# UCSD TritonSizer Software User Guide

Version 0.2

Release 2018.11

**VLSI CAD Laboratory**

**UC San Diego**

http://vlsicad.ucsd.edu

# Contents

# 1. Introduction

This documentation explains how to use the UCSD gate-sizing tool, *TritonSizer*.

## Version History

| Date | Updates |
|---|---|
| 11/28/2018 | Execution command updated<br>Input files updated<br>Note for liberty parser added |
| 9/14/2018 | Initial draft |

# 2. Getting Started

TritonSizer requires Tcl package (v8.4) and a golden (signoff) timer (e.g., OpenSTA, Synopsys PrimeTime, Cadence Tempus, etc.). The TritonSizer binary is compiled with gcc v4.8/6.2.

We provide a tcl script for golden timers. This file must exist at the working directory with the name of "sizer.tbc" (a compiled Tcl binary). Users can define their own scripts to enable any golden signoff timer in "sizer.tcl".

TritonSizer takes a post-routed design as an input, and provides an optimized sizing solution as .sizes file. With this file, users can run ECO flow with a P&R implementation tool (e.g., Cadence Innovus).

## Preparations

**How to complie**

TritonSizer software is executed with the following command:

```
$ git clone --recursive https://github.com/abk-openroad/TritonSizer.git
$ cd ~/TritonSizer
$ make clean
$ make
```

**Execution command**

TritonSizer software is executed with the following command:

```
$ source load.sh
$ cd src
$ sizer -env <environment file> -f <command file> | tee log

//<environment file> is used to set environment variables and library
information. <command file> contains information for input/output files and
command options.
```

<environment file> is used to set environment variables and library information. <command file> contains information for input/output files and command options.

**Note for the Liberty parser in TritonSizer**

The Liberty parser in TritonSizer is implemented based on the Liberty parser provided by the ISPD13 Discrete Gate Sizing contest (link: http://www.ispd.cc/contests/13/ispd2013_contest.html). We have extended and improved the original Liberty parser significantly to enable support for Liberty files from various foundries including TSMC65, TSMC45, ST28, TSMC16 and ASAP7. However, as we support a limited set of keywords for parsing only the information consumable by TritonSizer, some Liberty files might not work.

Here is the list of foundry libraries that we have verified: TSMC65 TSMC45 ST28 TSMC16 ASAP7

To obtain support for another library, please provide us access to the Liberty files along with the TritonSizer log file and the error messages. Alternatively, you make code changes in ckt.cpp files and run the regression tests in the bench directory (with ASAP7 library), then issue a pull request.

Contact: Minsoo Kim, mik226@eng.ucsd.edu

# Inputs

| Input type | Description |
|---|---|
| .db/.lib | Synopsys DB (.db) and liberty (.lib) files contain timing and power characterization data for cell library. |
| .sdc | The SDC file contains timing constraint information that drives optimization decisions. |
| .spef (optional) | The SPEF file contains segment level parasitic information that is used for delay calculation during optimization. |
| .v | The Verilog netlist contains connectivity information used during timing analysis. |
| .tcf (optional) | Switching activity file to be used for total power calculation |

# Outputs

The outputs are located at <working_dir>/results .

| Output type | Description |
|---|---|
| <design>.sizes | The output size/vt list for the optimized design |
| <design>.change.sizes | The changed size/vt list for the optimized design |
| sum.rpt | The power reduction result, runtime, memory information |
| pwr.list | .csv file for power and timing trajectory;<br>format: <TNS> <WNS> <total power> <leakage> |

## Environment File

In the environment file, following information is specified. An example script is given in Appendix A.

```
-dbpath           <directory path of .db files>
-libpath          <directory path of .lib files>
-db               <.db library files>
-lib              <.lib library files>
-mindb            <.db library files for hold check>
-minlib           <.lib library files for hold check>
-envlib           <.lib files for macro cells>
-suffix_lvt       [<name suffix for lower Vt ]
-suffix_hvt       [<name suffix for higher Vt ]
-numVt            [#Vt options (1~3)]
```

**Library path and files (-dbpath, -db, -libpath, -lib)**

Synopsys DB (.db) and liberty (.lib) files are specified with this option.

**Suffix of library cell name (-suffix_lvt, -suffix_nvt, -suffix_hvt)**

Naming convention for LVT, NVT and HVT cells is specified with "-suffix_lvt", "-suffix_nvt" and "-suffix_hvt", respectively. Base on the suffix of the library cell name, TritonSizer differentiates VTs.

Example) If INVD1_LVT, INVD1_NVT, INVD1_HVT are LVT, NVT, HVT cells respectively,

```
-suffix_lvt LVT
-suffix_nvt NVT
-suffix_hvt HVT
```

NOTE: "-suffix_hvt" should be specified by default. Please give the highest VT among VT options with "-suffix_hvt". E.g., if you want to use NVT and LVT, set as follows:

```
-suffix_nvt LVT
-suffix_hvt NVT
-num vt 2
```

NOTE: TritonSizer supports up to 3 VT types.

## Command File

In the command file, following information is specified. An example script is given in Appendix B.

```
# input design
-top         <design name>                          # top-level instance
name
-v           <Verilog input file>                   # default: {design
name}.v
-spef        <SPEF input file>                # default: {design name}.spef
-sdc         <SDC file>                       # default: {design name}.sdc
-tcf         <TCF file>                       # for dynamic power
calculation
-hold                                   # check hold timer violation;
default:false
-vt_only                                      # Vt swap only
-size_only                              # upsizing/downsizing only
-slack_margin                                 # add a margin to slack
-prft_only                        # default: true; will be removed in the
next version
-iso_tns                                                                #
set a target tns (positive value (ns))
                  # example: "-iso_tns 1.0" will give a solution with -1.0ns
            total negative slack
-power_opt_wns <value (ns)>    # set a wns margin during power reduction stage
# example: "-power_opt_wns -0.1" allows power reduction until wns becomes
-0.1ns
-power_opt_gb <value (ns)> # set a guardband during power reduction stage
# example: "-power_opt_gb -0.01" allows cells with -0.01ns slack to be
downsized
-no_seq_opt # do not allow changing sequential cells
-peephole <num> # enable peephole optimization and #cells to be optimized
simultaneously
(default: 3)
-peephole_iter <num> # the threshold value of get-stuck numbers to determine
whether to enter into peephole optimization or not (should be less than kick
max iteration to enable peephole optimization)
-all_pin_chk # when performing timing correlation, check special pins of FF
(e.g., enable pin, etc.); recommended
-all_move # if specified, all sizing moves (downsizing, using higher VT) are
considered during power reduction; recommended
# socket interface with a golden timer
-ptport <port number> # port number to be used for the socket interface between
sizer and a golden timer

-fix_slew                          # default:false; recommended for timing
recovery
-fix_global                       # default:false; recommended for timing
recovery
- macros
```

```
-dont_touch_inst  <instance name>  # specifies an instance name that should not
be sized

-alpha <alpha (0~1.0)>                            # a weighting factor for power
-sf <sensitivity function number>          # sensitivity function for power
reduction
-sft <sensitivity function number for timing>    # sensitivity function for
timing recovery
-corr_ratio <(0.0~1.0)>              # specifies the frequency of timing
correlation with a golden
                                # timer; 0.01 is default (Y)
-trial_rate <(0.0~1.0)>              # specifies % of total candidates to be
tried during power
                                # reduction (X)



-useTempus                # must be enabled to use Tempus as a golden timer
-useOpenSTA               # must be enabled to use OpenSTA as a golden timer
-dont_touch_cell <cell master name> # specifies a cell master name that should
not be sized. E.g., macros, etc.
-dont_touch_inst  <instance name>  # specifies an instance name that should not
be sized
-dont_touch_cell_list <file> # specifies the file that has a list of cell
master names that should not be sized. E.g., macros, etc.
-dont_touch_list <file>  # specifies the file that has a list of instance
names that should not be sized
-dont_use_list <file>  # specifies the file that has a list of cell master
names that should not be used
-alpha <alpha (0~1.0)>                            # a weighting factor for power
-sf <sensitivity function number>          # sensitivity function for power
reduction
-sft <sensitivity function number for timing>    # sensitivity function for
timing recovery
-corr_ratio <(0.0~1.0)>              # specifies the frequency of timing
correlation with a golden
                                # timer; 0.01 is default (Y)
-trial_rate <(0.0~1.0)>              # specifies % of total candidates to be
tried during power
-trial_move_num                         # specifies the number of total
candidates to be tried during power
# control kick move
-kick_ratio <kick ratio (0~1.0)> # default = 0.05; control the number of cells
to be changed in kick move
-kick_max_iter <kick max iteration> # default = 4; the number of kick move
iterations
-slew_th <threshold ratio> # default = 0.2; control upsizing in kick move
-kick_step <number> # default = 1; the number of upsizing step
-kick_max_iter <number> # default = 4; defines the number of kick move
iterations
# GWTW control
-GWTW_max_iter <gwtw iteration> # the number of GWTW iterations (K, default =
4)
```

```
-GWTW_div <#variation> # the number of different optimization instances
(default = 4)
-GWTW_num_start <#starts> # the number of starting points at the beginning
(default = 4)
-opt_effort <number> # the number of iterations of power optimization> (N,
default = 2)
-thread <number of threads>                # the number of cpu cores to be used
for sizer;
# if you are using GWTW, it should be matched with <#variation> * <#starts>
-prft_ptnum <number of golden timers>            #should be matched to the
number of threads
```

**NOTE:** TritonSizer with “-size_only” option try sizing all VT cell. If you want to specify a certain VT type to be sized, use “-dont_touch_cell” or “-dont_touch_inst” options together. E.g., to enable NVT-only sizing, one can set “don’t touch” to all HVT cells, and apply sizing only optimization. The following Tcl script for Encounter or ETS can be used to extract and set don’t touch to all HVT cells in a design. (Note that the generated file should be appended to the cmd file.)

<Tcl script (prefix == “SEN_”)>

```
#! /usr/bin/tclsh
set outFile [open dont_touch_HVT.tcl w]
foreach_in_collection cell [get_cells *] {
    set cellName [get_property $cell full_name]
    set refName  [get_property $cell ref_name]
    if {[regexp {SEN_} $refName]} {
        puts $outFile “-dont_touch_inst $cellName”
    }
}
close $outFile
```

## Sensitivity functions

| Sensitivity function | Description for sensitivity function (leakage optimization) |
|---|---|
| SF0 | $\Delta power \cdot slack$ |
| SF1 | $\Delta power / \Delta delay$ |
| SF3 | $\Delta power \cdot slack / \#paths$ |

| | |
|---|---|
| SF4 | $\Delta power{\cdot}slack/(\Delta delay{\cdot}\#paths)$ |
| SF10 | $\Delta power{\cdot}slack\ /path\_depth$ |
| SF11 | $\Delta power{\cdot}slack/\#intrinsic\_paths$ |
| SF12 | $\Delta power{\cdot}slack/(\#\ paths + \#\ intrinsic\_paths)$ |
| SF31 | $power \cdot (min((slack_i - delay_i)\ ,\ \gamma{\cdot}(max\_transition\_slack_i)\ ))$ |
| SF32 *<br>(for power reduction) | $\frac{power \cdot (min((slack_i - delay_i)\ ,\ \gamma{\cdot}(max\_transition\_slack_i)\ ))}{cell\ load}$ |
| SF5 | $\Delta power/(\Delta slack{\cdot}\#paths)$ |
| SF8 *<br>(for timing recovery) | $\Delta power/(\Delta delay{\cdot}\#paths)$ |

## Multi Modes Multi Corners (MMMC) Support

TritonSizer supports MMMC feature. Users can turn on the MMMC option with MMMC input files in <cmd> file.

<Example of cmd file for mmmc setup>

```
-mmmc
-mmmcFile <mmmc file1>
-mmmcFile <mmmc file2>
…
```

Library information (.lib files), RC information (.spef file) and timing constraints (.sdc file) need to be specified in MMMC input files.

The input script for timer can be specified in MMMC input file. It specified, the input script is used when running a signoff timer instead of a generated input script by *TritonSizer.*

<Example of MMMC input file>

```
-lib <.lib file>
-staScr <input script for timer>
-spef <.spef file>
-sdc <.sdc file>
```

# 3. Appendix A. Environment File (Example)

```
-libpath /home/_YOUR_DIRECTORY_/lib
-dbpath /home/_YOUR_DIRECTORY_/lib
-lib test_rvt.lib
-lib test_lvt.lib
-lib test_hvt.lib
-db test_rvt.db
-db test_lvt.db
-db test_hvt.db
-suffix_lvt LVT
-suffix_hvt HVT
```

# 4. Appendix B. Command File (Example)

```
# GWTW with 4 threads
#-prft_ptnum 4
#-thread 4
#-GWTW_max_iter 2
#-GWTW_div 2
#-GWTW_num_start 2

# Single thread
-prft_ptnum 1
-thread 1

-prft_only
-ptport 7474
-top test
-v ./test.v
-spef ./test.spef
-sdc ./test.sdc
-tcf ./test.tcf
-noDEF
-ptLog
-useTempus
-alpha 0.5
-dont_touch_cell TIEL
-dont_touch_cell TIEH
-sf 1
-sft 5
-kick_ratio 1.0
-kick_max_iter 3
-opt_effort 2
-fix_slew
-fix_global
-trial_rate 0.3
-slack_margin 0
```

# To be added

- should we provide example scripts to enable any signoff timer?
- add new options (e.g., useOpenSTA) and explanations
- compilation guide
- code review ?
- ...