [-exclude_elements exclude_elements] \
    [-source source] \
    [-sink sink] \
    [-use_functional_equivalence use_functional_equivalence] \
    [-applies_to applies_to] \
    [-applies_to_boundary applies_to_boundary] \
    [-rule rule] \
    [-threshold threshold] \
    [-no_shift] \
    [-force_shift] \
    [-location location] \
    [-input_supply input_supply] \
    [-output_supply output_supply] \
    [-internal_supply internal_supply] \
    [-name_prefix name_prefix] \
    [-name_suffix name_suffix] \
    [-instance instance] \
    [-update] \
```

(continues on next page)

(continued from previous page)

```
    [-use_equivalence use_equivalence] \
    name
```

Options

Switch Name	Description
-domain	
-elements	
-exclude_elements	
-source	
-sink	
-use_functional_equivalence	
-applies_to	
-applies_to_boundary	
-rule	
-threshold	
-no_shift	
-force_shift	
-location	
-input_supply	
-output_supply	
-internal_supply	
-name_prefix	
-name_suffix	
-instance	
-update	
-use_equivalence	
name	

Set Domain Voltage

This command sets the voltage of a power domain.

```
set_domain_voltage
    [-domain domain] \
    [-voltage voltage]
```

Options

Switch Name	Description
`-domain`	Power domain name.
`-voltage`	Power domain voltage. The allowed values are `float`, default value is `0.0`.

Set Level Shifter Cell

This command sets the library cell used for level shifter.

```
set_level_shifter_cell
    [-level_shifter level_shifter] \
    [-cell_name cell_name] \
    [-input_port input_port] \
    [-output_port output_port]
```

Options

Switch Name	Description
`-level_shifter`	KIV.
`-cell_name`	KIV.
`-input_port`	KIV.
`-output_port`	KIV.

Example scripts

Example script demonstrating how to run `upf` related commands can be found here:

```
./test/upf_test.tcl
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

License

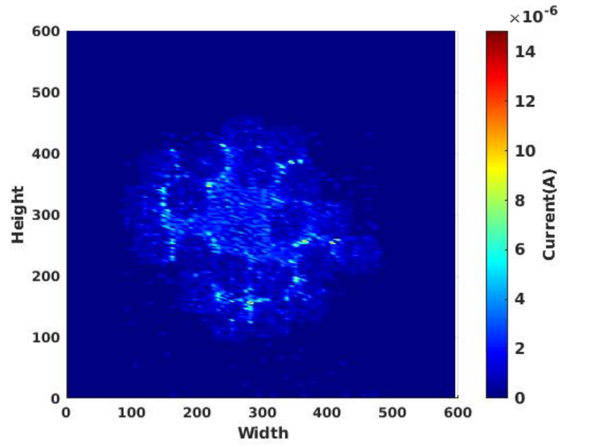
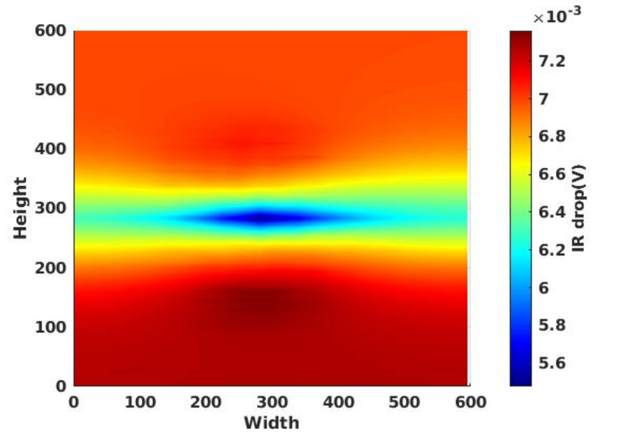
BSD 3-Clause License. See *LICENSE* file.

IR Drop Analysis

The IR Drop Analysis module in OpenROAD (`psm`) is based on PDNSim, an open-source static IR analyzer.

Features:

- Report worst IR drop.
- Report worst current density over all nodes and wire segments in the power distribution network, given a placed and PDN-synthesized design.
- Check for floating PDN stripes on the power and ground nets.
- Spice netlist writer for power distribution network wire segments.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Analyze Power Grid

This command analyzes power grid.

```
analyze_power_grid
    [-vsrc vsrc_file]
    [-outfile out_file]
    [-error_file err_file]
    [-enable_em]
    [-em_outfile em_out_file]
    [-net net_name]
    [-dx bump_pitch_x]
    [-dy bump_pitch_y]
    [-node_density val_node_density]
    [-node_density_factor val_node_density_factor]
    [-corner corner]
```

Options

Switch Name	Description
`-vsrc`	File to set the location of the power C4 bumps/IO pins. Vsrc_aes.loc file for an example with a description specified *here*.
`-dx,-dy`	These arguments set the bump pitch to decide the voltage source location in the absence of a vsrc file. Default bump pitch of 140um used in absence of these arguments and vsrc.
`-net`	Name of the net to analyze, power or ground net name.
`-enable_em`	Report current per power grid segment.
`-outfile`	Write per-instance voltage into the file.
`-em_outfile`	Write the per-segment current values into a file. This option is only available if used in combination with `-enable_em`.
`-voltage`	Sets the voltage on a specific net. If this option is not set, the Liberty file's voltage value is obtained from operating conditions.
`-node_density`	Node density (in microns) on the standard cell rails. It cannot be used together with `-node_density_factor`.
`-node_density_factor`	Factor which is multiplied by standard cell height to determine the node density on the std cell rails. It cannot be used together with `-node_density`. The default value is `5`, and the allowed values are integers `[0, MAX_INT]`.
`-corner`	Corner to use for analysis.

Check Power Grid

This command checks power grid.

```
check_power_grid
    [-net net_name]
    [-error_file error_file]
```

Options

Switch Name	Description
`-net`	Name of the net to analyze. Must be a power or ground net name.
`-error_file`	Name of the error file to save output to.

Write Spice Power Grid

This command writes the `spice` file for power grid.

```
write_pg_spice
    [-vsrc vsrc_file]
    [-outfile out_file]
    [-net net_name]
    [-dx bump_pitch_x]
    [-dy bump_pitch_y]
    [-corner corner]
```

Options

Switch Name	Description
`-vsrc`	File to set the location of the power C4 bumps/IO pins. See Vsrc_aes.loc file for an example and its *description*.
`-dx,-dy`	Set the bump pitch to decide the voltage source location in the absence of a `vsrc` file. The default bump pitch is 140um if neither these arguments nor a `vsrc` file are given.
`-net`	Name of the net to analyze. Must be a power or ground net name.
`-outfile`	Write per-instance voltage written into the file.
`-corner`	Corner to use for analysis.

Set PDNSim Net voltage

This command sets PDNSim net voltage.

```
set_pdnsim_net_voltage
    [-net net_name]
    [-voltage volt]
```

Options

Switch Name	Description
-net	Name of the net to analyze. It must be a power or ground net name.
-voltage	Sets the voltage on a specific net. If this option is not given, the Liberty file's voltage value is obtained from operating conditions.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

Command Name	Description
find_net	Get a reference to net name.

Example scripts

Example scripts demonstrating how to run PDNSim on a sample design on aes as follows:

```
./test/aes_test_vdd.tcl
./test/aes_test_vss.tcl
```

Regression tests

There are a set of regression tests in ./test. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

References

1. PDNSIM documentation
2. Chhabria, V.A. and Sapatnekar, S.S. (no date) The-openroad-project/pdnsim: Power Grid Analysis, GitHub. Available at: https://github.com/The-OpenROAD-Project/PDNSim (Accessed: 24 July 2023). (link)

License

BSD 3-Clause License. See LICENSE file.

Voltage source location file description

This file specifies the description of the C4 bump configurations file. The file is a csv as described below:

```
<x_coordinate>, <y_coordinate>, <octagonal_c4_bump_edge_length>, <voltage_value>
```

The x and y coordinate specify the center location of the voltage C4 bumps in micro meter.

The octagonal c4_edge_length specifies the edge length of the C4 to determine the pitch of the RDL layer in micron

Voltage_value specifies the value of voltage source at the C4 bump. In case there is a need to specify voltage drop in micron

Example file

```
250,250,20,1.1
130,170,20,1.1
370,410,10,1.1
410,450,10,1.1
```

5.2.17 Global Placement

The global placement module in OpenROAD (`gpl`) is based on the open-source RePlAce tool, from the paper "Advancing Solution Quality and Routability Validation in Global Placement".

Features:

- Analytic and nonlinear placement algorithm. Solves electrostatic force equations using Nesterov's method. (link)
- Verified with various commercial technologies and research enablements using OpenDB (7/14/16/28/45/55/65nm).
- Verified deterministic solution generation with various compilers and OS.
- Supports Mixed-size placement mode.

Visualized examples from ISPD 2006 contest; adaptec2.inf	Real-world Design: Coyote (TSMC16 7.5T)

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Global Placement

When using the `-timing_driven` flag, `gpl` does a virtual `repair_design` to find slacks and weight nets with low slack. It adjusts the worst slacks (modified with `-timing_driven_nets_percentage`) using a multiplier (modified with `-timing_driven_net_weight_max`). The multiplier is scaled from the full value for the worst slack, to 1.0 at the `timing_driven_nets_percentage` point. Use the `set_wire_rc` command to set resistance and capacitance of estimated wires used for timing.

Routability-driven arguments

- They begin with `-routability`.
- `-routability_check_overflow`, `-routability_max_density`, `-routability_max_bloat_iter`, `-routability_max_inflation_iter`, `-routability_target_rc_metric`, `-routability_inflation_ratio_coef`, `-routability_max_inflation_ratio`, `-routability_rc_coefficients`

Timing-driven arguments

- They begin with `-timing_driven`.
- `-timing_driven_net_reweight_overflow`, `-timing_driven_net_weight_max`, `-timing_driven_nets_percentage`

```
global_placement
    [-timing_driven]
    [-routability_driven]
    [-disable_timing_driven]
    [-disable_routability_driven]
    [-skip_initial_place]
    [-incremental]
    [-bin_grid_count grid_count]
    [-density target_density]
    [-init_density_penalty init_density_penalty]
    [-init_wirelength_coef init_wirelength_coef]
    [-min_phi_coef min_phi_conef]
    [-max_phi_coef max_phi_coef]
    [-reference_hpwl reference_hpwl]
    [-overflow overflow]
    [-initial_place_max_iter initial_place_max_iter]
    [-initial_place_max_fanout initial_place_max_fanout]
    [-pad_left pad_left]
```

(continues on next page)

(continued from previous page)

```
    [-pad_right pad_right]
    [-force_cpu]
    [-skip_io]
    [-skip_nesterov_place]
    [-routability_check_overflow routability_check_overflow]
    [-routability_max_density routability_max_density]
    [-routability_max_bloat_iter routability_max_bloat_iter]
    [-routability_max_inflation_iter routability_max_inflation_iter]
    [-routability_target_rc_metric routability_target_rc_metric]
    [-routability_inflation_ratio_coef routability_inflation_ratio_coef]
    [-routability_max_inflation_ratio routability_max_inflation_ratio]
    [-routability_rc_coefficients routability_rc_coefficients]
    [-timing_driven_net_reweight_overflow]
    [-timing_driven_net_weight_max]
    [-timing_driven_nets_percentage]
```

Options

Switch Name	Description
`-timing_driven`	Enable timing-driven mode. See *link* for timing-specific arguments.
`-routability_driven`	Enable routability-driven mode. See *link* for routability-specific arguments.
`-skip_initial_place`	Skip the initial placement (Biconjugate gradient stabilized, or BiCGSTAB solving) before Nesterov placement. Initial placement improves HPWL by ~5% on large designs. Equivalent to `-initial_place_max_iter 0`.
`-incremental`	Enable the incremental global placement. Users would need to tune other parameters (e.g., `init_density_penalty`) with pre-placed solutions.
`-bin_grid_count`	Set bin grid's counts. The internal heuristic defines the default value. Allowed values are integers `[64,128,256,512,...]`.
`-density`	Set target density. The default value is `0.7` (i.e., 70%). Allowed values are floats `[0, 1]`.
`-init_density_penalty`	Set initial density penalty. The default value is `8e-5`. Allowed values are floats `[1e-6, 1e6]`.
`-init_wirelength_coef`	Set initial wirelength coefficient. The default value is `0.25`. Allowed values are floats.
`-min_phi_coef`	Set `pcof_min` (μ_k Lower Bound). The default value is `0.95`. Allowed values are floats `[0.95, 1.05]`.
`-max_phi_coef`	Set `pcof_max` (μ_k Upper Bound). Default value is 1.05. Allowed values are `[1.00-1.20, float]`.
`-overflow`	Set target overflow for termination condition. The default value is `0.1`. Allowed values are floats `[0, 1]`.
`-initial_place_max_iter`	Set maximum iterations in the initial place. The default value is 20. Allowed values are integers `[0, MAX_INT]`.
`-initial_place_max_fanout`	Set net escape condition in initial place when $fanout \geq initial\_place\_max\_fanout$. The default value is 200. Allowed values are integers `[1, MAX_INT]`.
`-pad_left`	Set left padding in terms of number of sites. The default value is 0, and the allowed values are integers `[1, MAX_INT]`
`-pad_right`	Set right padding in terms of number of sites. The default value is 0, and the allowed values are integers `[1, MAX_INT]`
`-force_cpu`	Force to use the CPU solver even if the GPU is available.
`-skip_io`	Flag to ignore the IO ports when computing wirelength during placement. The default value is False, allowed values are boolean.

Routability-Driven Arguments

Switch Name	Description
`-routability_check_overflow`	Set overflow threshold for routability mode. The default value is `0.2`, and the allowed values are floats `[0, 1]`.
`-routability_max_density`	Set density threshold for routability mode. The default value is `0.99`, and the allowed values are floats `[0, 1]`.
`-routability_max_bloat_iter`	Set bloat iteration threshold for routability mode. The default value is `1`, and the allowed values are integers `[1, MAX_INT]`.
`-routability_max_inflation_iter`	Set inflation iteration threshold for routability mode. The default value is `4`, and the allowed values are integers `[1, MAX_INT]`.
`-routability_target_rc_metric`	Set target RC metric for routability mode. The default value is `1.0`, and the allowed values are floats.
`-routability_inflation_ratio_coef`	Set inflation ratio coefficient for routability mode. The default value is `2.5`, and the allowed values are floats.
`-routability_max_inflation_ratio`	Set inflation ratio threshold for routability mode. The default value is `2.5`, and the allowed values are floats.
`-routability_rc_coefficients`	Set routability RC coefficients. It comes in the form of a Tcl List `{k1, k2, k3, k4}`. The default value for each coefficient is `{1.0, 1.0, 0.0, 0.0}` respectively, and the allowed values are floats.

Timing-Driven Arguments

Switch Name	Description
`-timing_driven_net_reweight_overflow`	Set overflow threshold for timing-driven net reweighting. Allowed value is a Tcl list of integers where each number is `[0, 100]`.
`-timing_driven_net_weight_max`	Set the multiplier for the most timing-critical nets. The default value is `1.9`, and the allowed values are floats.
`-timing_driven_nets_percentage`	Set the reweighted percentage of nets in timing-driven mode. The default value is 10. Allowed values are floats `[0, 100]`.

Cluster Flops

This command does flop clustering based on parameters.

```
cluster_flops
    [-tray_weight tray_weight]\
    [-timing_weight timing_weight]\
    [-max_split_size max_split_size]\
    [-num_paths num_paths]
```

Options

Switch Name	Description
`-tray_weight`	Tray weight, default value is 20.0, type `float`.
`-timing_weight`	Timing weight, default value is 1.0, type `float`.
`-max_split_size`	Maximum split size, default value is -1, type `int`.
`-num_paths`	KIV, default value is 0, type `int`.

Debug Mode

The `global_placement_debug` command initiates a debug mode, enabling real-time visualization of the algorithm's progress on the layout. Use the command prior to executing the `global_placement` command, for example in the `global_place.tcl` script.

```
global_placement_debug
    [-pause]
    [-update]
    [-inst]
    [-draw_bins]
    [-initial]
```

Options

Switch Name	Description
`-pause`	Number of iterations between pauses during debugging. Allows for visualization of the current state. Useful for closely monitoring the progression of the placement algorithm. Allowed values are integers, default is 10.
`-update`	Defines the frequency (in iterations) at which the tool refreshes its layout output to display the latest state during debugging. Allowed values are integers, default is 10.
`-inst`	Targets a specific instance name for debugging focus. Allowed value is a string, the default behavior focuses on no specific instance.
`-draw_bins`	Activates visualization of placement bins, showcasing their density (indicated by the shade of white) and the direction of forces acting on them (depicted in red). The default setting is disabled.
`-initial`	Pauses the debug process during the initial placement phase. The default setting is disabled.

Example: `global_placement_debug -pause 100 -update 1 -initial -draw_bins -inst _614_` This command configures the debugger to pause every 100 iterations, with layout updates occurring every iteration. It enables initial placement stage visualization, bin drawing, and specifically highlights instance 614.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

```
# adds padding and gets global placement uniform target density
get_global_placement_uniform_density -pad_left -pad_right
```

Example scripts demonstrating how to run `gpl` on a sample design on `core01` as follows:

```
./test/core01.tcl
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

Using the Python interface to gpl

This API tries to stay close to the API defined in `C++` class `Replace` that is located here.

When initializing a design, a sequence of Python commands might look like the following:

```
from openroad import Design, Tech
tech = Tech()
tech.readLef(...)
design = Design(tech)
design.readDef(...)
gpl = design.getReplace()
```

Here is an example of some options / configurations to the global placer. (See Replace.h for a complete list)

```
gpl.setInitialPlaceMaxIter(iter)
gpl.setSkipIoMode(skip_io)
gpl.setTimingDrivenMode(timing_driven)
gpl.setTimingNetWeightMax(weight)
```

There are some useful Python functions located in the file *grt_aux.py* but these are not considered a part of the *final* API and they may change.

FAQs

Check out GitHub discussion about this tool.

References

- C.-K. Cheng, A. B. Kahng, I. Kang and L. Wang, "RePlAce: Advancing Solution Quality and Routability Validation in Global Placement", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 38(9) (2019), pp. 1717-1730. (.pdf)
- J. Lu, P. Chen, C.-C. Chang, L. Sha, D. J.-H. Huang, C.-C. Teng and C.-K. Cheng, "ePlace: Electrostatics based Placement using Fast Fourier Transform and Nesterov's Method", ACM TODAES 20(2) (2015), article 17. (.pdf)
- J. Lu, H. Zhuang, P. Chen, H. Chang, C.-C. Chang, Y.-C. Wong, L. Sha, D. J.-H. Huang, Y. Luo, C.-C. Teng and C.-K. Cheng, "ePlace-MS: Electrostatics based Placement for Mixed-Size Circuits", IEEE TCAD 34(5) (2015), pp. 685-698. (.pdf)
- A. B. Kahng, J. Li and L. Wang,
  "Improved Flop Tray-Based Design Implementation for Power Reduction",
  IEEE/ACM ICCAD, 2016, pp. 20:1-20:8.
- The timing-driven mode has been implemented by Mingyu Woo (only available in legacy repo in standalone branch.)
- The routability-driven mode has been implemented by Mingyu Woo.
- Timing-driven mode re-implementation is ongoing with the current clean-code structure.

Authors

- Authors/maintainer since Jan 2020: Mingyu Woo (Ph.D. Advisor: Andrew. B. Kahng)
- Original open-sourcing of RePlAce: August 2018, by Ilgweon Kang (Ph.D. Advisor: Chung-Kuan Cheng), Lutong Wang (Ph.D. Advisor: Andrew B. Kahng), and Mingyu Woo (Ph.D. Advisor: Andrew B. Kahng).
- Also thanks to Dr. Jingwei Lu for open-sourcing the previous ePlace-MS/ePlace project code.

License

BSD 3-Clause License. See LICENSE file.

5.2.18 Gate Resizer

Gate Resizer commands are described below. The `resizer` commands stop when the design area is `-max_utilization util` percent of the core area. `util` is between 0 and 100. The `resizer` stops and reports an error if the max utilization is exceeded.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Set Wire RC

The `set_wire_rc` command sets the resistance and capacitance used to estimate delay of routing wires. Separate values can be specified for clock and data nets with the `-signal` and `-clock` flags. Without either `-signal` or `-clock` the resistance and capacitance for clocks and data nets are set.

```
# Either run
set_wire_rc -clock ... -signal ... -layer ...

# Or
set_wire_rc -resistance ... -capacitance ...
```

```
set_wire_rc
    [-clock]
    [-signal]
    [-data]
    [-corner corner]
    [-layer layer_name]
    [-resistance res]
    [-capacitance cap]
```

Options

Switch Name	Description
`-clock`	Enable setting of RC for clock nets.
`-signal`	Enable setting of RC for signal nets.
`-layer`	Use the LEF technology resistance and area/edge capacitance values for the layer. This is used for a default width wire on the layer.
`-resistance`	Resistance per unit length, units are from the first Liberty file read, usually in the form of $\frac{resistanceUnit}{distanceUnit}$. Usually k/µm.
`-capacitance`	Capacitance per unit length, units are from the first Liberty file read, usually in the form of $\frac{capacitanceUnit}{distanceUnit}$. Usually pF/µm.

Set Layer RC

The `set_layer_rc` command can be used to set the resistance and capacitance for a layer or via. This is useful if these values are missing from the LEF file, or to override the values in the LEF.

```
set_layer_rc
    [-layer layer]
    [-via via_layer]
    [-resistance res]
    [-capacitance cap]
    [-corner corner]
```

Options

Switch Name	Description
`-layer`	Set layer name to modify. Note that the layer must be a routing layer.
`-via`	Select via layer name. Note that via resistance is per cut/via, not area-based.
`-resistance`	Resistance per unit length, same convention as `set_wire_rc`.
`-capacitance`	Capacitance per unit length, same convention as `set_wire_rc`.
`-corner`	Process corner to use.

Estimate Parasitics

Estimate RC parasitics based on placed component pin locations. If there are no component locations, then no parasitics are added. The resistance and capacitance values are per distance unit of a routing wire. Use the `set_units` command to check units or `set_cmd_units` to change units. The goal is to represent "average" routing layer resistance and capacitance. If the set_wire_rc command is not called before resizing, then the default_wireload model specified in the first Liberty file read or with the SDC set_wire_load command is used to make parasitics.

After the `global_route` command has been called, the global routing topology and layers can be used to estimate parasitics with the `-global_routing` flag.

```
estimate_parasitics
    -placement|-global_routing
```

Options

Switch Name	Description
`-placement` or `-global_routing`	Either of these flags must be set. Parasitics are estimated based after placement stage versus after global routing stage.

Set Don't Use

The `set_dont_use` command removes library cells from consideration by the `resizer` engine and the CTS engine. `lib_cells` is a list of cells returned by `get_lib_cells` or a list of cell names (`wildcards` allowed). For example, `DLY*` says do not use cells with names that begin with `DLY` in all libraries.

```
set_dont_use lib_cells
```

Unset Don't Use

The `unset_dont_use` command reverses the `set_dont_use` command.

```
unset_dont_use lib_cells
```

Set Don't Touch

The `set_dont_touch` command prevents the resizer commands from modifying instances or nets.

```
set_dont_touch instances_nets
```

Unset Don't Touch

The `unset_dont_touch` command reverse the `set_dont_touch` command.

```
unset_dont_touch instances_nets
```

Buffer Ports

The `buffer_ports -inputs` command adds a buffer between the input and its loads. The `buffer_ports -outputs` adds a buffer between the port driver and the output port. Inserting buffers on input and output ports makes the block input capacitances and output drives independent of the block internals.

```
buffer_ports
    [-inputs]
    [-outputs]
    [-max_utilization util]
    [-buffer_cell buf_cell]
```

Options

Switch Name	Description
`-inputs`, `-outputs`	Insert a buffer between the input and load, output and load respectively. The default behavior is `-inputs` and `-outputs` set if neither is specified.
`-max_utilization`	Defines the percentage of core area used.

Remove Buffers

Use the `remove_buffers` command to remove buffers inserted by synthesis. This step is recommended before using `repair_design` so that there is more flexibility in buffering nets.

```
remove_buffers
```

Balance Row Usage

Command description pending.

```
balance_row_usage
```

Repair Design

The `repair_design` command inserts buffers on nets to repair max slew, max capacitance and max fanout violations, and on long wires to reduce RC delay in the wire. It also resizes gates to normalize slews. Use `estimate_parasitics -placement` before `repair_design` to estimate parasitics considered during repair. Placement-based parasitics cannot accurately predict routed parasitics, so a margin can be used to "over-repair" the design to compensate.

```
repair_design
    [-max_wire_length max_length]
    [-slew_margin slew_margin]
    [-cap_margin cap_margin]
    [-max_utilization util]
    [-verbose]
```

Options

Switch Name	Description
`-max_wire_length`	Maximum length of wires (in microns), defaults to a value that minimizes the wire delay for the wire RC values specified by `set_wire_rc`.
`-slew_margin`	Add a slew margin. The default value is `0`, the allowed values are integers `[0, 100]`.
`-cap_margin`	Add a capactitance margin. The default value is `0`, the allowed values are integers `[0, 100]`.
`-max_utilization`	Defines the percentage of core area used.
`-verbose`	Enable verbose logging on progress of the repair.

Repair Tie Fanout

The `repair_tie_fanout` command connects each tie high/low load to a copy of the tie high/low cell.

```
repair_tie_fanout
    [-separation dist]
    [-max_fanout fanout]
    [-verbose]
    lib_port
```

Options

Switch Name	Description
-separation	Tie high/low insts are separated from the load by this value (Liberty units, usually microns).
-verbose	Enable verbose logging of repair progress.
lib_port	Tie high/low port, which can be a library/cell/port name or object returned by get_lib_pins.

Repair Timing

The repair_timing command repairs setup and hold violations. It should be run after clock tree synthesis with propagated clocks. Setup repair is done before hold repair so that hold repair does not cause setup checks to fail.

The worst setup path is always repaired. Next, violating paths to endpoints are repaired to reduced the total negative slack.

```
repair_timing
    [-setup]
    [-hold]
    [-recover_power percent_of_paths_with_slack]
    [-setup_margin setup_margin]
    [-hold_margin hold_margin]
    [-slack_margin slack_margin]
    [-libraries libs]
    [-allow_setup_violations]
    [-skip_pin_swap]
    [-skip_gate_cloning]
    [-repair_tns tns_end_percent]
    [-max_passes passes]
    [-max_utilization util]
    [-max_buffer_percent buffer_percent]
    [-verbose]
```

Options

Switch Name	Description
`-setup`	Repair setup timing.
`-hold`	Repair hold timing.
`-recover_power`	Set the percentage of paths to recover power for. The default value is `0`, and the allowed values are floats `(0, 100]`.
`-setup_margin`	Add additional setup slack margin.
`-hold_margin`	Add additional hold slack margin.
`-allow_setup_violations`	While repairing hold violations, buffers are not inserted that will cause setup violations unless `-allow_setup_violations` is specified.
`-skip_pin_swap`	Flag to skip pin swap. The default value is `False`, and the allowed values are bools.
`-skip_gate_cloning`	Flag to skip gate cloning. The default value is `False`, and the allowed values are bools.
`-repair_tns`	Percentage of violating endpoints to repair (0-100). When `tns_end_percent` is zero (the default), only the worst endpoint is repaired. When `tns_end_percent` is 100, all violating endpoints are repaired.
`-max_utilization`	Defines the percentage of core area used.
`-max_buffer_percent`	Specify a maximum number of buffers to insert to repair hold violations as a percentage of the number of instances in the design. The default value is `20`, and the allowed values are integers `[0, 100]`.
`-verbose`	Enable verbose logging of the repair progress.

Use`-recover_power` to specify the percent of paths with positive slack which will be considered for gate resizing to save power. It is recommended that this option be used with global routing based parasitics.

Repair Clock Nets

The `clock_tree_synthesis` command inserts a clock tree in the design but may leave a long wire from the clock input pin to the clock tree root buffer.

The `repair_clock_nets` command inserts buffers in the wire from the clock input pin to the clock root buffer.

```
repair_clock_nets
    [-max_wire_length max_wire_length]
```

Options

Switch Name	Description
`-max_wire_length`	Maximum length of wires (in microns), defaults to a value that minimizes the wire delay for the wire RC values specified by `set_wire_rc`.

Repair Clock Inverters

The repair_clock_inverters command replaces an inverter in the clock tree with multiple fanouts with one inverter per fanout. This prevents the inverter from splitting up the clock tree seen by CTS. It should be run before clock_tree_synthesis.

```
repair_clock_inverters
```

Report Design Area

The `report_design_area` command reports the area of the design's components and the utilization.

```
report_design_area
```

Report Floating Nets

The `report_floating_nets` command reports nets with only one pin connection.

```
report_floating_nets
    [-verbose]
```

Options

Switch Name	Description
`-verbose`	Print the net names.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

Command Name	Description
`repair_setup_pin`	Repair setup pin violation.
`check_parasitics`	Check if the `estimate_parasitics` command has been called.
`parse_time_margin_arg`	Get the raw value for timing margin (e.g. `slack_margin`, `setup_margin`, `hold_margin`)
`parse_percent_margin_arg`	Get the above margin in perentage format.
`parse_margin_arg`	Same as `parse_percent_margin_arg`.
`parse_max_util`	Check maximum utilization.
`parse_max_wire_length`	Get maximum wirelength.
`check_corner_wire_caps`	Check wire capacitance for corner.
`check_max_wire_length`	Check if wirelength is allowed by rsz for minimum delay.
`dblayer_wire_rc`	Get layer RC values.
`set_dblayer_wire_rc`	Set layer RC values.

Example scripts

A typical `resizer` command file (after a design and Liberty libraries have been read) is shown below.

```
read_sdc gcd.sdc

set_wire_rc -layer metal2

set_dont_use {CLKBUF_* AOI211_X1 OAI211_X1}

buffer_ports
repair_design -max_wire_length 100
repair_tie_fanout LOGIC0_X1/Z
repair_tie_fanout LOGIC1_X1/Z
# clock tree synthesis...
repair_timing
```

Note that OpenSTA commands can be used to report timing metrics before or after resizing the design.

```
set_wire_rc -layer metal2
report_checks
report_tns
report_wns
report_checks

repair_design

report_checks
report_tns
report_wns
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

License

BSD 3-Clause License. See *LICENSE* file.

5.2.19 Detailed Placement

The detailed placement module in OpenROAD (dpl) is based on OpenDP, or Open-Source Detailed Placement Engine. Its key features are:

- Fence region.
- Fragmented ROWs.

Commands

Note:

- Parameters in square brackets [-param param] are optional.
- Parameters without square brackets -param2 param2 are required.

Detailed Placement

The detailed_placement command performs detailed placement of instances to legal locations after global placement.

```
detailed_placement
    [-max_displacement disp|{disp_x disp_y}]
    [-disallow_one_site_gaps]
    [-report_file_name filename]
```

Options

Switch Name	Description
-max_displacement	Max distance that an instance can be moved (in microns) when finding a site where it can be placed. Either set one value for both directions or set {disp_x disp_y} for individual directions. The default values are {0, 0}, and the allowed values within are integers [0, MAX_INT].
-disallow_one_site_gaps	Disable one site gap during placement check.
-report_file_name	File name for saving the report to (e.g. report.json.)

Set Placement Padding

The `set_placement_padding` command sets left and right padding in multiples of the row site width. Use the `set_placement_padding` command before legalizing placement to leave room for routing. Use the `-global` flag for padding that applies to all instances. Use `-instances` for instance-specific padding. The instances `insts` can be a list of instance names, or an instance object returned by the SDC `get_cells` command. To specify padding for all instances of a common master, use the `-filter` “ref_name == ” option to `get_cells`.

```
set_placement_padding
    -global|-masters masters|-instances insts
    [-right site_count]
    [-left site_count]
```

Options

Warning: Either one of these flags must be set: `-global | -masters | -instances`. The order of preference is `global > masters > instances`

Switch Name	Description
`-global`	Set padding globally using `left` and `right` values.
`-masters`	Set padding only for these masters using `left` and `right` values.
`-instances`	For `-instances`, you will set padding only for these insts using `left` and `right` values.
`-left`	Left padding (in site count).
`-right`	Right padding (in site count).
`instances`	Set padding for these list of instances. Not to be confused with the `-instances` switch above.

Filler Placement

The `filler_placement` command fills gaps between detail-placed instances to connect the power and ground rails in the rows. `filler_masters` is a list of master/macro names to use for filling the gaps. Wildcard matching is supported, so `FILL*` will match, e.g., `FILLCELL_X1 FILLCELL_X16 FILLCELL_X2 FILLCELL_X32 FILLCELL_X4 FILLCELL_X8`. To specify a different naming prefix from `FILLER_` use `-prefix <new prefix>`.

```
filler_placement
    [-prefix prefix]
    filler_masters
```

Options

Switch Name	Description
`-prefix`	Prefix to name the filler cells. The default value is `FILLER_`.
`filler_masters`	Filler master cells.

Remove Fillers

This command removes all filler cells.

```
remove_fillers
```

Check Placement

The `check_placement` command checks the placement legality. It returns `0` if the placement is legal.

```
check_placement
    [-verbose]
    [-disallow_one_site_gaps]
    [-report_file_name filename]
```

Options

Switch Name	Description
`-verbose`	Enable verbose logging.
`-disallow_one_site_gaps`	Disable one site gap during placement check.
`-report_file_name`	File name for saving the report to (e.g. `report.json`.

Optimize Mirroring

The `optimize_mirroring` command mirrors instances about the Y axis in a weak attempt to reduce the total half-perimeter wirelength (HPWL).

```
optimize_mirroring
```

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

Command Name	Description
`detailed_placement_debug`	Debug detailed placement.
`get_masters_arg`	Get masters from a design.
`get_inst_bbox`	Get bounding box of an instance.
`get_inst_grid_bbox`	Get grid bounding box of an instance.
`format_grid`	Format grid (takes in length `x` and site width `w` as inputs).
`get_row_site`	Get row site name.

Example scripts

Examples scripts demonstrating how to run `dpl` on a sample design of `aes` as follows:

```
./test/aes.tcl
```

Regression tests

There are a set of regression tests in `./test`. Refer to this *section* for more information.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

Authors

- SangGi Do and Mingyu Woo (respective Ph. D. advisors: Seokhyeong Kang, Andrew B. Kahng).
- Rewrite and port to OpenDB/OpenROAD by James Cherry, Parallax Software

References

1. Do, S., Woo, M., & Kang, S. (2019, May). Fence-region-aware mixed-height standard cell legalization. In Proceedings of the 2019 on Great Lakes Symposium on VLSI (pp. 259-262). (.pdf)

License

BSD 3-Clause License. See LICENSE file.

5.2.20 Clock Tree Synthesis

The clock tree synthesis module in OpenROAD (`cts`) is based on TritonCTS 2.0. It is available from the `clock_tree_synthesis` command. TritonCTS 2.0 performs on-the-fly characterization. Thus, there is no need to generate characterization data. The on-the-fly characterization feature can be optionally controlled by parameters specified by the `configure_cts_characterization` command. Use `set_wire_rc` command to set the clock routing layer.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Configure CTS Characterization

Configure key CTS characterization parameters, for example maximum slew and capacitance, as well as the number of steps they will be divided for characterization.

```
configure_cts_characterization
    [-max_slew max_slew]
    [-max_cap max_cap]
    [-slew_steps slew_steps]
    [-cap_steps cap_steps]
```

Options

Switch Name	Description
`-max_slew`	Max slew value (in the current time unit) that the characterization will test. If this parameter is omitted, the code would use max slew value for specified buffer in `buf_list` from liberty file.
`-max_cap`	Max capacitance value (in the current capacitance unit) that the characterization will test. If this parameter is omitted, the code would use max cap value for specified buffer in `buf_list` from liberty file.
`-slew_steps`	Number of steps that `max_slew` will be divided into for characterization. The default value is `12`, and the allowed values are integers `[0, MAX_INT]`.
`-cap_steps`	Number of steps that `max_cap` will be divided into for characterization. The default value is `34`, and the allowed values are integers `[0, MAX_INT]`.

Clock Tree Synthesis

Perform clock tree synthesis.

```
clock_tree_synthesis
    [-wire_unit wire_unit]
    [-buf_list <list_of_buffers>]
    [-root_buf root_buf]
    [-clk_nets <list_of_clk_nets>]
    [-tree_buf <buf>]
    [-distance_between_buffers]
    [-branching_point_buffers_distance]
    [-clustering_exponent]
    [-clustering_unbalance_ratio]
    [-sink_clustering_size cluster_size]
    [-sink_clustering_max_diameter max_diameter]
```

(continues on next page)

(continued from previous page)

```
    [-sink_clustering_enable]
    [-balance_levels]
    [-sink_clustering_levels levels]
    [-num_static_layers]
    [-sink_clustering_buffer]
    [-obstruction_aware]
    [-apply_ndr]
    [-insertion_delay]
    [-use_dummy_load]
    [-insertion_delay]
    [-sink_buffer_max_cap_derate derate_value]
    [-delay_buffer_derate derate_value]
```

Options

Switch Name	Description
`-buf_list`	Tcl list of master cells (buffers) that will be considered when making the wire segments (e.g. `{BUFXX, BUFYY}`).
`-root_buffer`	The master cell of the buffer that serves as root for the clock tree. If this parameter is omitted, the first master cell from `-buf_list` is taken.
`-wire_unit`	Minimum unit distance between buffers for a specific wire. If this parameter is omitted, the code gets the value from ten times the height of `-root_buffer`.
`-clk_nets`	String containing the names of the clock roots. If this parameter is omitted, `cts` automatically looks for the clock roots automatically.
`-distance_between_buffers`	Distance (in microns) between buffers that `cts` should use when creating the tree. When using this parameter, the clock tree algorithm is simplified and only uses a fraction of the segments from the LUT.
`-branching_point_buffers_distance`	Distance (in microns) that a branch has to have in order for a buffer to be inserted on a branch end-point. This requires the `-distance_between_buffers` value to be set.
`-clustering_exponent`	Value that determines the power used on the difference between sink and means on the CKMeans clustering algorithm. The default value is `4`, and the allowed values are integers `[0, MAX_INT]`.
`-clustering_unbalance_ratio`	Value determines each cluster's maximum capacity during CKMeans. A value of `0.5` (i.e., 50%) means that each cluster will have exactly half of all sinks for a specific region (half for each branch). The default value is `0.6`, and the allowed values are floats `[0, 1.0]`.
`-sink_clustering_enable`	Enables pre-clustering of sinks to create one level of sub-tree before building H-tree. Each cluster is driven by buffer which becomes end point of H-tree structure.
`-sink_clustering_size`	Specifies the maximum number of sinks per cluster. The default value is `20`, and the allowed values are integers `[0, MAX_INT]`.
`-sink_clustering_max_diameter`	Specifies maximum diameter (in microns) of sink cluster. The default value is `50`, and the allowed values are integers `[0, MAX_INT]`.
`-balance_levels`	Attempt to keep a similar number of levels in the clock tree across non-register cells (e.g., clock-gate or inverter). The default value is `False`, and the allowed values are bool.
`-clk_nets`	String containing the names of the clock roots. If this parameter is omitted, `cts` looks for the clock roots automatically.
`-num_static_layers`	Set the number of static layers. The default value is `0`, and the allowed values are integers `[0, MAX_INT]`.
`-sink_clustering_buffer`	Set the sink clustering buffer(s) to be used.
`-obstruction_aware`	Enables obstruction-aware buffering such that clock buffers are not placed on top of blockages or hard macros. This option may reduce legalizer displacement, leading to better latency, skew or timing QoR. The default value is `False`, and the allowed values are bool.
`-apply_ndr`	Applies 2X spacing non-default rule to all clock nets except leaf-level nets. The default value is `False`.
`-no_insertion_delay`	Don't consider insertion delays in macro timing models in balancing latencies between macro cells and registers. This option prevents construction of separate clock trees for macro cells and registers. The default value is `False`.
`-use_dummy_load`	Applies dummy buffer or inverter cells at clock tree leaves to balance loads. The default values is `False`.
`-sink_buffer_max_cap_derate`	Use this option to control automatic buffer selection. To favor strong(weak) drive strength buffers use a small(large) value. The default value is `0.01`, meaning that buffers are selected by derating max cap limit by 0.01. The value of 1.0 means no derating of max cap limit.
`-delay_buffer_derate`	This option is used with -insertion_delay option that balances latencies between macro cells and registers by inserting delay buffers. The default values is `1.0`, meaning all needed delay buffers are inserted. Value of 0.5 means only half of necessary delay buffers are inserted. Value of 0.0 means no insertion of delay buffers.

Report CTS

This command is used to extract the following metrics after a successful `clock_tree_synthesis` run.

- Number of Clock Roots
- Number of Buffers Inserted
- Number of Clock Subnets
- Number of Sinks.

```
report_cts
    [-out_file file]
```

Options

Switch Name	Description
`-out_file`	The file to save `cts` reports. If this parameter is omitted, the report is streamed to `stdout` and not saved.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

Command Name	Description
`clock_tree_synthesis_debug`	Option to plot the CTS to GUI.

Example scripts

```
clock_tree_synthesis -root_buf "BUF_X4" \
                     -buf_list "BUF_X4" \
                     -wire_unit 20
report_cts "file.txt"
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

References

1. LEMON - **L**ibrary for **E**fficient **M**odeling and **O**ptimization in **N**etworks
2. Kahng, A. B., Li, J., & Wang, L. (2016, November). Improved flop tray-based design implementation for power reduction. In 2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD) (pp. 1-8). IEEE. (.pdf)

Authors

TritonCTS 2.0 is written by Mateus Fogaça, PhD student in the Graduate Program on Microelectronics from the Federal University of Rio Grande do Sul (UFRGS), Brazil. Mr. Fogaça's advisor is Prof. Ricardo Reis.

Many guidance provided by (alphabetic order):

- Andrew B. Kahng
- Jiajia Li
- Kwangsoo Han
- Tom Spyrou

License

BSD 3-Clause License. See LICENSE file.

5.2.21 Global Routing

The global routing module in OpenROAD (`grt`) is based on FastRoute, an open-source global router originally derived from Iowa State University's FastRoute4.1 algorithm.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Global Route

This command performs global routing with the option to use a `guide_file`. You may also choose to use incremental global routing using `-start_incremental`.

```
global_route
    [-guide_file out_file]
    [-congestion_iterations iterations]
    [-congestion_report_file file_name]
    [-congestion_report_iter_step steps]
    [-grid_origin {x y}]
    [-critical_nets_percentage percent]
    [-allow_congestion]
    [-allow_overflow]
    [-overflow_iterations]
    [-verbose]
    [-start_incremental]
    [-end_incremental]
```

Options

Switch Name	Description
`-guide_file`	Set the output guides file name (e.g., `route.guide`).
`-congestion_iterations`	Set the number of iterations made to remove the overflow of the routing. The default value is `50`, and the allowed values are integers `[0, MAX_INT]`.
`-congestion_report_file`	Set the file name to save the congestion report. The file generated can be read by the DRC viewer in the GUI (e.g., `report_file.rpt`).
`-congestion_report_iter_step`	Set the number of iterations to report. The default value is `0`, and the allowed values are integers `[0, MAX_INT]`.
`-grid_origin`	Set the (x, y) origin of the routing grid in DBU. For example, `-grid_origin {1 1}` corresponds to the die (0, 0) + 1 DBU in each x–, y- direction.
`-critical_nets_percentage`	Set the percentage of nets with the worst slack value that are considered timing critical, having preference over other nets during congestion iterations (e.g. `-critical_nets_percentage 30`). The default value is `0`, and the allowed values are integers `[0, MAX_INT]`.
`-allow_congestion`	Allow global routing results to be generated with remaining congestion. The default is false.
`-verbose`	This flag enables the full reporting of the global routing.
`-start_incremental`	This flag initializes the GRT listener to get the net modified. The default is false.
`-end_incremental`	This flag run incremental GRT with the nets modified. The default is false.

Set Routing Layers

This command sets the minimum and maximum routing layers for signal and clock nets. Example: `set_routing_layers -signal Metal2-Metal10 -clock Metal6-Metal9`

```
set_routing_layers
    [-signal min-max]
    [-clock min-max]
```

Options

Switch Name	Description
`-signal`	Set the min and max routing signal layer (names) in this format "%s-%s".
`-clock`	Set the min and max routing clock layer (names) in this format "%s-%s".

Set Macro Extension

This command sets the halo (in terms of GCells) along the boundaries of macros. A `GCell` is typically defined in terms of `Mx` routing tracks. Example: `set_macro_extension 2`

```
set_macro_extension extension
```

Options

Argument Name	Description
`extension`	Number of `GCells` added to the blockage boundaries from macros. The default `GCell` size is 15 `M3` pitches.

Set Pin Offset

This command sets the pin offset distance.

```
set_pin_offset offset
```

Options

Argument Name	Description
`offset`	Pin offset in microns (must be a positive integer).

Set Global Routing Layer Adjustment

The `set_global_routing_layer_adjustment` command sets routing resource adjustments in the routing layers of the design. Such adjustments reduce the number of routing tracks that the global router assumes to exist. This promotes the spreading of routing and reduces peak congestion, to reduce challenges for detailed routing.

You can set adjustment for a specific layer, e.g., `set_global_routing_layer_adjustment Metal4 0.5` reduces the routing resources of routing layer `Metal4` by 50%. You can also set adjustment for all layers at once using `*`, e.g., `set_global_routing_layer_adjustment * 0.3` reduces the routing resources of all routing layers by 30%. And, you can also set resource adjustment for a layer range, e.g.: `set_global_routing_layer_adjustment Metal4-Metal8 0.3` reduces the routing resources of routing layers `Metal4`, `Metal5`, `Metal6`, `Metal7` and `Metal8` by 30%.

```
set_global_routing_layer_adjustment layer adjustment
```

Options

Argument Name	Description
`layer`	Integer for the layer number (e.g. for M1 you would use 1).
`adjustment`	Float indicating the percentage reduction of each edge in the specified layer.

Set Global Routing Region Adjustment

Set global routing region adjustment. Example: `set_global_routing_region_adjustment {1.5 2 20 30.5} -layer Metal4 -adjustment 0.7`

```
set_global_routing_region_adjustment
    {lower_left_x lower_left_y upper_right_x upper_right_y}
    -layer layer
    -adjustment adjustment
```

Options

Switch Name	Description
`lower_left_x`, `lower_left_y`, `upper_right_x` , `upper_right_y`	Bounding box to consider.
`-layer`	Integer for the layer number (e.g. for M1 you would use 1).
`-adjustment`	Float indicating the percentage reduction of each edge in the specified layer.

Set Global Routing Randomness

The command randomizes global routing by shuffling the order of the nets and randomly subtracts or adds to the capacities of a random set of edges.

Example: `set_global_routing_random -seed 42 \ -capacities_perturbation_percentage 50 \ -perturbation_amount 2`

```
set_global_routing_random
    [-seed seed]
    [-capacities_perturbation_percentage percent]
    [-perturbation_amount value]
```

Options

Switch Name	Description
`-seed`	Sets the random seed (must be non-zero for randomization).
`-capacities_perturbation_percentage`	Sets the percentage of edges whose capacities are perturbed. By default, the edge capacities are perturbed by adding or subtracting 1 (track) from the original capacity.
`-perturbation_amount`	Sets the perturbation value of the edge capacities. This option is only meaningful when `-capacities_perturbation_percentage` is used.

Set Specific Nets to Route

The `set_nets_to_route` command defines a list of nets to route. Only the nets defined in this command are routed, leaving the remaining nets without any global route guides.

```
set_nets_to_route
    net_names
```

Options

Switch Name	Description
`net_names`	Tcl list of set of nets (e.g. `{net1, net2}`).

Repair Antennas

The `repair_antennas` command checks the global routing for antenna violations and repairs the violations by inserting diodes near the gates of the violating nets. By default the command runs only one iteration to repair antennas. Filler instances added by the `filler_placement` command should NOT be in the database when `repair_antennas` is called.

See LEF/DEF 5.8 Language Reference, Appendix C, "Calculating and Fixing Process Antenna Violations" for a *description* of antenna violations.

If no `diode_cell` argument is specified the LEF cell with class CORE, ANTENNACELL will be used. If any repairs are made the filler instances are remove and must be placed with the `filler_placement` command.

If the LEF technology layer `ANTENNADIFFSIDEAREARATIO` properties are constant instead of PWL, inserting diodes will not improve the antenna ratios, and thus, no diodes are inserted. The following warning message will be reported:

```
[WARNING GRT-0243] Unable to repair antennas on net with diodes.
```

```
repair_antennas
    [diode_cell]
    [-iterations iterations]
    [-ratio_margin margin]
```

Options

Switch Name	Description
`diode_cell`	Diode cell to fix antenna violations.
`-iterations`	Number of iterations. The default value is 1, and the allowed values are integers `[0, MAX_INT]`.
`-ratio_margin`	Add a margin to the antenna ratios. The default value is `0`, and the allowed values are integers `[0, 100]`.

Plot Global Routing Guides

The `draw_route_guides` command plots the route guides for a set of nets. To erase the route guides from the GUI, pass an empty list to this command: `draw_route_guides {}`.

```
draw_route_guides
    net_names
    [-show_pin_locations]
```

Options

Switch Name	Description
`net_names`	Tcl list of set of nets (e.g. `{net1, net2}`).
`-show_pin_locations`	Draw circles for the pin positions on the routing grid.

Report Wirelength

The `report_wire_length` command reports the wire length of the nets. Use the `-global_route` and the `-detailed_route` flags to report the wire length from global and detailed routing, respectively. If none of these flags are used, the tool will identify the state of the design and report the wire length accordingly.

Example: `report_wire_length -net {clk net60} -global_route -detailed_route -verbose -file out.csv`

```
report_wire_length
    [-net net_list]
    [-file file]
    [-global_route]
    [-detailed_route]
    [-verbose]
```

Options

Switch Name	Description
-net	List of nets to report the wirelength. Use * to report the wire length for all nets of the design.
-file	The name of the file for the wirelength report.
-global_route	Report the wire length of the global routing.
-detailed_route	Report the wire length of the detailed routing.
-verbose	This flag enables the full reporting of the layer-wise wirelength information.

Global Route Debug Mode

The `global_route_debug` command allows you to start a debug mode to view the status of the Steiner Trees. It also allows you to dump the input positions for the Steiner tree creation of a net. This must be used before calling the `global_route` command. Set the name of the net and the trees that you want to visualize.

```
global_route_debug
    [-st]
    [-rst]
    [-tree2D]
    [-tree3D]
    [-saveSttInput file_name]
    [-net net_name]
```

Options

Switch Name	Description
-st	Show the Steiner Tree generated by `stt`.
-rst	Show the Rectilinear Steiner Tree generated by `grt`.
-tree2D	Show the Rectilinear Steiner Tree generated by `grt` after the overflow iterations.
-tree3D	Show the 3D Rectilinear Steiner Tree post-layer assignment.
-saveSttInput	File name to save `stt` input of a net.
-net	The name of the net name to be displayed.

Read Global Routing Guides

This command reads global routing guides.

```
read_guides file_name
```

Options

Switch Name	Description
`file_name`	Path to global routing guide.

Example scripts

Examples scripts demonstrating how to run FastRoute on a sample design of `gcd` as follows:

```
./test/gcd.tcl
```

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

Command Name	Description
`check_routing_layer`	Check if the layer is within the min/max routing layer specified.
`parse_layer_name`	Get routing layer number from layer name
`parse_layer_range`	Parses a range from `layer_range` argument of format (%s-%s). `cmd` argument is not used.
`check_region`	Checks the defined region if its within the die area.
`define_layer_range`	Provide a Tcl list of layers and automatically generate the min and max layers for signal routing.
`define_clock_layer_range`	Provide a Tcl list of layers and automatically generate the min and max layers for clock routing.
`have_detailed_route`	Checks if block has detailed route already.

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

Using the Python interface to grt

Warning: The `Python` interface is currently in development and is subject to change.

The `Python` API tries to stay close to the API defined in the `C++` class `GlobalRouter` that is located here

When initializing a design, a sequence of `Python` commands might look like the following:

```
from openroad import Design, Tech
tech = Tech()
tech.readLef(...)
design = Design(tech)
design.readDef(...)
gr = design.getGlobalRouter()
```

Here are some options to the `global_route` command. (See `GlobalRouter.h` for a complete list)

```
gr.setGridOrigin(x, y)                     # int, default 0,0
gr.setCongestionReportFile(file_name)      # string
gr.setOverflowIterations(n)                # int, default 50
gr.setAllowCongestion(allowCongestion)     # boolean, default False
gr.setCriticalNetsPercentage(percentage)   # float
gr.setMinRoutingLayer(minLayer)            # int
gr.setMaxRoutingLayer(maxLayer)            # int
gr.setMinLayerForClock(minLayer)           # int
gr.setMaxLayerForClock(maxLayer)           # int
gr.setVerbose(v)                           # boolean, default False
```

and when ready to actually do the global route:

```
gr.globalRoute(save_guides)                # boolean, default False
```

If you have set `save_guides` to True, you can then save the guides in `file_name` with:

```
design.getBlock().writeGuides(file_name)
```

You can find the index of a named layer with

```
lindex = tech.getDB().getTech().findLayer(layer_name)
```

or, if you only have the `Python` design object

```
lindex = design.getTech().getDB().getTech().findLayer(layer_name)
```

Be aware that much of the error checking is done in `Tcl`, so that with the current `C++` / `Python` API, that might be an issue to deal with. There are also some useful `Python` functions located in the `grt_aux.py` file but these are not considered a part of the *final* API and may be subject to change.

FAQs

Check out GitHub discussion about this tool.

References

- Database comes from OpenDB
- FastRoute 4.1 documentation. The FastRoute4.1 version was received from Yue Xu on June 15, 2019.
- Min Pan, Yue Xu, Yanheng Zhang and Chris Chu. “FastRoute: An Efficient and High-Quality Global Router. VLSI Design, Article ID 608362, 2012.” Available here.
- C. J. Alpert, T. C. Hu, J. H. Huang, A. B. Kahng and D. Karger, “Prim-Dijkstra Tradeoffs for Improved Performance-Driven Global Routing”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 14(7) (1995), pp. 890-896. Available here.

License

BSD 3-Clause License. See LICENSE file.

Flute3

Flute3 is an open-source rectilinear Steiner minimum tree heuristic with improvements made by UFRGS students and James Cherry. This tool is used for the calculation of wirelength in `grt` and `rsz`.

The version in this repository uses CMake and C++ namespace, and has dynamic memory allocation. Flute3 can handle nets with any degree.

External references (Optional)

The algorithm base is Flute3.1, extracted from the FastRoute4.1 version that was received from yuexu@iastate.edu on June 15, 2019, with the BSD-3 open source license as given in the FastRoute website.

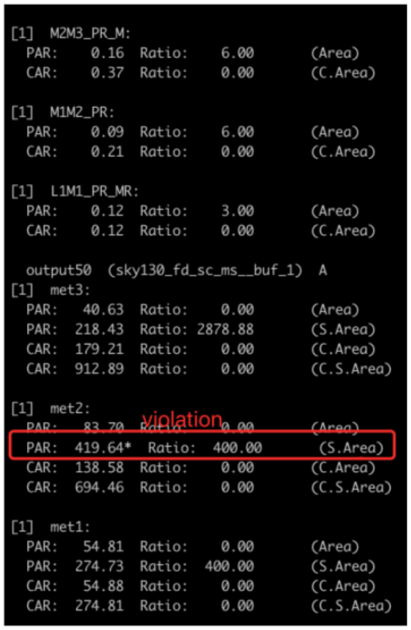
License

See LICENSE file.

5.2.22 Antenna Rule Checker

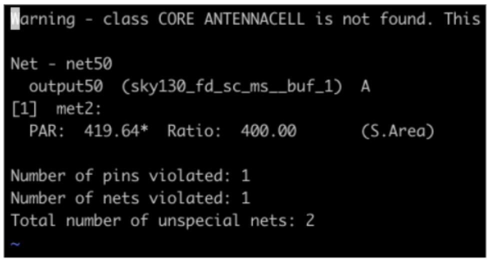
This tool checks antenna violations and generates a report to indicate violated nets. See LEF/DEF 5.8 Language Reference, Appendix C, “Calculating and Fixing Process Antenna Violations” (p.389) for a description of antenna violations.

This is an example of the detailed and simple reports of the antenna checker:



```
[1]  M2M3_PR_M:
  PAR:    0.16  Ratio:    6.00       (Area)
  CAR:    0.37  Ratio:    0.00       (C.Area)

[1]  M1M2_PR:
  PAR:    0.09  Ratio:    6.00       (Area)
  CAR:    0.21  Ratio:    0.00       (C.Area)

[1]  L1M1_PR_MR:
  PAR:    0.12  Ratio:    3.00       (Area)
  CAR:    0.12  Ratio:    0.00       (C.Area)

  output50  (sky130_fd_sc_ms__buf_1)  A
[1]  met3:
  PAR:   40.63  Ratio:    0.00       (Area)
  PAR:  218.43  Ratio: 2878.88       (S.Area)
  CAR:  179.21  Ratio:    0.00       (C.Area)
  CAR:  912.89  Ratio:    0.00       (C.S.Area)

[1]  met2:
  PAR:   83.70  Ratio:    0.00       (Area)
  PAR:  419.64*  Ratio:  400.00       (S.Area)
  CAR:  138.58  Ratio:    0.00       (C.Area)
  CAR:  694.46  Ratio:    0.00       (C.S.Area)

[1]  met1:
  PAR:   54.81  Ratio:    0.00       (Area)
  PAR:  274.73  Ratio:  400.00       (S.Area)
  CAR:   54.88  Ratio:    0.00       (C.Area)
  CAR:  274.81  Ratio:    0.00       (C.S.Area)
```

Full report



```
Warning - class CORE ANTENNACELL is not found. This

Net - net50
  output50  (sky130_fd_sc_ms__buf_1)  A
[1]  met2:
  PAR:  419.64*  Ratio:  400.00       (S.Area)

Number of pins violated: 1
Number of nets violated: 1
Total number of unspecial nets: 2
~
```

Simple report

|

|

Abbreviations Index:

- `PAR`: Partial Area Ratio
- `CAR`: Cumulative Area Ratio
- `Area`: Gate Area
- `S. Area`: Side Diffusion Area
- `C. Area`: Cumulative Gate Area
- `C. S. Area`: Cumulative Side (Diffusion) Area

Antenna violations can be repaired after global routing with the `repair_design` command.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Check Antennas

The `check_antennas` command will check for antenna violations.

```
check_antennas
    [-net net]
    [-verbose]
```

Options

Switch Name	Description
`-verbose`	Report all antenna calculations for violating nets.
`-net`	Check antennas on the specified net.

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

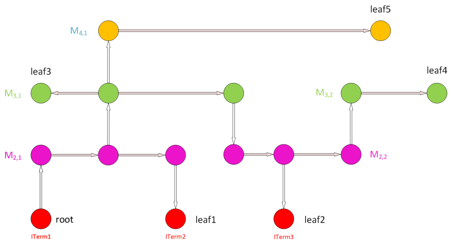
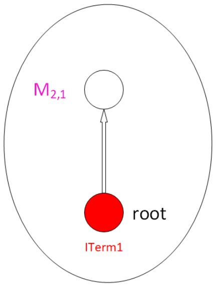
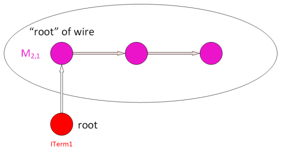
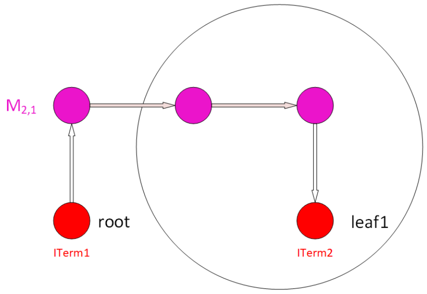
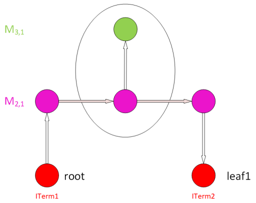
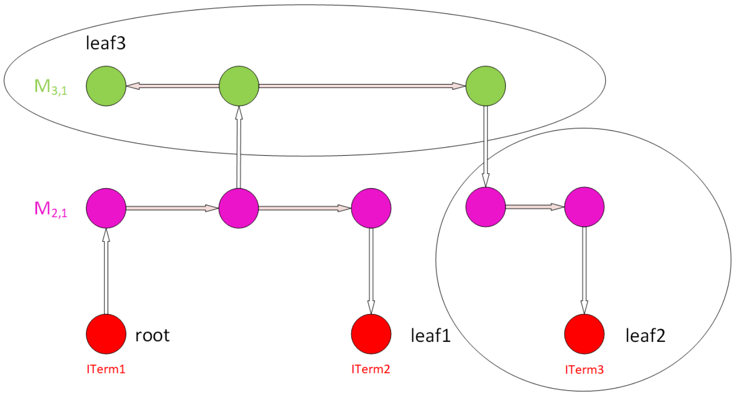
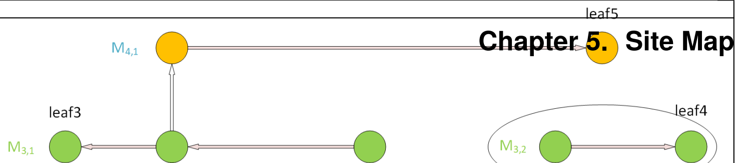
```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

Algorithm

<table>
<tr>
<td>
</td>
<td>
</td>
</tr>
<tr>
<td><b>Antenna Checker Algorithm</b>: Wire-Graph Example</td>
<td><b>Step 1</b>: (a) Start from the root node (ITerm) using upper Via to find a node for a new wire. (b) Save the ITerm area for cumulative gate/diffusion area.</td>
</tr>
<tr>
<td>
</td>
<td>
</td>
</tr>
<tr>
<td><b>Step 2</b>: From the node of the wire, find all the nodes in the wire through segment wires and find the “root” node of this wire.</td>
<td><b>Step 3</b>: (a) From the “root” node of the wire, along the outgoing segment edge that goes to other nodes belonging to this wire, calculate the area of this wire. (b) Then, find all the ITerms below these nodes, except for the root node (directly use an ITerm or lower Vias to find ITerms for lower metals). © Sum up the areas of all the ITerms found with the cumulative areas and calculate the PAR of this wire. (d) Add the PAR value and the wire info (layer, Index) into the PAR table. Add the new area to the cumulative areas.</td>
</tr>
<tr>
<td>
</td>
<td>
</td>
</tr>
<tr>
<td><b>Step 4</b>: Find all the upper Vias on this wire (for all the nodes on this wire), and go to the higher-level metal.</td>
<td><b>Step 5</b>: Repeat Steps 2 and 3 for new-found upper-level wires.</td>
</tr>
<tr>
<td>
</td>
<td>
</td>
</tr>
</table>

License

BSD 3-Clause License. See LICENSE file.

5.2.23 Detailed Routing

The Detailed Routing (drt) module in OpenROAD is based on the open-source detailed router, TritonRoute. TritonRoute consists of several main building blocks, including pin access analysis, track assignment, initial detailed routing, search and repair, and a DRC engine. The initial development of the router is inspired by the ISPD-2018 initial detailed routing contest. However, the current framework differs and is built from scratch, aiming for an industrial-oriented scalable and flexible flow.

TritonRoute provides industry-standard LEF/DEF interface with support of ISPD-2018 and ISPD-2019 contest-compatible route guide format.

Commands

Note:

- Parameters in square brackets [-param param] are optional.
- Parameters without square brackets -param2 param2 are required.

Detailed Route

This command performs detailed routing.

Developer arguments

- -or_seed, -or_k

Distributed arguments

- -distributed , -remote_host, -remote_port, -shared_volume, -cloud_size

```
detailed_route
    [-output_maze filename]
    [-output_drc filename]
    [-output_cmap filename]
    [-output_guide_coverage filename]
    [-drc_report_iter_step step]
    [-db_process_node name]
    [-disable_via_gen]
    [-droute_end_iter iter]
    [-via_in_pin_bottom_layer layer]
    [-via_in_pin_top_layer layer]
    [-or_seed seed]
    [-or_k k]
    [-bottom_routing_layer layer]
    [-top_routing_layer layer]
    [-verbose level]
    [-distributed]
```

(continues on next page)

(continued from previous page)

```
    [-remote_host rhost]
    [-remote_port rport]
    [-shared_volume vol]
    [-cloud_size sz]
    [-clean_patches]
    [-no_pin_access]
    [-min_access_points count]
    [-save_guide_updates]
    [-repair_pdn_vias layer]
    [-single_step_dr]
```

Options

Switch Name	Description
`-output_maze`	Path to output maze log file (e.g. `output_maze.log`).
`-output_drc`	Path to output DRC report file (e.g. `output_drc.rpt`).
`-output_cmap`	Path to output congestion map file (e.g. `output.cmap`).
`-output_guide_coverage`	Path to output guide coverage file (e.g. `sample_coverage.csv`).
`-drc_report_iter_step`	Report DRC on each iteration which is a multiple of this step. The default value is `0`, and the allowed values are integers `[0, MAX_INT]`.
`-db_process_node`	Specify the process node.
`-disable_via_gen`	Option to diable via generation with bottom and top routing layer. The default value is disabled.
`-droute_end_iter`	Number of detailed routing iterations. The default value is `-1`, and the allowed values are integers `[1, 64]`.
`-via_in_pin_bottom_layer`	Via-in pin bottom layer name.
`-via_in_pin_top_layer`	Via-in pin top layer name.
`-or_seed`	Refer to developer arguments *here*.
`-or_k`	Refer to developer arguments *here*.
`-bottom_routing_layer`	Bottommost routing layer name.
`-top_routing_layer`	Topmost routing layer name.
`-verbose`	Sets verbose mode if the value is greater than 1, else non-verbose mode (must be integer, or error will be triggered.)
`-distributed`	Refer to distributed arguments *here*.
`-clean_patches`	Clean unneeded patches during detailed routing.
`-no_pin_access`	Disables pin access for routing.
`-min_access_points`	Minimum access points for standard cell and macro cell pins.
`-save_guide_updates`	Flag to save guides updates.
`-repair_pdn_vias`	This option is used for PDKs where M1 and M2 power rails run in parallel.

Developer arguments

Some arguments that are helpful for developers are listed here.

Switch Name	Description
-or_seed	Random seed for the order of nets to reroute. The default value is -1, and the allowed values are integers [0, MAX_INT].
-or_k	Number of swaps is given by $k * sizeof(rerouteNets)$. The default value is 0, and the allowed values are integers [0, MAX_INT].

Detailed Route Debugging

The following command and arguments are useful when debugging error messages from drt and to understand its behavior.

```
detailed_route_debug
    [-pa]
    [-ta]
    [-dr]
    [-maze]
    [-net name]
    [-pin name]
    [-box x1 y1 x2 y2]
    [-iter iter]
    [-pa_markers]
    [-dump_dr]
    [-dump_dir dir]
    [-dump_last_worker]
    [-pa_edge]
    [-pa_commit]
    [-write_net_tracks]
```

Options

Switch Name	Description
`-pa`	Enable debug for pin access.
`-ta`	Enable debug for track assignment.
`-dr`	Enable debug for detailed routing.
`-maze`	Enable debug for maze routing.
`-net`	Enable debug for net name.
`-pin`	Enable debug for pin name.
`-box`	Set the box for debugging given by lower left/upper right coordinates.
`-worker`	Debugs routes that pass through the point `{x, y}`.
`-iter`	Specifies the number of debug iterations. The default value is `0`, and the accepted values are integers `[0, MAX_INT`.
`-pa_markers`	Enable pin access markers.
`-dump_dr`	Filename for detailed routing dump.
`-dump_dir`	Directory for detailed routing dump.
`-pa_edge`	Enable visibility of pin access edges.
`-pa_commit`	Enable visibility of pin access commits.
`-write_net_tracks`	Enable writing of net track assigments.

Check Pin Access

This function checks pin access.

```
pin_access
    [-db_process_node name]
    [-bottom_routing_layer layer]
    [-top_routing_layer layer]
    [-min_access_points count]
    [-verbose level]
    [-distributed]
    [-remote_host rhost]
    [-remote_port rport]
    [-shared_volume vol]
    [-cloud_size sz]
```

Options

Switch Name	Description
`-db_process_node`	Specify process node.
`-bottom_routing_layer`	Bottommost routing layer.
`-top_routing_layer`	Topmost routing layer.
`-min_access_points`	Minimum number of access points per pin.
`-verbose`	Sets verbose mode if the value is greater than 1, else non-verbose mode (must be integer, or error will be triggered.)
`-distributed`	Refer to distributed arguments *here*.

Distributed Arguments

We have compiled all distributed arguments in this section.

Note: Additional setup is required. Please refer to this guide.

Switch Name	Description
`-distributed`	Enable distributed mode with Kubernetes and Google Cloud.
`-remote_host`	The host IP.
`-remote_port`	The value of the port to access from.
`-shared_volume`	The mount path of the nfs shared folder.
`-cloud_size`	The number of workers.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

Command Name	Description
`detailed_route_set_default_via`	Set default via.
`detailed_route_set_unidirectional_layer`	Set unidirectional layer.
`step_dr`	Refer to function `detailed_route_step_drt`.
`check_drc`	Refer to function `check_drc_cmd`.

Example scripts

Example script demonstrating how to run TritonRoute on a sample design of `gcd` in the Nangate45 technology node.

```
# single machine example
./test/gcd_nangate45.tcl

# distributed example
./test/gcd_nangate45_distributed.tcl
```

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

References

Please cite the following paper(s) for publication:

1. A. B. Kahng, L. Wang and B. Xu, "TritonRoute: The Open Source Detailed Router", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2020), doi:10.1109/TCAD.2020.3003234. (.pdf)
2. A. B. Kahng, L. Wang and B. Xu, "The Tao of PAO: Anatomy of a Pin Access Oracle for Detailed Routing", Proc. ACM/IEEE Design Automation Conf., 2020, pp. 1-6. (.pdf)

Authors

TritonRoute was developed by graduate students Lutong Wang and Bangqi Xu from UC San Diego, and serves as the detailed router in the OpenROAD project.

License

BSD 3-Clause License. See LICENSE file.

5.2.24 Metal fill

This module inserts floating metal fill shapes to meet metal density design rules while obeying DRC constraints. It is driven by a `json` configuration file.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Density Fill

This command performs density fill to meet metal density DRC rules.

```
density_fill
    [-rules rules_file]
    [-area {lx ly ux uy}]
```

Options

Switch Name	Description
`-rules`	Specify `json` rule file.
`-area`	Optional. If not specified, the core area will be used.

Example scripts

The rules `json` file controls fill and you can see an example here.

The schema for the `json` is:

```
{
  "layers": {
    "<group_name>": {
      "layers": "<list of integer gds layers>",
      "names": "<list of name strings>",
      "opc": {
        "datatype":  "<list of integer gds datatypes>",
        "width":   "<list of widths in microns>",
        "height":   "<list of heightsin microns>",
        "space_to_fill": "<real: spacing between fills in microns>",
        "space_to_non_fill": "<real: spacing to non-fill shapes in microns>",
        "space_line_end": "<real: spacing to end of line in microns>"
      },
      "non-opc": {
        "datatype":  "<list of integer gds datatypes>",
        "width":   "<list of widths in microns>",
        "height":   "<list of heightsin microns>",
        "space_to_fill": "<real: spacing between fills in microns>",
        "space_to_non_fill": "<real: spacing to non-fill shapes in microns>"
      }
    }, ...
  }
}
```

The `opc` section is optional depending on your process.

The width/height lists are effectively parallel arrays of shapes to try in left to right order (generally larger to smaller).

The layer grouping is for convenience. For example in some technologies many layers have similar rules so it is convenient to have a `Mx`, `Cx` group.

This all started out in `klayout` so there are some obsolete fields that the parser accepts but ignores (e.g., `space_to_outline`).

Regression tests

There are a set of regression tests in `./test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

License

BSD 3-Clause License. See LICENSE file.

5.2.25 Parasitics Extraction

The parasitics extraction module in OpenROAD (`rcx`) is based on the open-source OpenRCX, a Parasitic Extraction (PEX, or RCX) tool that works on OpenDB design APIs. It extracts routed designs based on the LEF/DEF layout model.

OpenRCX extracts both Resistance and Capacitance for wires, based on coupling distance to the nearest wire and the track density context over and/or under the wire of interest, as well as cell abstracts. The capacitance and resistance measurements are based on equations of coupling distance interpolated on exact measurements from a calibration file, called the Extraction Rules file. The Extraction Rules file (RC technology file) is generated once for every process node and corner, using a provided utility for DEF wire pattern generation and regression modeling.

OpenRCX stores resistance, coupling capacitance and ground (i.e., grounded) capacitance on OpenDB objects with direct pointers to the associated wire and via db objects. Optionally, OpenRCX can generate a `.spef` file.

Commands

Note:

- Parameters in square brackets `[-param param]` are optional.
- Parameters without square brackets `-param2 param2` are required.

Define Process Corner

This command defines proccess corner.

```
define_process_corner
    [-ext_model_index index]
    filename
```

Options

Switch Name	Description
`-ext_model_index`	Extraction model index. Expects 2 inputs (an index, and corner name).
`filename`	Path to process corner file `rcx_patterns.rules`.

Extract Parasitics

The `extract_parasitics` command performs parasitic extraction based on the routed design. If there are no information on routed design, no parasitics are returned.

```
extract_parasitics
    [-ext_model_file filename]
    [-corner_cnt count]
    [-max_res ohms]
    [-coupling_threshold fF]
    [-debug_net_id id]
    [-lef_res]
    [-cc_model track]
    [-context_depth depth]
    [-no_merge_via_res]
```

Options

Switch Name	Description
`-ext_model_file`	Specify the Extraction Rules file used for the extraction.
`-corner_cnt`	Defines the number of corners used during the parasitic extraction.
`-max_res`	Combines resistors in series up to the threshold value.
`-coupling_threshold`	Coupling below this threshold is grounded. The default value is `0.1`, units are in `fF`, accepted values are floats.
`-debug_net_id`	*Developer Option*: Net ID to evaluate.
`-lef_res`	Override LEF resistance per unit.
`-cc_model`	Specify the maximum number of tracks of lateral context that the tool considers on the same routing level. The default value is `10`, and the allowed values are integers `[0, MAX_INT]`.
`-context_depth`	Specify the number of levels of vertical context that OpenRCX needs to consider for the over/under context overlap for capacitance calculation. The default value is `5`, and the allowed values are integers `[0, MAX_INT]`.
`-no_merge_via_res`	Separates the via resistance from the wire resistance.

Write SPEF

The `write_spef` command writes the `.spef` output of the parasitics stored in the database.

```
write_spef
    [-net_id net_id]
    [-nets nets]
    [-coordinates]
    filename
```

Options

Switch Name	Description
`-net_id`	Output the parasitics info for specific net IDs.
`-nets`	Net name.
`coordinates`	Coordinates TBC.
`filename`	Output filename.

Scale RC

Use the `adjust_rc` command to scale the resistance, ground, and coupling capacitance.

```
adjust_rc
    [-res_factor res]
    [-cc_factor cc]
    [-gndc_factor gndc]
```

Options

Switch Name	Description
`-res_factor`	Scale factor for resistance.
`-cc_factor`	Scale factor for coupling capacitance.
`-gndc_factor`	Scale factor for ground capacitance.

Comparing different SPEF files

The `diff_spef` command compares the parasitics in the reference database `<filename>.spef`. The output of this command is `diff_spef.out` and contains the RC numbers from the parasitics in the database and the `<filename>.spef`, and the percentage RC difference of the two data.

```
diff_spef
    [-file filename]
    [-r_res]
    [-r_cap]
    [-r_cc_cap]
    [-r_conn]
```

Options

Switch Name	Description
-file	Path to the input .spef filename.
-r_res	Read resistance.
-r_cap	Read capacitance.
-r_cc_cap	Read coupled capacitance.
r_conn	Read connections.

Extraction Rules File Generation

The bench_wires command produces a layout which contains various patterns that are used to characterize per-unit length R and C values. The generated patterns model the lateral, vertical, and diagonal coupling capacitances, as well as ground capacitance effects. This command generates a .def file that contains a number of wire patterns.

This command is specifically intended for the Extraction Rules file generation only.

```
bench_wires
    [-met_cnt mcnt]
    [-cnt count]
    [-len wire_len]
    [-over]
    [-diag]
    [-all]
    [-db_only]
    [-under_met layer]
    [-w_list width]
    [-s_list space]
    [-over_dist dist]
    [-under_dist dist]
```

Options

Switch Name	Description
-met_cnt	Number of layers used in each pattern. The default value is -1, meaning it is not set, and the allowed values are integers [0, MAX_INT].
-cnt	Number of wires in each pattern. The default value is 5, and the default values are integers [0, MAX_INT].
-len	Wirelength in microns in the pattern. The default value is 100, and the allowed values are integers [0, MAX_INT].
-all	Consider all different pattern geometries (over, under, over_under, and diagonal).
-db_only	Run with db values only. All parameters in bench_wires are ignored.
-under_met	Consider under metal layer.
-w_list	Lists of wire width multipliers from the minimum spacing defined in the LEF.
-s_list	Lists of wire spacing multipliers from the minimum spacing defined in the LEF. The list will be the input index on the OpenRCX RC table (Extraction Rules file).
-over_dist, -under_dist	Consider over and under metal distance respectively.

Generate verilog netlist

bench_verilog is used after the bench_wires command. This command generates a Verilog netlist of the generated pattern layout by the bench_wires command.

This command is optional when running the Extraction Rules generation flow. This step is required if the favorite extraction tool (i.e., reference extractor) requires a Verilog netlist to extract parasitics of the pattern layout.

```
bench_verilog
    [filename]
```

Options

Switch Name	Description
filename	Name for the Verilog output file (e.g., output.v).

Read SPEF

The bench_read_spef command reads a <filename>.spef file and stores the parasitics into the database.

```
bench_read_spef
    [filename]
```

Options

Switch Name	Description
filename	Path to the input .spef file.

Write Rule File

The write_rules command writes the Extraction Rules file (RC technology file) for OpenRCX. It processes the parasitics data from the layout patterns that are generated using the bench_wires command, and writes the Extraction Rules file with <filename> as the output file.

This command is specifically intended for the purpose of Extraction Rules file generation.

```
write_rules
  [-file filename]
  [-dir dir]
  [-name name]
  [-pattern pattern]
  [-db]
```

Options

Switch Name	Description
`-file`	Output file name.
`-dir`	Output file directory.
`-name`	Name of rule.
`-pattern`	Flag to write the pattern to rulefile (0/1).
`-db`	DB tbc.

Example scripts

Example scripts demonstrating how to run OpenRCX in the OpenROAD environment on sample designs can be found in /test. An example flow test taking a sample design from synthesizable RTL Verilog to final-routed layout in an open-source SKY130 technology is shown below.

```
./test/gcd.tcl
```

Example scripts demonstrating how to run the Extraction Rules file generation can be found in this directory.

```
./calibration/script/generate_patterns.tcl     # generate patterns
./calibration/script/generate_rules.tcl        # generate the Extraction Rules file
./calibration/script/ext_patterns.tcl          # check the accuracy of OpenRCX
```

Regression tests

There are a set of regression tests in `/test`. For more information, refer to this *section*.

Simply run the following script:

```
./test/regression
```

Extraction Rules File Generation

This flow generates an Extraction Rules file (RC tech file, or RC table) for OpenRCX. This file provides resistance and capacitance tables used for RC extraction for a specific process corner.

The Extraction Rules file (RC technology file) is generated once for every process node and corner automatically.

The detailed documentation can be found *here*.

Limitations

FAQs

Check out GitHub discussion about this tool.

License

BSD 3-Clause License. See LICENSE file.

Extraction Rules Generation Flow for OpenRCX

This flow generates the RC tech file for OpenRCX. The RC tech file provides resistance and capacitance tables used for RC extraction for a specific process corner.

The flow involves:

A. Running OpenRCX `generate_patterns.tcl` to generate layout patterns.

- Input: tech LEF
- Output: `patterns.def`, `patterns.v`
- Script: `generate_patterns.tcl`
- Desc: OpenRCX generates many pattern geometries to model various types of capacitance and resistance (i.e., multi-conductor) geometric configurations.

B. Running your favorite extraction tool (i.e., reference extractor) to extract parasitics of the layout patterns.

- Input: `patterns.def`, `patterns.v` (if required), and additional files required by the reference extractor.
- Output: `patterns.spef`
- Script: Not provided
- Desc: Extract parasitics of the patterns generated by OpenRCX using a reference extractor. This one-time step provides the parasitics of various types of pattern geometries as reference for fitted per-unit length R, C calculation.

C. Running OpenRCX to convert `patterns.spef` to RC tech file.

- Input: `patterns.spef`
- Output: RC tech file
- Script: `generate_rules.tcl`
- Desc: OpenRCX takes the `.spef` from the reference extractor and performs calculations to produce capacitance and resistance tables for a wide range of wire geometries. The output of this flow is a custom RC tech file for OpenRCX.

D. Benchmarking - test the accuracy of OpenRCX on the patterns layout.

- Input: `patterns.def` and RC tech file
- Output: `rcx.spef`, `diff_spef.out`
- Script: `ext_patterns.tcl`
- Desc: Perform parasitic extraction on pattern layout for the calibration using the generated RC tech file. OpenRCX then compares the extracted parasitics with the golden parasitics that had been extracted by the reference extractor in Step (B) above.

How to run:

1. Go to OpenRCX home directory (`./OpenROAD/src/rcx`).
2. Navigate to calibration folder `cd calibration`
3. Modify the `user_env.tcl` script in the script directory.

- TECH_LEF: points to the directory of the tech LEF
- PROCESS_NODE: the technology node
- extRules: the name and the location of the OpenRCX tech file

4. Run the executable script `run.sh` –> run Steps (A) through (D) of the flow above.

- `source run.sh` or `./run.sh`

5. The OpenRCX RC tech file can be found in the directory that is specified in the extRules variable.

5.2.26 OpenROAD Messages Glossary

Listed below are the OpenROAD warning/errors you may encounter while using the application.

Tool	Code	Filename:Line Number	Type	Information
ANT	0001	AntennaChecker.cc:1776	INFO	-
ANT	0002	AntennaChecker.cc:1774	INFO	-
ANT	0008	AntennaChecker.cc:1733	ERROR	-
ANT	0009	AntennaChecker.cc:1996	WARN	-
ANT	0010	AntennaChecker.tcl:57	WARN	-
ANT	0011	AntennaChecker.tcl:60	WARN	-
ANT	0012	AntennaChecker.i:66	ERROR	-
ANT	0013	AntennaChecker.cc:205	WARN	-
ANT	0014	AntennaChecker.cc:1758	ERROR	-
CTS	0001	TritonCTS.cpp:155	INFO	-
CTS	0003	TritonCTS.cpp:445	INFO	-
CTS	0004	TritonCTS.cpp:449	INFO	-
CTS	0005	TritonCTS.cpp:453	INFO	-
CTS	0006	TritonCTS.cpp:457	INFO	-
CTS	0007	TritonCTS.cpp:841	INFO	-
CTS	0008	TritonCTS.cpp:866	INFO	-
CTS	0010	TritonCTS.cpp:985	INFO	-
CTS	0011	TritonCTS.cpp:1081	INFO	-
CTS	0012	TritonCTS.cpp:1287	INFO	-
CTS	0013	TritonCTS.cpp:1289	INFO	-
CTS	0014	TritonCTS.cpp:1293	INFO	-
CTS	0015	TritonCTS.cpp:1297	INFO	-
CTS	0016	TritonCTS.cpp:1306	INFO	-
CTS	0017	TritonCTS.cpp:1310	INFO	-
CTS	0018	TritonCTS.cpp:1460	INFO	-
CTS	0019	HTreeBuilder.cpp:271	INFO	-
CTS	0020	HTreeBuilder.cpp:283	INFO	-
CTS	0021	HTreeBuilder.cpp:292	INFO	-
CTS	0022	HTreeBuilder.cpp:300	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
CTS	0023	HTreeBuilder.cpp:327	INFO	-
CTS	0024	HTreeBuilder.cpp:344	INFO	-
CTS	0025	HTreeBuilder.cpp:345	INFO	-
CTS	0026	HTreeBuilder.cpp:346	INFO	-
CTS	0027	HTreeBuilder.cpp:1125	INFO	-
CTS	0028	HTreeBuilder.cpp:1127	INFO	-
CTS	0029	HTreeBuilder.cpp:1133	INFO	-
CTS	0030	HTreeBuilder.cpp:1141	INFO	-
CTS	0031	HTreeBuilder.cpp:1163	INFO	-
CTS	0032	HTreeBuilder.cpp:1175	INFO	-
CTS	0034	HTreeBuilder.cpp:1329	INFO	-
CTS	0035	HTreeBuilder.cpp:1897	INFO	-
CTS	0039	TechChar.cpp:1678	INFO	-
CTS	0040	TritonCTS.cpp:853	WARN	-
CTS	0041	TritonCTS.cpp:978	WARN	-
CTS	0043	TechChar.cpp:137	WARN	-
CTS	0045	TechChar.cpp:387	WARN	-
CTS	0046	TechChar.cpp:142	INFO	-
CTS	0047	TechChar.cpp:144	INFO	-
CTS	0048	TechChar.cpp:149	INFO	-
CTS	0049	TechChar.cpp:514	INFO	-
CTS	0050	TechChar.cpp:680	INFO	-
CTS	0051	TechChar.cpp:681	INFO	-
CTS	0052	TechChar.cpp:753	INFO	-
CTS	0053	TechChar.cpp:683	INFO	-
CTS	0056	TritonCTS.tcl:175	ERROR	-
CTS	0058	SinkClustering.cpp:174	ERROR	-
CTS	0065	TechChar.cpp:182	ERROR	-
CTS	0073	TechChar.cpp:470	ERROR	-
CTS	0074	TechChar.cpp:697	ERROR	-
CTS	0075	TechChar.cpp:576	ERROR	-
CTS	0076	TechChar.cpp:624	ERROR	-
CTS	0077	TreeBuilder.cpp:94	ERROR	-
CTS	0078	TechChar.cpp:668	ERROR	-
CTS	0079	HTreeBuilder.cpp:200	ERROR	-
CTS	0080	HTreeBuilder.cpp:1889	ERROR	-
CTS	0081	TritonCTS.cpp:140	ERROR	-
CTS	0082	TritonCTS.cpp:98	WARN	-
CTS	0083	TritonCTS.cpp:863	WARN	-
CTS	0085	TritonCTS.cpp:1265	ERROR	-
CTS	0087	TritonCTS.cpp:431	ERROR	-
CTS	0090	HTreeBuilder.cpp:1130	INFO	-
CTS	0093	LevelBalancer.cpp:52	INFO	-
CTS	0095	TritonCTS.cpp:839	INFO	-
CTS	0096	TechChar.cpp:597	ERROR	-
CTS	0097	TritonCTS.cpp:146	INFO	-
CTS	0098	TritonCTS.cpp:402	INFO	-
CTS	0099	TritonCTS.cpp:403	INFO	-
CTS	0100	TritonCTS.cpp:404	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
CTS	0101	TritonCTS.cpp:406	INFO	-
CTS	0102	TritonCTS.cpp:407	INFO	-
CTS	0103	TritonCTS.tcl:227	ERROR	-
CTS	0104	TechChar.cpp:450	WARN	-
CTS	0105	TritonCTS.cpp:1043	WARN	-
CTS	0106	TechChar.cpp:702	ERROR	-
CTS	0107	TechChar.cpp:611	ERROR	-
CTS	0108	TechChar.cpp:618	ERROR	-
CTS	0109	TritonCTS.tcl:201	ERROR	-
CTS	0110	TritonCTS.cpp:580	ERROR	-
CTS	0111	TechChar.cpp:495	ERROR	-
CTS	0112	TritonCTS.cpp:756	ERROR	-
CTS	0113	TechChar.cpp:508	ERROR	-
CTS	0114	TritonCTS.cpp:823	ERROR	-
CTS	0115	TritonCTS.tcl:112	WARN	-
CTS	0116	TritonCTS.cpp:192	INFO	-
CTS	0117	TritonCTS.cpp:750	ERROR	-
CTS	0118	TritonCTS.cpp:1856	ERROR	-
CTS	0119	TritonCTS.cpp:1883	ERROR	-
CTS	0120	TritonCTS.cpp:1841	ERROR	-
CTS	0121	TritonCTS.cpp:330	INFO	-
CTS	0122	TritonCTS.cpp:880	INFO	-
CTS	0123	TritonCTS.tcl:209	ERROR	-
CTS	0124	TritonCTS.cpp:387	INFO	-
CTS	0125	TritonCTS.cpp:389	INFO	-
CTS	0200	TreeBuilder.cpp:56	INFO	-
CTS	0201	TreeBuilder.cpp:71	INFO	-
CTS	0202	TritonCTS.cpp:1362	INFO	-
CTS	0203	TritonCTS.cpp:1598	WARN	-
CTS	0204	HTreeBuilder.cpp:160	INFO	-
CTS	0205	HTreeBuilder.cpp:167	INFO	-
CTS	0206	HTreeBuilder.cpp:173	INFO	-
CTS	0207	TritonCTS.cpp:409	INFO	-
CTS	0534	TechChar.cpp:532	ERROR	-
CTS	0541	TechChar.cpp:537	ERROR	-
CTS	0542	CtsOptions.h:162	ERROR	-
CTS	0543	CtsOptions.h:178	ERROR	-
DFT	0002	ScanReplace.cpp:384	WARN	-
DFT	0003	ScanReplace.cpp:374	INFO	-
DFT	0004	ClockDomain.cpp:52	ERROR	-
DFT	0005	ScanCellFactory.cpp:137	WARN	-
DFT	0006	dft.i:80	ERROR	-
DFT	0007	ScanCellFactory.cpp:146	WARN	-
DPL	0001	FillerPlacement.cpp:132	INFO	-
DPL	0002	FillerPlacement.cpp:184	ERROR	-
DPL	0012	dbToOpendp.cpp:141	ERROR	-
DPL	0013	Grid.cpp:865	ERROR	-
DPL	0014	Place.cpp:902	ERROR	-
DPL	0015	Place.cpp:314	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DPL	0016	Place.cpp:432	ERROR	-
DPL	0017	Place.cpp:486	ERROR	-
DPL	0018	Place.cpp:625	ERROR	-
DPL	0019	Place.cpp:1116	ERROR	-
DPL	0020	OptMirror.cpp:112	INFO	-
DPL	0021	OptMirror.cpp:114	INFO	-
DPL	0022	OptMirror.cpp:116	INFO	-
DPL	0023	OptMirror.cpp:121	INFO	-
DPL	0026	Place.cpp:1362	CRITICAL	-
DPL	0027	Opendp.tcl:80	ERROR	-
DPL	0028	Opendp.tcl:217	WARN	-
DPL	0029	Opendp.tcl:234	ERROR	-
DPL	0030	Opendp.tcl:250	ERROR	-
DPL	0031	Opendp.tcl:56	ERROR	-
DPL	0032	Opendp.tcl:192	ERROR	-
DPL	0033	CheckPlacement.cpp:144	ERROR	-
DPL	0034	Opendp.cpp:176	INFO	-
DPL	0035	Opendp.cpp:181	INFO	-
DPL	0036	Opendp.cpp:188	ERROR	-
DPL	0037	Opendp.cpp:147	WARN	-
DPL	0038	Opendp.cpp:163	WARN	-
DPL	0039	Opendp.tcl:221	ERROR	-
DPL	0040	CheckPlacement.cpp:225	ERROR	-
DPL	0041	Grid.cpp:912	ERROR	-
DPL	0042	Grid.cpp:691	WARN	-
DPL	0043	Opendp.cpp:551	ERROR	-
DPL	0044	Opendp.cpp:557	ERROR	-
DPL	0045	CheckPlacement.cpp:287	ERROR	-
DPL	0046	Opendp.cpp:570	ERROR	-
DPL	0048	CheckPlacement.cpp:166	ERROR	-
DPL	0049	dbToOpendp.cpp:144	ERROR	-
DPL	0050	FillerPlacement.cpp:250	ERROR	-
DPL	0128	Grid.cpp:185	ERROR	-
DPL	1599	Place.cpp:1029	ERROR	-
DPL	4219	Opendp.cpp:587	ERROR	-
DPL	5211	Opendp.cpp:579	ERROR	-
DPO	0001	detailed.cxx:174	ERROR	-
DPO	0031	Optdp.tcl:59	ERROR	-
DPO	0100	Optdp.cpp:433	INFO	-
DPO	0101	Optdp.cpp:562	ERROR	-
DPO	0102	Optdp.cpp:596	ERROR	-
DPO	0103	Optdp.cpp:624	ERROR	-
DPO	0104	Optdp.cpp:637	ERROR	-
DPO	0105	Optdp.cpp:653	ERROR	-
DPO	0106	Optdp.cpp:663	ERROR	-
DPO	0107	Optdp.cpp:670	ERROR	-
DPO	0108	Optdp.cpp:750	WARN	-
DPO	0109	Optdp.cpp:677	INFO	-
DPO	0110	Optdp.cpp:929	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DPO	0200	legalize_shift.cxx:198	WARN	-
DPO	0201	legalize_shift.cxx:214	WARN	-
DPO	0202	detailed_mis.cxx:185	INFO	-
DPO	0203	detailed_random.cxx:398	INFO	-
DPO	0300	detailed_mis.cxx:164	INFO	-
DPO	0301	detailed_mis.cxx:194	INFO	-
DPO	0302	detailed_mis.cxx:226	INFO	-
DPO	0303	detailed.cxx:176	INFO	-
DPO	0304	detailed_reorder.cxx:113	INFO	-
DPO	0305	detailed_reorder.cxx:127	INFO	-
DPO	0306	detailed_global.cxx:133	INFO	-
DPO	0307	detailed_global.cxx:141	INFO	-
DPO	0308	detailed_vertical.cxx:135	INFO	-
DPO	0309	detailed_vertical.cxx:147	INFO	-
DPO	0310	detailed_manager.cxx:949	INFO	-
DPO	0311	detailed_manager.cxx:1351	INFO	-
DPO	0312	detailed_manager.cxx:1406	INFO	-
DPO	0313	detailed_manager.cxx:1587	INFO	-
DPO	0314	detailed_manager.cxx:1645	INFO	-
DPO	0315	detailed_manager.cxx:1677	INFO	-
DPO	0317	detailed_abu.cxx:422	INFO	-
DPO	0318	detailed_manager.cxx:1187	INFO	-
DPO	0319	detailed_manager.cxx:1231	INFO	-
DPO	0320	detailed_manager.cxx:1261	INFO	-
DPO	0321	detailed_manager.cxx:1293	INFO	-
DPO	0322	detailed_manager.cxx:303	INFO	-
DPO	0323	detailed.cxx:131	WARN	-
DPO	0324	detailed_random.cxx:192	INFO	-
DPO	0325	detailed_random.cxx:242	INFO	-
DPO	0326	detailed_random.cxx:264	INFO	-
DPO	0327	detailed_random.cxx:302	INFO	-
DPO	0328	detailed_random.cxx:321	INFO	-
DPO	0329	detailed_random.cxx:390	INFO	-
DPO	0330	detailed_random.cxx:428	INFO	-
DPO	0332	detailed_random.cxx:501	INFO	-
DPO	0333	detailed_random.cxx:515	INFO	-
DPO	0334	detailed_global.cxx:542	INFO	-
DPO	0335	detailed_random.cxx:671	INFO	-
DPO	0336	detailed_vertical.cxx:535	INFO	-
DPO	0337	detailed_random.cxx:843	INFO	-
DPO	0338	detailed_random.cxx:534	INFO	-
DPO	0339	detailed_manager.cxx:1467	WARN	-
DPO	0340	detailed_manager.cxx:1477	WARN	-
DPO	0380	detailed_orient.cxx:104	INFO	-
DPO	0381	detailed_orient.cxx:112	WARN	-
DPO	0382	detailed_orient.cxx:118	INFO	-
DPO	0383	detailed_orient.cxx:127	INFO	-
DPO	0384	detailed_orient.cxx:132	INFO	-
DPO	0385	detailed_random.cxx:589	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DPO	0400	detailed_manager.cxx:171	ERROR	-
DPO	0401	detailed_manager.cxx:134	INFO	-
DPO	0402	detailed_manager.cxx:152	INFO	-
DRT	0000	FlexDR_init.cpp:352	ERROR	-
DRT	0001	TritonRoute.cpp:1054	ERROR	-
DRT	0002	TritonRoute.cpp:203	ERROR	-
DRT	0003	io.cpp:3570	ERROR	-
DRT	0004	io.cpp:3576	ERROR	-
DRT	0005	frRegionQuery.cpp:120	ERROR	-
DRT	0006	frRegionQuery.cpp:131	ERROR	-
DRT	0007	frRegionQuery.cpp:149	ERROR	-
DRT	0008	frRegionQuery.cpp:321	ERROR	-
DRT	0009	frRegionQuery.cpp:347	ERROR	-
DRT	0010	frRegionQuery.cpp:357	ERROR	-
DRT	0011	frRegionQuery.cpp:381	ERROR	-
DRT	0012	frRegionQuery.cpp:392	ERROR	-
DRT	0013	frRegionQuery.cpp:403	ERROR	-
DRT	0014	frRegionQuery.cpp:451	ERROR	-
DRT	0015	frRegionQuery.cpp:468	ERROR	-
DRT	0016	frRegionQuery.cpp:513	ERROR	-
DRT	0017	frRegionQuery.cpp:532	ERROR	-
DRT	0018	frRegionQuery.cpp:736	INFO	-
DRT	0019	frRegionQuery.cpp:740	INFO	-
DRT	0020	frRegionQuery.cpp:752	INFO	-
DRT	0021	frRegionQuery.cpp:756	INFO	-
DRT	0022	frRegionQuery.cpp:773	INFO	-
DRT	0023	frRegionQuery.cpp:784	INFO	-
DRT	0024	frRegionQuery.cpp:794	INFO	-
DRT	0026	frRegionQuery.cpp:824	INFO	-
DRT	0027	frRegionQuery.cpp:828	INFO	-
DRT	0028	frRegionQuery.cpp:839	INFO	-
DRT	0029	frRegionQuery.cpp:868	INFO	-
DRT	0030	frRegionQuery.cpp:872	INFO	-
DRT	0031	frRegionQuery.cpp:160	ERROR	-
DRT	0032	frRegionQuery.cpp:1001	INFO	-
DRT	0033	frRegionQuery.cpp:1015	INFO	-
DRT	0034	frRegionQuery.cpp:1043	INFO	-
DRT	0035	frRegionQuery.cpp:882	INFO	-
DRT	0036	frRegionQuery.cpp:1029	INFO	-
DRT	0037	FlexGC_init.cpp:88	ERROR	-
DRT	0038	FlexGC_init.cpp:99	ERROR	-
DRT	0039	FlexGC_init.cpp:105	ERROR	-
DRT	0041	FlexGC_main.cpp:151	WARN	-
DRT	0042	FlexGC_main.cpp:225	ERROR	-
DRT	0043	FlexGC_main.cpp:305	WARN	-
DRT	0044	FlexGC_main.cpp:2706	WARN	-
DRT	0045	FlexGC_main.cpp:2849	WARN	-
DRT	0046	FlexGC_main.cpp:2856	WARN	-
DRT	0047	FlexGC_main.cpp:2863	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DRT	0048	FlexGC_main.cpp:2870	WARN	-
DRT	0050	FlexGC_main.cpp:2877	WARN	-
DRT	0051	FlexGC_main.cpp:2884	WARN	-
DRT	0052	FlexGC_main.cpp:2891	WARN	-
DRT	0053	FlexGC_init.cpp:995	ERROR	-
DRT	0054	FlexGC_main.cpp:2981	WARN	-
DRT	0055	FlexGC_main.cpp:2986	WARN	-
DRT	0056	FlexGC_main.cpp:2994	WARN	-
DRT	0057	FlexGC_main.cpp:3002	WARN	-
DRT	0058	FlexGC_main.cpp:3007	WARN	-
DRT	0059	FlexGC_main.cpp:3012	WARN	-
DRT	0060	FlexGC_main.cpp:3018	WARN	-
DRT	0061	FlexGC_main.cpp:3025	WARN	-
DRT	0062	FlexGC_main.cpp:3030	WARN	-
DRT	0063	FlexGC_main.cpp:3035	WARN	-
DRT	0065	FlexPA_unique.cpp:89	ERROR	-
DRT	0066	FlexPA_unique.cpp:98	WARN	-
DRT	0067	FlexPA_prep.cpp:116	ERROR	-
DRT	0068	FlexPA_prep.cpp:458	ERROR	-
DRT	0069	FlexPA_unique.cpp:253	ERROR	-
DRT	0070	FlexPA_prep.cpp:806	ERROR	-
DRT	0073	FlexPA_prep.cpp:1563	ERROR	-
DRT	0074	FlexPA_prep.cpp:1612	ERROR	-
DRT	0076	FlexPA_prep.cpp:1575	INFO	-
DRT	0077	FlexPA_prep.cpp:1579	INFO	-
DRT	0078	FlexPA_prep.cpp:1623	INFO	-
DRT	0079	FlexPA_prep.cpp:1790	INFO	-
DRT	0080	FlexPA_prep.cpp:1795	INFO	-
DRT	0081	FlexPA_prep.cpp:1807	INFO	-
DRT	0082	FlexPA_prep.cpp:1720	INFO	-
DRT	0083	FlexPA_prep.cpp:1724	INFO	-
DRT	0084	FlexPA_prep.cpp:1736	INFO	-
DRT	0085	FlexPA_prep.cpp:2034	ERROR	-
DRT	0086	FlexPA_prep.cpp:2292	ERROR	-
DRT	0087	FlexPA_prep.cpp:1776	WARN	-
DRT	0089	FlexPA_prep.cpp:2479	WARN	-
DRT	0090	FlexPA_prep.cpp:2727	ERROR	-
DRT	0091	FlexPA_prep.cpp:2800	ERROR	-
DRT	0092	FlexRP_prep.cpp:1522	WARN	-
DRT	0093	FlexRP_prep.cpp:1546	WARN	-
DRT	0094	io.cpp:86	ERROR	-
DRT	0095	io.cpp:129	ERROR	-
DRT	0096	io.cpp:134	ERROR	-
DRT	0097	io.cpp:214	ERROR	-
DRT	0098	io.cpp:224	ERROR	-
DRT	0099	io.cpp:245	ERROR	-
DRT	0100	io.cpp:357	ERROR	-
DRT	0101	io.cpp:360	ERROR	-
DRT	0102	io.cpp:507	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DRT	0103	io.cpp:537	ERROR	-
DRT	0104	io.cpp:586	ERROR	-
DRT	0105	io.cpp:612	ERROR	-
DRT	0106	io.cpp:619	ERROR	-
DRT	0107	io.cpp:708	ERROR	-
DRT	0108	io.cpp:851	ERROR	-
DRT	0109	io.cpp:915	ERROR	-
DRT	0110	FlexPA_prep.cpp:1674	INFO	-
DRT	0111	FlexPA_prep.cpp:1678	INFO	-
DRT	0112	io.cpp:1048	ERROR	-
DRT	0113	io.cpp:3241	ERROR	-
DRT	0114	io.cpp:3377	ERROR	-
DRT	0115	FlexPA_graphics.cpp:66	INFO	-
DRT	0116	io.cpp:1212	ERROR	-
DRT	0117	io.cpp:1216	ERROR	-
DRT	0118	TritonRoute.tcl:272	ERROR	-
DRT	0119	FlexPA_graphics.cpp:271	INFO	-
DRT	0122	io.cpp:2398	WARN	-
DRT	0123	io.cpp:2431	WARN	-
DRT	0124	io.cpp:2662	WARN	-
DRT	0125	io.cpp:2686	ERROR	-
DRT	0126	io.cpp:2694	ERROR	-
DRT	0127	io.cpp:2705	ERROR	-
DRT	0128	io.cpp:2543	ERROR	-
DRT	0129	io.cpp:2550	ERROR	-
DRT	0130	io.cpp:2564	ERROR	-
DRT	0131	io.cpp:2585	WARN	-
DRT	0132	io.cpp:2606	WARN	-
DRT	0133	io.cpp:2616	WARN	-
DRT	0134	io.cpp:2631	WARN	-
DRT	0135	io.cpp:2641	WARN	-
DRT	0136	io.cpp:2759	ERROR	-
DRT	0139	io.cpp:1958	WARN	-
DRT	0141	io.cpp:1999	WARN	-
DRT	0142	io.cpp:2001	WARN	-
DRT	0143	io.cpp:2003	WARN	-
DRT	0147	io.cpp:2248	WARN	-
DRT	0149	io.cpp:2754	INFO	-
DRT	0150	io.cpp:1207	INFO	-
DRT	0153	io.cpp:2823	ERROR	-
DRT	0154	io.cpp:2828	ERROR	-
DRT	0155	io.cpp:2846	ERROR	-
DRT	0156	io.cpp:2869	INFO	-
DRT	0157	io.cpp:2873	INFO	-
DRT	0160	io_parser_helper.cpp:167	WARN	-
DRT	0161	io.cpp:1687	WARN	-
DRT	0162	io_pin.cpp:36	INFO	-
DRT	0163	io_pin.cpp:82	INFO	-
DRT	0164	io_pin.cpp:136	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DRT	0165	FlexPA.cpp:180	INFO	-
DRT	0166	FlexPA.cpp:217	INFO	-
DRT	0167	frTechObject.h:263	INFO	-
DRT	0168	io_parser_helper.cpp:812	INFO	-
DRT	0169	io_parser_helper.cpp:825	INFO	-
DRT	0170	io_parser_helper.cpp:1007	ERROR	-
DRT	0171	io_parser_helper.cpp:1012	ERROR	-
DRT	0172	io_parser_helper.cpp:1062	ERROR	-
DRT	0173	io_parser_helper.cpp:1067	ERROR	-
DRT	0174	io_parser_helper.cpp:1107	ERROR	-
DRT	0175	io_parser_helper.cpp:1123	ERROR	-
DRT	0176	io_parser_helper.cpp:1129	INFO	-
DRT	0177	io_parser_helper.cpp:1135	INFO	-
DRT	0178	io_parser_helper.cpp:863	INFO	-
DRT	0179	io_parser_helper.cpp:866	INFO	-
DRT	0180	io.cpp:3216	INFO	-
DRT	0181	FlexTA.cpp:333	INFO	-
DRT	0182	FlexTA.cpp:346	INFO	-
DRT	0183	FlexTA.cpp:223	INFO	-
DRT	0184	FlexTA.cpp:241	INFO	-
DRT	0185	io_parser_helper.cpp:882	INFO	-
DRT	0186	FlexTA.cpp:307	INFO	-
DRT	0187	FlexDR.cpp:460	INFO	-
DRT	0190	io.cpp:231	WARN	-
DRT	0191	io.cpp:251	WARN	-
DRT	0192	io.cpp:2671	WARN	-
DRT	0193	io.cpp:2370	WARN	-
DRT	0194	FlexDR.cpp:474	INFO	-
DRT	0195	FlexDR.cpp:555	INFO	-
DRT	0198	FlexDR.cpp:836	INFO	-
DRT	0199	FlexDR.cpp:774	INFO	-
DRT	0201	FlexGR.cpp:658	ERROR	-
DRT	0202	FlexGR.cpp:687	WARN	-
DRT	0203	FlexGR.cpp:666	WARN	-
DRT	0204	TritonRoute.cpp:1112	WARN	-
DRT	0206	FlexDR_conn.cpp:1328	ERROR	-
DRT	0207	FlexDR_graphics.cpp:761	INFO	-
DRT	0210	io.cpp:1869	WARN	-
DRT	0214	io_guide.cpp:896	WARN	-
DRT	0215	io_guide.cpp:905	WARN	-
DRT	0216	io_guide.cpp:914	WARN	-
DRT	0217	io_guide.cpp:919	WARN	-
DRT	0218	io_guide.cpp:956	ERROR	-
DRT	0219	io_guide.cpp:958	ERROR	-
DRT	0220	io_guide.cpp:1005	WARN	-
DRT	0221	io_guide.cpp:1024	WARN	-
DRT	0222	io_guide.cpp:1032	WARN	-
DRT	0223	io_guide.cpp:1086	ERROR	-
DRT	0224	io_guide.cpp:1291	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DRT	0225	io_guide.cpp:1301	WARN	-
DRT	0226	FlexGC_eol.cpp:487	ERROR	-
DRT	0228	io_guide.cpp:379	ERROR	-
DRT	0229	io_guide.cpp:510	ERROR	-
DRT	0230	io_guide.cpp:601	WARN	-
DRT	0231	io_guide.cpp:617	WARN	-
DRT	0232	io_guide.cpp:558	ERROR	-
DRT	0233	io_parser_helper.cpp:133	ERROR	-
DRT	0234	io_parser_helper.cpp:125	ERROR	-
DRT	0235	io_parser_helper.cpp:242	WARN	-
DRT	0236	io_parser_helper.cpp:249	INFO	-
DRT	0237	io_parser_helper.cpp:267	WARN	-
DRT	0238	io_parser_helper.cpp:274	INFO	-
DRT	0239	io_parser_helper.cpp:331	ERROR	-
DRT	0242	io_parser_helper.cpp:338	ERROR	-
DRT	0243	io_parser_helper.cpp:349	ERROR	-
DRT	0245	io_parser_helper.cpp:871	INFO	-
DRT	0246	io_parser_helper.cpp:916	WARN	-
DRT	0247	io.cpp:2901	WARN	-
DRT	0248	io_pin.cpp:71	WARN	-
DRT	0249	FlexDR_graphics.cpp:682	INFO	-
DRT	0250	FlexDR_graphics.cpp:685	INFO	-
DRT	0253	FlexGC_init.cpp:179	ERROR	-
DRT	0255	FlexDR_maze.cpp:1820	ERROR	-
DRT	0256	io.cpp:399	WARN	-
DRT	0258	io.cpp:1680	WARN	-
DRT	0259	io.cpp:1694	WARN	-
DRT	0261	io.cpp:1715	WARN	-
DRT	0262	io.cpp:1722	WARN	-
DRT	0263	io.cpp:1729	WARN	-
DRT	0267	frTime.cpp:47	INFO	-
DRT	0268	FlexTA.cpp:289	INFO	-
DRT	0269	FlexGC_eol.cpp:58	ERROR	-
DRT	0270	FlexGC_eol.cpp:240	ERROR	-
DRT	0271	FlexGC_eol.cpp:412	ERROR	-
DRT	0272	io.cpp:2772	WARN	-
DRT	0273	io.cpp:2784	WARN	-
DRT	0275	FlexDR_graphics.cpp:688	INFO	-
DRT	0276	FlexPA_prep.cpp:2102	ERROR	-
DRT	0277	FlexPA_prep.cpp:2879	ERROR	-
DRT	0278	FlexPA_prep.cpp:2917	ERROR	-
DRT	0279	io.cpp:1741	WARN	-
DRT	0280	FlexPA_graphics.cpp:344	WARN	-
DRT	0281	FlexPA_graphics.cpp:350	INFO	-
DRT	0282	io.cpp:175	WARN	-
DRT	0290	TritonRoute.cpp:1318	WARN	-
DRT	0291	TritonRoute.cpp:1387	ERROR	-
DRT	0292	FlexPA_graphics.cpp:306	INFO	-
DRT	0293	FlexPA_graphics.cpp:74	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DRT	0294	io.cpp:3465	ERROR	-
DRT	0295	io.cpp:3470	ERROR	-
DRT	0296	io.cpp:3474	ERROR	-
DRT	0297	io.cpp:3501	ERROR	-
DRT	0298	io.cpp:3508	ERROR	-
DRT	0299	io.cpp:3512	ERROR	-
DRT	0300	io.cpp:3525	ERROR	-
DRT	0301	io.cpp:3537	ERROR	-
DRT	0302	io.cpp:1011	ERROR	-
DRT	0303	io.cpp:3546	ERROR	-
DRT	0304	TritonRoute.cpp:855	ERROR	-
DRT	0305	io.cpp:552	ERROR	-
DRT	0306	io.cpp:575	ERROR	-
DRT	0307	io.cpp:601	ERROR	-
DRT	0308	TritonRoute.tcl:468	ERROR	-
DRT	0311	FlexGC_main.cpp:2044	WARN	-
DRT	0312	FlexGC_main.cpp:2051	WARN	-
DRT	0313	FlexGC_main.cpp:2056	WARN	-
DRT	0314	FlexGC_main.cpp:2059	WARN	-
DRT	0315	FlexGC_main.cpp:2062	WARN	-
DRT	0316	FlexGC_main.cpp:2065	WARN	-
DRT	0317	io.cpp:2143	WARN	-
DRT	0318	io.cpp:2151	WARN	-
DRT	0319	io.cpp:2159	WARN	-
DRT	0320	FlexPA_unique.cpp:222	ERROR	-
DRT	0321	FlexPA_unique.cpp:232	ERROR	-
DRT	0322	FlexPA_unique.cpp:240	ERROR	-
DRT	0324	io.cpp:1783	WARN	-
DRT	0325	io.cpp:1791	WARN	-
DRT	0326	io.cpp:1799	WARN	-
DRT	0327	io.cpp:1807	WARN	-
DRT	0328	io.cpp:1815	WARN	-
DRT	0329	FlexPA_prep.cpp:1656	ERROR	-
DRT	0330	FlexPA_prep.cpp:1823	ERROR	-
DRT	0331	FlexPA.cpp:152	ERROR	-
DRT	0332	FlexPA_prep.cpp:1702	ERROR	-
DRT	0333	io.cpp:1476	WARN	-
DRT	0334	io.cpp:1481	WARN	-
DRT	0335	io.cpp:1511	WARN	-
DRT	0336	io_parser_helper.cpp:213	ERROR	-
DRT	0337	io.cpp:342	ERROR	-
DRT	0338	io.cpp:389	ERROR	-
DRT	0339	io.cpp:2745	ERROR	-
DRT	0400	io.cpp:1359	WARN	-
DRT	0401	io.cpp:1367	WARN	-
DRT	0403	io.cpp:1375	WARN	-
DRT	0404	io.cpp:1137	ERROR	-
DRT	0405	io.cpp:1141	ERROR	-
DRT	0406	FlexTA_assign.cpp:624	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DRT	0410	FlexGC_main.cpp:2183	ERROR	-
DRT	0411	FlexGC_main.cpp:2188	ERROR	-
DRT	0412	FlexTA_assign.cpp:802	ERROR	-
DRT	0415	TritonRoute.cpp:1146	ERROR	-
DRT	0416	io_parser_helper.cpp:629	ERROR	-
DRT	0417	io_parser_helper.cpp:681	ERROR	-
DRT	0418	io_parser_helper.cpp:787	WARN	-
DRT	0419	io_parser_helper.cpp:790	WARN	-
DRT	0421	io_parser_helper.cpp:749	WARN	-
DRT	0422	io_parser_helper.cpp:752	WARN	-
DRT	0423	FlexDR_maze.cpp:2144	ERROR	-
DRT	0500	FlexDR.cpp:1288	ERROR	-
DRT	0506	TritonRoute.tcl:170	ERROR	-
DRT	0507	TritonRoute.tcl:175	ERROR	-
DRT	0508	TritonRoute.tcl:180	ERROR	-
DRT	0512	frRegionQuery.cpp:306	ERROR	-
DRT	0513	frRegionQuery.cpp:233	ERROR	-
DRT	0516	TritonRoute.tcl:185	ERROR	-
DRT	0517	TritonRoute.tcl:376	ERROR	-
DRT	0519	FlexPA_prep.cpp:950	WARN	-
DRT	0520	TritonRoute.tcl:382	ERROR	-
DRT	0550	FlexGridGraph.h:1044	ERROR	-
DRT	0551	FlexGridGraph.h:1056	ERROR	-
DRT	0552	TritonRoute.tcl:339	ERROR	-
DRT	0553	TritonRoute.tcl:344	ERROR	-
DRT	0554	TritonRoute.tcl:349	ERROR	-
DRT	0555	TritonRoute.tcl:354	ERROR	-
DRT	0606	TritonRoute.cpp:559	WARN	-
DRT	0607	TritonRoute.cpp:571	WARN	-
DRT	0608	io_parser_helper.cpp:85	ERROR	-
DRT	0610	TritonRoute.cpp:1214	ERROR	-
DRT	0611	TritonRoute.cpp:1220	ERROR	-
DRT	0612	TritonRoute.tcl:487	ERROR	-
DRT	0613	TritonRoute.tcl:494	ERROR	-
DRT	0615	TritonRoute.cpp:1229	ERROR	-
DRT	0616	TritonRoute.cpp:1234	ERROR	-
DRT	0617	TritonRoute.cpp:583	WARN	-
DRT	0618	TritonRoute.cpp:1237	ERROR	-
DRT	0999	FlexDR.cpp:1306	ERROR	-
DRT	1000	io_guide.cpp:103	INFO	-
DRT	1001	io_guide.cpp:153	WARN	-
DRT	1002	io_guide.cpp:221	ERROR	-
DRT	1003	FlexPA_prep.cpp:402	ERROR	-
DRT	1004	FlexPA_prep.cpp:410	ERROR	-
DRT	1005	FlexPA_prep.cpp:421	ERROR	-
DRT	1006	FlexDR_maze.cpp:1876	ERROR	-
DRT	1007	io_guide.cpp:101	ERROR	-
DRT	1008	io_guide.cpp:282	ERROR	-
DRT	1009	FlexDR_init.cpp:1325	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DRT	1010	FlexDR_init.cpp:66	ERROR	-
DRT	1011	io.cpp:1181	ERROR	-
DRT	12304	TritonRoute.cpp:775	ERROR	-
DRT	2000	FlexDR_maze.cpp:2174	INFO	-
DRT	2001	FlexDR_maze.cpp:1584	INFO	-
DRT	2002	FlexDR_maze.cpp:1798	INFO	-
DRT	2003	FlexDR_maze.cpp:1864	INFO	-
DRT	2005	FlexDR_maze.cpp:2054	INFO	-
DRT	2006	FlexDR_maze.cpp:2058	INFO	-
DRT	2007	FlexDR_maze.cpp:2092	INFO	-
DRT	2008	TritonRoute.tcl:260	ERROR	-
DRT	3000	io_guide.cpp:302	ERROR	-
DRT	4000	FlexPA_graphics.cpp:79	INFO	-
DRT	4500	FlexGC_main.cpp:2158	ERROR	-
DRT	4501	FlexGC_main.cpp:2171	ERROR	-
DRT	5000	FlexPA_graphics.cpp:86	WARN	-
DRT	6000	FlexPA_prep.cpp:773	WARN	-
DRT	6001	FlexDR_conn.cpp:1028	WARN	-
DRT	7461	FlexDR.cpp:1264	ERROR	-
DRT	9199	TritonRoute.cpp:482	ERROR	-
DRT	9504	TritonRoute.cpp:809	ERROR	-
DRT	9999	TritonRoute.cpp:418	ERROR	-
DST	0001	Distributed.cc:92	ERROR	-
DST	0002	Distributed.tcl:42	ERROR	-
DST	0003	Distributed.tcl:47	ERROR	-
DST	0004	WorkerConnection.cc:122	WARN	-
DST	0005	WorkerConnection.cc:112	WARN	-
DST	0006	BalancerConnection.cc:102	WARN	-
DST	0007	LoadBalancer.cc:43	INFO	-
DST	0008	BalancerConnection.cc:232	WARN	-
DST	0009	Distributed.cc:110	ERROR	-
DST	0010	Distributed.tcl:66	ERROR	-
DST	0011	Distributed.tcl:71	ERROR	-
DST	0012	Distributed.cc:226	WARN	-
DST	0013	Distributed.cc:236	WARN	-
DST	0014	Distributed.cc:270	WARN	-
DST	0016	Distributed.tcl:91	ERROR	-
DST	0017	Distributed.tcl:96	ERROR	-
DST	0020	Distributed.cc:279	WARN	-
DST	0022	Distributed.cc:289	WARN	-
DST	0041	WorkerConnection.cc:78	WARN	-
DST	0042	BalancerConnection.cc:87	WARN	-
DST	0112	Distributed.cc:165	WARN	-
DST	0113	Distributed.cc:175	WARN	-
DST	0114	Distributed.cc:201	WARN	-
DST	0203	LoadBalancer.cc:199	WARN	-
DST	0204	BalancerConnection.cc:137	WARN	-
DST	0205	BalancerConnection.cc:147	WARN	-
DST	0207	BalancerConnection.cc:210	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
DST	9999	Distributed.cc:256	ERROR	-
FIN	0001	DensityFill.cpp:179	ERROR	-
FIN	0002	DensityFill.cpp:190	ERROR	-
FIN	0003	DensityFill.cpp:450	INFO	-
FIN	0004	DensityFill.cpp:488	INFO	-
FIN	0005	DensityFill.cpp:507	INFO	-
FIN	0006	DensityFill.cpp:513	INFO	-
FIN	0007	finale.tcl:49	ERROR	-
FIN	0008	finale.tcl:55	ERROR	-
FIN	0009	DensityFill.cpp:475	INFO	-
FIN	0010	DensityFill.cpp:533	WARN	-
GPL	0002	placerBase.cpp:778	INFO	-
GPL	0003	placerBase.cpp:806	INFO	-
GPL	0004	placerBase.cpp:807	INFO	-
GPL	0005	placerBase.cpp:808	INFO	-
GPL	0006	placerBase.cpp:1263	INFO	-
GPL	0007	placerBase.cpp:1267	INFO	-
GPL	0008	placerBase.cpp:1268	INFO	-
GPL	0009	placerBase.cpp:1269	INFO	-
GPL	0010	placerBase.cpp:1270	INFO	-
GPL	0011	placerBase.cpp:1271	INFO	-
GPL	0012	placerBase.cpp:1273	INFO	-
GPL	0013	placerBase.cpp:1274	INFO	-
GPL	0014	placerBase.cpp:1275	INFO	-
GPL	0015	placerBase.cpp:1276	INFO	-
GPL	0016	placerBase.cpp:1282	INFO	-
GPL	0017	placerBase.cpp:1283	INFO	-
GPL	0018	placerBase.cpp:1285	INFO	-
GPL	0019	placerBase.cpp:1286	INFO	-
GPL	0020	placerBase.cpp:1288	INFO	-
GPL	0021	placerBase.cpp:1289	INFO	-
GPL	0023	nesterovBase.cpp:614	INFO	-
GPL	0024	nesterovBase.cpp:615	INFO	-
GPL	0025	nesterovBase.cpp:616	INFO	-
GPL	0026	nesterovBase.cpp:617	INFO	-
GPL	0027	nesterovBase.cpp:618	INFO	-
GPL	0028	nesterovBase.cpp:648	INFO	-
GPL	0029	nesterovBase.cpp:653	INFO	-
GPL	0030	nesterovBase.cpp:670	INFO	-
GPL	0031	nesterovBase.cpp:1292	INFO	-
GPL	0032	nesterovBase.cpp:1293	INFO	-
GPL	0033	nesterovBase.cpp:1294	INFO	-
GPL	0034	nesterovBase.cpp:1651	INFO	-
GPL	0035	nesterovBase.cpp:1669	INFO	-
GPL	0036	routeBase.cpp:149	INFO	-
GPL	0037	routeBase.cpp:150	INFO	-
GPL	0038	routeBase.cpp:151	INFO	-
GPL	0039	routeBase.cpp:152	INFO	-
GPL	0040	routeBase.cpp:174	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
GPL	0045	routeBase.cpp:608	INFO	-
GPL	0046	routeBase.cpp:609	INFO	-
GPL	0047	routeBase.cpp:627	INFO	-
GPL	0048	routeBase.cpp:628	INFO	-
GPL	0049	routeBase.cpp:640	INFO	-
GPL	0050	routeBase.cpp:641	INFO	-
GPL	0051	routeBase.cpp:642	INFO	-
GPL	0052	routeBase.cpp:643	INFO	-
GPL	0053	routeBase.cpp:647	INFO	-
GPL	0054	routeBase.cpp:666	INFO	-
GPL	0055	routeBase.cpp:667	INFO	-
GPL	0056	routeBase.cpp:668	INFO	-
GPL	0057	routeBase.cpp:669	INFO	-
GPL	0058	routeBase.cpp:671	INFO	-
GPL	0059	routeBase.cpp:672	INFO	-
GPL	0063	routeBase.cpp:738	INFO	-
GPL	0064	routeBase.cpp:739	INFO	-
GPL	0065	routeBase.cpp:740	INFO	-
GPL	0066	routeBase.cpp:795	INFO	-
GPL	0067	routeBase.cpp:796	INFO	-
GPL	0068	routeBase.cpp:797	INFO	-
GPL	0069	routeBase.cpp:798	INFO	-
GPL	0070	routeBase.cpp:800	INFO	-
GPL	0071	routeBase.cpp:801	INFO	-
GPL	0072	routeBase.cpp:802	INFO	-
GPL	0073	routeBase.cpp:803	INFO	-
GPL	0074	routeBase.cpp:811	INFO	-
GPL	0075	routeBase.cpp:824	INFO	-
GPL	0100	timingBase.cpp:163	INFO	-
GPL	0102	timingBase.cpp:166	WARN	-
GPL	0103	timingBase.cpp:204	INFO	-
GPL	0114	timingBase.cpp:153	WARN	-
GPL	0115	replace.tcl:146	WARN	-
GPL	0116	replace.tcl:158	WARN	-
GPL	0118	placerBase.cpp:797	ERROR	-
GPL	0119	placerBase.cpp:840	ERROR	-
GPL	0120	placerBase.cpp:844	ERROR	-
GPL	0121	replace.tcl:118	ERROR	-
GPL	0122	mbff.cpp:164	ERROR	-
GPL	0130	replace.tcl:317	ERROR	-
GPL	0131	replace.tcl:418	ERROR	-
GPL	0132	replace.cpp:186	INFO	-
GPL	0133	replace.cpp:203	INFO	-
GPL	0134	placerBase.cpp:109	WARN	-
GPL	0135	replace.tcl:180	ERROR	-
GPL	0136	replace.cpp:290	WARN	-
GPL	0150	replace.tcl:122	WARN	-
GPL	0151	replace.tcl:153	WARN	-
GPL	0250	initialPlace.cpp:104	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
GPL	0251	initialPlace.cpp:109	WARN	-
GPL	0301	placerBase.cpp:1292	ERROR	-
GPL	0302	nesterovBase.cpp:1401	ERROR	-
GPL	0303	nesterovBase.cpp:1633	ERROR	-
GPL	0304	nesterovPlace.cpp:267	ERROR	-
GPL	0305	placerBase.cpp:791	ERROR	-
GPL	9032	mbff.cpp:546	ERROR	-
GRT	0001	GlobalRouter.cpp:765	INFO	-
GRT	0002	GlobalRouter.cpp:766	INFO	-
GRT	0003	GlobalRouter.cpp:3379	INFO	-
GRT	0004	GlobalRouter.cpp:3101	INFO	-
GRT	0005	GlobalRouter.tcl:526	ERROR	-
GRT	0006	GlobalRouter.cpp:323	INFO	-
GRT	0009	GlobalRouter.cpp:4011	INFO	-
GRT	0010	GlobalRouter.cpp:2948	ERROR	-
GRT	0011	GlobalRouter.cpp:3021	ERROR	-
GRT	0012	RepairAntennas.cpp:94	INFO	-
GRT	0014	GlobalRouter.cpp:280	INFO	-
GRT	0015	GlobalRouter.cpp:330	INFO	-
GRT	0018	GlobalRouter.cpp:2227	INFO	-
GRT	0019	GlobalRouter.cpp:2888	INFO	-
GRT	0020	GlobalRouter.cpp:3604	INFO	-
GRT	0021	GlobalRouter.cpp:3605	INFO	-
GRT	0022	GlobalRouter.cpp:3606	INFO	-
GRT	0023	GlobalRouter.cpp:3607	INFO	-
GRT	0025	MakeWireParasitics.cpp:237	WARN	-
GRT	0026	MakeWireParasitics.cpp:344	WARN	-
GRT	0027	RepairAntennas.cpp:417	WARN	-
GRT	0028	GlobalRouter.cpp:3374	ERROR	-
GRT	0029	GlobalRouter.cpp:2986	ERROR	-
GRT	0030	GlobalRouter.cpp:1391	WARN	-
GRT	0031	GlobalRouter.cpp:2181	WARN	-
GRT	0033	GlobalRouter.cpp:2452	WARN	-
GRT	0034	GlobalRouter.cpp:2937	WARN	-
GRT	0035	GlobalRouter.cpp:2969	WARN	-
GRT	0036	GlobalRouter.cpp:3032	WARN	-
GRT	0037	GlobalRouter.cpp:3168	WARN	-
GRT	0038	GlobalRouter.cpp:3320	WARN	-
GRT	0039	GlobalRouter.cpp:3357	WARN	-
GRT	0040	GlobalRouter.cpp:3414	WARN	-
GRT	0041	GlobalRouter.cpp:3447	WARN	-
GRT	0042	GlobalRouter.cpp:3057	ERROR	-
GRT	0043	GlobalRouter.cpp:3537	INFO	-
GRT	0044	GlobalRouter.tcl:66	ERROR	-
GRT	0045	GlobalRouter.tcl:356	ERROR	-
GRT	0047	GlobalRouter.tcl:80	ERROR	-
GRT	0048	GlobalRouter.tcl:88	ERROR	-
GRT	0051	GlobalRouter.tcl:231	ERROR	-
GRT	0052	GlobalRouter.tcl:235	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
GRT	0053	GlobalRouter.cpp:3619	INFO	-
GRT	0054	GlobalRouter.cpp:335	INFO	-
GRT	0055	GlobalRouter.tcl:246	ERROR	-
GRT	0057	GlobalRouter.tcl:585	ERROR	-
GRT	0059	GlobalRouter.tcl:498	ERROR	-
GRT	0062	GlobalRouter.tcl:541	ERROR	-
GRT	0063	GlobalRouter.tcl:548	ERROR	-
GRT	0064	GlobalRouter.tcl:554	ERROR	-
GRT	0065	GlobalRouter.tcl:558	ERROR	-
GRT	0066	GlobalRouter.tcl:562	ERROR	-
GRT	0067	GlobalRouter.tcl:566	ERROR	-
GRT	0068	RepairAntennas.cpp:145	ERROR	-
GRT	0069	GlobalRouter.tcl:321	ERROR	-
GRT	0071	GlobalRouter.cpp:1040	ERROR	-
GRT	0072	GlobalRouter.cpp:1253	ERROR	-
GRT	0073	GlobalRouter.tcl:334	ERROR	-
GRT	0074	GlobalRouter.cpp:1928	ERROR	-
GRT	0075	GlobalRouter.cpp:1906	ERROR	-
GRT	0076	GlobalRouter.cpp:1984	ERROR	-
GRT	0078	GlobalRouter.cpp:2066	ERROR	-
GRT	0079	GlobalRouter.cpp:2168	ERROR	-
GRT	0080	GlobalRouter.cpp:2197	ERROR	-
GRT	0084	GlobalRouter.cpp:461	ERROR	-
GRT	0085	GlobalRouter.cpp:456	ERROR	-
GRT	0086	GlobalRouter.cpp:2717	ERROR	-
GRT	0088	GlobalRouter.cpp:2732	INFO	-
GRT	0090	GlobalRouter.cpp:2792	ERROR	-
GRT	0094	GlobalRouter.cpp:3388	ERROR	-
GRT	0096	GlobalRouter.cpp:3671	INFO	-
GRT	0101	FastRoute.cpp:1005	INFO	-
GRT	0103	FastRoute.cpp:1139	INFO	-
GRT	0111	FastRoute.cpp:1312	INFO	-
GRT	0112	FastRoute.cpp:1313	INFO	-
GRT	0113	FastRoute.cpp:391	WARN	-
GRT	0114	FastRoute.cpp:421	WARN	-
GRT	0115	GlobalRouter.cpp:272	WARN	-
GRT	0118	GlobalRouter.cpp:1574	ERROR	Helpful Information-Do refer to the GUI guide and global routin
GRT	0119	GlobalRouter.cpp:1567	ERROR	Helpful Information-Do refer to the GUI guide and global routin
GRT	0122	RipUp.cpp:450	ERROR	-
GRT	0123	RipUp.cpp:572	ERROR	-
GRT	0125	maze.cpp:820	ERROR	-
GRT	0126	GlobalRouter.cpp:481	ERROR	-
GRT	0146	GlobalRouter.tcl:286	WARN	-
GRT	0149	utility.cpp:2045	ERROR	-
GRT	0150	maze.cpp:1838	ERROR	-
GRT	0151	maze.cpp:1872	WARN	-
GRT	0152	maze.cpp:1993	WARN	-
GRT	0153	maze.cpp:2028	WARN	-
GRT	0164	utility.cpp:1764	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
GRT	0165	utility.cpp:1778	WARN	-
GRT	0166	utility.cpp:1794	WARN	-
GRT	0167	utility.cpp:1808	WARN	-
GRT	0169	maze.cpp:1020	ERROR	-
GRT	0170	maze.cpp:1198	WARN	-
GRT	0171	maze3D.cpp:413	ERROR	-
GRT	0172	maze3D.cpp:737	ERROR	-
GRT	0179	route.cpp:329	WARN	-
GRT	0181	route.cpp:936	WARN	-
GRT	0183	maze3D.cpp:1124	ERROR	-
GRT	0184	maze3D.cpp:715	WARN	-
GRT	0187	maze3D.cpp:399	ERROR	-
GRT	0188	RSMT.cpp:139	ERROR	-
GRT	0189	RSMT.cpp:147	ERROR	-
GRT	0197	utility.cpp:361	INFO	-
GRT	0198	utility.cpp:362	INFO	-
GRT	0199	utility.cpp:363	INFO	-
GRT	0200	utility.cpp:601	WARN	-
GRT	0201	maze.cpp:891	ERROR	-
GRT	0202	utility.cpp:706	ERROR	-
GRT	0203	utility.cpp:1112	ERROR	-
GRT	0204	utility.cpp:1195	ERROR	-
GRT	0206	utility.cpp:1331	ERROR	-
GRT	0207	utility.cpp:1745	WARN	-
GRT	0208	utility.cpp:1753	WARN	-
GRT	0209	GlobalRouter.cpp:3037	ERROR	-
GRT	0214	FastRoute.cpp:597	ERROR	-
GRT	0215	GlobalRouter.tcl:350	WARN	-
GRT	0219	GlobalRouter.tcl:154	ERROR	-
GRT	0220	GlobalRouter.tcl:169	ERROR	-
GRT	0221	RepairAntennas.cpp:228	ERROR	-
GRT	0222	GlobalRouter.tcl:521	ERROR	-
GRT	0223	GlobalRouter.tcl:401	ERROR	-
GRT	0224	GlobalRouter.tcl:462	ERROR	-
GRT	0225	RipUp.cpp:176	ERROR	-
GRT	0226	RipUp.cpp:267	ERROR	-
GRT	0228	utility.cpp:1641	ERROR	-
GRT	0229	utility.cpp:1655	ERROR	-
GRT	0230	FastRoute.cpp:1283	WARN	-
GRT	0231	GlobalRouter.tcl:444	ERROR	-
GRT	0232	GlobalRouter.cpp:4064	ERROR	-
GRT	0233	GlobalRouter.cpp:1601	ERROR	-
GRT	0234	GlobalRouter.cpp:1624	ERROR	-
GRT	0235	GlobalRouter.cpp:1644	ERROR	-
GRT	0236	GlobalRouter.cpp:1652	ERROR	-
GRT	0237	GlobalRouter.cpp:3809	INFO	-
GRT	0238	GlobalRouter.tcl:490	ERROR	-
GRT	0239	GlobalRouter.cpp:3833	WARN	-
GRT	0240	GlobalRouter.cpp:3839	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
GRT	0241	GlobalRouter.cpp:3804	WARN	-
GRT	0242	GlobalRouter.tcl:189	ERROR	-
GRT	0243	RepairAntennas.cpp:454	WARN	-
GRT	0244	GlobalRouter.cpp:312	ERROR	-
GRT	0245	GlobalRouter.tcl:337	ERROR	-
GRT	0246	GlobalRouter.cpp:306	WARN	-
GRT	0247	utility.cpp:1611	ERROR	-
GRT	0248	utility.cpp:1622	ERROR	-
GRT	0249	GlobalRouter.cpp:1588	ERROR	-
GRT	0250	GlobalRouter.cpp:1633	WARN	-
GRT	0251	GlobalRouter.cpp:235	ERROR	-
GRT	0252	GlobalRouter.tcl:370	ERROR	-
GRT	0253	GlobalRouter.cpp:1792	ERROR	-
GRT	0254	utility.cpp:248	ERROR	-
GRT	0300	GlobalRouter.cpp:1546	WARN	-
GRT	0301	GlobalRouter.cpp:1374	WARN	-
GRT	0350	MakeWireParasitics.cpp:453	WARN	-
GRT	0500	RipUp.cpp:337	ERROR	-
GRT	0600	utility.cpp:2550	ERROR	-
GRT	1247	utility.cpp:1498	ERROR	-
GRT	1248	utility.cpp:1511	ERROR	-
GUI	0001	gui.i:47	INFO	-
GUI	0002	gui.i:53	ERROR	-
GUI	0003	gui.i:64	ERROR	-
GUI	0005	gui.i:68	ERROR	-
GUI	0006	gui.i:72	ERROR	-
GUI	0007	gui.i:337	ERROR	-
GUI	0008	gui.cpp:1249	WARN	-
GUI	0009	gui.i:360	ERROR	-
GUI	0010	gui_utils.cpp:71	WARN	-
GUI	0011	gui_utils.cpp:121	WARN	-
GUI	0012	gui_utils.cpp:126	WARN	-
GUI	0013	displayControls.cpp:1090	ERROR	-
GUI	0014	displayControls.cpp:1111	WARN	-
GUI	0015	gui.cpp:683	ERROR	-
GUI	0016	gui.cpp:716	ERROR	-
GUI	0017	gui.tcl:132	ERROR	-
GUI	0018	gui.tcl:146	ERROR	-
GUI	0019	gui.tcl:118	ERROR	-
GUI	0020	gui.tcl:48	ERROR	-
GUI	0021	gui.tcl:53	ERROR	-
GUI	0022	mainWindow.cpp:723	ERROR	-
GUI	0023	mainWindow.cpp:1207	WARN	-
GUI	0024	mainWindow.cpp:1028	WARN	-
GUI	0025	mainWindow.cpp:814	ERROR	-
GUI	0026	gui.tcl:79	ERROR	-
GUI	0027	gui.tcl:84	ERROR	-
GUI	0028	gui.cpp:879	ERROR	-
GUI	0029	gui.cpp:908	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
GUI	0030	drcWidget.cpp:517	ERROR	-
GUI	0031	gui.tcl:138	WARN	-
GUI	0032	drcWidget.cpp:500	ERROR	-
GUI	0033	gui.cpp:291	WARN	-
GUI	0034	displayControls.cpp:1123	WARN	-
GUI	0035	gui.cpp:504	ERROR	-
GUI	0036	gui.i:497	ERROR	-
GUI	0037	gui.i:503	ERROR	-
GUI	0038	gui.tcl:224	ERROR	-
GUI	0039	gui.tcl:252	WARN	-
GUI	0040	drcWidget.cpp:594	WARN	-
GUI	0041	drcWidget.cpp:688	WARN	-
GUI	0042	drcWidget.cpp:677	WARN	-
GUI	0043	drcWidget.cpp:666	WARN	-
GUI	0044	drcWidget.cpp:655	WARN	-
GUI	0045	drcWidget.cpp:548	ERROR	-
GUI	0046	drcWidget.cpp:563	ERROR	-
GUI	0047	drcWidget.cpp:583	ERROR	-
GUI	0048	drcWidget.cpp:613	ERROR	-
GUI	0049	drcWidget.cpp:619	ERROR	-
GUI	0050	drcWidget.cpp:626	ERROR	-
GUI	0051	drcWidget.cpp:715	WARN	-
GUI	0052	drcWidget.cpp:709	WARN	-
GUI	0053	gui.cpp:1152	ERROR	-
GUI	0054	displayControls.cpp:813	WARN	-
GUI	0055	drcWidget.cpp:755	ERROR	-
GUI	0056	gui.tcl:243	ERROR	-
GUI	0057	displayControls.cpp:735	WARN	-
GUI	0058	drcWidget.cpp:823	ERROR	-
GUI	0059	gui.cpp:484	ERROR	-
GUI	0060	gui.cpp:927	ERROR	-
GUI	0061	gui.cpp:936	ERROR	-
GUI	0062	gui.cpp:945	ERROR	-
GUI	0063	gui.cpp:952	ERROR	-
GUI	0064	gui.cpp:696	ERROR	-
GUI	0065	gui.cpp:701	ERROR	-
GUI	0066	heatMap.cpp:649	WARN	-
GUI	0067	gui.tcl:286	ERROR	-
GUI	0068	gui.tcl:295	ERROR	-
GUI	0069	gui.tcl:298	ERROR	-
GUI	0070	gui.tcl:329	ERROR	-
GUI	0071	gui.tcl:336	ERROR	-
GUI	0072	heatMap.cpp:96	ERROR	-
GUI	0073	heatMap.cpp:101	ERROR	-
GUI	0074	clockWidget.cpp:1517	ERROR	-
GUI	0075	gui.cpp:487	ERROR	-
GUI	0076	mainWindow.cpp:1601	ERROR	-
GUI	0077	mainWindow.cpp:1595	WARN	-
GUI	0078	layoutViewer.cpp:2099	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
GUI	0079	drcWidget.cpp:791	WARN	-
GUI	0080	drcWidget.cpp:881	WARN	-
GUI	0081	drcWidget.cpp:874	WARN	-
GUI	0082	drcWidget.cpp:861	WARN	-
GUI	0083	drcWidget.cpp:854	WARN	-
GUI	0084	drcWidget.cpp:846	WARN	-
GUI	0085	drcWidget.cpp:839	WARN	-
GUI	0086	drcWidget.cpp:779	ERROR	-
GUI	0087	drcWidget.cpp:765	ERROR	-
GUI	0088	gui.tcl:202	ERROR	-
GUI	0089	clockWidget.cpp:1481	ERROR	-
GUI	0090	gui.i:591	ERROR	-
GUI	0091	gui.i:604	ERROR	-
GUI	0092	gui.i:617	ERROR	-
GUI	0093	gui.i:630	ERROR	-
GUI	0094	layoutViewer.cpp:2062	WARN	-
GUI	0095	gui.cpp:978	ERROR	-
GUI	0096	gui.tcl:153	ERROR	-
GUI	0097	chartsWidget.cpp:163	WARN	-
GUI	0098	gui.tcl:158	ERROR	-
GUI	0099	drcWidget.cpp:805	WARN	-
IFP	0001	InitFloorplan.cc:437	INFO	-
IFP	0010	InitFloorplan.tcl:165	ERROR	-
IFP	0011	InitFloorplan.tcl:56	WARN	-
IFP	0013	InitFloorplan.tcl:84	ERROR	-
IFP	0015	InitFloorplan.tcl:108	ERROR	-
IFP	0016	InitFloorplan.tcl:120	ERROR	-
IFP	0017	InitFloorplan.tcl:137	ERROR	-
IFP	0018	InitFloorplan.i:149	ERROR	-
IFP	0019	InitFloorplan.tcl:140	ERROR	-
IFP	0021	InitFloorplan.cc:711	WARN	-
IFP	0022	InitFloorplan.cc:719	WARN	-
IFP	0025	InitFloorplan.tcl:168	ERROR	-
IFP	0026	InitFloorplan.cc:314	WARN	-
IFP	0027	InitFloorplan.cc:345	WARN	-
IFP	0028	InitFloorplan.cc:208	WARN	-
IFP	0029	InitFloorplan.cc:632	ERROR	-
IFP	0030	InitFloorplan.cc:657	INFO	-
IFP	0031	InitFloorplan.tcl:234	ERROR	-
IFP	0032	InitFloorplan.tcl:239	ERROR	-
IFP	0038	InitFloorplan.i:66	ERROR	-
IFP	0039	InitFloorplan.cc:616	ERROR	-
IFP	0040	InitFloorplan.cc:446	ERROR	-
IFP	0043	InitFloorplan.cc:397	WARN	-
IFP	0044	InitFloorplan.cc:776	ERROR	-
IFP	0045	InitFloorplan.cc:789	ERROR	-
IFP	0048	InitFloorplan.cc:508	ERROR	-
IFP	0049	InitFloorplan.cc:545	INFO	-
IFP	0050	InitFloorplan.cc:571	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
MPL	0001	mpl.tcl:83	ERROR	-
MPL	0002	hier_rtlmp.cpp:923	ERROR	-
MPL	0003	hier_rtlmp.cpp:2796	ERROR	-
MPL	0004	hier_rtlmp.cpp:3020	ERROR	-
MPL	0005	hier_rtlmp.cpp:3804	ERROR	-
MPL	0006	hier_rtlmp.cpp:4056	ERROR	-
MPL	0007	hier_rtlmp.cpp:5209	ERROR	-
MPL	0008	hier_rtlmp.cpp:5228	ERROR	-
MPL	0009	hier_rtlmp.cpp:5375	ERROR	-
MPL	0010	hier_rtlmp.cpp:5512	ERROR	-
MPL	0011	hier_rtlmp.cpp:6413	ERROR	-
MPL	0012	mpl.tcl:280	ERROR	-
MPL	0015	graphics.cpp:335	ERROR	-
MPL	0016	hier_rtlmp.cpp:462	ERROR	-
MPL	0017	hier_rtlmp.cpp:457	INFO	-
MPL	0018	mpl.tcl:290	WARN	-
MPL	0019	mpl.tcl:268	ERROR	-
MPL	0020	mpl.tcl:303	ERROR	-
MPL	0021	mpl.tcl:305	ERROR	-
MPL	0022	mpl.tcl:276	ERROR	-
MPL	0025	hier_rtlmp.cpp:525	WARN	-
MPL	0026	hier_rtlmp.cpp:954	WARN	-
MPL	0027	hier_rtlmp.cpp:612	WARN	-
MPL	0034	rtl_mp.cpp:143	ERROR	-
MPL	0035	rtl_mp.cpp:177	INFO	-
MPL	0036	rtl_mp.cpp:167	WARN	-
MPL	0038	object.cpp:626	ERROR	-
MPL	0039	hier_rtlmp.cpp:6405	WARN	-
MPL	0040	hier_rtlmp.cpp:5077	ERROR	-
MPL	0061	MacroPlacer.cpp:432	WARN	-
MPL	0064	MacroPlacer.cpp:709	ERROR	-
MPL	0065	MacroPlacer.cpp:1287	WARN	-
MPL	0066	MacroPlacer.cpp:232	WARN	-
MPL	0067	MacroPlacer.cpp:208	INFO	-
MPL	0068	MacroPlacer.cpp:230	INFO	-
MPL	0069	MacroPlacer.cpp:279	INFO	-
MPL	0070	MacroPlacer.cpp:404	INFO	-
MPL	0071	MacroPlacer.cpp:452	INFO	-
MPL	0072	MacroPlacer.cpp:489	WARN	-
MPL	0073	MacroPlacer.cpp:482	INFO	-
MPL	0076	MacroPlacer.cpp:575	INFO	-
MPL	0077	MacroPlacer.cpp:628	INFO	-
MPL	0079	MacroPlacer.cpp:640	INFO	-
MPL	0080	MacroPlacer.cpp:776	INFO	-
MPL	0085	MacroPlacer.tcl:86	WARN	-
MPL	0089	MacroPlacer.tcl:69	ERROR	-
MPL	0092	MacroPlacer.tcl:49	ERROR	-
MPL	0093	MacroPlacer.tcl:60	ERROR	-
MPL	0095	MacroPlacer.tcl:103	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
MPL	0096	MacroPlacer.tcl:116	ERROR	-
MPL	0097	MacroPlacer.cpp:522	ERROR	-
MPL	0098	MacroPlacer.cpp:164	WARN	-
MPL	0099	MacroPlacer.cpp:808	ERROR	-
MPL	0100	MacroPlacer.cpp:829	WARN	-
MPL	0101	MacroPlacer.cpp:833	INFO	-
MPL	0102	MacroPlacer.cpp:196	INFO	-
ODB	0000	dbWireCodec.cpp:1506	ERROR	-
ODB	0002	dbDatabase.cpp:560	ERROR	-
ODB	0005	dbBlock.cpp:2165	WARN	-
ODB	0006	dbBlock.cpp:2192	WARN	-
ODB	0007	dbBlock.cpp:2219	WARN	-
ODB	0008	dbBlock.cpp:2819	ERROR	-
ODB	0009	dbBlock.cpp:2823	WARN	-
ODB	0010	dbBlock.cpp:3079	INFO	-
ODB	0011	dbBlock.cpp:3230	WARN	-
ODB	0012	dbBlock.cpp:3240	ERROR	-
ODB	0013	dbBlock.cpp:3402	ERROR	-
ODB	0014	dbBlock.cpp:3581	WARN	-
ODB	0015	tmg_conn_w.cpp:98	INFO	-
ODB	0016	tmg_conn.cpp:1659	ERROR	-
ODB	0018	tmg_conn.cpp:1882	ERROR	-
ODB	0019	dbBlock.cpp:3664	WARN	-
ODB	0021	dbCCSeg.cpp:418	INFO	-
ODB	0022	dbCCSeg.cpp:444	INFO	-
ODB	0023	dbCCSeg.cpp:774	ERROR	-
ODB	0024	dbCapNode.cpp:197	WARN	-
ODB	0025	dbCapNode.cpp:1114	INFO	-
ODB	0034	dbITerm.cpp:692	WARN	-
ODB	0036	dbInst.cpp:726	INFO	-
ODB	0037	dbInst.cpp:1054	INFO	-
ODB	0038	dbInst.cpp:1076	WARN	-
ODB	0039	dbInst.cpp:1081	WARN	-
ODB	0040	dbInst.cpp:1089	WARN	-
ODB	0041	dbInst.cpp:1094	WARN	-
ODB	0042	dbInst.cpp:1102	WARN	-
ODB	0043	dbInst.cpp:1111	WARN	-
ODB	0044	dbInst.cpp:1224	WARN	-
ODB	0045	dbInst.cpp:1275	WARN	-
ODB	0046	dbInst.cpp:1303	WARN	-
ODB	0047	dbInst.cpp:1678	WARN	-
ODB	0048	dbNet.cpp:1070	WARN	-
ODB	0049	dbNet.cpp:2230	ERROR	-
ODB	0050	dbNet.cpp:2237	ERROR	-
ODB	0051	dbNet.cpp:2263	ERROR	-
ODB	0052	dbNet.cpp:2692	WARN	-
ODB	0053	dbNet.cpp:2741	WARN	-
ODB	0054	dbRSeg.cpp:571	INFO	-
ODB	0056	dbRSeg.cpp:577	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
ODB	0057	dbRSeg.cpp:859	WARN	-
ODB	0058	dbTech.cpp:1008	WARN	-
ODB	0059	dbTech.cpp:1017	ERROR	-
ODB	0060	dbTech.cpp:1027	ERROR	-
ODB	0061	dbTech.cpp:1037	ERROR	-
ODB	0062	dbWire.cpp:1846	WARN	-
ODB	0063	dbWireCodec.cpp:1241	WARN	-
ODB	0064	dbWireCodec.cpp:1250	INFO	-
ODB	0065	dbWireCodec.cpp:1259	INFO	-
ODB	0066	dbWireCodec.cpp:1271	INFO	-
ODB	0067	dbWireCodec.cpp:1280	INFO	-
ODB	0068	dbWireCodec.cpp:1303	INFO	-
ODB	0069	dbWireCodec.cpp:1315	INFO	-
ODB	0070	dbWireCodec.cpp:1328	INFO	-
ODB	0071	dbWireCodec.cpp:1338	INFO	-
ODB	0072	dbWireCodec.cpp:1348	WARN	-
ODB	0073	dbWireCodec.cpp:1367	INFO	-
ODB	0074	dbWireCodec.cpp:1374	INFO	-
ODB	0075	dbWireCodec.cpp:1390	INFO	-
ODB	0076	dbWireCodec.cpp:1397	INFO	-
ODB	0077	dbWireCodec.cpp:1405	INFO	-
ODB	0078	dbWireCodec.cpp:1411	INFO	-
ODB	0079	dbWireCodec.cpp:1417	INFO	-
ODB	0080	dbWireCodec.cpp:1426	INFO	-
ODB	0081	dbWireCodec.cpp:1440	INFO	-
ODB	0082	dbWireCodec.cpp:1445	INFO	-
ODB	0083	dbWireCodec.cpp:1453	ERROR	-
ODB	0084	dbWireCodec.cpp:1462	INFO	-
ODB	0085	dbWireCodec.cpp:1471	INFO	-
ODB	0086	dbWireCodec.cpp:1479	ERROR	-
ODB	0087	dbWirePathItr.cpp:521	WARN	-
ODB	0088	definBlockage.cpp:73	WARN	-
ODB	0089	definBlockage.cpp:84	WARN	-
ODB	0090	definBlockage.cpp:219	WARN	-
ODB	0091	definBlockage.cpp:240	WARN	-
ODB	0092	definComponent.cpp:155	WARN	-
ODB	0093	definComponent.cpp:177	WARN	-
ODB	0094	definComponent.cpp:183	INFO	-
ODB	0095	definFill.cpp:64	WARN	-
ODB	0096	definNet.cpp:127	WARN	-
ODB	0097	definNet.cpp:154	INFO	-
ODB	0098	definNet.cpp:169	WARN	-
ODB	0099	definNet.cpp:210	WARN	-
ODB	0100	definNet.cpp:220	WARN	-
ODB	0102	definNet.cpp:276	WARN	-
ODB	0103	definNet.cpp:290	WARN	-
ODB	0104	definNet.cpp:384	WARN	-
ODB	0105	definNet.cpp:399	WARN	-
ODB	0106	definNet.cpp:426	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
ODB	0107	definNet.cpp:544	WARN	-
ODB	0108	definNet.cpp:578	WARN	-
ODB	0109	definNet.cpp:603	WARN	-
ODB	0110	definNet.cpp:620	WARN	-
ODB	0111	definNonDefaultRule.cpp:64	WARN	-
ODB	0112	definNonDefaultRule.cpp:87	WARN	-
ODB	0113	definNonDefaultRule.cpp:104	WARN	-
ODB	0114	definNonDefaultRule.cpp:122	WARN	-
ODB	0115	definNonDefaultRule.cpp:139	WARN	-
ODB	0116	definNonDefaultRule.cpp:147	WARN	-
ODB	0117	definPin.cpp:120	WARN	-
ODB	0118	definPin.cpp:126	WARN	-
ODB	0119	definPin.cpp:208	WARN	-
ODB	0120	definPin.cpp:223	WARN	-
ODB	0121	definPin.cpp:249	WARN	-
ODB	0122	definPin.cpp:391	WARN	-
ODB	0123	definPin.cpp:404	WARN	-
ODB	0124	definReader.cpp:561	WARN	-
ODB	0125	definReader.cpp:1679	INFO	-
ODB	0126	definReader.cpp:1684	WARN	-
ODB	0127	definReader.cpp:1714	INFO	-
ODB	0128	definReader.cpp:371	INFO	-
ODB	0129	definReader.cpp:1718	WARN	-
ODB	0130	definReader.cpp:1723	INFO	-
ODB	0131	definReader.cpp:1731	INFO	-
ODB	0132	definReader.cpp:1744	INFO	-
ODB	0133	definReader.cpp:1752	INFO	-
ODB	0134	definReader.cpp:1767	INFO	-
ODB	0135	definReader.cpp:1780	INFO	-
ODB	0136	definNet.cpp:259	WARN	-
ODB	0137	definReader.cpp:1784	WARN	-
ODB	0138	definReader.cpp:1789	INFO	-
ODB	0139	definReader.cpp:1793	INFO	-
ODB	0140	definReader.cpp:1801	INFO	-
ODB	0141	definReader.cpp:1809	INFO	-
ODB	0142	definReader.cpp:1816	INFO	-
ODB	0143	definReader.cpp:1827	INFO	-
ODB	0144	definReader.cpp:1831	WARN	-
ODB	0145	definReader.cpp:1836	INFO	-
ODB	0146	definReader.cpp:1841	INFO	-
ODB	0147	definReader.cpp:1844	INFO	-
ODB	0148	definReader.cpp:1914	WARN	-
ODB	0149	definReader.cpp:1935	WARN	-
ODB	0150	definReader.cpp:1950	WARN	-
ODB	0151	definReader.cpp:1971	WARN	-
ODB	0152	definRegion.cpp:65	WARN	-
ODB	0153	dbPowerSwitch.cpp:346	ERROR	-
ODB	0154	dbPowerSwitch.cpp:356	ERROR	-
ODB	0155	definRow.cpp:113	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
ODB	0156	definSNet.cpp:128	WARN	-
ODB	0157	definSNet.cpp:185	WARN	-
ODB	0158	definSNet.cpp:195	WARN	-
ODB	0159	definSNet.cpp:261	WARN	-
ODB	0160	definSNet.cpp:281	WARN	-
ODB	0161	definSNet.cpp:317	WARN	-
ODB	0162	definSNet.cpp:336	WARN	-
ODB	0163	definSNet.cpp:481	WARN	-
ODB	0164	definSNet.cpp:548	WARN	-
ODB	0165	definTracks.cpp:70	WARN	-
ODB	0166	definVia.cpp:67	WARN	-
ODB	0167	definVia.cpp:81	WARN	-
ODB	0168	definVia.cpp:121	WARN	-
ODB	0169	definVia.cpp:130	WARN	-
ODB	0170	definVia.cpp:139	WARN	-
ODB	0171	definVia.cpp:248	WARN	-
ODB	0172	defout_impl.cpp:158	WARN	-
ODB	0173	defout_impl.cpp:972	WARN	-
ODB	0174	defout_impl.cpp:1748	WARN	-
ODB	0175	lefin.cpp:199	WARN	-
ODB	0176	lefin.cpp:221	WARN	-
ODB	0177	lefin.cpp:412	WARN	-
ODB	0178	lefin.cpp:432	WARN	-
ODB	0179	lefin.cpp:545	ERROR	-
ODB	0180	lefin.cpp:623	WARN	-
ODB	0181	lefin.cpp:644	WARN	-
ODB	0182	lefin.cpp:653	WARN	-
ODB	0183	lefin.cpp:861	ERROR	-
ODB	0184	lefin.cpp:1228	WARN	-
ODB	0185	lefin.cpp:1237	WARN	-
ODB	0186	lefin.cpp:1260	WARN	-
ODB	0187	lefin.cpp:1311	WARN	-
ODB	0188	lefin.cpp:1322	WARN	-
ODB	0189	lefin.cpp:1359	WARN	-
ODB	0190	lefin.cpp:1367	WARN	-
ODB	0191	lefin.cpp:1390	WARN	-
ODB	0192	lefin.cpp:1403	WARN	-
ODB	0193	lefin.cpp:1417	WARN	-
ODB	0194	lefin.cpp:1471	WARN	-
ODB	0195	lefin.cpp:1496	WARN	-
ODB	0196	lefin.cpp:1513	WARN	-
ODB	0197	lefin.cpp:1530	WARN	-
ODB	0198	lefin.cpp:1547	WARN	-
ODB	0199	lefin.cpp:1573	WARN	-
ODB	0200	lefin.cpp:1589	WARN	-
ODB	0201	lefin.cpp:1605	WARN	-
ODB	0202	lefin.cpp:1622	WARN	-
ODB	0203	lefin.cpp:1747	WARN	-
ODB	0204	lefin.cpp:1753	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
ODB	0205	lefin.cpp:1796	WARN	-
ODB	0206	lefin.cpp:1826	WARN	-
ODB	0207	lefin.cpp:1843	WARN	-
ODB	0208	lefin.cpp:1726	WARN	-
ODB	0209	lefin.cpp:1899	WARN	-
ODB	0210	lefin.cpp:1929	WARN	-
ODB	0211	lefin.cpp:1945	WARN	-
ODB	0212	lefin.cpp:1954	WARN	-
ODB	0213	lefin.cpp:1962	WARN	-
ODB	0214	lefin.cpp:2016	WARN	-
ODB	0215	lefin.cpp:2026	WARN	-
ODB	0216	lefin.cpp:2056	WARN	-
ODB	0217	lefin.cpp:2075	WARN	-
ODB	0218	lefin.cpp:2085	WARN	-
ODB	0219	dbPowerSwitch.cpp:373	ERROR	-
ODB	0220	dbPowerSwitch.cpp:383	ERROR	-
ODB	0221	lefin.cpp:2153	INFO	-
ODB	0222	lefin.cpp:2161	INFO	-
ODB	0228	lefin.cpp:2309	WARN	-
ODB	0229	lefin.cpp:2314	WARN	-
ODB	0230	lefin.cpp:2346	WARN	-
ODB	0231	lefin.cpp:2352	WARN	-
ODB	0240	reader.cpp:557	WARN	-
ODB	0241	create_box.cpp:172	WARN	-
ODB	0242	create_box.cpp:202	WARN	-
ODB	0243	create_box.cpp:205	INFO	-
ODB	0244	create_box.cpp:241	WARN	-
ODB	0245	create_box.cpp:271	WARN	-
ODB	0246	lefin.cpp:2234	ERROR	-
ODB	0247	definReader.cpp:959	WARN	-
ODB	0248	definReader.cpp:476	WARN	-
ODB	0249	definReader.cpp:1493	WARN	-
ODB	0250	definReader.cpp:1703	ERROR	-
ODB	0252	definReader.cpp:1727	INFO	-
ODB	0253	definReader.cpp:1739	INFO	-
ODB	0254	definReader.cpp:1760	INFO	-
ODB	0260	definReader.cpp:70	WARN	-
ODB	0261	definReader.cpp:341	WARN	-
ODB	0270	reader.cpp:548	WARN	-
ODB	0271	definReader.cpp:1923	WARN	-
ODB	0273	dbUtil.cpp:961	INFO	-
ODB	0274	create_box.cpp:60	WARN	-
ODB	0275	definReader.cpp:720	WARN	-
ODB	0276	create_box.cpp:276	INFO	-
ODB	0277	lefin.cpp:2178	WARN	-
ODB	0279	lefin.cpp:766	WARN	-
ODB	0280	lefin.cpp:2194	WARN	-
ODB	0282	dbTechLayer.cpp:2047	ERROR	-
ODB	0283	cdl.cpp:117	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
ODB	0284	cdl.cpp:148	WARN	-
ODB	0285	cdl.cpp:152	WARN	-
ODB	0286	cdl.cpp:163	WARN	-
ODB	0287	cdl.cpp:225	ERROR	-
ODB	0288	lefin.cpp:2294	ERROR	-
ODB	0289	lefin.cpp:2374	ERROR	-
ODB	0292	lefin.cpp:2331	ERROR	-
ODB	0293	definReader.cpp:1334	WARN	-
ODB	0297	dbModule.cpp:237	ERROR	-
ODB	0298	dbModule.cpp:383	ERROR	-
ODB	0299	definVia.cpp:276	ERROR	-
ODB	0300	definVia.cpp:286	ERROR	-
ODB	0301	definVia.cpp:296	ERROR	-
ODB	0302	definVia.cpp:312	ERROR	-
ODB	0303	util.cpp:458	INFO	-
ODB	0304	definGroup.cpp:61	WARN	-
ODB	0305	definGroup.cpp:74	WARN	-
ODB	0306	definGroup.cpp:99	WARN	-
ODB	0307	dbBlock.cpp:3691	ERROR	-
ODB	0308	odb.tcl:10	ERROR	-
ODB	0309	odb.tcl:15	ERROR	-
ODB	0310	odb.tcl:27	ERROR	-
ODB	0311	odb.tcl:35	ERROR	-
ODB	0312	odb.tcl:38	ERROR	-
ODB	0313	odb.tcl:47	ERROR	-
ODB	0314	odb.tcl:59	ERROR	-
ODB	0315	odb.tcl:205	ERROR	-
ODB	0316	odb.tcl:213	ERROR	-
ODB	0317	odb.tcl:220	ERROR	-
ODB	0318	odb.tcl:225	ERROR	-
ODB	0319	odb.tcl:232	ERROR	-
ODB	0320	odb.tcl:245	ERROR	-
ODB	0321	odb.tcl:250	ERROR	-
ODB	0322	odb.tcl:253	ERROR	-
ODB	0323	odb.tcl:267	ERROR	-
ODB	0324	odb.tcl:272	ERROR	-
ODB	0325	odb.tcl:275	ERROR	-
ODB	0326	odb.tcl:287	ERROR	-
ODB	0327	odb.tcl:292	ERROR	-
ODB	0328	odb.tcl:297	ERROR	-
ODB	0329	odb.tcl:303	ERROR	-
ODB	0330	odb.tcl:306	ERROR	-
ODB	0331	odb.tcl:309	ERROR	-
ODB	0332	odb.tcl:321	ERROR	-
ODB	0333	odb.tcl:326	ERROR	-
ODB	0334	odb.tcl:331	ERROR	-
ODB	0335	odb.tcl:337	ERROR	-
ODB	0336	odb.tcl:340	ERROR	-
ODB	0337	odb.tcl:343	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
ODB	0338	odb.tcl:357	ERROR	-
ODB	0339	odb.tcl:362	ERROR	-
ODB	0340	odb.tcl:365	ERROR	-
ODB	0341	odb.tcl:370	ERROR	-
ODB	0342	odb.tcl:377	ERROR	-
ODB	0343	odb.tcl:384	ERROR	-
ODB	0344	odb.tcl:387	ERROR	-
ODB	0345	odb.tcl:402	ERROR	-
ODB	0346	odb.tcl:407	ERROR	-
ODB	0347	odb.tcl:410	ERROR	-
ODB	0348	odb.tcl:415	ERROR	-
ODB	0349	odb.tcl:419	ERROR	-
ODB	0350	odb.tcl:426	ERROR	-
ODB	0351	odb.tcl:433	ERROR	-
ODB	0352	odb.tcl:436	ERROR	-
ODB	0353	odb.tcl:449	ERROR	-
ODB	0354	odb.tcl:468	ERROR	-
ODB	0355	odb.tcl:533	ERROR	-
ODB	0356	lefin.cpp:2210	WARN	-
ODB	0357	cdl.cpp:219	WARN	-
ODB	0358	cdl.cpp:193	ERROR	-
ODB	0359	dbInst.cpp:512	ERROR	-
ODB	0360	dbInst.cpp:609	ERROR	-
ODB	0361	lefin.cpp:2224	WARN	-
ODB	0362	dbInst.cpp:1541	ERROR	-
ODB	0364	dbNet.cpp:3111	ERROR	-
ODB	0367	dbModule.cpp:250	ERROR	-
ODB	0368	dbInst.cpp:1216	ERROR	-
ODB	0369	dbITerm.cpp:400	ERROR	-
ODB	0370	dbITerm.cpp:521	ERROR	-
ODB	0371	dbModule.cpp:287	ERROR	-
ODB	0372	dbITerm.cpp:531	ERROR	-
ODB	0373	dbITerm.cpp:408	ERROR	-
ODB	0374	dbBTerm.cpp:804	ERROR	-
ODB	0375	dbBTerm.cpp:519	ERROR	-
ODB	0376	dbBTerm.cpp:701	ERROR	-
ODB	0377	dbBTerm.cpp:484	ERROR	-
ODB	0378	dbBlock.cpp:4222	WARN	-
ODB	0379	dbGlobalConnect.cpp:307	WARN	-
ODB	0380	dbGlobalConnect.cpp:336	WARN	-
ODB	0381	dbBlock.cpp:4196	ERROR	-
ODB	0382	dbBlock.cpp:4200	WARN	-
ODB	0383	dbBlock.cpp:4268	WARN	-
ODB	0384	dbGlobalConnect.cpp:267	ERROR	-
ODB	0385	dbInst.cpp:1402	ERROR	-
ODB	0386	util.cpp:439	WARN	-
ODB	0387	definNonDefaultRule.cpp:238	WARN	-
ODB	0388	lefin.cpp:773	INFO	-
ODB	0389	dbUtil.cpp:95	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
ODB	0390	tmg_conn.cpp:1205	ERROR	-
ODB	0391	tmg_conn.cpp:1235	ERROR	-
ODB	0392	tmg_conn.cpp:1271	ERROR	-
ODB	0393	tmg_conn.cpp:1742	ERROR	-
ODB	0394	lefin.cpp:1688	INFO	-
ODB	0395	tmg_conn.cpp:1557	WARN	-
ODB	0396	tmg_conn.cpp:1639	WARN	-
ODB	0397	dbITerm.cpp:492	ERROR	-
ODB	0398	dbUtil.cpp:753	WARN	-
ODB	0399	dbUtil.cpp:795	WARN	-
ODB	0400	dbUtil.cpp:1058	WARN	-
ODB	0401	dbUtil.cpp:1061	WARN	-
ODB	0402	dbUtil.cpp:1111	WARN	-
ODB	0403	dbUtil.cpp:1138	WARN	-
ODB	0404	dbUtil.cpp:1363	WARN	-
ODB	0405	dbUtil.cpp:1366	WARN	-
ODB	0406	dbUtil.cpp:1442	WARN	-
ODB	0407	dbUtil.cpp:1456	WARN	-
ODB	0408	dbUtil.cpp:1572	WARN	-
ODB	0409	dbUtil.cpp:1579	WARN	-
ODB	0410	dbUtil.cpp:1601	WARN	-
ODB	0411	dbUtil.cpp:1609	WARN	-
ODB	0412	dbUtil.cpp:1635	WARN	-
ODB	0413	dbUtil.cpp:1643	WARN	-
ODB	0414	dbTrackGrid.cpp:276	ERROR	-
ODB	0415	dbTrackGrid.cpp:289	ERROR	-
ODB	0416	dbTrackGrid.cpp:295	ERROR	-
ODB	0417	dbUtil.cpp:135	WARN	-
ODB	0418	dbTrackGrid.cpp:265	ERROR	-
ODB	0420	tmg_conn.cpp:602	ERROR	-
ODB	0421	definReader.cpp:1933	ERROR	-
ODB	0422	definReader.cpp:1969	ERROR	-
ODB	0423	lefin.cpp:629	WARN	-
ODB	0424	parse.cpp:65	ERROR	-
ODB	0425	definComponentMaskShift.cpp:55	WARN	-
ODB	0428	parse.cpp:50	ERROR	-
ODB	0429	parse.cpp:412	ERROR	-
ODB	0430	dbBox.cpp:826	ERROR	-
ODB	0431	dbMaster.cpp:790	ERROR	-
ODB	0432	dbDatabase.cpp:467	ERROR	-
ODB	0433	dbITerm.cpp:415	ERROR	-
ODB	0434	dbBox.cpp:499	ERROR	-
ODB	0435	dbBox.cpp:804	ERROR	-
ODB	0436	dbInst.cpp:1499	ERROR	-
ODB	0437	definReader.cpp:999	WARN	-
ODB	0438	dbDatabase.cpp:162	CRITICAL	-
ODB	0439	odb.tcl:548	ERROR	-
ODB	0440	dbITerm.cpp:390	ERROR	-
ODB	1000	odb.tcl:72	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
ODB	1001	odb.tcl:143	WARN	-
ODB	1002	odb.tcl:146	WARN	-
ODB	1004	odb.tcl:161	ERROR	-
ODB	1005	odb.tcl:168	ERROR	-
ODB	1006	odb.tcl:173	ERROR	-
ODB	1007	odb.tcl:180	ERROR	-
ODB	1008	odb.tcl:188	ERROR	-
ODB	1009	odb.tcl:92	WARN	-
ODB	1100	dbAccessPoint.cpp:316	ERROR	-
ODB	1101	dbAccessPoint.cpp:336	ERROR	-
ODB	1102	dbWireCodec.cpp:636	ERROR	-
ODB	1103	dbWireCodec.cpp:660	ERROR	-
ODB	2000	lefin.cpp:1212	WARN	-
ORD	0001	OpenRoad.tcl:45	ERROR	-
ORD	0002	OpenRoad.tcl:48	ERROR	-
ORD	0003	OpenRoad.tcl:85	ERROR	-
ORD	0004	OpenRoad.tcl:88	ERROR	-
ORD	0005	OpenRoad.tcl:94	ERROR	-
ORD	0006	OpenRoad.tcl:125	ERROR	-
ORD	0007	OpenRoad.tcl:191	ERROR	-
ORD	0008	OpenRoad.tcl:194	ERROR	-
ORD	0014	Metrics.tcl:64	ERROR	-
ORD	0015	OpenRoad.i:426	ERROR	-
ORD	0017	Metrics.tcl:81	ERROR	-
ORD	0018	Metrics.tcl:98	ERROR	-
ORD	0019	Metrics.tcl:48	ERROR	-
ORD	0030	OpenRoad.cc:592	INFO	-
ORD	0031	OpenRoad.cc:578	WARN	-
ORD	0032	OpenRoad.cc:609	WARN	-
ORD	0033	OpenRoad.tcl:97	WARN	-
ORD	0034	OpenRoad.cc:414	INFO	-
ORD	0036	Design.cc:66	ERROR	-
ORD	0037	Design.cc:127	ERROR	-
ORD	0038	Main.cc:280	WARN	-
ORD	0039	Main.cc:284	WARN	-
ORD	0042	OpenRoad.tcl:303	ERROR	-
ORD	0043	OpenRoad.tcl:311	ERROR	-
ORD	0045	OpenRoad.tcl:341	ERROR	-
ORD	0046	OpenRoad.tcl:333	WARN	-
ORD	0047	OpenRoad.cc:468	ERROR	-
ORD	0048	OpenRoad.cc:317	INFO	-
ORD	0049	OpenRoad.i:476	ERROR	-
ORD	0050	OpenRoad.i:487	ERROR	-
ORD	0051	OpenRoad.cc:295	ERROR	-
ORD	0052	OpenRoad.i:335	ERROR	-
ORD	0053	OpenRoad.cc:397	ERROR	-
ORD	0054	OpenRoad.cc:480	ERROR	-
ORD	0101	Design.cc:82	ERROR	-
ORD	0102	Design.cc:88	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
ORD	0104	Timing.cc:148	ERROR	-
ORD	0105	OpenRoad.cc:514	ERROR	-
ORD	0201	Resizer.tcl:48	ERROR	-
ORD	0202	Resizer.tcl:57	ERROR	-
ORD	0203	Resizer.tcl:61	ERROR	-
ORD	0204	Resizer.tcl:65	ERROR	-
ORD	0205	Resizer.tcl:98	ERROR	-
ORD	0206	Resizer.tcl:102	WARN	-
ORD	0208	Resizer.tcl:120	ERROR	-
ORD	0209	Resizer.tcl:123	ERROR	-
ORD	0560	OpenRoad.tcl:412	ERROR	-
ORD	1009	OpenRoad.tcl:212	ERROR	-
ORD	1010	OpenRoad.tcl:215	ERROR	-
ORD	1011	OpenRoad.tcl:221	ERROR	-
ORD	1012	OpenRoad.tcl:227	ERROR	-
ORD	1013	OpenRoad.tcl:174	ERROR	-
ORD	2001	dbNetwork.cc:916	WARN	-
ORD	2002	dbNetwork.cc:1039	WARN	-
ORD	2003	dbNetwork.cc:1216	CRITICAL	-
ORD	2004	dbNetwork.cc:1271	CRITICAL	-
ORD	2005	dbNetwork.cc:1276	CRITICAL	-
ORD	2006	dbNetwork.cc:1340	CRITICAL	-
ORD	2007	dbNetwork.cc:1397	CRITICAL	-
ORD	2008	dbNetwork.cc:1469	CRITICAL	-
ORD	2009	dbReadVerilog.tcl:51	ERROR	-
ORD	2010	dbReadVerilog.tcl:58	ERROR	-
ORD	2011	dbReadVerilog.cc:426	WARN	-
ORD	2012	dbReadVerilog.cc:432	WARN	-
ORD	2013	dbReadVerilog.cc:294	WARN	-
ORD	2014	dbReadVerilog.cc:276	WARN	-
ORD	2015	dbReadVerilog.cc:303	WARN	-
ORD	2016	dbNetwork.cc:1312	CRITICAL	-
PAD	0001	ICeWall.cpp:655	ERROR	-
PAD	0002	ICeWall.cpp:1134	ERROR	-
PAD	0003	RDLRouter.cpp:160	WARN	-
PAD	0004	RDLRouter.cpp:174	WARN	-
PAD	0005	RDLRouter.cpp:232	INFO	-
PAD	0006	RDLRouter.cpp:278	WARN	-
PAD	0007	RDLRouter.cpp:300	ERROR	-
PAD	0008	RDLRouter.cpp:347	ERROR	-
PAD	0009	RDLRouter.cpp:472	ERROR	-
PAD	0010	RDLRouter.cpp:1201	ERROR	-
PAD	0011	ICeWall.cpp:76	ERROR	-
PAD	0012	ICeWall.cpp:90	ERROR	-
PAD	0013	ICeWall.cpp:453	ERROR	-
PAD	0014	ICeWall.cpp:302	ERROR	-
PAD	0015	ICeWall.cpp:305	ERROR	-
PAD	0016	ICeWall.cpp:308	ERROR	-
PAD	0018	ICeWall.cpp:487	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PAD	0019	ICeWall.cpp:503	ERROR	-
PAD	0020	ICeWall.cpp:689	ERROR	-
PAD	0021	ICeWall.cpp:867	ERROR	-
PAD	0022	ICeWall.cpp:1324	ERROR	-
PAD	0023	ICeWall.cpp:73	ERROR	-
PAD	0024	ICeWall.cpp:196	ERROR	-
PAD	0025	ICeWall.cpp:201	ERROR	-
PAD	0026	ICeWall.cpp:771	ERROR	-
PAD	0027	ICeWall.cpp:892	ERROR	-
PAD	0028	ICeWall.cpp:446	ERROR	-
PAD	0029	ICeWall.cpp:562	ERROR	-
PAD	0030	ICeWall.cpp:847	ERROR	-
PAD	0031	ICeWall.cpp:928	WARN	-
PAD	0032	ICeWall.cpp:922	ERROR	-
PAD	0033	ICeWall.cpp:159	WARN	-
PAD	0034	ICeWall.cpp:166	ERROR	-
PAD	0035	ICeWall.cpp:221	ERROR	-
PAD	0036	ICeWall.cpp:227	ERROR	-
PAD	0037	RDLRouter.cpp:1299	ERROR	-
PAD	0100	pad.tcl:448	ERROR	-
PAD	0101	pad.tcl:456	ERROR	-
PAD	0102	pad.tcl:464	ERROR	-
PAD	0103	pad.tcl:472	ERROR	-
PAD	0104	pad.tcl:480	ERROR	-
PAD	0105	pad.tcl:394	ERROR	-
PAD	0106	pad.i:161	ERROR	-
PAD	0107	pad.tcl:400	ERROR	-
PAD	0108	pad.tcl:407	ERROR	-
PAD	0109	pad.tcl:488	ERROR	-
PAD	0110	pad.tcl:493	ERROR	-
PAD	0111	pad.tcl:498	ERROR	-
PAD	0112	pad.tcl:159	WARN	-
PAD	0113	pad.tcl:115	ERROR	-
PAD	0114	pad.tcl:121	ERROR	-
PAD	0115	ICeWall.cpp:1032	ERROR	-
PAD	0116	ICeWall.cpp:177	INFO	-
PAD	0117	pad.tcl:308	ERROR	-
PAD	0118	ICeWall.cpp:494	ERROR	-
PAD	0119	ICeWall.cpp:626	ERROR	-
PAD	0120	ICeWall.cpp:1042	ERROR	-
PAD	0232	grid.cpp:1546	WARN	-
PAD	9001	ICeWall.tcl:1926	ERROR	-
PAD	9002	ICeWall.tcl:1722	ERROR	-
PAD	9004	ICeWall.tcl:2509	WARN	-
PAD	9005	ICeWall.tcl:413	ERROR	-
PAD	9006	ICeWall.tcl:459	ERROR	-
PAD	9007	ICeWall.tcl:1896	ERROR	-
PAD	9008	ICeWall.tcl:1135	ERROR	-
PAD	9009	ICeWall.tcl:5021	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PAD	9010	ICeWall.tcl:5050	ERROR	-
PAD	9011	ICeWall.tcl:1684	WARN	-
PAD	9012	ICeWall.tcl:2553	WARN	-
PAD	9014	ICeWall.tcl:4672	ERROR	-
PAD	9015	ICeWall.tcl:4345	ERROR	-
PAD	9016	ICeWall.tcl:769	ERROR	-
PAD	9017	ICeWall.tcl:2039	ERROR	-
PAD	9018	ICeWall.tcl:4697	WARN	-
PAD	9019	ICeWall.tcl:2600	WARN	-
PAD	9021	ICeWall.tcl:1342	ERROR	-
PAD	9022	ICeWall.tcl:1324	ERROR	-
PAD	9023	ICeWall.tcl:1329	ERROR	-
PAD	9024	ICeWall.tcl:496	ERROR	-
PAD	9025	ICeWall.tcl:567	ERROR	-
PAD	9026	ICeWall.tcl:507	ERROR	-
PAD	9027	ICeWall.tcl:3277	ERROR	-
PAD	9028	ICeWall.tcl:3297	ERROR	-
PAD	9029	ICeWall.tcl:4749	ERROR	-
PAD	9030	ICeWall.tcl:4817	ERROR	-
PAD	9031	ICeWall.tcl:672	ERROR	-
PAD	9032	ICeWall.tcl:968	ERROR	-
PAD	9033	ICeWall.tcl:1241	ERROR	-
PAD	9034	ICeWall.tcl:2728	ERROR	-
PAD	9035	ICeWall.tcl:2814	ERROR	-
PAD	9036	ICeWall.tcl:2894	ERROR	-
PAD	9037	ICeWall.tcl:2866	ERROR	-
PAD	9038	ICeWall.tcl:2905	ERROR	-
PAD	9039	ICeWall.tcl:2917	ERROR	-
PAD	9040	ICeWall.tcl:2930	ERROR	-
PAD	9041	ICeWall.tcl:686	ERROR	-
PAD	9042	ICeWall.tcl:2096	WARN	-
PAD	9043	ICeWall.tcl:2099	ERROR	-
PAD	9044	ICeWall.tcl:2231	ERROR	-
PAD	9045	ICeWall.tcl:2247	ERROR	-
PAD	9046	ICeWall.tcl:2286	WARN	-
PAD	9047	ICeWall.tcl:2291	WARN	-
PAD	9048	ICeWall.tcl:2655	WARN	-
PAD	9049	ICeWall.tcl:1144	ERROR	-
PAD	9050	ICeWall.tcl:4662	WARN	-
PAD	9051	ICeWall.tcl:2540	INFO	-
PAD	9052	ICeWall.tcl:3832	INFO	-
PAD	9053	ICeWall.tcl:4703	INFO	-
PAD	9054	ICeWall.tcl:375	ERROR	-
PAD	9055	ICeWall.tcl:378	ERROR	-
PAD	9056	ICeWall.tcl:421	ERROR	-
PAD	9057	ICeWall.tcl:424	ERROR	-
PAD	9058	ICeWall.tcl:470	ERROR	-
PAD	9059	ICeWall.tcl:474	ERROR	-
PAD	9060	ICeWall.tcl:586	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PAD	9061	ICeWall.tcl:717	ERROR	-
PAD	9062	ICeWall.tcl:728	ERROR	-
PAD	9063	ICeWall.tcl:865	ERROR	-
PAD	9064	ICeWall.tcl:871	ERROR	-
PAD	9065	ICeWall.tcl:913	ERROR	-
PAD	9066	ICeWall.tcl:1013	ERROR	-
PAD	9070	ICeWall.tcl:1190	ERROR	-
PAD	9071	ICeWall.tcl:1204	ERROR	-
PAD	9072	ICeWall.tcl:1399	ERROR	-
PAD	9073	ICeWall.tcl:1403	ERROR	-
PAD	9074	ICeWall.tcl:1406	ERROR	-
PAD	9075	ICeWall.tcl:1443	ERROR	-
PAD	9076	ICeWall.tcl:1446	ERROR	-
PAD	9077	ICeWall.tcl:1449	ERROR	-
PAD	9078	ICeWall.tcl:2567	ERROR	-
PAD	9079	ICeWall.tcl:2969	ERROR	-
PAD	9080	ICeWall.tcl:4429	ERROR	-
PAD	9081	ICeWall.tcl:4439	ERROR	-
PAD	9082	ICeWall.tcl:4564	ERROR	-
PAD	9083	ICeWall.tcl:4568	ERROR	-
PAD	9084	ICeWall.tcl:4722	ERROR	-
PAD	9085	ICeWall.tcl:4726	ERROR	-
PAD	9086	ICeWall.tcl:4882	ERROR	-
PAD	9087	ICeWall.tcl:4886	ERROR	-
PAD	9091	ICeWall.tcl:5294	ERROR	-
PAD	9092	ICeWall.tcl:5292	ERROR	-
PAD	9093	ICeWall.tcl:5308	ERROR	-
PAD	9094	ICeWall.tcl:5324	ERROR	-
PAD	9095	ICeWall.tcl:5334	ERROR	-
PAD	9096	ICeWall.tcl:1146	ERROR	-
PAD	9097	ICeWall.tcl:1148	ERROR	-
PAD	9098	ICeWall.tcl:5337	ERROR	-
PAD	9099	ICeWall.tcl:5435	ERROR	-
PAD	9100	ICeWall.tcl:5442	ERROR	-
PAD	9101	ICeWall.tcl:5445	ERROR	-
PAD	9102	ICeWall.tcl:5450	ERROR	-
PAD	9103	ICeWall.tcl:5455	ERROR	-
PAD	9104	ICeWall.tcl:5458	ERROR	-
PAD	9105	ICeWall.tcl:5472	ERROR	-
PAD	9106	ICeWall.tcl:5481	ERROR	-
PAD	9109	ICeWall.tcl:5859	ERROR	-
PAD	9110	ICeWall.tcl:5877	ERROR	-
PAD	9111	ICeWall.tcl:5780	ERROR	-
PAD	9112	ICeWall.tcl:5910	ERROR	-
PAD	9113	ICeWall.tcl:6539	ERROR	-
PAD	9114	ICeWall.tcl:605	ERROR	-
PAD	9115	ICeWall.tcl:625	ERROR	-
PAD	9116	ICeWall.tcl:788	ERROR	-
PAD	9117	ICeWall.tcl:804	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PAD	9119	ICeWall.tcl:809	ERROR	-
PAD	9120	ICeWall.tcl:1168	ERROR	-
PAD	9121	ICeWall.tcl:1171	ERROR	-
PAD	9122	ICeWall.tcl:5540	ERROR	-
PAD	9123	ICeWall.tcl:5926	ERROR	-
PAD	9124	ICeWall.tcl:5956	ERROR	-
PAD	9125	ICeWall.tcl:5963	ERROR	-
PAD	9126	ICeWall.tcl:5969	ERROR	-
PAD	9127	ICeWall.tcl:5988	ERROR	-
PAD	9128	ICeWall.tcl:5995	ERROR	-
PAD	9129	ICeWall.tcl:6013	ERROR	-
PAD	9130	ICeWall.tcl:6019	ERROR	-
PAD	9131	ICeWall.tcl:6032	ERROR	-
PAD	9132	ICeWall.tcl:6038	ERROR	-
PAD	9133	ICeWall.tcl:6052	ERROR	-
PAD	9134	ICeWall.tcl:6063	ERROR	-
PAD	9135	ICeWall.tcl:6067	WARN	-
PAD	9136	ICeWall.tcl:6078	ERROR	-
PAD	9137	ICeWall.tcl:6082	ERROR	-
PAD	9140	ICeWall.tcl:5950	ERROR	-
PAD	9141	ICeWall.tcl:2026	ERROR	-
PAD	9142	ICeWall.tcl:5602	ERROR	-
PAD	9143	ICeWall.tcl:5607	ERROR	-
PAD	9144	ICeWall.tcl:5627	ERROR	-
PAD	9145	ICeWall.tcl:5632	ERROR	-
PAD	9146	ICeWall.tcl:5731	ERROR	-
PAD	9147	ICeWall.tcl:5708	ERROR	-
PAD	9159	ICeWall.tcl:6006	INFO	-
PAD	9160	ICeWall.tcl:5717	ERROR	-
PAD	9161	ICeWall.tcl:1061	ERROR	-
PAD	9162	ICeWall.tcl:2978	ERROR	-
PAD	9163	ICeWall.tcl:6111	ERROR	-
PAD	9164	ICeWall.tcl:6114	ERROR	-
PAD	9165	ICeWall.tcl:6166	ERROR	-
PAD	9166	ICeWall.tcl:6179	ERROR	-
PAD	9167	ICeWall.tcl:6185	ERROR	-
PAD	9168	ICeWall.tcl:6192	ERROR	-
PAD	9169	ICeWall.tcl:6198	ERROR	-
PAD	9170	ICeWall.tcl:6208	ERROR	-
PAD	9171	ICeWall.tcl:6214	ERROR	-
PAD	9173	ICeWall.tcl:6229	ERROR	-
PAD	9174	ICeWall.tcl:6313	ERROR	-
PAD	9175	ICeWall.tcl:6324	ERROR	-
PAD	9176	ICeWall.tcl:6344	ERROR	-
PAD	9177	ICeWall.tcl:6348	ERROR	-
PAD	9178	ICeWall.tcl:6364	ERROR	-
PAD	9179	ICeWall.tcl:6372	ERROR	-
PAD	9180	ICeWall.tcl:6411	ERROR	-
PAD	9181	ICeWall.tcl:6427	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PAD	9182	ICeWall.tcl:6438	ERROR	-
PAD	9183	ICeWall.tcl:6479	ERROR	-
PAD	9184	ICeWall.tcl:6493	ERROR	-
PAD	9185	ICeWall.tcl:6513	ERROR	-
PAD	9187	ICeWall.tcl:366	ERROR	-
PAD	9188	ICeWall.tcl:6617	ERROR	-
PAD	9189	ICeWall.tcl:6626	ERROR	-
PAD	9190	ICeWall.tcl:6633	ERROR	-
PAD	9191	ICeWall.tcl:6641	ERROR	-
PAD	9192	ICeWall.tcl:6644	ERROR	-
PAD	9193	ICeWall.tcl:6651	ERROR	-
PAD	9194	ICeWall.tcl:6654	ERROR	-
PAD	9195	ICeWall.tcl:6657	ERROR	-
PAD	9196	ICeWall.tcl:6660	ERROR	-
PAD	9197	ICeWall.tcl:6667	ERROR	-
PAD	9198	ICeWall.tcl:6672	ERROR	-
PAD	9199	ICeWall.tcl:6678	ERROR	-
PAD	9200	ICeWall.tcl:5902	ERROR	-
PAD	9201	ICeWall.tcl:6390	ERROR	-
PAD	9202	ICeWall.tcl:6398	ERROR	-
PAD	9203	ICeWall.tcl:359	ERROR	-
PAD	9204	ICeWall.tcl:362	ERROR	-
PAD	9205	ICeWall.tcl:5686	ERROR	-
PAD	9207	ICeWall.tcl:4780	ERROR	-
PAD	9208	ICeWall.tcl:206	ERROR	-
PAD	9209	ICeWall.tcl:5790	ERROR	-
PAD	9210	ICeWall.tcl:5795	ERROR	-
PAD	9211	ICeWall.tcl:5808	ERROR	-
PAD	9212	ICeWall.tcl:5819	ERROR	-
PAD	9213	ICeWall.tcl:5822	ERROR	-
PAD	9214	ICeWall.tcl:5834	ERROR	-
PAD	9215	ICeWall.tcl:5841	ERROR	-
PAD	9216	ICeWall.tcl:5881	ERROR	-
PAD	9217	ICeWall.tcl:2382	ERROR	-
PAD	9218	ICeWall.tcl:52	ERROR	-
PAD	9219	ICeWall.tcl:125	ERROR	-
PAD	9220	ICeWall.tcl:196	ERROR	-
PAD	9221	ICeWall.tcl:236	ERROR	-
PAD	9222	ICeWall.tcl:251	ERROR	-
PAD	9223	ICeWall.tcl:2354	ERROR	-
PAD	9224	ICeWall.tcl:232	ERROR	-
PAD	9225	ICeWall.tcl:192	ERROR	-
PAD	9226	ICeWall.tcl:117	ERROR	-
PAD	9227	ICeWall.tcl:244	ERROR	-
PAD	9228	ICeWall.tcl:6605	ERROR	-
PAD	9229	ICeWall.tcl:5422	ERROR	-
PAD	9230	ICeWall.tcl:5425	ERROR	-
PAD	9231	ICeWall.tcl:61	ERROR	-
PAD	9232	ICeWall.tcl:84	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PAD	9233	ICeWall.tcl:73	ERROR	-
PAD	9234	ICeWall.tcl:76	ERROR	-
PAD	9235	ICeWall.tcl:3737	ERROR	-
PAD	9236	ICeWall.tcl:3864	ERROR	-
PAD	9237	ICeWall.tcl:69	ERROR	-
PAD	9238	ICeWall.tcl:3710	ERROR	-
PAD	9239	ICeWall.tcl:5352	ERROR	-
PAD	9240	ICeWall.tcl:5345	ERROR	-
PAD	9241	ICeWall.tcl:5376	ERROR	-
PAD	9242	ICeWall.tcl:5380	ERROR	-
PAD	9243	ICeWall.tcl:5409	ERROR	-
PAD	9244	ICeWall.tcl:5412	ERROR	-
PAD	9245	ICeWall.tcl:5415	ERROR	-
PAD	9246	ICeWall.tcl:5778	INFO	-
PAD	9247	ICeWall.tcl:4222	ERROR	-
PAD	9248	ICeWall.tcl:4273	ERROR	-
PAD	9249	ICeWall.tcl:4241	ERROR	-
PAD	9250	ICeWall.tcl:4034	ERROR	-
PAD	9251	ICeWall.tcl:1124	WARN	-
PAD	9252	ICeWall.tcl:2643	ERROR	-
PAD	9253	ICeWall.tcl:3815	ERROR	-
PAD	9254	ICeWall.tcl:5082	WARN	-
PAD	9255	ICeWall.tcl:5084	WARN	-
PAD	9256	ICeWall.tcl:5087	ERROR	-
PAD	9257	ICeWall.tcl:2622	ERROR	-
PAD	9258	ICeWall.tcl:5750	ERROR	-
PAD	9259	ICeWall.tcl:5740	ERROR	-
PAD	9260	ICeWall.tcl:3466	ERROR	-
PAD	9261	ICeWall.tcl:3501	ERROR	-
PAD	9262	ICeWall.tcl:3442	WARN	-
PAD	9263	ICeWall.tcl:3446	WARN	-
PAD	9264	ICeWall.tcl:3588	WARN	-
PAD	9265	ICeWall.tcl:3626	WARN	-
PAD	9266	ICeWall.tcl:3628	WARN	-
PAD	9267	ICeWall.tcl:3657	WARN	-
PAD	9268	ICeWall.tcl:3659	WARN	-
PAD	9269	ICeWall.tcl:3666	WARN	-
PAD	9270	ICeWall.tcl:3668	WARN	-
PAD	9271	ICeWall.tcl:3678	WARN	-
PAR	0001	PartitionMgr.cpp:770	INFO	-
PAR	0002	TritonPart.cpp:234	INFO	-
PAR	0003	TritonPart.cpp:275	WARN	-
PAR	0004	TritonPart.cpp:323	INFO	-
PAR	0005	TritonPart.cpp:392	INFO	-
PAR	0006	TritonPart.cpp:446	INFO	-
PAR	0007	TritonPart.cpp:481	INFO	-
PAR	0008	TritonPart.cpp:521	ERROR	-
PAR	0009	TritonPart.cpp:532	WARN	-
PAR	0010	TritonPart.cpp:692	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PAR	0011	TritonPart.cpp:769	INFO	-
PAR	0012	TritonPart.cpp:886	INFO	-
PAR	0013	TritonPart.cpp:892	INFO	-
PAR	0015	PartitionMgr.cpp:779	WARN	-
PAR	0017	TritonPart.cpp:1216	INFO	-
PAR	0018	TritonPart.cpp:1527	INFO	-
PAR	0019	TritonPart.cpp:1736	WARN	-
PAR	0021	TritonPart.cpp:1351	WARN	-
PAR	0022	TritonPart.cpp:1378	WARN	-
PAR	0023	TritonPart.cpp:1405	WARN	-
PAR	0024	TritonPart.cpp:1544	WARN	-
PAR	0025	PartitionMgr.cpp:850	ERROR	-
PAR	0026	PartitionMgr.cpp:859	ERROR	-
PAR	0027	PartitionMgr.cpp:882	ERROR	-
PAR	0028	PartitionMgr.cpp:893	ERROR	-
PAR	0029	TritonPart.cpp:901	ERROR	-
PAR	0030	TritonPart.cpp:997	ERROR	-
PAR	0031	TritonPart.cpp:1078	ERROR	-
PAR	0032	TritonPart.cpp:1096	ERROR	-
PAR	0033	TritonPart.cpp:1117	ERROR	-
PAR	0034	TritonPart.cpp:1140	ERROR	-
PAR	0035	TritonPart.cpp:1501	ERROR	-
PAR	0036	PartitionMgr.cpp:872	ERROR	-
PAR	0037	TritonPart.cpp:1496	INFO	-
PAR	0051	partitionmgr.tcl:1015	ERROR	-
PAR	0142	Evaluator.cpp:286	WARN	-
PAR	0924	partitionmgr.tcl:114	ERROR	-
PAR	0925	partitionmgr.tcl:350	ERROR	-
PDN	0001	grid.cpp:134	INFO	-
PDN	0100	domain.cpp:260	ERROR	-
PDN	0101	domain.cpp:282	INFO	-
PDN	0102	domain.cpp:291	INFO	-
PDN	0103	domain.cpp:84	ERROR	-
PDN	0104	domain.cpp:87	ERROR	-
PDN	0105	rings.cpp:123	WARN	-
PDN	0106	grid_component.cpp:367	ERROR	-
PDN	0107	grid_component.cpp:379	ERROR	-
PDN	0108	grid_component.cpp:449	ERROR	-
PDN	0109	straps.cpp:470	ERROR	-
PDN	0110	via.cpp:986	WARN	-
PDN	0113	PdnGen.tcl:576	ERROR	-
PDN	0114	grid_component.cpp:427	ERROR	-
PDN	0115	sroute.cpp:95	ERROR	-
PDN	0116	sroute.cpp:151	ERROR	-
PDN	0174	PdnGen.tcl:6612	ERROR	-
PDN	0175	straps.cpp:102	ERROR	-
PDN	0178	straps.cpp:2233	WARN	-
PDN	0179	straps.cpp:2242	ERROR	-
PDN	0180	rings.cpp:63	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PDN	0181	domain.cpp:267	ERROR	-
PDN	0182	PdnGen.cc:523	WARN	-
PDN	0183	PdnGen.cc:319	WARN	-
PDN	0184	PdnGen.cc:347	ERROR	-
PDN	0185	straps.cpp:127	ERROR	-
PDN	0186	grid.cpp:116	ERROR	-
PDN	0187	straps.cpp:82	ERROR	-
PDN	0188	grid.cpp:1686	ERROR	-
PDN	0189	PdnGen.cc:917	WARN	-
PDN	0190	straps.cpp:344	ERROR	-
PDN	0191	techlayer.cpp:147	ERROR	-
PDN	0192	grid.cpp:616	ERROR	-
PDN	0193	grid.cpp:624	ERROR	-
PDN	0194	grid.cpp:648	ERROR	-
PDN	0195	via.cpp:2868	WARN	-
PDN	0196	PdnGen.cc:389	ERROR	-
PDN	0197	power_cells.cpp:153	ERROR	-
PDN	0198	power_cells.cpp:124	ERROR	-
PDN	0199	PdnGen.cc:360	ERROR	-
PDN	0200	shape.cpp:367	WARN	-
PDN	0201	PdnGen.cc:461	ERROR	-
PDN	0202	PdnGen.cc:465	ERROR	-
PDN	0203	PdnGen.cc:987	ERROR	-
PDN	0204	PdnGen.cc:996	ERROR	-
PDN	0205	PdnGen.cc:1027	ERROR	-
PDN	0206	PdnGen.cc:1034	ERROR	-
PDN	0207	PdnGen.cc:1041	ERROR	-
PDN	0208	PdnGen.cc:1048	ERROR	-
PDN	0209	PdnGen.cc:1062	WARN	-
PDN	0210	PdnGen.cc:326	ERROR	-
PDN	0220	power_cells.cpp:243	ERROR	-
PDN	0221	power_cells.cpp:296	ERROR	-
PDN	0222	power_cells.cpp:232	WARN	-
PDN	0223	power_cells.cpp:324	WARN	-
PDN	0224	grid_component.cpp:475	ERROR	-
PDN	0225	pdn.tcl:345	ERROR	-
PDN	0226	via_repair.cpp:130	WARN	-
PDN	0227	via.cpp:907	WARN	-
PDN	0228	PdnGen.cc:800	WARN	-
PDN	0229	PdnGen.cc:343	ERROR	-
PDN	0230	pdn.tcl:475	ERROR	-
PDN	0231	grid.cpp:1519	WARN	-
PDN	0232	PdnGen.cc:168	WARN	-
PDN	0233	PdnGen.cc:175	ERROR	-
PDN	0234	PdnGen.cc:932	WARN	-
PDN	0235	PdnGen.cc:937	ERROR	-
PDN	0236	PdnGen.cc:1117	ERROR	-
PDN	0237	PdnGen.cc:1125	ERROR	-
PDN	0238	PdnGen.cc:1057	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PDN	1001	pdn.tcl:100	ERROR	-
PDN	1002	pdn.tcl:104	ERROR	-
PDN	1003	pdn.tcl:109	ERROR	-
PDN	1004	pdn.tcl:113	ERROR	-
PDN	1005	pdn.tcl:122	ERROR	-
PDN	1006	pdn.tcl:137	ERROR	-
PDN	1007	pdn.tcl:295	ERROR	-
PDN	1008	pdn.tcl:300	ERROR	-
PDN	1009	pdn.tcl:306	ERROR	-
PDN	1011	pdn.tcl:426	ERROR	-
PDN	1013	pdn.tcl:436	ERROR	-
PDN	1014	pdn.tcl:442	ERROR	-
PDN	1015	pdn.tcl:448	ERROR	-
PDN	1016	pdn.tcl:450	ERROR	-
PDN	1017	pdn.tcl:462	ERROR	-
PDN	1019	pdn.tcl:568	ERROR	-
PDN	1020	pdn.tcl:570	ERROR	-
PDN	1021	pdn.tcl:606	ERROR	-
PDN	1022	pdn.tcl:828	ERROR	-
PDN	1023	pdn.tcl:842	ERROR	-
PDN	1024	pdn.tcl:851	WARN	-
PDN	1025	pdn.tcl:873	ERROR	-
PDN	1026	pdn.tcl:966	ERROR	-
PDN	1027	pdn.tcl:982	ERROR	-
PDN	1028	pdn.tcl:986	ERROR	-
PDN	1029	pdn.tcl:995	ERROR	-
PDN	1030	pdn.tcl:1039	ERROR	-
PDN	1032	pdn.tcl:1107	ERROR	-
PDN	1033	pdn.tcl:1154	ERROR	-
PDN	1034	pdn.tcl:1167	ERROR	-
PDN	1035	pdn.tcl:1186	ERROR	-
PDN	1036	pdn.tcl:1210	ERROR	-
PDN	1037	pdn.tcl:50	ERROR	-
PDN	1038	pdn.tcl:57	ERROR	-
PDN	1039	pdn.tcl:64	ERROR	-
PDN	1040	pdn.tcl:1503	ERROR	-
PDN	1041	pdn.tcl:1506	ERROR	-
PDN	1042	pdn.tcl:163	WARN	-
PDN	1043	pdn.tcl:877	ERROR	-
PDN	1044	pdn.tcl:998	ERROR	-
PDN	1045	pdn.tcl:906	ERROR	-
PDN	1046	pdn.tcl:228	ERROR	-
PDN	1047	pdn.tcl:1193	ERROR	-
PDN	1048	pdn.tcl:902	ERROR	-
PDN	1049	pdn.tcl:910	ERROR	-
PDN	1183	pdn.tcl:224	ERROR	-
PDN	1184	pdn.tcl:233	ERROR	-
PDN	1186	pdn.tcl:244	ERROR	-
PDN	1187	pdn.tcl:250	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PDN	1188	pdn.tcl:256	ERROR	-
PDN	1190	pdn.tcl:784	ERROR	-
PDN	1191	pdn.tcl:774	ERROR	-
PDN	1192	pdn.tcl:777	ERROR	-
PDN	9002	PdnGen.tcl:3146	WARN	-
PDN	9003	PdnGen.tcl:3150	WARN	-
PDN	9004	PdnGen.tcl:3160	WARN	-
PDN	9006	PdnGen.tcl:5364	WARN	-
PDN	9008	PdnGen.tcl:4932	INFO	-
PDN	9009	PdnGen.tcl:4979	INFO	-
PDN	9010	PdnGen.tcl:6787	INFO	-
PDN	9011	PdnGen.tcl:6823	INFO	-
PDN	9012	PdnGen.tcl:6833	INFO	-
PDN	9013	PdnGen.tcl:6837	INFO	-
PDN	9014	PdnGen.tcl:6839	INFO	-
PDN	9015	PdnGen.tcl:6868	INFO	-
PDN	9016	PdnGen.tcl:6888	INFO	-
PDN	9017	PdnGen.tcl:4234	ERROR	-
PDN	9018	PdnGen.tcl:6820	WARN	-
PDN	9019	PdnGen.tcl:1472	ERROR	-
PDN	9020	PdnGen.tcl:1644	ERROR	-
PDN	9021	PdnGen.tcl:1709	ERROR	-
PDN	9022	PdnGen.tcl:2922	ERROR	-
PDN	9023	PdnGen.tcl:2923	ERROR	-
PDN	9024	PdnGen.tcl:2930	ERROR	-
PDN	9025	PdnGen.tcl:3316	ERROR	-
PDN	9026	PdnGen.tcl:3462	ERROR	-
PDN	9027	PdnGen.tcl:4742	ERROR	-
PDN	9028	PdnGen.tcl:6898	ERROR	-
PDN	9029	PdnGen.tcl:4885	ERROR	-
PDN	9030	PdnGen.tcl:5024	ERROR	-
PDN	9032	PdnGen.tcl:5644	INFO	-
PDN	9033	PdnGen.tcl:1380	ERROR	-
PDN	9034	PdnGen.tcl:6790	INFO	-
PDN	9035	PdnGen.tcl:5724	WARN	-
PDN	9036	PdnGen.tcl:2756	WARN	-
PDN	9037	PdnGen.tcl:4786	ERROR	-
PDN	9038	PdnGen.tcl:2729	WARN	-
PDN	9039	PdnGen.tcl:2750	WARN	-
PDN	9040	PdnGen.tcl:3179	WARN	-
PDN	9041	PdnGen.tcl:3185	WARN	-
PDN	9042	PdnGen.tcl:3193	WARN	-
PDN	9043	PdnGen.tcl:3199	WARN	-
PDN	9044	PdnGen.tcl:3080	ERROR	-
PDN	9045	PdnGen.tcl:3101	ERROR	-
PDN	9048	PdnGen.tcl:3983	ERROR	-
PDN	9051	PdnGen.tcl:5970	ERROR	-
PDN	9052	PdnGen.tcl:3042	ERROR	-
PDN	9055	PdnGen.tcl:5089	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PDN	9056	PdnGen.tcl:5093	WARN	-
PDN	9062	PdnGen.tcl:6894	ERROR	-
PDN	9063	PdnGen.tcl:2909	WARN	-
PDN	9064	PdnGen.tcl:4043	WARN	-
PDN	9065	PdnGen.tcl:4046	WARN	-
PDN	9066	PdnGen.tcl:4049	WARN	-
PDN	9067	PdnGen.tcl:4052	WARN	-
PDN	9068	PdnGen.tcl:4055	ERROR	-
PDN	9069	PdnGen.tcl:4603	ERROR	-
PDN	9070	PdnGen.tcl:4648	ERROR	-
PDN	9071	PdnGen.tcl:4654	ERROR	-
PDN	9072	PdnGen.tcl:87	ERROR	-
PDN	9075	PdnGen.tcl:116	ERROR	-
PDN	9076	PdnGen.tcl:119	ERROR	-
PDN	9079	PdnGen.tcl:151	ERROR	-
PDN	9081	PdnGen.tcl:158	ERROR	-
PDN	9083	PdnGen.tcl:177	ERROR	-
PDN	9084	PdnGen.tcl:184	ERROR	-
PDN	9085	PdnGen.tcl:198	ERROR	-
PDN	9086	PdnGen.tcl:201	ERROR	-
PDN	9087	PdnGen.tcl:210	ERROR	-
PDN	9088	PdnGen.tcl:549	ERROR	-
PDN	9090	PdnGen.tcl:1195	ERROR	-
PDN	9095	PdnGen.tcl:277	ERROR	-
PDN	9109	PdnGen.tcl:233	ERROR	-
PDN	9110	PdnGen.tcl:288	ERROR	-
PDN	9111	PdnGen.tcl:300	ERROR	-
PDN	9112	PdnGen.tcl:310	WARN	-
PDN	9114	PdnGen.tcl:706	ERROR	-
PDN	9115	PdnGen.tcl:911	ERROR	-
PDN	9116	PdnGen.tcl:924	ERROR	-
PDN	9117	PdnGen.tcl:937	ERROR	-
PDN	9118	PdnGen.tcl:950	ERROR	-
PDN	9119	PdnGen.tcl:977	ERROR	-
PDN	9121	PdnGen.tcl:241	ERROR	-
PDN	9124	PdnGen.tcl:740	ERROR	-
PDN	9125	PdnGen.tcl:963	ERROR	-
PDN	9126	PdnGen.tcl:1008	ERROR	-
PDN	9127	PdnGen.tcl:321	ERROR	-
PDN	9128	PdnGen.tcl:362	ERROR	-
PDN	9129	PdnGen.tcl:399	ERROR	-
PDN	9130	PdnGen.tcl:446	ERROR	-
PDN	9138	PdnGen.tcl:456	ERROR	-
PDN	9139	PdnGen.tcl:261	ERROR	-
PDN	9140	PdnGen.tcl:267	ERROR	-
PDN	9141	PdnGen.tcl:269	ERROR	-
PDN	9146	PdnGen.tcl:257	ERROR	-
PDN	9147	PdnGen.tcl:1143	ERROR	-
PDN	9148	PdnGen.tcl:1146	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PDN	9149	PdnGen.tcl:835	ERROR	-
PDN	9150	PdnGen.tcl:849	WARN	-
PDN	9151	PdnGen.tcl:868	ERROR	-
PDN	9152	PdnGen.tcl:882	WARN	-
PDN	9153	PdnGen.tcl:1155	ERROR	-
PDN	9154	PdnGen.tcl:1158	ERROR	-
PDN	9155	PdnGen.tcl:1168	ERROR	-
PDN	9156	PdnGen.tcl:1171	ERROR	-
PDN	9158	PdnGen.tcl:757	ERROR	-
PDN	9159	PdnGen.tcl:768	ERROR	-
PDN	9160	PdnGen.tcl:5691	ERROR	-
PDN	9164	PdnGen.tcl:500	ERROR	-
PDN	9165	PdnGen.tcl:1097	ERROR	-
PDN	9166	PdnGen.tcl:1281	ERROR	-
PDN	9168	PdnGen.tcl:2998	ERROR	-
PDN	9169	PdnGen.tcl:6136	WARN	-
PDN	9170	PdnGen.tcl:6156	WARN	-
PDN	9171	PdnGen.tcl:6164	WARN	-
PDN	9172	PdnGen.tcl:6173	WARN	-
PDN	9176	PdnGen.tcl:381	ERROR	-
PDN	9177	PdnGen.tcl:4792	WARN	-
PDN	9178	PdnGen.tcl:490	ERROR	-
PDN	9179	PdnGen.tcl:480	ERROR	-
PDN	9180	PdnGen.tcl:346	ERROR	-
PDN	9181	PdnGen.tcl:330	WARN	-
PDN	9190	PdnGen.tcl:421	ERROR	-
PDN	9191	PdnGen.tcl:3808	ERROR	-
PDN	9192	PdnGen.tcl:3813	ERROR	-
PDN	9193	PdnGen.tcl:3826	ERROR	-
PDN	9194	PdnGen.tcl:509	ERROR	-
PDN	9195	PdnGen.tcl:521	ERROR	-
PDN	9196	PdnGen.tcl:3863	ERROR	-
PDN	9197	PdnGen.tcl:3854	ERROR	-
PDN	9248	PdnGen.tcl:5658	ERROR	-
PPL	0001	IOPlacer.cpp:1380	INFO	-
PPL	0002	IOPlacer.cpp:1385	INFO	-
PPL	0003	IOPlacer.cpp:1387	INFO	-
PPL	0004	IOPlacer.cpp:1391	INFO	-
PPL	0005	IOPlacer.cpp:1393	INFO	-
PPL	0006	IOPlacer.cpp:1394	INFO	-
PPL	0007	IOPlacer.cpp:2073	INFO	-
PPL	0008	IOPlacer.cpp:1327	INFO	-
PPL	0009	IOPlacer.cpp:1330	INFO	-
PPL	0010	IOPlacer.cpp:1405	INFO	-
PPL	0011	IOPlacer.cpp:949	ERROR	-
PPL	0012	IOPlacer.cpp:2276	INFO	-
PPL	0013	IOPlacer.cpp:1318	ERROR	-
PPL	0015	IOPlacer.tcl:486	WARN	-
PPL	0016	IOPlacer.tcl:167	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PPL	0017	IOPlacer.tcl:504	ERROR	-
PPL	0018	IOPlacer.tcl:510	ERROR	-
PPL	0019	IOPlacer.tcl:539	ERROR	-
PPL	0020	IOPlacer.cpp:1472	ERROR	-
PPL	0021	IOPlacer.tcl:547	ERROR	-
PPL	0023	IOPlacer.tcl:565	ERROR	-
PPL	0024	IOPlacer.cpp:937	ERROR	-
PPL	0025	IOPlacer.tcl:605	ERROR	-
PPL	0027	IOPlacer.tcl:648	ERROR	-
PPL	0028	IOPlacer.tcl:661	ERROR	-
PPL	0029	IOPlacer.tcl:674	ERROR	-
PPL	0030	IOPlacer.tcl:682	ERROR	-
PPL	0031	IOPlacer.tcl:471	ERROR	-
PPL	0032	IOPlacer.tcl:476	ERROR	-
PPL	0033	HungarianMatching.cpp:124	WARN	-
PPL	0034	IOPlacer.cpp:2329	ERROR	-
PPL	0035	IOPlacer.cpp:699	ERROR	-
PPL	0036	IOPlacer.cpp:1412	WARN	-
PPL	0037	IOPlacer.cpp:1421	WARN	-
PPL	0038	IOPlacer.cpp:2752	WARN	-
PPL	0039	IOPlacer.cpp:2140	ERROR	-
PPL	0040	IOPlacer.cpp:981	ERROR	-
PPL	0041	IOPlacer.tcl:617	INFO	-
PPL	0042	IOPlacer.cpp:1263	WARN	-
PPL	0043	IOPlacer.tcl:624	WARN	-
PPL	0044	IOPlacer.cpp:1992	INFO	-
PPL	0045	IOPlacer.tcl:551	ERROR	-
PPL	0046	IOPlacer.tcl:569	ERROR	-
PPL	0047	IOPlacer.tcl:225	WARN	-
PPL	0048	IOPlacer.cpp:1644	INFO	-
PPL	0050	IOPlacer.tcl:699	ERROR	-
PPL	0051	IOPlacer.tcl:704	ERROR	-
PPL	0052	IOPlacer.tcl:55	ERROR	-
PPL	0053	IOPlacer.tcl:58	ERROR	-
PPL	0054	IOPlacer.tcl:65	ERROR	-
PPL	0055	IOPlacer.tcl:89	ERROR	-
PPL	0056	IOPlacer.tcl:95	ERROR	-
PPL	0057	IOPlacer.tcl:101	ERROR	-
PPL	0058	IOPlacer.tcl:214	ERROR	-
PPL	0059	IOPlacer.tcl:198	ERROR	-
PPL	0061	IOPlacer.tcl:740	ERROR	-
PPL	0062	IOPlacer.cpp:1382	INFO	-
PPL	0063	IOPlacer.tcl:85	ERROR	-
PPL	0064	IOPlacer.tcl:390	ERROR	-
PPL	0065	IOPlacer.tcl:396	ERROR	-
PPL	0066	IOPlacer.tcl:402	ERROR	-
PPL	0068	IOPlacer.tcl:406	ERROR	-
PPL	0069	IOPlacer.tcl:419	ERROR	-
PPL	0070	IOPlacer.cpp:2412	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PPL	0071	IOPlacer.tcl:427	ERROR	-
PPL	0072	IOPlacer.cpp:237	ERROR	-
PPL	0073	IOPlacer.tcl:206	WARN	-
PPL	0075	IOPlacer.cpp:1760	WARN	-
PPL	0076	IOPlacer.cpp:1829	ERROR	-
PPL	0077	IOPlacer.cpp:1480	ERROR	-
PPL	0078	IOPlacer.cpp:1248	WARN	-
PPL	0079	IOPlacer.cpp:1551	ERROR	-
PPL	0081	IOPlacer.tcl:240	ERROR	-
PPL	0082	HungarianMatching.cpp:173	ERROR	-
PPL	0083	IOPlacer.tcl:137	ERROR	-
PPL	0084	IOPlacer.cpp:1744	WARN	-
PPL	0085	IOPlacer.cpp:316	ERROR	-
PPL	0086	HungarianMatching.cpp:326	ERROR	-
PPL	0087	IOPlacer.tcl:141	ERROR	-
PPL	0088	IOPlacer.cpp:2106	ERROR	-
PPL	0089	HungarianMatching.cpp:266	ERROR	-
PPL	0090	IOPlacer.cpp:358	ERROR	-
PPL	0091	IOPlacer.cpp:452	ERROR	-
PPL	0092	IOPlacer.cpp:2082	WARN	-
PPL	0093	IOPlacer.cpp:384	ERROR	-
PPL	0094	IOPlacer.cpp:2808	ERROR	-
PPL	0095	IOPlacer.tcl:234	ERROR	-
PPL	0096	IOPlacer.cpp:229	WARN	-
PPL	0097	IOPlacer.cpp:530	WARN	-
PPL	0098	IOPlacer.cpp:1893	ERROR	-
PPL	0100	IOPlacer.cpp:566	INFO	-
PPL	0101	IOPlacer.cpp:477	ERROR	-
PPL	0104	IOPlacer.cpp:1922	ERROR	-
PPL	0106	IOPlacer.cpp:2252	WARN	-
PPL	0107	IOPlacer.cpp:2268	ERROR	-
PPL	0108	IOPlacer.tcl:368	ERROR	-
PPL	0109	IOPlacer.cpp:409	ERROR	-
PPL	0110	IOPlacer.cpp:1813	WARN	-
PPL	0111	IOPlacer.cpp:1845	ERROR	-
PPL	0112	SimulatedAnnealing.cpp:854	ERROR	-
PSM	0001	pdnsim.cpp:89	INFO	-
PSM	0004	ir_solver.cpp:1915	INFO	-
PSM	0008	ir_solver.cpp:167	WARN	-
PSM	0010	ir_solver.cpp:198	ERROR	-
PSM	0012	ir_solver.cpp:209	ERROR	-
PSM	0014	ir_solver.cpp:390	ERROR	-
PSM	0015	ir_solver.cpp:444	INFO	-
PSM	0016	ir_solver.cpp:479	WARN	-
PSM	0017	ir_solver.cpp:494	WARN	-
PSM	0018	ir_solver.cpp:502	WARN	-
PSM	0022	ir_solver.cpp:421	INFO	-
PSM	0024	ir_solver.cpp:757	WARN	-
PSM	0028	pdnsim.tcl:205	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PSM	0030	ir_solver.cpp:1300	WARN	-
PSM	0031	ir_solver.cpp:1404	INFO	-
PSM	0032	ir_solver.cpp:1177	WARN	-
PSM	0033	ir_solver.cpp:1188	WARN	-
PSM	0035	ir_solver.cpp:1157	ERROR	-
PSM	0036	ir_solver.cpp:1210	ERROR	-
PSM	0037	ir_solver.cpp:1231	ERROR	-
PSM	0038	ir_solver.cpp:1478	WARN	-
PSM	0039	ir_solver.cpp:1488	WARN	-
PSM	0040	ir_solver.cpp:1510	INFO	-
PSM	0041	ir_solver.cpp:1829	ERROR	-
PSM	0042	ir_solver.cpp:738	ERROR	-
PSM	0045	gmat.cpp:191	ERROR	-
PSM	0046	gmat.cpp:204	ERROR	-
PSM	0047	gmat.cpp:206	ERROR	-
PSM	0048	gmat.cpp:289	INFO	-
PSM	0049	gmat.cpp:341	ERROR	-
PSM	0050	gmat.cpp:395	WARN	-
PSM	0051	gmat.cpp:570	ERROR	-
PSM	0052	gmat.cpp:596	ERROR	-
PSM	0054	pdnsim.tcl:61	ERROR	-
PSM	0055	pdnsim.tcl:100	ERROR	-
PSM	0057	pdnsim.tcl:121	ERROR	-
PSM	0058	pdnsim.tcl:133	ERROR	-
PSM	0059	pdnsim.tcl:154	ERROR	-
PSM	0060	pdnsim.tcl:163	ERROR	-
PSM	0063	ir_solver.cpp:512	WARN	-
PSM	0064	ir_solver.cpp:392	INFO	-
PSM	0065	ir_solver.cpp:535	WARN	-
PSM	0066	ir_solver.cpp:1254	ERROR	-
PSM	0067	ir_solver.cpp:1320	WARN	-
PSM	0068	pdnsim.cpp:255	ERROR	-
PSM	0069	pdnsim.tcl:129	ERROR	-
PSM	0070	ir_solver.cpp:1392	WARN	-
PSM	0071	ir_solver.cpp:720	WARN	-
PSM	0073	ir_solver.cpp:1350	INFO	-
PSM	0075	ir_solver.cpp:469	ERROR	-
PSM	0076	ir_solver.cpp:1371	INFO	-
PSM	0078	pdnsim.cpp:167	ERROR	-
PSM	0079	pdnsim.cpp:286	ERROR	-
PSM	0080	gmat.cpp:111	WARN	-
PSM	0081	ir_solver.cpp:906	ERROR	-
PSM	0082	ir_solver.cpp:1365	ERROR	-
PSM	0084	pdnsim.cpp:101	ERROR	-
PSM	0085	pdnsim.tcl:158	ERROR	-
PSM	0086	ir_solver.cpp:139	ERROR	-
PSM	0087	ir_solver.cpp:129	ERROR	-
PSM	0088	ir_solver.cpp:126	ERROR	-
PSM	0089	ir_solver.cpp:441	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
PSM	0090	ir_solver.cpp:300	ERROR	-
PSM	0091	ir_solver.cpp:335	ERROR	-
PSM	0092	ir_solver.cpp:1743	ERROR	-
PSM	0093	ir_solver.cpp:412	ERROR	-
PSM	0094	ir_solver.cpp:1503	WARN	-
RCX	0001	ext.cpp:323	INFO	-
RCX	0002	ext.cpp:331	ERROR	-
RCX	0003	netRC.cpp:2263	WARN	-
RCX	0004	netRC.cpp:2259	WARN	-
RCX	0005	netRC.cpp:2275	WARN	-
RCX	0007	extBench.cpp:486	INFO	-
RCX	0008	ext.cpp:237	INFO	-
RCX	0015	ext.cpp:254	INFO	-
RCX	0016	ext.cpp:292	INFO	-
RCX	0017	ext.cpp:317	INFO	-
RCX	0019	ext.cpp:381	INFO	-
RCX	0021	ext.cpp:434	INFO	-
RCX	0029	ext.cpp:190	INFO	-
RCX	0030	ext.cpp:201	ERROR	-
RCX	0031	ext.cpp:210	INFO	-
RCX	0040	netRC.cpp:1815	INFO	-
RCX	0042	extSpef.cpp:1585	INFO	-
RCX	0043	extFlow.cpp:1236	INFO	-
RCX	0044	extSpefIn.cpp:2682	WARN	-
RCX	0045	netRC.cpp:1920	INFO	-
RCX	0047	extSpef.cpp:1624	INFO	-
RCX	0048	extSpefIn.cpp:2673	WARN	-
RCX	0049	extSpefIn.cpp:2676	WARN	-
RCX	0050	extSpefIn.cpp:2679	WARN	-
RCX	0052	extSpefIn.cpp:2686	ERROR	-
RCX	0055	extBench.cpp:295	INFO	-
RCX	0057	extBench.cpp:355	INFO	-
RCX	0058	extBench.cpp:417	INFO	-
RCX	0060	extSpefIn.cpp:2629	INFO	-
RCX	0065	extmain.cpp:657	INFO	-
RCX	0069	extRCmodel.cpp:3448	INFO	-
RCX	0072	extRCmodel.cpp:3566	INFO	-
RCX	0074	extmeasure.cpp:81	WARN	-
RCX	0076	extSpefIn.cpp:587	WARN	-
RCX	0077	extSpefIn.cpp:758	WARN	-
RCX	0078	extSpefIn.cpp:2254	WARN	-
RCX	0079	extmeasure.cpp:1858	INFO	-
RCX	0081	extFlow.cpp:312	ERROR	-
RCX	0082	extFlow.cpp:354	ERROR	-
RCX	0107	netRC.cpp:1928	WARN	-
RCX	0110	netRC.cpp:577	INFO	-
RCX	0111	netRC.cpp:625	WARN	-
RCX	0112	netRC.cpp:723	ERROR	-
RCX	0113	netRC.cpp:728	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
RCX	0114	netRC.cpp:736	WARN	-
RCX	0115	netRC.cpp:743	ERROR	-
RCX	0120	netRC.cpp:1157	WARN	-
RCX	0121	netRC.cpp:1303	INFO	-
RCX	0122	netRC.cpp:1319	INFO	-
RCX	0127	netRC.cpp:1727	WARN	-
RCX	0128	netRC.cpp:1749	INFO	-
RCX	0129	netRC.cpp:1753	WARN	-
RCX	0134	netRC.cpp:2157	INFO	-
RCX	0135	netRC.cpp:2082	WARN	-
RCX	0136	netRC.cpp:2106	INFO	-
RCX	0137	netRC.cpp:2176	INFO	-
RCX	0138	extmain.cpp:335	WARN	-
RCX	0139	extmain.cpp:327	WARN	-
RCX	0140	extmain.cpp:550	INFO	-
RCX	0141	extmain.cpp:635	INFO	-
RCX	0142	extmain.cpp:647	INFO	-
RCX	0143	extmain.cpp:659	INFO	-
RCX	0147	ext.cpp:175	ERROR	-
RCX	0148	ext.cpp:229	WARN	-
RCX	0149	OpenRCX.tcl:337	WARN	-
RCX	0152	extprocess.cpp:294	WARN	-
RCX	0153	extprocess.cpp:325	WARN	-
RCX	0154	extprocess.cpp:530	WARN	-
RCX	0158	extprocess.cpp:269	WARN	-
RCX	0159	extprocess.cpp:547	ERROR	-
RCX	0171	extSpef.cpp:193	ERROR	-
RCX	0172	extSpef.cpp:198	ERROR	-
RCX	0175	extSpef.cpp:1142	WARN	-
RCX	0176	extSpef.cpp:1420	INFO	-
RCX	0178	extSpefIn.cpp:2337	INFO	-
RCX	0208	extRCmodel.cpp:149	INFO	-
RCX	0216	extRCmodel.cpp:3578	INFO	-
RCX	0217	extRCmodel.cpp:3584	INFO	-
RCX	0218	extRCmodel.cpp:3633	INFO	-
RCX	0219	extRCmodel.cpp:3646	INFO	-
RCX	0220	extRCmodel.cpp:3669	INFO	-
RCX	0221	extRCmodel.cpp:3682	INFO	-
RCX	0222	extRCmodel.cpp:3928	WARN	-
RCX	0223	extRCmodel.cpp:3952	WARN	-
RCX	0239	extFlow.cpp:1386	WARN	-
RCX	0240	extFlow.cpp:1397	WARN	-
RCX	0252	extmain.cpp:63	ERROR	-
RCX	0258	extSpefIn.cpp:156	WARN	-
RCX	0259	extSpefIn.cpp:188	ERROR	-
RCX	0260	extSpefIn.cpp:464	ERROR	-
RCX	0261	extSpefIn.cpp:641	WARN	-
RCX	0262	extSpefIn.cpp:558	ERROR	-
RCX	0263	extSpefIn.cpp:614	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
RCX	0264	extSpefIn.cpp:902	WARN	-
RCX	0265	extSpefIn.cpp:1095	INFO	-
RCX	0266	extSpefIn.cpp:1148	INFO	-
RCX	0267	extSpefIn.cpp:1349	INFO	-
RCX	0269	extSpefIn.cpp:1504	WARN	-
RCX	0270	extSpefIn.cpp:1545	WARN	-
RCX	0271	extSpefIn.cpp:1622	WARN	-
RCX	0272	extSpefIn.cpp:1639	WARN	-
RCX	0273	extSpefIn.cpp:1684	ERROR	-
RCX	0274	extSpefIn.cpp:1691	WARN	-
RCX	0275	extSpefIn.cpp:1699	WARN	-
RCX	0276	extSpefIn.cpp:1701	WARN	-
RCX	0277	extSpefIn.cpp:1745	WARN	-
RCX	0278	extSpefIn.cpp:1772	WARN	-
RCX	0279	extSpefIn.cpp:1799	WARN	-
RCX	0280	extSpefIn.cpp:1857	WARN	-
RCX	0281	extSpefIn.cpp:1913	WARN	-
RCX	0282	extSpefIn.cpp:1998	WARN	-
RCX	0283	extSpefIn.cpp:2243	INFO	-
RCX	0284	extSpefIn.cpp:2251	WARN	-
RCX	0285	extSpefIn.cpp:2585	WARN	-
RCX	0286	extSpefIn.cpp:2382	INFO	-
RCX	0287	extSpefIn.cpp:2402	INFO	-
RCX	0288	extSpefIn.cpp:2409	INFO	-
RCX	0289	extSpefIn.cpp:2423	INFO	-
RCX	0290	extSpefIn.cpp:2433	INFO	-
RCX	0291	extSpefIn.cpp:2473	INFO	-
RCX	0292	extSpefIn.cpp:2582	WARN	-
RCX	0293	extSpefIn.cpp:2700	INFO	-
RCX	0294	extSpefIn.cpp:2718	INFO	-
RCX	0295	extSpefIn.cpp:2822	WARN	-
RCX	0296	extSpefIn.cpp:2852	WARN	-
RCX	0297	extSpefIn.cpp:2857	WARN	-
RCX	0298	extSpefIn.cpp:2859	WARN	-
RCX	0357	OpenRCX.tcl:236	ERROR	-
RCX	0358	extRCmodel.cpp:3572	INFO	-
RCX	0374	extSpefIn.cpp:1607	WARN	-
RCX	0376	netRC.cpp:2325	INFO	-
RCX	0378	ext.cpp:179	ERROR	-
RCX	0380	ext.cpp:378	ERROR	-
RCX	0381	ext.cpp:429	ERROR	-
RCX	0404	extSpefIn.cpp:2393	INFO	-
RCX	0405	extSpefIn.cpp:1791	WARN	-
RCX	0406	extSpefIn.cpp:1738	WARN	-
RCX	0407	extSpefIn.cpp:1780	WARN	-
RCX	0410	extRCmodel.cpp:3620	INFO	-
RCX	0411	extRCmodel.cpp:2773	INFO	-
RCX	0414	extRCmodel.cpp:2767	INFO	-
RCX	0416	extRCmodel.cpp:2749	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
RCX	0417	extRCmodel.cpp:2731	INFO	-
RCX	0418	extRCmodel.cpp:2707	WARN	-
RCX	0431	netRC.cpp:1260	INFO	-
RCX	0433	netRC.cpp:1236	INFO	-
RCX	0434	netRC.cpp:1266	INFO	-
RCX	0435	netRC.cpp:1605	INFO	-
RCX	0436	netRC.cpp:1787	INFO	-
RCX	0437	netRC.cpp:100	INFO	-
RCX	0438	netRC.cpp:89	INFO	-
RCX	0439	netRC.cpp:1819	INFO	-
RCX	0440	netRC.cpp:1830	INFO	-
RCX	0442	extFlow.cpp:1356	INFO	-
RCX	0443	extSpef.cpp:1591	INFO	-
RCX	0444	extSpefIn.cpp:2264	INFO	-
RCX	0445	extSpefIn.cpp:2555	INFO	-
RCX	0447	extSpefIn.cpp:2661	WARN	-
RCX	0448	extSpefIn.cpp:2649	WARN	-
RCX	0449	extSpefIn.cpp:2707	INFO	-
RCX	0451	extSpefIn.cpp:2710	INFO	-
RCX	0452	extSpefIn.cpp:82	WARN	-
RCX	0456	extmeasure.cpp:962	INFO	-
RCX	0458	extmeasure.cpp:1031	INFO	-
RCX	0459	extmeasure.cpp:641	INFO	-
RCX	0460	extmeasure.cpp:91	WARN	-
RCX	0463	extSpefIn.cpp:2619	INFO	-
RCX	0464	extSpefIn.cpp:2632	INFO	-
RCX	0465	extSpef.cpp:1621	INFO	-
RCX	0468	netRC.cpp:1610	ERROR	-
RCX	0472	netRC.cpp:1286	INFO	-
RCX	0474	netRC.cpp:2044	INFO	-
RCX	0475	netRC.cpp:2141	INFO	-
RCX	0476	extmain.cpp:649	INFO	-
RCX	0480	netRC.cpp:2343	INFO	-
RCX	0485	extRCmodel.cpp:2643	INFO	-
RCX	0487	netRC.cpp:1632	ERROR	-
RCX	0489	extRCmodel.cpp:3155	ERROR	-
RCX	0490	extRCmodel.cpp:3450	ERROR	-
RCX	0491	extRCmodel.cpp:3459	ERROR	-
RCX	0497	extmain.cpp:370	ERROR	-
RMP	0001	blif.cpp:105	ERROR	-
RMP	0002	blif.cpp:360	INFO	-
RMP	0003	blif.cpp:384	ERROR	-
RMP	0004	blif.cpp:407	ERROR	-
RMP	0005	blif.cpp:429	INFO	-
RMP	0006	blif.cpp:433	INFO	-
RMP	0007	blif.cpp:458	INFO	-
RMP	0008	blif.cpp:477	INFO	-
RMP	0009	blif.cpp:521	INFO	-
RMP	0010	blif.cpp:574	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
RMP	0012	rmp.tcl:97	ERROR	-
RMP	0016	Restructure.cpp:620	ERROR	-
RMP	0020	Restructure.cpp:485	ERROR	-
RMP	0021	Restructure.cpp:292	INFO	-
RMP	0025	Restructure.cpp:642	WARN	-
RMP	0026	Restructure.cpp:225	ERROR	-
RMP	0032	rmp.tcl:113	WARN	-
RMP	0033	rmp.tcl:129	WARN	-
RMP	0034	blif.cpp:421	ERROR	-
RMP	0035	Restructure.cpp:453	WARN	-
RMP	0036	Restructure.cpp:594	WARN	-
RMP	0037	Restructure.cpp:299	ERROR	-
RMP	0076	blif.cpp:541	ERROR	-
RMP	0146	blif.cpp:595	INFO	-
RSZ	0001	Resizer.tcl:145	ERROR	-
RSZ	0002	Resizer.tcl:150	ERROR	-
RSZ	0003	Resizer.tcl:231	ERROR	-
RSZ	0004	Resizer.tcl:607	ERROR	-
RSZ	0005	Resizer.tcl:228	ERROR	-
RSZ	0010	Resizer.tcl:192	WARN	-
RSZ	0011	Resizer.tcl:195	WARN	-
RSZ	0020	Resizer.tcl:523	WARN	-
RSZ	0021	Resizer.tcl:574	WARN	-
RSZ	0022	Resizer.cc:471	ERROR	-
RSZ	0023	Resizer.cc:2920	ERROR	-
RSZ	0025	RepairSetup.cc:294	ERROR	-
RSZ	0026	Resizer.cc:278	INFO	-
RSZ	0027	Resizer.cc:522	INFO	-
RSZ	0028	Resizer.cc:652	INFO	-
RSZ	0030	RepairSetup.cc:318	INFO	-
RSZ	0031	RepairSetup.cc:321	INFO	-
RSZ	0032	RepairHold.cc:307	INFO	-
RSZ	0033	RepairHold.cc:318	INFO	-
RSZ	0034	RepairDesign.cc:125	INFO	-
RSZ	0035	RepairDesign.cc:128	INFO	-
RSZ	0036	RepairDesign.cc:131	INFO	-
RSZ	0037	RepairDesign.cc:134	INFO	-
RSZ	0038	RepairDesign.cc:137	INFO	-
RSZ	0039	RepairDesign.cc:142	INFO	-
RSZ	0040	RepairSetup.cc:273	INFO	-
RSZ	0041	RepairSetup.cc:281	INFO	-
RSZ	0042	Resizer.cc:1870	INFO	-
RSZ	0043	RepairSetup.cc:284	INFO	-
RSZ	0044	RepairSetup.cc:324	INFO	-
RSZ	0045	RepairSetup.cc:276	INFO	-
RSZ	0046	RepairHold.cc:265	INFO	-
RSZ	0047	RepairDesign.cc:271	INFO	-
RSZ	0048	RepairDesign.cc:274	INFO	-
RSZ	0049	RepairSetup.cc:287	INFO	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
RSZ	0050	RepairHold.cc:314	ERROR	-
RSZ	0051	RepairDesign.cc:321	INFO	-
RSZ	0052	RepairDesign.cc:324	INFO	-
RSZ	0053	RepairDesign.cc:327	INFO	-
RSZ	0054	RepairDesign.cc:330	INFO	-
RSZ	0055	RepairDesign.cc:333	INFO	-
RSZ	0056	RepairDesign.cc:339	INFO	-
RSZ	0057	RepairDesign.cc:342	INFO	-
RSZ	0058	Resizer.tcl:648	INFO	-
RSZ	0060	RepairHold.cc:311	ERROR	-
RSZ	0062	RepairSetup.cc:291	WARN	-
RSZ	0064	RepairHold.cc:303	WARN	-
RSZ	0065	Resizer.tcl:643	WARN	-
RSZ	0066	RepairHold.cc:301	WARN	-
RSZ	0067	Resizer.tcl:585	WARN	-
RSZ	0068	Resizer.cc:1602	ERROR	-
RSZ	0069	SteinerTree.cc:91	WARN	-
RSZ	0070	Resizer.cc:2375	ERROR	-
RSZ	0071	Rebuffer.cc:297	CRITICAL	-
RSZ	0072	RepairDesign.cc:745	CRITICAL	-
RSZ	0073	BufferedNet.cc:670	WARN	-
RSZ	0074	BufferedNet.cc:674	WARN	-
RSZ	0075	Rebuffer.cc:132	WARN	-
RSZ	0076	Resizer.tcl:435	WARN	-
RSZ	0077	Resizer.cc:3133	WARN	-
RSZ	0078	BufferedNet.cc:83	CRITICAL	-
RSZ	0079	BufferedNet.cc:118	CRITICAL	-
RSZ	0080	BufferedNet.cc:146	CRITICAL	-
RSZ	0081	BufferedNet.cc:176	CRITICAL	-
RSZ	0082	BufferedNet.cc:346	CRITICAL	-
RSZ	0083	RepairDesign.cc:1146	CRITICAL	-
RSZ	0084	Resizer.cc:670	WARN	-
RSZ	0085	Resizer.cc:573	WARN	-
RSZ	0086	Resizer.cc:1477	ERROR	-
RSZ	0088	Resizer.cc:2332	WARN	-
RSZ	0089	Resizer.cc:2347	ERROR	-
RSZ	0090	PreChecks.cc:81	ERROR	-
RSZ	0091	RepairSetup.cc:915	ERROR	-
RSZ	0092	SteinerTree.cc:394	ERROR	-
RSZ	0093	SteinerTree.cc:323	ERROR	-
RSZ	0094	RepairSetup.cc:142	INFO	-
RSZ	0095	Resizer.tcl:531	WARN	-
RSZ	0096	Resizer.cc:973	WARN	-
RSZ	0125	RecoverPower.cc:201	ERROR	-
RSZ	0141	RecoverPower.cc:198	INFO	-
RSZ	0142	RecoverPower.cc:176	INFO	-
RSZ	3111	RecoverPower.cc:222	INFO	-
STA	1000	dbSta.cc:712	ERROR	-
STT	0001	SteinerTreeBuilder.tcl:53	ERROR	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
STT	0004	SteinerTreeBuilder.cpp:248	ERROR	-
STT	0005	SteinerTreeBuilder.cpp:267	ERROR	-
STT	0006	SteinerTreeBuilder.tcl:92	ERROR	-
STT	0007	SteinerTreeBuilder.cpp:304	ERROR	-
TAP	0003	tapcell.cpp:541	INFO	-
TAP	0004	tapcell.cpp:544	INFO	-
TAP	0005	tapcell.cpp:168	INFO	-
TAP	0010	tapcell.tcl:171	ERROR	-
TAP	0011	tapcell.tcl:181	ERROR	-
TAP	0014	tapcell.tcl:75	WARN	-
TAP	0015	tapcell.tcl:144	WARN	-
TAP	0016	tapcell.tcl:148	WARN	-
TAP	0020	tapcell.cpp:1218	ERROR	-
TAP	0032	tapcell.cpp:344	WARN	-
TAP	0033	tapcell.cpp:379	ERROR	-
TAP	0034	tapcell.tcl:219	ERROR	-
TAP	0035	tapcell.i:62	ERROR	-
TAP	0100	tapcell.tcl:252	INFO	-
TAP	0101	tapcell.tcl:254	INFO	-
TAP	0102	tapcell.tcl:387	ERROR	-
TAP	0103	tapcell.tcl:414	ERROR	-
TAP	0104	tapcell.cpp:1419	ERROR	-
UPF	0001	upf.cpp:51	WARN	-
UPF	0002	upf.cpp:67	WARN	-
UPF	0003	upf.cpp:84	WARN	-
UPF	0004	upf.cpp:97	WARN	-
UPF	0005	upf.cpp:108	WARN	-
UPF	0006	upf.cpp:125	WARN	-
UPF	0007	upf.cpp:169	WARN	-
UPF	0008	upf.cpp:189	WARN	-
UPF	0009	upf.cpp:209	WARN	-
UPF	0010	upf.cpp:228	WARN	-
UPF	0011	upf.cpp:247	WARN	-
UPF	0012	upf.cpp:271	WARN	-
UPF	0013	upf.cpp:282	WARN	-
UPF	0014	upf.cpp:327	WARN	-
UPF	0015	upf.cpp:338	WARN	-
UPF	0016	upf.cpp:358	WARN	-
UPF	0017	upf.cpp:428	WARN	-
UPF	0018	upf.cpp:447	WARN	-
UPF	0019	upf.cpp:472	ERROR	-
UPF	0020	upf.cpp:491	WARN	-
UPF	0021	upf.cpp:501	WARN	-
UPF	0022	upf.cpp:509	WARN	-
UPF	0023	upf.cpp:528	WARN	-
UPF	0024	upf.cpp:632	WARN	-
UPF	0025	upf.cpp:656	WARN	-
UPF	0026	upf.cpp:692	WARN	-
UPF	0027	upf.cpp:814	WARN	-

continu

Table 3 – continued from previous page

Tool	Code	Filename:Line Number	Type	Information
UPF	0028	upf.cpp:864	WARN	-
UPF	0029	upf.cpp:1261	ERROR	-
UPF	0030	upf.cpp:964	WARN	-
UPF	0031	upf.cpp:992	WARN	-
UPF	0032	upf.cpp:1377	WARN	-
UPF	0033	upf.tcl:634	ERROR	-
UPF	0034	upf.tcl:640	ERROR	-
UPF	0035	upf.cpp:1187	WARN	-
UPF	0036	upf.tcl:304	ERROR	-
UPF	0037	upf.tcl:312	ERROR	-
UPF	0038	upf.cpp:1385	WARN	-
UPF	0039	upf.cpp:1471	WARN	-
UPF	0040	upf.tcl:368	ERROR	-
UPF	0041	upf.cpp:1492	WARN	-
UPF	0042	upf.cpp:1513	WARN	-
UPF	0043	upf.i:146	ERROR	-
UPF	0044	upf.cpp:1394	WARN	-
UPF	0053	upf.cpp:1081	WARN	-
UPF	0054	upf.cpp:1143	WARN	-
UPF	0055	upf.cpp:1163	WARN	-
UPF	0056	upf.cpp:1222	WARN	-
UPF	0057	upf.tcl:507	WARN	-
UPF	0059	upf.cpp:1533	WARN	-
UPF	0060	upf.cpp:1556	WARN	-
UPF	0061	upf.cpp:147	WARN	-
UPF	0071	upf.cpp:807	WARN	-
UPF	0072	writer.cpp:46	ERROR	-
UTL	0001	CFileUtils.cpp:8	ERROR	-
UTL	0002	CFileUtils.cpp:25	ERROR	-
UTL	0003	CFileUtils.cpp:35	ERROR	-
UTL	0004	CFileUtils.cpp:56	ERROR	-
UTL	0005	ScopedTemporaryFile.cpp:12	ERROR	-
UTL	0006	ScopedTemporaryFile.cpp:15	INFO	-
UTL	0007	ScopedTemporaryFile.cpp:19	ERROR	-
UTL	0008	ScopedTemporaryFile.cpp:27	WARN	-
UTL	0009	ScopedTemporaryFile.cpp:31	WARN	-
UTL	0100	Utl.tcl:127	ERROR	-

5.3 Getting Involved

Thank you for taking the time to read this document and to contribute. The OpenROAD project will not reach all of its objectives without help!

Possible ways to contribute to the OpenROAD application:

- Tool improvements
- New tools
- Improvements to documentation, including this document

- Star our project and repos so we can see the number of people who are interested

5.3.1 Licensing Contributions

As much as possible, all contributions should be licensed using the BSD3 license. You can propose another license if you must, but contributions made with BSD3 fit best with the spirit of OpenROAD's permissive open-source philosophy. We do have exceptions in the project, but over time we hope that all contributions will be BSD3, or some other permissive license such as MIT or Apache2.0.

5.3.2 Contributing Scripts and Code

We follow the Google C++ style guide. If you find code in our project that does *not* follow this guide, then within each file that you edit, follow the style in that file.

Please pay careful attention to the *tool checklist* for all code. If you want to add or improve functionality in OpenROAD, please start with the top-level app repo. You can see in the `src` directory that submodules exist pointing to tested versions of the other relevant repos in the project. Please look at the tool workflow in the developer guide *document* to work with the app and its submodule repos in an efficient way.

Please run clang-format on all the C++ source files that you change, before committing. In the root directory of the OpenROAD repository there is the file `.clang-format` that defines all coding formatting rules.

Please pay attention to the test directory and be sure to add tests for any code changes that you make, using open-source PDK and design information. We provide the `nangate45` PDK in the OpenROAD-flow-scripts repo to help with this. Pull requests with code changes are unlikely to be accepted without accompanying test cases. There are many examples tests. Each repo has a test directory as well with tests you should run and add to if you modify something in one of the submodules.

For changes that claim to improve QoR or PPA, please run many tests and ensure that the improvement is not design-specific. There are designs in the OpenROAD-flow-scripts repo which can be used unless the improvement is technology-specific.

Do not add runtime or build dependencies without serious thought. For a project like OpenROAD with many application subcomponents, the software architecture can quickly get out of control. Changes with lots of new dependencies which are not necessary are less likely to be integrated.

If you want to add Tcl code to define a new tool command, look at pdngen as an example of how to do so. Take a look at the CMake file which automatically sources the Tcl code and the Tcl file itself.

To accept contributions, we require each commit to be made with a DCO (Developer Certificate of Origin) attached. When you commit you add the `-s` flag to your commit. For example:

```
git commit -s -m "test dco with -s"
```

This will append a statement to your commit comment that attests to the DCO. GitHub has built in the `-s` option to its command line since use of this is so pervasive. The promise is very basic, certifying that you know that you have the right to commit the code. Please read the full statement here.

5.3.3 Questions

Please refer to our *FAQs*.

5.3.4 Developer Guide

Tool Philosophy

OpenROAD is a tool to build a chip from synthesizable RTL (Verilog) to completed physical layout (manufacturable, tapeout-clean GDSII).

The unifying principle behind the design of OpenROAD is for all of the tools to reside in one tool, with one process, and one database. All tools in the flow should use Tcl commands exclusively to control them instead of external "configuration files". File-based communication between tools and forking processes is strongly discouraged. This architecture streamlines the construction of a flexible tool flow and minimizes the overhead of invoking each tool in the flow.

Tool File Organization

Every tool follows the following file structure, grouping sources, tests and headers together.

- `src/` This folder contains the source files for individual tools.

`src`	Purpose
`CMakeLists.txt`	`add_subdirectory` for each tool
`tool/src`	sources and private headers
`tool/src/CMakeLists.txt`	tool specific CMake file
`tool/include/tool`	exported headers
`tool/test`	tool tests
`tool/regression`	tool unit tests

- OpenROAD repository: This folder contains the top-level files for overall compilation. OpenROAD uses swig that acts as a wrapper for C/C++ programs to be callable in higher-level languages, such as Python and Tcl.

`OpenROAD`	Purpose
`CMakeLists.txt`	top-level CMake file
`src/Main.cc`	main file
`src/OpenROAD.cc`	OpenROAD class functions
`src/OpenROAD.i`	top-level swig, includes, tool swig files
`src/OpenROAD.tcl`	basic read/write lef/def/db commands
`include/ord/OpenROAD.hh`	OpenROAD top-level class, has instances of tools

Some tools such as OpenSTA are submodules, which are simply subdirectories in `src/` that are pointers to the git submodule. They are intentionally not segregated into a separate module.

The use of submodules for new code integrated into OpenROAD is strongly discouraged. Submodules make changes to the underlying infrastructure (e.g., OpenSTA) difficult to propagate across the dependent submodule repositories.

Where external/third-party code that a tool depends on should be placed depends on the nature of the dependency.

- Libraries - code packaged as a linkable library. Examples are `tcl`, `boost`, `zlib`, `eigen`, `lemon`, `spdlog`.

These should be installed in the build environment and linked by OpenROAD. Document these dependencies in the top-level `README.md` file. The `Dockerfile` should be updated to illustrate where to find the library and how to install it. Adding libraries to the build environment requires coordination with system administrators, so that continuous integration hosts ensure that environments include the dependency. Advance notification should also be given to the development team so that their private build environments can be updated.

Each tool CMake file builds a library that is linked by the OpenROAD application. The tools should not define a `main()` function. If the tool is Tcl only and has no C++ code, it does not need to have a CMake file. Tool CMake files should **not** include the following:

- `cmake_minimum_required`
- `GCC_COVERAGE_COMPILE_FLAGS`
- `GCC_COVERAGE_LINK_FLAGS`
- `CMAKE_CXX_FLAGS`
- `CMAKE_EXE_LINKER_FLAGS`

None of the tools have commands to read or write LEF, DEF, Verilog or database files. For consistency, these functions are all provided by the OpenROAD framework.

Tools should package all of their state in a single class. An instance of each tool class resides in the top-level OpenROAD object. This allows multiple tools to exist at the same time. If any tool keeps state in global variables (even static), then only one tool can exist at a time. Many of the tools being integrated were not built with this goal in mind and will only work on one design at a time.

Each tool should use a unique namespace for all of its code. The same namespace should be used for Tcl functions, including those defined by a swig interface file. Internal Tcl commands stay inside the namespace, and user visible Tcl commands should be defined in the global namespace. User commands should be simple Tcl commands such as `global_placement` that do not create tool instances that must be based to the commands. Defining Tcl commands for a tool class is fine for internal commands, but not for user visible commands. Commands have an implicit argument of the current OpenROAD class object. Functions to get individual tools from the OpenROAD object can be defined.

Initialization (C++ tools only)

The OpenROAD class has pointers to each tool, with functions to get each tool. Each tool has (at a minimum) a function to make an instance of the tool class, an initialization function that is called after all of the tools have been made, and a function to delete the tool. This small header does **not** include the class definition for the tool so that the OpenROAD framework does not have to know anything about the tool internals or include a gigantic header file.

`MakeTool.hh` defines the following:

```
Tool *makeTool();
void initTool(OpenRoad *openroad);
void deleteTool(Tool *tool);
```

The `OpenRoad::init()` function calls all of the `makeTool` functions and then all of the `initTool()` functions. The `init` functions are called from the bottom of the tool dependencies. Each `init` function grabs the state it needs out of the `OpenRoad` instance.

Commands

Tools should provide Tcl commands to control them. Tcl object based tool interfaces are not user-friendly. Define Tcl procedures that take keyword arguments that reference the `OpenRoad` object to get tool state. OpenSTA has Tcl utilities to parse keyword arguments (`sta::parse_keyword_args`). See `OpenSTA/tcl/*.tcl` for examples. Use swig to define internal functions to C++ functionality.

Tcl files can be included by encoding them in CMake into a string that is evaluated at run time (See `Resizer::init()`).

Note: Please refer to the top-level Tcl formatting *guide*. Our top-level Tcl files, in particular, have to be formatted in this specific manner because of the automatic parsing used to convert the READMEs into manpages.

Errors

Tools should report errors to the user using the `ord::error` function defined in `include/openroad/Error.hh`. `ord::error` throws `ord::Exception`. The variables `ord::exit_on_error` and `ord::file_continue_on_error` control how the error is handled. If `ord::exit_on_error` is `true` then OpenROAD reports the error and exits. If the error is encountered while reading a file with the `source` or `read_sdc` commands and `ord::file_continue_on_error` is `false` then no other commands are read from the file. The default value is `false` for both variables.

Test

Each "tool" has a `/test` directory containing a script named `regression` to run "unit" tests. With no arguments it should run default unit tests.

No database files should be in tests. Read LEF/DEF/Verilog to make a database.

The regression script should not depend on the current working directory. It should be able to be run from any directory. Use filenames relative to the script name rather the current working directory.

Regression scripts should print a concise summary of test failures. The regression script should return an exit code of 0 if there are no errors and 1 if there are errors. The script should **not** print thousands of lines of internal tool information.

Regression scripts should pass the `-no_init` option to `openroad` so that a user's `init` file is not sourced before the tests runs.

Regression scripts should add output files or directories to `.gitignore` so that running does not leave the source repository "dirty".

The Nangate45 open-source library data used by many tests is in `test/Nangate45`. Use the following command to add a link in the tool command:

```
cd src/<tool>/test
ln -s ../../../test/Nangate45
```

After the link is installed, the test script can read the Liberty file with the command shown below.

```
read_liberty Nangate45/Nangate45_typ.lib
```

Building

Instructions for building are available *here*.

Example of Adding a Tool to OpenROAD

The patch file "AddTool.patch" illustrates how to add a tool to OpenROAD. Use the following commands to add a sample tool:

```
# first, update existing config files
patch -p1 < docs/misc/AddTool.patch

# next, create the additional source files of the tool using this command
patch -p1 < docs/misc/AddToolFiles.patch

# finally, create the regression tests as follows
cd src/tool/test
ln -s ../../../test/regression.tcl regression.tcl
```

This adds a directory `OpenRoad/src/tool` that illustrates a tool named "Tool" that uses the file structure described above and defines a command to run the tool with keyword and flag arguments as illustrated below:

```
> toolize foo
Helping 23/6
Gotta positional_argument1 foo
Gotta param1 0.000000
Gotta flag1 false

> toolize -flag1 -key1 2.0 bar
Helping 23/6
Gotta positional_argument2 bar
Gotta param1 2.000000
Gotta flag1 true

> help toolize
toolize [-key1 key1] [-flag1] positional_argument1
```

Documentation

Tool commands should be documented in the top-level OpenROAD `README.md` file. Detailed documentation should be the `tool/README.md` file.

Note: Please refer to the README formatting *guide*. Our top-level READMEs, in particular, have to be formatted in this specific manner because of the automatic parsing used to convert the READMEs into manpages.

Tool Flow Namespace

Tool namespaces are usually three-lettered lowercase letters.

- Verilog to DB (dbSTA)
- OpenDB: Open Database (*odb*)
- TritonPart: constraints-driven paritioner (*par*)
- Floorplan Initialization (*ifp*)
- ICeWall chip-level connections (*pad*)
- I/O Placement (*ppl*)
- PDN Generation (*pdn*)
- Tapcell and Welltie Insertion (*tap*)
- Triton Macro Placer (*mpl*)
- Hierarchical Automatic Macro Placer (*mpl2*)
- RePlAce Global Placer (*gpl*)
- Gate resizing and buffering (*rsz*)
- Detailed placement (*dpl*)
- Clock tree synthesis (*cts*)
- FastRoute Global routing (*grt*)
- Antenna check and diode insertion (*ant*)
- TritonRoute Detailed routing (*drt*)
- Metal fill insertion (*fin*)
- Design for Test (dft)
- OpenRCX Parasitic Extraction (*rcx*)
- OpenSTA timing/power analyzer (sta
- Graphical User Interface (*gui*)
- Static IR analyzer (*psm*)

Tool Checklist

Tools should make every attempt to minimize external dependencies. Linking libraries other than those currently in use complicates the builds and sacrifices the portability of OpenROAD. OpenROAD should be portable to many different compiler/operating system versions and dependencies make this vastly more complicated.

1. OpenROAD submodules reference tool `openroad` branch head. No git `develop`, `openroad_app`, or `openroad_build` branches.
2. Submodules used by more than one tool belong in `src/`, not duplicated in each tool repo.
3. `CMakeLists.txt` does not use add_compile_options include_directories link_directories link_libraries. Use target_ versions instead. See tips here.
4. `CMakeLists.txt` does not use glob. Use explicit lists of source files and headers instead.

5. CMakeLists.txt does not define CFLAGS CMAKE_CXX_FLAGS CMAKE_CXX_FLAGS_DEBUG CMAKE_CXX_FLAGS_RELEASE. Let the top level and defaults control these.
6. No main.cpp or main procedure.
7. No compiler warnings for GCC or Clang with optimization enabled.
8. Does not call flute::readLUT (called once by openroad).
9. Tcl command(s) documented in top level README.md in flow order.
10. Command line tool documentation in tool README.
11. Conforms to Tcl command naming standards (no camel case).
12. Does not read configuration files. Use command arguments or support commands.
13. .clang-format at tool root directory to aid foreign programmers.
14. No jenkins/, Jenkinsfile, Dockerfile in tool directory.
15. regression script named test/regression with no arguments that runs tests. Not tests/regression-tcl.sh, not test/run_tests.py etc.
16. regression script should run independent of current directory. For example, ../test/regression should work.
17. regression should only print test results or summary, not belch 1000s of lines of output.
18. Test scripts use OpenROAD tcl commands (not itcl, not internal accessors).
19. regression script should only write files in a directory that is in the tool's .gitignore so the hierarchy does not have modified files in it as a result or running the regressions.
20. Regressions report no memory errors with valgrind (stretch goal).
21. Regressions report no memory leaks with valgrind (difficult).
22. Ensure the top-level README and Tcl format are compliant.

Code Linting and Formatting

OpenROAD uses both clang-tidy and clang-format to perform automatic linting and formatting whenever a pull request is submitted. To run these locally, please first setup Clang Tooling using this guide. Thereafter, you may run these commands:

```
cmake . -B build  # generate build files
# typically only run these commands on files you changed.
clang-tidy -p ./build source_file.cpp
clang-format -i -style=file:.clang-format source_file.cpp
```

Guidelines

1. Internally, the code should use int for all database units and int64_t for all area calculations. Refer to this link for a more detailed writeup on the reasons why this approach is preferred. The only place that the database distance units should appear in any program should be in the user interface, as microns are easier for humans than DBUs.

5.3.5 Coding Practices

List of coding practices.

Note: This is a compilation of many idioms in OpenROAD code that are considered undesirable.

C++

Practice #1

Don't comment out code, instead remove it. `git` provides a complete history of the code if you want to look backwards. Huge chunks of commented-out code make it difficult to read.

Practice #2

Don't use prefixes on function names or variables. That's what namespaces are for.

```
namespace fr {
  class frConstraint
  class frLef58CutClassConstraint
  class frShortConstraint
  class frNonSufficientMetalConstraint
  class frOffGridConstraint
  class frMinEnclosedAreaConstraint
  class frMinStepConstraint
  class frMinimumcutConstraint
  class frAreaConstraint
  class frMinWidthConstraint
  class frLef58SpacingEndOfLineWithinEndToEndConstraint
  class frLef58SpacingEndOfLineWithinParallelEdgeConstraint
  class frLef58SpacingEndOfLineWithinMaxMinLengthConstraint
  class frLef58SpacingEndOfLineWithinConstraint
  class frLef58SpacingEndOfLineConstraint
}
```

Practice #3

Namespaces should be all lower case and short. This is an example of a poor choice: `namespace TritonCTS`

Practice #4

Don't use `extern` on function definitions. It is pointless in a world with prototypes.

```
namespace fr {
  extern frCoord getGCELLGRIDX();
  extern frCoord
  getGCELLGRIDY();
  extern frCoord getGCELLOFFSETX();
  extern frCoord
  getGCELLOFFSETY();
}
```

Practice #5

Don't use prefixes on file names. That's what directories are for.

```
frDRC.h frDRC_init.cpp frDRC_main.cpp frDRC_setup.cpp frDRC_util.cpp
```

Practice #6

Don't name variables `theThingy`, `curThingy` or `myThingy`. It is just distracting extraneous verbiage. Just use `thingy`.

```
float currXSize;
float currYSize;
float currArea;
float currWS;
float currWL;
float currWLnoWts;
```

Practice #7

Do not use global variables. All state should be inside of classes. Global variables make multi-threading next to impossible and preclude having multiple copies of a tool running in the same process. The only global variable in `openroad` should be the singleton that Tcl commands reference.

```
extern std::string DEF_FILE;
extern std::string GUIDE_FILE;
extern std::string OUTGUIDE_FILE;
extern std::string LEF_FILE;
extern std::string OUTTA_FILE;
extern std::string OUT_FILE;
extern std::string DBPROCESSNODE;
extern std::string OUT_MAZE_FILE;
extern std::string DRC_RPT_FILE;
extern int MAX_THREADS ;
extern int VERBOSE ;
extern int BOTTOM_ROUTING_LAYER;
extern bool ALLOW_PIN_AS_FEEDTHROUGH;
```

(continues on next page)

(continued from previous page)

```
extern bool USENONPREFTRACKS;
extern bool USEMINSPACING_OBS;
extern bool RESERVE_VIA_ACCESS;
extern bool ENABLE_BOUNDARY_MAR_FIX;
```

Practice #8

Do not use strings (names) to refer to database or sta objects except in user interface code. DEF, SDC, and Verilog all use different names for netlist instances and nets, so the names will not always match.

Practice #9

Do not use continue. Wrap the body in an if instead.

```
// instead of
for(dbInst* inst : block->getInsts() ) {
  // Skip for standard cells
  if (inst->getBBox()->getDY() <= cellHeight) { continue; }
  // code
}
// use
for(dbInst* inst : block->getInsts() ){
  // Skip for standard cells
  if (inst->getBBox()->getDY() > cellHeight) {
    // code
  }
}
```

Practice #10

Don't put magic numbers in the code. Use a variable with a name that captures the intent. Document the units if they exist.

```
referenceHpwl_= 446000000;
coeffV = 1.36;
coeffV = 1.2;
double nearest_dist = 99999999999;
if (dist < rowHeight * 2) {}
for(int i = 9; i > -1; i--) {}
if(design_util > 0.6 || num_fixed_nodes > 0) div = 1;
avail_region_area += (theRect->xUR - theRect->xLL - (int)theRect->xUR % 200 + (int)t ␣
↪heRect->xLL % 200 - 200) * (theRect->yUR - theRect->yLL - (int)theRect->yUR % 2000 +␣
↪(int)theRect->yLL % 2000 - 2000);
```

Practice #11

Don’t copy code fragments. Write functions.

```
// 10x
int x_pos = (int)floor(theCell->x_coord / wsite + 0.5);
// 15x
int y_pos = (int)floor(y_coord / rowHeight + 0.5);

// This
nets[newnetID]->netIDorg = netID;
nets[newnetID]->numPins = numPins;
nets[newnetID]->deg = pinInd;
nets[newnetID]->pinX = (short *)malloc(pinInd* sizeof(short));
nets[newnetID]->pinY = (short *)malloc(pinInd* sizeof(short));
nets[newnetID]->pinL = (short *)malloc(pinInd* sizeof(short));
nets[newnetID]->alpha = alpha;

// Should factor out the array lookup.
Net *net = nets[newnetID];
net->netIDorg = netID;
net->numPins = numPins;
net->deg = pinInd;
net->pinX = (short*)malloc(pinInd* sizeof(short));
net->pinY = (short *)malloc(pinInd* sizeof(short));
net->pinL = (short *)malloc(pinInd* sizeof(short));
net->alpha = alpha;

// Same here:
if (grid[j][k].group != UINT_MAX) {
  if (grid[j][k].isValid) {
    if (groups[grid[j][k].group].name == theGroup->name)
      area += wsite * rowHeight;
  }
}
```

Practice #12

Don’t use logical operators to test for null pointers.

```
if (!net) {
  // code
}

// should be
if (net != nullptr) {
  // code
}
```

Practice #13

Don’t use `malloc`. Use `new`. We are writing C++, not C.

Practice #14

Don’t use C style arrays. There is no bounds checks for them so they invite subtle memory errors to unwitting programmers who fail to use `valgrind`. Use `std::vector` or `std::array`.

Practice #15

Break long functions into smaller ones, preferably that fit on one screen.

Practice #16

Don’t reinvent functions like `round`, `floor`, `abs`, `min`, `max`. Use the std versions.

```
int size_x = (int)floor(theCell->width / wsite + 0.5);
```

Practice #17

Don’t use C’s stdlib.h `abs`, `fabs` or `fabsf`. They fail miserably if the wrong arg type is passed to them. Use `std::abs`.

Practice #18

Fold code common to multiple loops into the same loop. Each of these functions loops over every instance like this:

```
legal &= row_check(log);
legal &= site_check(log);
for(int i = 0; i < cells.size(); i++) {
  cell* theCell = &cells[i];
  legal &= power_line_check(log);
  legal &= edge_check(log);
  legal &= placed_check(log);
  legal &= overlap_check(log);
}
// with this loop
for(int i = 0; i < cells.size(); i++) {
  cell* theCell = &cells[i];
}
```

Instead make one pass over the instances doing each check.

Practice #19

Don't use == `true`, or == `false`. Boolean expressions already have a value of true or false.

```
if(found.first == true) {
  // code
}
// is simply
if(found.first) {
  // code
}
// and
if(found.first == false) {
  // code
}
// is simply
if(!found.first) {
 // code
}
```

Practice #20

Don't nest if statements. Use && on the clauses instead.

```
if(grid[j][k].group != UINT_MAX)
  if(grid[j][k].isValid == true)
    if(groups[grid[j][k].group].name == theGroup->name)
```

is simply

```
if(grid[j][k].group != UINT_MAX
   && grid[j][k].isValid
   && groups[grid[j][k].group].name == theGroup->name)
```

Practice #21

Don't call return at the end of a function that does not return a value.

Practice #22

Don't use <> to include anything but system headers. Your project's headers should never be in <>.

1. GCC Include Syntax
2. StackOverflow discussion on "filename" vs <filename>

These are all wrong:

```
#include <odb/db.h>
#include <sta/liberty/Liberty.hh>
#include <odb/db.h>
```

(continues on next page)

(continued from previous page)

```
#include <odb/dbTypes.h>
#include <odb/defin.h>
#include <odb/defout.h>
#include <odb/lefin.h>
```

Practice #23

Don't make "include the kitchen sink" headers and include them in every source file. This is convenient but slows the builds down for everyone. Make each source file include just the headers it actually needs.

```
// Types.hpp
#include <sta/liberty/Liberty.hh>
#include <odb/db.h>
#include <odb/dbTypes.h>
// It should be obvious that every source file is not reading def.
#include <odb/defin.h>
// or writing it.
#include <odb/defout.h>
#include <odb/lefin.h>
#include "db_sta/dbNetwork.hh"
#include "db_sta/dbSta.hh"
```

Note this example also incorrectly uses `<>'s` around OpenROAD headers.

Header files should only include files to support the header. Include files necessary for code in the code file, not the header.

In the example below NONE of the system files listed are necessary for the header file.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <limits.h>

unsigned num_nets = 1000;
unsigned num_terminals = 64;
unsigned verbose = 0;
float alpha1 = 1;
float alpha2 = 0.45;
float alpha3 = 0;
float alpha4 = 0;
float margin = 1.1;
unsigned seed = 0;
unsigned root_idx = 0;
unsigned dist = 2;
float beta = 1.4;
bool runOneNet = false;
unsigned net_num = 0;
```

Practice #24

Use class declarations if you are only referring to objects by pointer instead of including their complete class definition. This can vastly reduce the code the compiler has to process.

```
class Network;
// instead of
#include "Network.hh"
```

Practice #25

Use pragma once instead of `#define` to protect headers from being read more than once. The #define symbol has to be unique, which is difficult to guarantee.

```
// Instead of:
#ifndef __MACRO_PLACER_HASH_UTIL__
#define __MACRO_PLACER_HASH_UTIL__
#endif
// use
#pragma once
```

Practice #26

Don't put `using namespace` inside a function.

Practice #27

Don't nest namespaces.

Practice #28

Avoid `using namespace`. It increases the likelihood of conflicts and doesn't explicity declare what in the namespace is being used. Use `using namespace::symbol;` instead. And especially do not use `using namespace std`.

The following is especially confused because it is trying to "use" the symbols in code that are already in the MacroPlace namespace.

```
using namespace MacroPlace;

namespace MacroPlace { }
```

Practice #29

Use nullptr instead of NULL. This is the C++ approved version of the ancient C #define.

Practice #30

Use range iteration. C++ iterators are ugly and verbose.

```
// Instead of
odb::dbSet::iterator nIter;
for (nIter = nets.begin(); nIter != nets.end(); ++nIter) {
  odb::dbNet* currNet = *nIter;
  // code
}
// use
for (odb::dbNet* currNet : nets) {
  // code
}
```

Practice #31

Don't use end of line comments unless they are very short.

```
for (int x = firstTile._x; x <= lastTile._x; x++) { // Setting capacities of edges␣
↪completely inside the adjust region according the percentage of reduction
  // code
}
```

Practice #32

Don't std::pow for powers of 2 or for decimal constants.

```
// This
double newCapPerSqr = (_options->getCapPerSqr() * std::pow(10.0, -12));
// Should be
double newCapPerSqr = _options->getCapPerSqr() * 1E-12;

// This
unsigned numberOfTopologies = std::pow(2, numberOfNodes);
// Should be
unsigned numberOfTopologies = 1 << numberOfNodes;
```

Git

Practice #33

Don’t put /’s in .gitignore directory names. test/

Practice #34

Don’t put file names in .gitignore ignored directories. test/results test/results/diffs

Practice #35

Don’t list compile artifacts in .gitignore. They all end up in the build directory so each file type does not have to appear in .gitignore.

All of the following are to be avoided:

Compiled Object files

*.slo *.lo *.o *.obj

Precompiled Headers

*.gch *.pch

Compiled Dynamic libraries

*.so *.dylib *.dll

Fortran module files

*.mod *.smod

Compiled Static libraries

*.lai *.la *.a *.lib

CMake

Practice #35

Don’t change compile flags in cmake files. These are set at the top level and should not be overridden.

```
set(CMAKE_CXX_FLAGS "-O3")
set(CMAKE_CXX_FLAGS_DEBUG "-g -ggdb")
set(CMAKE_CXX_FLAGS_RELEASE "-O3")
```

Practice #36

Don’t put /’s in CMake directory names. CMake knows they are directories.

```
target_include_directories( ABKCommon PUBLIC ${ABKCOMMON_HOME} src/ )
```

Practice #37

Don’t use `glob`. Explictly list the files in a group.

```
# Instead of
file(GLOB_RECURSE SRC_FILES ${CMAKE_CURRENT_SOURCE_DIR}/src/*.cpp)
# should be
list(REMOVE_ITEM SRC_FILES ${CMAKE_CURRENT_SOURCE_DIR}/src/Main.cpp)
list(REMOVE_ITEM SRC_FILES ${CMAKE_CURRENT_SOURCE_DIR}/src/Parameters.h)
list(REMOVE_ITEM SRC_FILES ${CMAKE_CURRENT_SOURCE_DIR}/src/Parameters.cpp)
```

5.3.6 Using the Logging Infrastructure

OpenROAD uses spdlog as part of logging infrastructure in order to ensure a clear, consistent messaging and complete messaging interface. A wrapper formats the prefix in the recommended messaging style and limit. A message format is as follows:

```
<tool id>-<message id>  <Message body>.
```

For example,

```
[INFO ODB-0127] Reading DEF file: ./results/asap7/aes/base/4_cts.def
```

All output from OpenROAD tools should be directed through the logging API to ensure that redirection, file logging and execution control flow are handled consistently. This also includes messages from any third-party tool. Use the ‘ord’ message ID for third-party tools.

The logging infrastructure also supports generating a JSON file containing design metrics (e.g., area or slack). This output is directed to a user-specified file. The OpenROAD application has a `-metrics` command line argument to specify the file.

Handling Messages

OpenROAD supports multiple levels of severity for message outputs: critical, error, warning, information and debug. These are supported by automatic calls to the logger which will then prefix the appropriate severity type to the message.

C++20 Requirements

In C++20 the logger messages are checked during compile time which introduces restrictions around rutime format strings. See docs.

OpenROAD uses spdlog which uses fmt_lib under the hood. Below is an example of what is no longer allowed.

In order to make use of runtime format strings, we have introduced a FMT_RUNTIME macro in Logger.h. You should use this macro any time you pass a dynamic string as the format string

```
logger_->info("{} {}", a, b); // OK

void blah(std::string template& a) {
  logger_->info(a, c); // Illegal
  logger_->info(FMT_RUNTIME(a), c); // Ok
}
```

Messaging Guidelines

In addition to the proper use of message types, follow the guidelines below to compose messages for clarity, consistency and other guidelines:

Grammar

Start with a capital letter and end with a period, besides well-known exceptions. Use capital letters for file formats and tool proper names, e.g., LEF, DEF, SPICE, FLUTE.

After the first word's capitalization, do not use capital letters (aside from obvious exceptions, such as RSMT, hCut, etc.).

Do not use exclamations. Severity must be communicated by message severity and clear implied or explicit action.

Avoid long, verbose messages. Use commas to list and separate clauses in messages.

Spellcheck all messages using American English spellings.

Use ellipsis ... only to indicate a pause, as when some tool is running or being initialized.

Abbreviations and Shortcuts

Use single-word versions when well-accepted / well-understood by users and developers. Examples: stdcell, cutline, wirelength, flipchip, padring, bondpad, wirebond, libcell, viarule.

Do not abbreviate or truncate English words; expand for the sake of clarity.

```
Incorrect:   Num, #;  Tot.
```

```
Correct:     Number;  Total
```

Use acceptable, well-understood abbreviations for brevity. Examples: db, tech, lib, inst, term, params, etc.

Avoid contractions of action words:

```
Incorrect:   Can't, Can not;  Don't
```

```
Correct:     Cannot;  Do not
```

Actionability

Messages should communicate a clear, implied or explicit action that is necessary for flow continuation or improved quality of results.

```
Example:
A value for core_area must be specified in the footprint specification, or in the␣
↪environment variable CORE_AREA.
```

Clarity

Messages must be clear and complete, so as to communicate necessary and sufficient information and actions. Elaborate specific variables, options, and/or parameters to avoid any ambiguity.

```
Example:
Unrecognized argument $arg, should be one of -pitch, -bump_pin_name, -spacing_to_edge, -
↪cell_name, -bumps_per_tile, -rdl_layer, -rdl_width, -rdl_spacing.
```

Specify objects clearly in the local context:

```
Example:
cutWithin is smaller than cutSpacing for ADJACENTCUTS on layer {}. Please check your␣
↪rule definition.

Incomplete:
Warning: {} does not have viaDef aligned with layer.
```

Make any assumptions or use of default values explicit:

```
Example:
No net slacks found.
Timing-driven mode disabled.

Incomplete, missing default:
Utilization exceeds 100%.
```

Use simple language, and avoid repetitions:

```
Example:
Missing orientation for cell $cell_ref.

Incorrect:
No orientation available for orientation of $cell_ref.
```

Message Types

OpenROAD supports the following levels of severity through the logger: report, debug, information, warning, error and critical.

Report

Report messages are output by the tool in the form of a report to the user. Examples include timing paths or power analysis results.

Example report message:

```
Path startpoint: $startpoint
```

Debug

Debug messages are only of use to tool developers and not to end users. These messages are not shown unless explicitly enabled.

Information

Information messages may be used to report metrics, quality of results, or program status to the user. Any message which indicates runtime problems, such as potential faulty input or other internal program issues, should be issued at a higher status level.

Example information messages:

```
Number of input ports: 47

Running optimization iteration 2

Current cell site utilization: 57.1567%
```

Warning

Warnings should be used to indicate atypical runtime conditions that may affect quality, but not correctness, of the output. Any conditions that affect correctness should be issued at a higher status level.

Example warning messages:

```
Core area utilization is greater than 90%. The generated cell placement may not be␣
↪routable.

14 outputs are not constrained for max capacitance.

Pin 'A[0]' on instance 'mem01' does not contain antenna information and will not be␣
↪checked for antenna violations.
```

Error

Error messages should be used to indicate correctness problems. Problems with command arguments are a good example of where error messages are appropriate. Errors exit the current command by throwing an exception that is converted to an error in Tcl. Errors that occur while reading a command file stop execution of the script commands.

Example error messages:

```
Invalid selection: net 'test0' does not exist in the design.

Cell placement cannot be run before floorplanning.

Argument 'max_routing_layer' expects an integer value from 1 to 10.
```

Critical

Critical messages should be used to indicate correctness problems that the program is not able to work around or ignore, and that require immediate exiting of the program (abort).

Example critical messages:

```
Database 'chip' has been corrupted and is not recoverable.

Unable to allocate heap memory for array 'vertexIndices'. The required memory size may exceed host machine limits.

Assertion failed: 'nodeVisited == false' on line 122 of example.cpp. Please file a Github issue and attach a testcase.
```

Coding

Each status message requires:

- The three letter tool ID
- The message ID
- The message string
- Optionally, additional arguments to fill in placeholders in the message string

Reporting is simply printing and does not require a tool or message ID. The tool ID comes from a fixed enumeration of all the tools in the system. This enumeration is in `Logger.h`. New abbreviations should be added after discussion with the OpenROAD system architects. The abbreviation matches the C++ namespace for the tool.

Message IDs are integers. They are expected to be unique for each tool. This has the benefit that a message can be mapped to the source code unambiguously even if the text is not unique. Maintaining this invariant is the tool owner's responsibility. To ensure that the IDs are unique, each tool should maintain a file named 'messages.txt' in the top-level tool directory, listing the message IDs along with the format string. When code that uses a message ID is removed, the ID should be retired by removing it from 'messages.txt'. See the utility `etc/find_messages.py` to scan a tool directory and write a `messages.txt` file.

Spdlog comes with the `fmt` library which supports message formatting in a python or C++20 like style.

The message string should not include the tool ID or message ID which will automatically be prepended. A trailing newline will automatically be added, and hence messages should not end with one. Messages should be written as complete sentences and end in a period. Multi-line messages may contain embedded new lines.

Some examples:

```
logger->report("Path startpoint: {}", startpoint);

logger->error(ODB, 25, "Unable to open LEF file {}.", file_name);

logger->info(DRT, 42, "Routed {} nets in {:3.2f}s.", net_count, elapsed_time);
```

Tcl functions for reporting messages are defined in the OpenROAD swig file `OpenRoad.i`. The message is simply a Tcl string (no C++20 formatting).

```
utl::report "Path startpoint: $startpoint"

utl::error ODB 25 "Unable to open LEF file $file_name."

utl::info DRT 42 "Routed $net_count nets in [format %3.2f $elapsed_time]."
```

`utl::report` should be used instead of ‘puts’ so that all output is logged.

Calls to the Tcl functions `utl::warn` and `utl::error` with a single message argument report with tool `ID UKN` and message `ID 0000`.

Tools use `#include utl/Logger.h` that defines the logger API. The Logger instance is owned by the OpenROAD instance. Each tool should retrieve the logger instance in the tool init function called after the tool make function by the OpenROAD application.

Every tool swig file must include src/Exception.i so that errors thrown by `utl::error` are caught at the Tcl command level. Use the following swig command before `%inline`.

```
%include "../../Exception.i"
```

The logger functions are shown below.

```
Logger::report(const std::string& message,
               const Args&... args)
Logger::info(ToolId tool,
             int id,
             const std::string& message,
             const Args&... args)
Logger::warn(ToolId tool,
             int id,
             const std::string& message,
             const Args&... args)
Logger::error(ToolId tool,
              int id,
              const std::string& message,
              const Args&... args)
Logger::critical(ToolId tool,
                 int id,
                 const std::string& message,
                 const Args&... args)
```

The corresponding Tcl functions are shown below.

```
utl::report message
utl::info tool id message
```

(continues on next page)

(continued from previous page)

```
utl::warn tool id message
utl::error tool id message
utl::critical tool id message
```

Although there is a `utl::critical` function, it is really difficult to imagine any circumstances that would justify aborting execution of the application in a tcl function.

Debug Messages

Debug messages have a different programming model. As they are most often *not* issued the concern is to avoid slowing down normal execution. For this reason such messages are issued by using the debugPrint macro. This macro will avoid evaluating its arguments if they are not going to be printed. The API is:

```
debugPrint(logger, tool, group, level, message, ...);
```

The `debug()` method of the Logger class should not be called directly. No message id is used as these messages are not intended for end users. The level is printed as the message id in the output.

The argument types are as for the info/warn/error/critical messages. The one additional argument is group which is a `const char*`. Its purpose is to allow the enabling of subsets of messages within one tool.

Debug messages are enabled with the tcl command: `set_debug_level <tool> <group> <level>`

Metrics

The metrics logging uses a more restricted API since JSON only supports specific types. There are a set of overloaded methods of the form:

```
metric(ToolId tool,
       const std::string_view metric,
       <type> value)
```

where `<type>` can be `int, double, string, or bool`. This will result in the generated JSON:

```
"<tool>-<metric>" : value
```

String values will be enclosed in double-quotes automatically.

Converting to Logger

The error functions in `include/openroad/Error.hh` should no longer be included or used. Use the corresponding logger functions.

All uses of the tcl functions ord::error and ord::warn should be updated call the `utl::error/warn` with a tool ID and message ID. For compatibility these are defaulted to `UKN` and `0000` until they are updated.

Regression tests should not have any `UKN-0000` messages in their ok files. A simple grep should indicate that you still have pending calls to pre-logger error/warn functions.

The `cmake` file for the tool must also be updated to include spdlog in the link libraries so it can find the header files if they are not in the normal system directories.

Tip: At UCSD, dfm.ucsd.edu is an example of this problem; it has an ancient version of spdlog in '/usr/include/spdlog'. Use `module` to install spdlog 1.8.1 on dfm.ucsd.edu and check your build there.

```
target_link_libraries(<library_target>
  PUBLIC
    utl
)
```

Useful Information

As tool developers, we can also choose to include useful information to the end user - be it in the form on debugging tips, or solutions to fix the errors/warnings. We compile a list of such errors in this *table*. The good thing about this page is the ability to encode rich formatting using Markdown, enabling you to convey more information than what can be said from the limited messages in code.

To format the information, refer to this sample GRT information file. In addition, make sure you create the corresponding `docs/messages` folder under the `tool` folder, before creating your Markdown file with the corresponding `NUM`.

```
cd src/<tool> && mkdir -p doc/messages
cd doc/messages && touch <NUM>.md
```

OpenROAD Tool List

A full list of tool namespaces can be found *here*.

5.3.7 CI Guide

This document describes the pipelines available to the developers and code maintainers in the Jenkins server. Note that pipelines with the suffix `*-Private` are only available to code maintainers and The OpenROAD Project members as they can contain confidential information. Thus, to access Private pipelines one needs to have authorization to access confidential data and be logged in the Jenkins website.

Below there is a list of the available features. Instructions on how to navigate Jenkins to access these features are available here.

- Find your build through Jenkins website or from GitHub.
- See test status: Pass/Fail.
- Log files for each test.
- Build artifacts to reproduce failures.
- HTML reports about code coverage and metrics.

OpenROAD App

- OpenROAD-Coverage-Public
  - Description: run dynamic code coverage tool `lconv`.
  - Target: master branch.
  - Report link here.
- OpenROAD-Coverity-Public
  - Description: compile and submit builds to Coverity static code analysis tool.
  - Target: master branch.
  - Report link here.
- OpenROAD-Nightly-Public
  - Description: `openroad` unit tests, docker builds, ISPD 2018 and 2019 benchmarks for DRT and large unit tests of GPL.
  - Target: master branch.
- OpenROAD-Public
  - Description: `openroad` unit tests and docker builds.
  - Target: all branches and open PRs.
- OpenROAD-Special-Private
  - Description: for developer testing, runs ISPD 2018 and 2019 benchmarks for DRT and large unit tests of GPL.
  - Target branches: `TR_*`, `secure-TR_*`, `TR-*`, `secure-TR-*`.
- OpenROAD-Private
  - Description: `openroad` unit tests and docker builds.
  - Target: all branches. Note that PRs will be run on public side after "Ready to Sync Public" workflow.

OpenROAD Flow

- Information about OpenROAD Flow CI jobs can be found here

OpenLane

- OpenLane-MPW-CI-Public
  - Description: test projects to older MPW shuttles with newer OpenLane versions.
  - Repo link.
- OpenLane-Public
  - Description: test OpenLane with latest commit from OpenROAD.
  - Repo link.

5.3.8 Tool Name

The top-level READMEs in each tool folder (~/OpenROAD/src/<tool>/README.md) has to be formatted in this particular manner. For most part, you can copy the format and replace the contents where necessary.

Commands

Note:

- Parameters in square brackets [-param param] are optional.
- Parameters without square brackets -param2 param2 are required.

Command ABC

The command_abc command performs…

The developer arguments are…

Note for commands, you must adhere to the top-level Tcl definition for the same command.

- Arguments: cannot be preceded with dashes
- Flags/Keys: verify if it is optional or required, then insert the necessary square brackets. Also, keys have to be followed with a specifier whereas flags only require the -flag itself.

As far as possible, do place the positional arguments last.

```
command_abc
    -key1 key1
    [-flag1]
    [-flagDev]
    arg1
```

Options

Switch Name	Description
arg1	Description for arg1.
-key1	Description for key1.
-flag1	Description for flag1.

Developer Arguments

If there are some developer arguments you want to highlight to the end user not to worry about - you can park them in the same level below the main Options category.

Switch Name	Description
-flagDev	Description for flagDev.

Useful Developer Commands

If you are a developer, you might find these useful. More details can be found in the source file or the swig file.

Command Name	Description
`command_abc_debug`	Debug something.

Example scripts

Examples scripts demonstrating …

```
./test/asdfg.tcl
```

Regression tests

There are a set of regression tests in `./test`. Refer to this section for more information.

Simply run the following script:

```
./test/regression
```

Limitations

FAQs

Check out GitHub discussion about this tool.

Authors

References

License

BSD 3-Clause License. See LICENSE file.

5.3.9 Tcl Format

The most important part to take note of are: `sta::define_cmd_args` - which defines what is printed when a user types `help command` in the OR shell; as well as `sta::parse_key_args` - which defines what keys/flags the command actually parses.

Let us use `check_antennas` command for an example.

Specification

```
sta::define_cmd_args "check_antennas" { [-verbose]\
                                        [-net net]}

proc check_antennas { args } {
  sta::parse_key_args "check_antennas" args \
    keys {-report_file -net} \
    flags {-verbose -report_violating_nets}
...
}
```

Do not compile

If you add `;# checker off` behind the command's `sta::define_cmd_args {}` and `sta::parse_key_args {}` the function will not be compiled in the Manpages and included in the doctests.

```
sta::define_cmd_args "check_antennas" { [-verbose]\
                                        [-net net]} ;# checker off

proc check_antennas { args } {
  sta::parse_key_args "check_antennas" args \
    keys {-report_file -net} \
    flags {-verbose -report_violating_nets};# checker off
...
}
```

5.4 Contributor Covenant Code of Conduct

5.4.1 Our Pledge

We as members, contributors, and leaders pledge to make participation in our community a harassment-free experience for everyone, regardless of age, body size, visible or invisible disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

We pledge to act and interact in ways that contribute to an open, welcoming, diverse, inclusive, and healthy community.

5.4.2 Our Standards

Examples of behavior that contributes to a positive environment for our community include:

- Demonstrating empathy and kindness toward other people
- Being respectful of differing opinions, viewpoints, and experiences
- Giving and gracefully accepting constructive feedback
- Accepting responsibility and apologizing to those affected by our mistakes, and learning from the experience
- Focusing on what is best not just for us as individuals, but for the overall community

Examples of unacceptable behavior include:

- The use of sexualized language or imagery, and sexual attention or advances of any kind
- Trolling, insulting or derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or email address, without their explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

5.4.3 Enforcement Responsibilities

Community leaders are responsible for clarifying and enforcing our standards of acceptable behavior and will take appropriate and fair corrective action in response to any behavior that they deem inappropriate, threatening, offensive, or harmful.

Community leaders have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, and will communicate reasons for moderation decisions when appropriate.

5.4.4 Scope

This Code of Conduct applies within all community spaces, and also applies when an individual is officially representing the community in public spaces. Examples of representing our community include using an official e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event.

5.4.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported to the community leaders responsible for enforcement at complaints@openroad.tools. All complaints will be reviewed and investigated promptly and fairly.

All community leaders are obligated to respect the privacy and security of the reporter of any incident.

5.4.6 Enforcement Guidelines

Community leaders will follow these Community Impact Guidelines in determining the consequences for any action they deem in violation of this Code of Conduct:

1. Correction

Community Impact: Use of inappropriate language or other behavior deemed unprofessional or unwelcome in the community.

Consequence: A private, written warning from community leaders, providing clarity around the nature of the violation and an explanation of why the behavior was inappropriate. A public apology may be requested.

2. Warning

Community Impact: A violation through a single incident or series of actions.

Consequence: A warning with consequences for continued behavior. No interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, for a specified period of time. This includes avoiding interactions in community spaces as well as external channels like social media. Violating these terms may lead to a temporary or permanent ban.

3. Temporary Ban

Community Impact: A serious violation of community standards, including sustained inappropriate behavior.

Consequence: A temporary ban from any sort of interaction or public communication with the community for a specified period of time. No public or private interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, is allowed during this period. Violating these terms may lead to a permanent ban.

4. Permanent Ban

Community Impact: Demonstrating a pattern of violation of community standards, including sustained inappropriate behavior, harassment of an individual, or aggression toward or disparagement of classes of individuals.

Consequence: A permanent ban from any sort of public interaction within the community.

5.4.7 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 2.0, available at https://www.contributor-covenant.org/version/2/0/code_of_conduct.html.

Community Impact Guidelines were inspired by Mozilla's code of conduct enforcement ladder.

For answers to common questions about this code of conduct, see the FAQ at https://www.contributor-covenant.org/faq. Translations are available at https://www.contributor-covenant.org/translations.

5.5 FAQs

If you cannot find your question/answer here, please file a GitHub issue to the appropriate repository or start a discussion.

- Issues and bugs:
  - OpenROAD: https://github.com/The-OpenROAD-Project/OpenROAD/issues
- Discussions:
  - OpenROAD: https://github.com/The-OpenROAD-Project/OpenROAD/discussions

5.5.1 How can I contribute?

Thank you for your willingness to contribute. Please see the *Getting Involved* guide.