OpenROAD Safe Names Conventions v1.0

The OpenROAD Project

December 5, 2019

Web: https://theopenroadproject.org/
GitHub: https://github.com/The-OpenROAD-Project

Introduction.

With integration of engines onto the new incremental substrate provided by the OpenDB database and the OpenSTA static timing engine, as well as the opening up to users and developers of Tcl and Python scripting interfaces, the OpenROAD project seeks to define and promulgate “safe names conventions” for the RTL-to-GDS space.

The purpose of this document is to give high-level guidance to our project members and to the open-source EDA developer community, so that open-source EDA tools’ scripting interfaces can remain clear of copyright infringement claims, particularly as they relate to “EDA tool APIs”. We hope that following this guidance will result in the creation of “safe names” for open-source EDA tool APIs in the RTL-to-GDS space.

The OpenROAD project thanks Avatar Integrated Systems for providing selected Aprisa user manual content that has helped us develop the guidance below.

This document accepts comments. Please give us your feedback and suggestions!

Disclaimer and Notice.

Organization of Document.

This document consists of two main sections.

- The first section addresses “how to create an EDA tool API name” -- e.g., for the actions of “analyze setup timing” or “import LEF”.
- The second section lists “verb”, “object”, and “modifier” terms that we believe are natural for tool developers and tool users within the RTL-to-GDS (synthesis,floorplanning, place-and-route, timing analysis, optimization, etc.) space. We classify these terms as “neutral/safe”, “not recommended”, and “recommended alternative”.

Section 1: How to Create an EDA Tool API Name.

Extension mechanisms (**Tcl**, Python) and naming conventions (**verb-modifier-object**-modifier, **underscore as delimiter**, **case-insensitive**) in EDA have converged to a uniform industry-wide style over the past several decades. Use of intuitive, straightforward terms ("report", "delay", "propagated") is natural, and avoids creating a "Tower of Babel" for users of EDA tools.

[As a side note: The style and conventions that we propose can be seen in the OpenROAD PI's academic projects such as the Metrics Dictionary at https://vlsicad.ucsd.edu/GSRC/metrics/ or the naming convention defined at https://vlsicad.ucsd.edu/GSRC/GTX/RULES/naming.html.]

Most API names involve **terms** or **literals** of the following types:

- (1) a "**verb**" or **action**, such as "run" or "load" or "check";
- (2) an "**object**", which is a noun such as "collection" or "arrival time"; and
- (3) zero or more "**modifiers**", which are qualifiers or adjectives such as "above" or "propagated" or "early".

It is possible for a literal to be a compound term, indicated by the use of an underscore character. An example: "number_of", which functions as a modifier in, e.g., "number_of_clocks".

Namespace literals may also be used, such as "db" or "cts".

OpenROAD's recipe for creating a new EDA tool API name is as follows:

- (1) choose the verb, such as "import".
- (2) choose the object, such as "def" or "def_file".
    - *At this point, you may already be done! E.g., "import_def" is a very reasonable API name. Similarly, "report_power" is also a reasonable API name.*
- (3) choose modifier(s) of the object, such as "total", "max", "fall" or "leakage".
    - *The last example modifier can induce either "report_leakage_power" or "report_power_leakage". Both verb-modifier-object and verb-object-modifier styles have been used in EDA tools. We believe **verb-modifier-object is more natural** (e.g., "get_all_instances", or "set_max_capacitance" (seen in public SDC syntax)), but will note when certain modifiers are felt to more naturally follow (certain) objects. For example, in the API name "report_ta_crpr", the literal "crpr" modifies "ta" (timing analysis) but is more natural after the "ta".*
    - *Multiple modifiers can be used, e.g., "unset_disable_inferred_clock_gating".*

Within OpenROAD itself, project architects will periodically review the consistency, intuitiveness and usability of tool API names. **It is a good idea to plan and discuss new APIs at the same time that you discuss new functions with OpenROAD project architects and/or repo owners.**

Section 2: Classes of “verb”, “object” and “modifier” Terms.

Note: This section of the document will evolve with community input. We look forward to receiving your comments and suggestions!

Subsection 2.1: VERBS

NOT RECOMMENDED	RECOMMENDED ALTERNATIVE	EXAMPLE API NAME	NEUTRAL/SAFE VERB LIST
all	**get_all**	**get_all**_instances	add, bisect, check, compare, compute, copy, create, declare, dont_use, enable, exchange, export, extract, generate, help, ignore, import, load, opt_, _optimize, place_, process, purge, quit, route_, set, show, sort, turn_off, undo, unset, verify,
create	**add**	**add**_clock	
define	**declare**	**declare**_custom_property	
derive	**create**	**create**_derived_clocks	
disable	**turn_off**	**turn_off**_case_analysis	
exclude	**ignore**	set_xtk_noise_analysis **-ignore**	
insert	**add**	**add**_scenario	
parse	**process**	**process**_proc_arguments	
read	**import**	**import**_def	
remove	**purge**	**purge**_assigned_delay	
split	**bisect**	**bisect**_object	
swap	**exchange**	**exchange**_cell	
update	**compute**	**compute**_timing	
write	**export**	**export**_verilog	

Subsection 2.2: OBJECTS

Note: (1) “objects” can usually be thought of as “nouns”; (2) for simplicity, elements of “variable/parameter” and “attribute” names are included here, since they typically build from “object” names.

NOT RECOMMENDED	**RECOMMENDED ALTERNATIVE**	**EXAMPLE API NAME**	**NEUTRAL/SAFE OBJECT LIST**
arrival	**arrival_time**	set_xtk_noise_analysis -ignore_**arrival_time**	arrival_time, cell, clock_object, clock_period, clock_skew, cgc, constraint, copy, corner, crpr, db_object, def, delay, end, end_pin, fanin, fanout, instance, inverter_pair, latch, latency, lef, lib, liberty, load, macro, mode, module, name, net, netlist, number_of_, nets, oblist, object_class, parasitics, path, pin, placement, property, routing, sdc, slack, spef, spice, start, start_pin, threshold, transition, uncertainty_from_clock, vcd, verilog, window, xtk,
attribute	**property**	get_**property**	
buffer	**inverter_pair**	add_**inverter_pair**	
class	**object_class**	get_property **-object_class**	
clock	**clock_object** **(not a strong injunction)**	report_ta_crpr -from_**clock_objects**	
clock_uncertainty	**uncertainty_from_clock**		
collection	**oblist**	copy_**oblist**	
_count	**number_of_**	**number_of**_pins	
design	**module**	get_**modules**	
endpoint	**end , end_pin**	get_all_fanouts -only_**end_pins**	
icg	**cgc**	is_**cgc**_enable	
library	**lib**	purge_**lib**	
name	**hier_name**	full_**hier_name**	
noise	**xtk**	purge_**xtk**_noise_analysis	
object	**db_object**		
period	**clock_period**		
si	**xtk**	purge_**xtk**_noise_analysis	
skew	**clock_skew**	report_ta_clock -**clock_skew**	
startpoint	**start , start_pin**	get_all_fanins -only_**start_pins**	
timing	**timing_analysis**	report_**timing_analysis**	
unit	**distance_unit**	export_def -**distance_unit**	

Subsection 2.3: MODIFIERS

NOT RECOMMENDED	RECOMMENDED ALTERNATIVE	EXAMPLE API NAME	NEUTRAL/SAFE MODIFIER LIST
actual	**<NONE>**		above, all (except as an option), assigned, bidir, _cone, detail, crpr, early, eco, every, fall, from_objects, full, generated, incremental, internal, inverted, propagated, late, logic_one, logic_zero, master, max, min, multicycle, number_of, orientation, rise, silent, ta, threshold, thru_objects, timing_sense, to_objects, toggle, total
all (if an option, else OK)	**every**	purge_clock_object -**every**	
annotated	**assigned**	set_**assigned**_delay	
_count	**number_of_**	**number_of**_pins	
delta	**incremental**	**incremental**_delay	
from	**from_objects**	get_all_fanouts -**from_objects**	
is_generated	**generated**	get_ta_**generated**_clocks	
high	**logic_one**	power_**logic_one**_default_static_probability	
inout	**bidir**	ignore_internal_cell_**bidir**_paths	
low	**logic_zero**	set_xtk_noise_analysis -**logic_zero**	
_max	**max_**	**max**_total_cap	
_min	**min_**	**min**_total_cap	
is_propagated	**propagated**	end_has_**propagated**_clock	
quiet	**silent**	set_property -**silent**	
ref_	**master_**	**master**_name	
rise_fall	**timing_sense**		
switching	**toggle**	get_**toggle**_activity_pins	
timing (analysis)	**ta**	get_**ta**_generated_clocks	
to	**to_objects**	get_all_fanins -**to_objects**	
through	**thru_objects**	get_ta_paths -fall_**thru_objects**	
transitive_	**_cone**	report_ta_fanin_**cone**	
verbose	**detail**	report_power_analysis -**detail**	