SpyGlass® LC Parser User Guide

Version N-2017.12-SP2, June 2018

SYNOPSYS®

Synopsys, Inc.
690 E. Middlefield Road
Mountain View, CA 94043
www.synopsys.com

Report an Error

The SpyGlass Technical Publications team welcomes your feedback and suggestions on this publication. Please provide specific feedback and, if possible, attach a snapshot. Send your feedback to *spyglass_support@synopsys.com*.

Contents

Preface

About This Book

The SpyGlass® Library Compiler Reference Guide describes details of SpyGlass Library Compiler.

Contents of This Book

The SpyGlass Library Compiler Reference Guide consists of the following sections:

Section	Description
Statements in a Liberty File	Describes different statements in a liberty file
Compiling Libraries using SpyGlass Library Compiler	Describes the purpose of SpyGlass Library Compiler
Parsing Library-File Statements by SpyGlass Library Compiler	Describes how SpyGlass Library Compiler parses statements in a library file
Interpreting Library-Attributes by SpyGlass Library Compiler	Describes how SpyGlass Library Compiler interprets different attribute types in a library file

Typographical Conventions

This document uses the following typographical conventions:

To indicate	Convention Used
Program code	`OUT <= IN;`
Object names	`OUT`
Variables representing objects names	*`<sig-name>`*
Message	Active low signal name '<sig-name>' must end with _X.
Message location	OUT <= IN;
Reworked example with message removed	OUT_X <= IN;
Important Information	**NOTE:** This rule...

The following table describes the syntax used in this document:

Syntax	Description
[ ] (Square brackets)	An optional entry
{ } (Curly braces)	An entry that can be specified once or multiple times
| (Vertical bar)	A list of choices out of which you can choose one
. . . (Horizontal ellipsis)	Other options that you can specify

Statements in a Liberty File

Liberty file statements are classified in the following categories:

- **Attribute statements**

  The syntax of an attribute statement is as follows:

  ```
  <attribute> : <value> ;
  ```

  Where,

  - <*attribute*> is an attribute.

    Examples of attributes are `capacitance`, `is_pad`, `function`, and `signal_type`.

  - <*value*> is the value of the attribute.

    The data type of the value can be integer, float, double, Boolean, Boolean expression, or string. In addition, there are certain attributes that have a particular format. The type of values accepted by SGLC for each of these categories is discussed in detail in the *Legend Table*.

- **Group statements**

  The syntax of a group statement is as follows:

  ```
  <keyword> ( <group_name> ) {
     Attribute Statements
     Group Statements
  ```

```
}
```

Where,

- ❒ *`<keyword>`* is the group that is defined.

  Examples of groups are `library`, `cell`, `pin`, and `timing`.
- ❒ *`<group_name>`* is the name of the group.

  This is a string value. For groups, such as, `timing` and `memory`, this is optional. For other groups, such as, `pin` and `timing`, this is a list of strings.

NOTE: *In a third type of statement, the attribute value is specified inside parenthesis. The syntax of this statement is as follows:*

```
index_1(float list) , values(list of float list)
```

Compiling Libraries using SpyGlass Library Compiler

Use the `spyglass_lc` utility to compile .lib files to *.sglib* files. For example, the following command compiles the *a.lib* and *b.lib* files:

```
spyglass_lc -gateslib a.lib b.lib -verilog des.v
```

While compiling libraries, you can specify various commands, such as `-lib`, `-gateslib`, and `-lef` with the `spyglass_lc` utility.

Parsing Library-File Statements by SpyGlass Library Compiler

The SpyGlass® Library Compiler (SGLC), invoked using the `spyglass_lc` utility, is primarily based on the syntax and semantics specified in the *Liberty User Guide* (LUG 2013.12).

Following are some properties of SGLC on how it parses a liberty file:

- *Parsing of Comment Statements*
- *Parsing of Attribute Statements*
- *Parsing of Group Statements*

Parsing of Comment Statements

SGLC parses comment statements in a liberty file according to the following rules:

- SGLC supports single line comments ("//") as in C language. LUG does not mention anything about single line comments in liberty files.

NOTE: *"//" specified inside strings (attribute values) is considered a part of the string and not a comment statement.*

- SGLC does not support nested comment structures. There is no mention or use of nested comments in LUG.

Parsing of Attribute Statements

SGLC parses the attribute statements in a liberty file according to the following rules:

- A single space on either side of a colon is not mandatory. For example:

```
pin (X) {
  direction:input;       <----no space
  capacitance  :   1.5 ; <----more than one space
}
```

- A single semicolon is considered as an empty statement. A semicolon is optional at the end of a statement. In addition, a semicolon can be specified after opening/closing curly brackets and start/end of statements. For example:

```
pin (A) { ;             <-----extra semicolon
  direction : input     <-----optional semicolon
  ; capacitance : 1.5 ; <-----extra semicolon at start
};                      <-----extra semicolon
;                       <-----empty statement
```

Categorizing Attributes by SpyGlass Library Compiler

SGLC can treat an attribute in the following four ways depending upon the type of the attribute:

- **Parsing the attribute and populating it in the object model**.

  There are certain attributes that are populated in the object model after syntax checking.

- **Parsing the attribute but not populating it in the object model**.

  There are certain attributes on which only syntax checking is done but they are not populated in the object model. Requirement of populating these attributes in the object model is based on the product requirements.

- **Not parsing the attribute but not displaying warning messages**.

  There are certain attributes/groups which, though exist in LUG, are ignored by SGLC. No warning is flagged for such attributes/groups. In addition, no syntax checking is performed on them. In case a particular group is ignored, the attributes/sub-groups specified in that group are also ignored.

- **Parsing the attribute and displaying warning messages.**

  In case of unknown attributes or new attributes, SGLC flags a warning (LIBWRN_78) and ignores those attributes.

  In addition, if an attribute is known but it is not specified in the required scope (for example, `direction` attribute specified in `cell` group), the attribute is ignored and a warning (LIBWRN_111) is flagged.

NOTE: *User-defined attributes and groups (declared using* `define` *and* `define_group` *statements) are parsed and stored in the object model.*

Example

```
library( mylib ) {
       cell ( mycell ) {
            direction : input ;
            pin ( mypin ) {
                 my_direction : input ;
            }
       }
}
```

The following warning messages are displayed:

ID	Rule	Alias	Severity	File	Line	Wt	Message
[0]	LIBWRN_111		Warning	a.lib	3	10 00	Construct 'direction' is not supported in the scope of cell group
[1]	LIBWRN_78		Warning	a.lib	5	10 00	Ignoring Unknown Attribute 'my_direction' in group 'pin'

Parsing of Group Statements

SGLC parses the group statements in a liberty file according to the following rules:

- SGLC allows the use of opening curly brackets in a new line whereas most EDA tools expect the curly brackets in the same line. For example:

  ```
  cell ( mycell )
  {                      <-----brackets in new line
      ........
  }
  ```

- SGLC supports the use of nested parenthesis. For example:

  ```
  pin ((( CIN ))) {       <----nested parenthesis
     .............
  }
  ```

Interpreting Library-Attributes by SpyGlass Library Compiler

This section helps you to understand how SpyGlass Library Compiler (SGLC) reads a library file. For better understanding, refer to the *Legend Table* that describes the different attribute value types. The subsequent sections describe how SGLC interprets each attribute in a liberty file, that is, the value type with which it associates each attribute. If any attribute in a liberty file is not specified with the expected value type, SGLC flags a syntax error (`LIBSTX_403`).

As discussed earlier, a group statement contains attribute-statements and (sub)group-statements. This document is organized based on groups. Within a group, the attribute statements are specified with their associated value type while the sub-group statements are present as links. Clicking a sub-group name takes you to that group.

Example

```
*  Library Group contains *
   library( complex_string ) {
     ...............
    input_threshold_pct_fall    : float ;   <-Value type for
                                              the attribute
     output_threshold_pct_fall  : float ;
     input_threshold_pct_rise   : float ;
```

```
          output_threshold_pct_rise  : float ;
          ................
          CellGroup
          ScalingFactorData
          EmLutTemplateGroup
          .........
       }
     * Cell Group contains *
       cell( complex_string ) {
           ...............
           PinGroup
           ...............
       }
```

As mentioned in the *Categorizing Attributes by SpyGlass Library Compiler* section, there are four different categories of attributes/groups. All the attributes/groups falling in Category 1 and 2 are described in this document. Attributes/groups of category 3 are captured separately as part of each attribute/group description and in the appendix. Please note that the attributes and sub-groups of the ignored group are also ignored. These attributes and sub-groups are not mentioned in this document.

Legend Table

The following table describes the attribute types and their values:

Attribute	Value
float	Integer and floating-point numbers. Integer numbers: Can start with +/- Floating-point numbers with the following properties: • Can start with - • Can be written in exponential form, for example, 1.6e-19 and -1.73E+23
float_list	List of float values
float_iter	List of float_list
double	8-byte floating-point numbers
boolean	true or false
boolexpr	Boolean expression with operators, such as, [* & ^ ! + | ']
unit	current: 1uA, 10uA, 100uA, 1mA, 10mA, 100mA, 1A leakage_power: 1pW, 10pW, 100pW, 1nW, 10nW, 100nW,1uW, 10uW, 100uW, 1mW pulling_resistance: 1ohm, 10ohm, 100ohm, 1kohm time: 1ps, 10ps, 100ps, 1ns voltage: 1mV, 10mV, 100mV, 1V
enum_string	Limited set of values as specified in the LUG
enum_string_list	List of enum_string
simple_string	String with the following properties: • Cannot start with the following characters: [. / ! -] • Cannot contain the following characters: [^ & * + | ( ) = { } \ : ; " ' ,]
simple_string_list	List of simple_string

Attribute	Value
complex_string	String with the following properties: • Cannot start with /* • Cannot contain the following characters: [( ) { } = \ ; " < > ,] • Every keyboard character can be specified within quotes. However, specifying spaces in case of pin/bus/bundle names or cell names will cause errors in further processing.
complex_string_list	List of complex_string

NOTE: *Groups for which group names are optional are given in square [ ] brackets.*

LibraryGroup

```
library ( complex_string ) {
```

default_cell_failure_rate	: float;
input_threshold_pct_fall	: float;
output_threshold_pct_fall	: float;
input_threshold_pct_rise	: float;
output_threshold_pct_rise	: float;
slew_lower_threshold_pct_fall	: float;
slew_upper_threshold_pct_fall	: float;
slew_lower_threshold_pct_rise	: float;
slew_upper_threshold_pct_rise	: float;
slew_derate_from_library	: float;
technology(enum_string);	
bus_naming_style	: complex_string;
comment	: complex_string;
current_unit	: unit ;
date	: "complex_string";
default_leakage_power_density	: float;
delay_model	: enum_string;
power_model	: enum_string;
in_place_swap_mode	: enum_string;
key_bit	: float;
key_feature	: complex_string;
key_file	: complex_string;
key_seed	: float;
key_version	: float;
leakage_power_unit	: unit;
lsi_pad_fall	: float;
lsi_pad_rise	: float;
lsi_rounding_cutoff	: float;
lsi_rounding_digit	: float;

nom_process	: float;
nom_temperature	: float;
nom_voltage	: float;
nom_calc_mode	: complex_string;
piece_type	: enum_string;
preferred_output_pad_slew_rate_control	: enum_string;
preferred_output_pad_voltage	: complex_string;
pulling_resistance_unit	: unit;
reference_capacitance	: float;
revision	: "complex_string";
simulation	: Boolean;
time_unit	: unit;
timing_report	: Boolean;
voltage_unit	: unit ;
capacitive_load_unit(float, simple_string);	
define_cell_area(simple_string, enum_string);	
library_features(complex_string_list);	
piece_define(float_list);	
routing_layers(complex_string_list);	
voltage_map(complex_string , float);	
DefaultData	
ScalingFactorData	
EmLutTemplateGroup	
CellGroup	
FallNetDelayGroup	
FallTransitionDegradationGroup	
InputVoltageGroup	
LuTableTemplateGroup	
ModelGroup	
OperatingConditionsGroup	
PowerSupplyGroup	
OutputVoltageGroup	
PolyTemplateGroup	

PowerPolyTemplateGroup
PowerLutTemplateGroup
RiseNetDelayGroup
RiseTransitionDegradationGroup
ScaledCellGroup
ScalingFactorGroup
TimingRangeGroup
TypeGroup
WireLoadGroup
WireLoadSelectionGroup
WireLoadTableGroup

Refer to the *Legend Table* for the attribute types and their values.

Ignored Attributes	preferred_input_pad_voltage resistance_unit distance_unit dist_conversion_factor va_parameters base_curve_type curve_x curve_y base_curve_type capacitance_conversion_factor curve_x curve_y default_fpga_isd default_ocv_derate_distance_group default_ocv_derate_group default_part define_group dist_con_factor dist_conversion_factor distance_unit fpga_domain_style fpga_isd fpga_technology k_temp_rise_propogation k_temp_rise_wire_resistance ocv_arc_depth ocv_derate scan_group va_parameters

Ignored Groups	base_curves ccs_lu_table_template ccs_timing_base_curve ccs_timing_base_curve_template compact_lut_template iv_lut_template maxcap_lut_template maxtrans_lut_template noise_lut_template normalized_driver_waveform output_current_template pg_current_template propagation_lut_template sensitization user_parameters dc_current_template faults_lut_template critical_area_lut_template model device_layer cont_layer poly_layer routing_layer cont_layer critical_area_lut_template dc_current_template device_layer faults_lut_template model ocv_table_template output_current part pg_current_template poly_layer routing_layer timing

CellGroup

cell(complex_string){

timing_model_type	: simple_string;
user_function_class	: simple_string;
retention_cell	: simple_string ;
is_macro_cell	: Boolean
CellData	
PinGroup	
BusGroup	
BundleGroup	
PGPinGroup	
FFGroup	
FFBankGroup	
LatchGroup	
LatchBankGroup	
SeqGroup	
SeqBankGroup	
StateGroup	
StatetableGroup	
TestCellGroup	
GeneratedClockGroup	
InternalPowerGroup	
LeakagePowerGroup	
LutGroup	
MemoryGroup	
ModeDefinitionGroup	
RoutingTrackGroup	
TypeGroup	

}

Refer to the *Legend Table* for the attribute types and their values.

Ignored Attributes	driver_waveform driver_waveform_fall driver_waveform_rise em_temp_degradation_factor failure_rate pin_name_map sensitization_master is_decap_cell is_filler_cell is_tap_cell is_pll_cell power_cell_type auxiliary_pad_cell physical_connection antenna_diode_type auxiliary_pad_cell base_name drive_type fpga_cell_type fpga_domain_style io_type is_decap_cell is_filler_cell is_pll_cell is_tap_cell ocv_arc_depth ocv_derate_distance_group ocv_derate_group physical_connection power_cell_type resource_usage scan_group slew_type

Ignored Groups	dc_current dynamic_current intrinsic_parasitic leakage_current cell_based_variation critical_area_table functional_yield_metric leakage_current cell_based_variation critical_area_table edif_name fpga_condition fpga_isd functional_yield_metric gate_leakage ocv_derate retention_condition

CellData

area	: float;
auxiliary_pad_cell	: Boolean;
bus_naming_style	: complex_string;
switch_cell_type	: enum_string;
cell_footprint	: complex_string;
cell_leakage_power	: double;
contention_condition	: boolexpr;
dont_fault	: enum_string;
dont_touch	: Boolean;
clock_gating_integrated_cell	: enum_string;
dont_use	: Boolean;
geometry_print	: complex_string;
power_gating_cell	: complex_string;
handle_negative_constraint	: Boolean;
ignore_verify_icg_type	: Boolean;
interface_timing	: Boolean;
is_clock_gating_cell	: Boolean;
map_only	: Boolean;
mpm_libname	: complex_string;
mpm_name	: complex_string;
observe_node	: complex_string;
pad_cell	: Boolean;
io_slots	: float;
bond_pads	: float;
pad_drivers	: float;
io_pads	: float;
pad_type	: complex_string; // clock type is only supported
preferred	: Boolean;
rail_connection(simple_string, simple_string);	

version	: "complex_string";
scaling_factors	: complex_string;
scan_group	: simple_string;
set_node	: complex_string;
use_for_size_only	: Boolean;
vhdl_name	: “simple_string”;
pin_equal(simple_string_list);	
pin_opposite(simple_string, simple_string);	
single_bit_degenerate	: complex_string;
threshold_voltage_group	: complex_string;
is_isolation_cell	: Boolean;
is_level_shifter	: Boolean;
level_shifter_type	: enum_string;
input_voltage_range(float, float);	
output_voltage_range(float, float);	
input_threshold_pct_fall	: float;
input_threshold_pct_rise	: float;
output_threshold_pct_fall	: float;
output_threshold_pct_rise	: float;
slew_lower_threshold_pct_fall	: float;
slew_lower_threshold_pct_rise	: float;
slew_upper_threshold_pct_fall	: float;
slew_upper_threshold_pct_rise	: float;

Refer to the *Legend Table* for the attribute types and their values.

PinGroup

```
pin( complex_string_list ) {
```

always_on	: Boolean;
has_builtin_pad	: Boolean;
max_input_noise_width	: float;
min_input_noise_width	: float;
is_isolated	: Boolean;
isolation_enable_condition	: boolexpr
fall_capacitance_range	: (float , float);
rise_capacitance_range	: (float , float);
rise_capacitance	: float
fall_capacitance	: float
PinData	
ElectroMigrationGroup	
EmMaxToggleRateGroup	
InternalPowerGroup	
MemoryReadGroup	
MemoryWriteGroup	
MinPulseWidthGroup	
MinimumPeriodGroup	
TimingGroup	
TLatchGroup	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

Ignored Attributes	bus_hold_function is_three_state open_drain_function open_source_function power_gating_pin pull_down_function pull_up_function resistive_0_function resistive_1_function resistive_function bias_connection pull_up_function pull_down_function bus_hold_function open_drain_function open_source_function resistive_function resistive_0_function resistive_1_function has_pass_gate data_in_type is_pll_reference_pin is_pll_feedback_pin is_pll_output_pin is_unbuffered pulse_clock is_analog physical_connection alive_during_partial_power_down antenna_diode_related_ground_pins antenna_diode_related_power_pins bias_connection data_in_type has_pass_gate is_pll_feedback_pin is_pll_output_pin is_pll_reference_pin is_unbuffered physical_connection pulse_clock

Ignored Groups	ccsn_first_stage ccsn_last_stage hyperbolic_noise_above_high hyperbolic_noise_below_low hyperbolic_noise_high hyperbolic_noise_low input_signal_swing max_cap max_trans output_signal_swing pin_based_variation pin_capacitance receiver_capacitance electromigration electromigration pin_based_variation

BusGroup

```
bus( complex_string ){
```

bus_type	: complex_string; //This must be the first attribute
always_on	: Boolean;
has_builtin_pad	: Boolean;
max_input_noise_width	: float;
min_input_noise_width	: float;
input_map_shift(simple_string, "+/-", simple_string);	
input_map	: “simple_string_list”;
pin_equal(simple_string_list);	
pin_opposite(simple_string, simple_string);	
is_isolated	: Boolean;
isolation_enable_condition	: boolexpr
fall_capacitance_range	: (float , float);
rise_capacitance_range	: (float , float);
rise_capacitance	: float
fall_capacitance	: float
PinData	
ElectroMigrationGroup	
EmMaxToggleRateGroup	
InternalPowerGroup	
MemoryReadGroup	
MemoryWriteGroup	
MinPulseWidthGroup	
MinimumPeriodGroup	
TimingGroup	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

Ignored Attributes	is_three_state
Ignored Groups	ccsn_last_stage hyperbolic_noise_above_high hyperbolic_noise_below_low hyperbolic_noise_high hyperbolic_noise_low max_cap max_trans pin_capacitance receiver_capacitance tlatch

BundleGroup

```
bundle( complex_string ){
```

`members( complex_string_list );`	`//This must be the first attribute`
`has_builtin_pad`	`: Boolean;`
`max_input_noise_width`	`: float ;`
`min_input_noise_width`	`: float;`
PinData	
`input_map_shift( simple_string, "+/-", simple_string );`	
`input_map`	`: “simple_string_list”;`
`pin_equal( simple_string_list );`	
`pin_opposite( simple_string, simple_string );`	
`is_isolated`	`: Boolean;`
`isolation_enable_condition`	`: boolexpr`
`fall_capacitance_range`	`: ( float , float );`
`rise_capacitance_range`	`: ( float , float );`
`rise_capacitance`	`: float`
`fall_capacitance`	`: float`
ElectroMigrationGroup	
EmMaxToggleRateGroup	
InternalPowerGroup	
MemoryReadGroup	
MemoryWriteGroup	
MinPulseWidthGroup	
MinimumPeriodGroup	
PinGroup	
TimingGroup	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

Ignored Attributes	is_three_state power_gating_pin physical_connection
Ignored Groups	ccsn_last_stage hyperbolic_noise_above_high hyperbolic_noise_below_low hyperbolic_noise_high hyperbolic_noise_low max_cap max_trans pin_capacitance receiver_capacitance tlatch

PGPinGroup

```
pg_pin( complex_string ){
```

pg_type	: enum_string;
user_pg_type	: complex_string;
pg_function	: boolexpr;
direction	: enum_string;
voltage_name	: complex_string;
switch_function	: boolexpr;
std_cell_main_rail	: Boolean;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

Ignored Attributes	bias_connection pull_up_function pull_down_function bus_hold_function open_drain_function open_source_function resistive_function resistive_0_function resistive_1_function has_pass_gate physical_connection bias_connection bus_hold_function has_pass_gate open_drain_function open_source_function physical_connection pull_down_function pull_up_function resistive_0_function resistive_1_function resistive_function

FFGroup

```
ff( simple_string, simple_string ){
```

clocked_on	: boolexpr;
next_state	: boolexpr;
clear	: boolexpr;
preset	: boolexpr;
clear_preset_var1	: enum_string;
clear_preset_var2	: enum_string;
clocked_on_also	: boolexpr;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

FFBankGroup

```
ff_bank( simple_string, simple_string, float ){
```

clocked_on	: boolexpr;
next_state	: boolexpr;
clear	: boolexpr;
preset	: boolexpr;
clear_preset_var1	: enum_string;
clear_preset_var2	: enum_string;
clocked_on_also	: boolexpr;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

LatchGroup

```
latch( simple_string, simple_string ){
```

enable	: boolexpr;
enable_also	: boolexpr;
data_in	: boolexpr;
clear	: boolexpr;
preset	: boolexpr;
clear_preset_var1	: enum_string;
clear_preset_var2	: enum_string ;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

LatchBankGroup

```
latch_bank( simple_string, simple_string, float ){
```

enable	: boolexpr;
enable_also	: boolexpr;
data_in	: boolexpr;
clear	: boolexpr;
preset	: boolexpr;
clear_preset_var1	: enum_string;
clear_preset_var2	: enum_string ;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

SeqGroup

```
seq( simple_string, simple_string ){
```

clear	: boolexpr;
clear_preset_var1	: enum_string;
clear_preset_var2	: enum_string;
clocked_on	: boolexpr;
clocked_on_also	: boolexpr;
data_in	: boolexpr;
enable	: boolexpr;
enable_also	: boolexpr;
next_state	: boolexpr;
preset	: boolexpr;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

SeqBankGroup

```
seq_bank( simple_string, simple_string, complex_string ){
```

clear	: boolexpr;
clear_preset_var1	: enum_string;
clear_preset_var2	: enum_string;
clocked_on	: boolexpr;
clocked_on_also	: boolexpr;
data_in	: boolexpr;
enable	: boolexpr;
enable_also	: boolexpr;
next_state	: boolexpr;
preset	: boolexpr;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

StateGroup

```
state( complex_string, complex_string ){
```

clocked_on	: boolexpr;
clocked_on_also	: boolexpr;
data_in	: boolexpr;
force_00	: boolexpr;
force_01	: boolexpr;
force_10	: boolexpr;
force_11	: boolexpr;
next_state	: boolexpr;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

StatetableGroup

```
statetable( simple_string, simple_string ){
table:
```

“ :	simple_string_list	simple_string_list	:	simple_string_list	,
:	simple_string_list	simple_string_list	:	simple_string_list	,
:	simple_string_list	simple_string_list	:	simple_string_list	,
	-------------	--------------		--------------	”

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

TestCellGroup

```
test_cell( [ complex_string ] ){
```

bus_naming_style	: complex_string ;
BusGroup	
BundleGroup	
FFGroup	
FFBankGroup	
GeneratedClockGroup	
InternalPowerGroup	
LatchGroup	
LatchBankGroup	
LeakagePowerGroup	
LuTableTemplateGroup	
MemoryGroup	
ModeDefinitionGroup	
PinGroup	
PGPinGroup	
RoutingTrackGroup	
SeqGroup	
SeqBankGroup	
StateGroup	
StatetableGroup	
TypeGroup	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

GeneratedClockGroup

```
generated_clock( complex_string ){
```

`clock_pin`	`: simple_string_list;`
`divided_by`	`: float;`
`duty_cycle`	`: float;`
`invert`	`: Boolean;`
`master_pin`	`: complex_string;`
`multiplied_by`	`: float;`
`edges( float, float, float );`	
`shifts( float, float, float );`	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

TimingGroup

```
timing( [complex_string_list ] ){
```

default_timing	: Boolean ;
edge_rate_sensitivity_f0	: float ;
edge_rate_sensitivity_f1	: float ;
edge_rate_sensitivity_r0	: float ;
edge_rate_sensitivity_r1	: float ;
fall_resistance	: float ;
hold_coefficient	: float ;
intrinsic_fall	: float ;
intrinsic_rise	: float ;
mode(simple_string, simple_string_list);	
related_bus_equivalent	: simple_string_list ;
related_pin	: simple_string_list ;
related_bus_pins	: simple_string_list ;
related_output_pin	: complex_string ;
rise_resistance	: float ;
sdf_cond	: complex_string ;
sdf_cond_end	: complex_string ;
sdf_cond_start	: complex_string ;
sdf_edges	: complex_string ;
setup_coefficient	: float ;
slope_fall	: float ;
slope_rise	: float ;
steady_state_resistance_float_max	: float ;
steady_state_resistance_float_min	: float ;
steady_state_resistance_high_max	: float ;
steady_state_resistance_high_min	: float ;
steady_state_resistance_low_max	: float ;
steady_state_resistance_low_min	: float ;
timing_type	: enum_string ;

timing_sense	: enum_string ;
when	: boolexpr ;
when_end	: boolexpr ;
when_start	: boolexpr ;
fall_delay_intercept(float, float);	
fall_pin_resistance(float, float);	
rise_delay_intercept(float, float);	
rise_pin_resistance(float, float);	
clock_gating_flag	: Boolean ;
CellDegradationGroup	
CellFallGroup	
CellRiseGroup	
DomainGroup	
FallPropagationGroup	
RisePropagationGroup	
FallTransitionGroup	
RetainingRiseGroup	
RetainingFallGroup	
RetainRiseSlewGroup	
RetainFallSlewGroup	
RiseTransitionGroup	
FallConstraintGroup	
RiseConstraintGroup	

}

Refer to the *Legend Table* for the attribute types and their values.

Ignored Attributes	interdependence_id pin_name_map sensitization_master steady_state_resistance_above_high steady_state_resistance_below_low steady_state_resistance_high steady_state_resistance_low tied_off wave_fall wave_rise cv_arc_depth

Ignored Groups	cell_fall_to_pct cell_rise_to_pct compact_ccs_fall compact_ccs_rise compressed_ccs_timing_rise noise_immunity_above_high noise_immunity_below_low noise_immunity_high noise_immunity_low output_current_fall output_current_rise propagated_noise_height_above_high propagated_noise_height_below_low propagated_noise_height_high propagated_noise_height_low propagated_noise_peak_time_ratio_above_high propagated_noise_peak_time_ratio_below_low propagated_noise_peak_time_ratio_high propagated_noise_peak_time_ratio_low propagated_noise_width_above_high propagated_noise_width_below_low propagated_noise_width_high propagated_noise_width_low receiver_capacitance1_fall receiver_capacitance1_rise receiver_capacitance2_fall receiver_capacitance2_rise steady_state_current_high steady_state_current_low steady_state_current_tristate timing_based_variation compact_ccs_retain_rise compact_ccs_retain_fall compact_ccs_rise compact_ccs_fall compact_ccs_retain_fall compact_ccs_retain_rise compact_ccs_rise ocv_sigma_cell_fall ocv_sigma_cell_rise

PinData

bit_width	: float ;
capacitance	: float ;
clock_gate_clock_pin	: Boolean ;
clock_gate_enable_pin	: Boolean ;
clock_gate_test_pin	: Boolean ;
clock_gate_obs_pin	: Boolean ;
clock_gate_out_pin	: Boolean ;
fault_model	: complex_string ;
complementary_pin	: simple_string ;
dcm_timing	: Boolean ;
clock	: Boolean ;
connection_class	: simple_string_list ;
direction	: enum_string ;
dont_fault	: enum_string ;
drive_current	: float ;
driver_type	: enum_string or "enum_string enum_string" ;
edge_rate_breakpoint_f0	: float ;
edge_rate_breakpoint_f1	: float ;
edge_rate_breakpoint_r0	: float ;
edge_rate_breakpoint_r1	: float ;
edge_rate_fall	: float ;
edge_rate_load_fall	: float ;
edge_rate_load_rise	: float ;
edge_rate_rise	: float ;
fall_capacitance	: float ;
fall_current_slope_after_threshold	: float ;
fall_current_slope_before_threshold	: float ;
fall_time_after_threshold	: float ;
fall_time_before_threshold	: float ;

fanout_load	: float ;
fsim_map	: complex_string ;
function	: boolexpr ;
hysteresis	: Boolean ;
input_map	: “simple_string_list” ;
input_signal_level	: complex_string ;
input_signal_level_low	: float ;
input_signal_level_high	: float ;
output_signal_level_low	: float ;
output_signal_level_high	: float ;
input_voltage	: complex_string ;
internal_node	: simple_string ;
inverted_output	: Boolean ;
is_pad	: Boolean ;
isolation_cell_data_pin	: Boolean ;
level_shifter_data_pin	: Boolean ;
level_shifter_enable_pin	: Boolean ;
isolation_cell_enable_pin	: Boolean ;
lsi_pad	: Boolean ;
max_time_borrow	: float ;
max_capacitance	: float ;
max_fanout	: float ;
max_transition	: float ;
min_capacitance	: float ;
min_fanout	: float ;
min_period	: float ;
min_transition	: float ;
min_pulse_width_high	: float ;
min_pulse_width_low	: float ;
multicell_pad_pin	: Boolean ;
nextstate_type	: enum_string ;
output_signal_level	: complex_string ;
output_voltage	: complex_string ;

pin_func_type	: enum_string ;
prefer_tied	: float ;
primary_output	: Boolean ;
pulling_current	: float ;
pulling_resistance	: float ;
reference_capacitance	: float ;
rise_capacitance	: float ;
rise_current_slope_after_threshold	: float ;
rise_current_slope_before_threshold	: float ;
rise_time_after_threshold	: float ;
rise_time_before_threshold	: float ;
signal_type	: enum_string ;
slew_control	: enum_string ;
state_function	: boolexpr ;
test_output_only	: Boolean ;
three_state	: boolexpr ;
vhdl_name	: “simple_string” ;
related_power_pin	: complex_string ;
related_ground_pin	: complex_string ;
switch_pin	: Boolean ;
retention_pin(enum_string, float);	
map_to_logic	: complex_string ;
x_function	: boolexpr ;
input_threshold_pct_fall	: float ;
input_threshold_pct_rise	: float ;
output_threshold_pct_fall	: float ;
output_threshold_pct_rise	: float ;
slew_lower_threshold_pct_fall	: float ;
slew_upper_threshold_pct_fall	: float ;
slew_lower_threshold_pct_rise	: float ;
slew_upper_threshold_pct_rise	: float ;
power_down_function	: boolexpr ;

Refer to the *Legend Table* for the attribute types and their values.

nextstate_type

To make SpyGlass library compiler infer a pin as enable (or load) in a multi-bit flip flop, include the appropriate specification of the `nextstate_type` attribute on that pin.

In a pin group, `nextstate_type` defines the type of a `next_state` attribute to be used in an FFGroup, a sequential group, or an ff_bank group. In general, `nextstate_type` attribute should be added on a pin to explicitly mention the type of the pin, as used in an FFGroup, a sequential group, or an ff_bank group.

If the result in not as intended, check the library for each input used in the `next_state` statement of a sequential group and add the `nextstate_type` attribute if necessary.

TimingData

`index_1( float_list );`
`index_2( float_list );`
`index_3( float_list );`
`intermediate_values( float_iter );`
`values( float_iter );`
`orders( float_list );`
`coefs ( float_list )`
`variable_1_range( float, float );`
`variable_2_range( float, float );`
`variable_3_range( float, float );`
`variable_4_range( float, float );`
`variable_5_range( float, float );`
`variable_6_range( float, float );`
`variable_7_range( float, float );`
DomainGroup

Refer to the *Legend Table* for the attribute types and their values.

ScalingFactorData

k_process_cell_degradation	:	float ;
k_process_cell_fall	:	float ;
k_process_cell_leakage_power	:	float ;
k_process_cell_rise	:	float ;
k_process_drive_current	:	float ;
k_process_drive_fall	:	float ;
k_process_drive_rise	:	float ;
k_process_fall_delay_intercept	:	float ;
k_process_fall_pin_resistance	:	float ;
k_process_fall_propagation	:	float ;
k_process_fall_transition	:	float ;
k_process_hold_fall	:	float ;
k_process_hold_rise	:	float ;
k_process_internal_power	:	float ;
k_process_intrinsic_fall	:	float ;
k_process_intrinsic_rise	:	float ;
k_process_min_period	:	float ;
k_process_min_pulse_width_high	:	float ;
k_process_min_pulse_width_low	:	float ;
k_process_nochange_fall	:	float ;
k_process_nochange_rise	:	float ;
k_process_pin_cap	:	float ;
k_process_pin_fall_cap	:	float ;
k_process_pin_rise_cap	:	float ;
k_process_recovery_fall	:	float ;
k_process_recovery_rise	:	float ;
k_process_removal_fall	:	float ;
k_process_removal_rise	:	float ;
k_process_rise_delay_intercept	:	float ;
k_process_rise_pin_resistance	:	float ;

k_process_rise_propagation	: float ;
k_process_rise_transition	: float ;
k_process_setup_fall	: float ;
k_process_setup_rise	: float ;
k_process_skew_fall	: float ;
k_process_slope_rise	: float ;
k_temp_cell_degradation	: float ;
k_temp_cell_fall	: float ;
k_temp_cell_leakage_power	: float ;
k_temp_cell_rise	: float ;
k_temp_drive_current	: float ;
k_temp_drive_fall	: float ;
k_temp_drive_rise	: float ;
k_temp_fall_delay_intercept	: float ;
k_temp_fall_pin_resistance	: float ;
k_temp_fall_propagation	: float ;
k_temp_fall_transition	: float ;
k_temp_hold_fall	: float ;
k_temp_hold_rise	: float ;
k_temp_internal_power	: float ;
k_temp_intrinsic_fall	: float ;
k_temp_intrinsic_rise	: float ;
k_temp_min_period	: float ;
k_temp_min_pulse_width_high	: float ;
k_temp_min_pulse_width_low	: float ;
k_temp_nochange_fall	: float ;
k_temp_nochange_rise	: float ;
k_temp_pin_cap	: float ;
k_temp_recovery_fall	: float ;
k_temp_recovery_rise	: float ;
k_temp_removal_fall	: float ;
k_temp_removal_rise	: float ;
k_temp_rise_delay_intercept	: float ;

k_temp_rise_pin_resistance	: float ;
k_temp_rise_propagation	: float ;
k_temp_rise_transition	: float ;
k_temp_setup_fall	: float ;
k_temp_setup_rise	: float ;
k_temp_skew_fall	: float ;
k_temp_skew_rise	: float ;
k_temp_slope_fall	: float ;
k_temp_slope_rise	: float ;
k_volt_cell_degradation	: float ;
k_volt_cell_fall	: float ;
k_volt_cell_leakage_power	: float ;
k_volt_cell_rise	: float ;
k_volt_drive_current	: float ;
k_volt_drive_fall	: float ;
k_volt_drive_rise	: float ;
k_volt_fall_delay_intercept	: float ;
k_volt_fall_pin_resistance	: float ;
k_volt_fall_propagation	: float ;
k_volt_fall_transition	: float ;
k_volt_hold_fall	: float ;
k_volt_hold_rise	: float ;
k_volt_internal_power	: float ;
k_volt_intrinsic_fall	: float ;
k_volt_intrinsic_rise	: float ;
k_volt_min_period	: float ;
k_volt_min_pulse_width_high	: float ;
k_volt_min_pulse_width_low	: float ;
k_volt_nochange_fall	: float ;
k_volt_nochange_rise	: float ;
k_volt_pin_cap	: float ;
k_volt_recovery_fall	: float ;
k_volt_recovery_rise	: float ;

`k_volt_removal_fall`	`: float ;`
`k_volt_removal_rise`	`: float ;`
`k_volt_rise_delay_intercept`	`: float ;`
`k_volt_rise_pin_resistance`	`: float ;`
`k_volt_rise_propagation`	`: float ;`
`k_volt_rise_transition`	`: float ;`
`k_volt_setup_fall`	`: float ;`
`k_volt_setup_rise`	`: float ;`
`k_volt_skew_fall`	`: float ;`
`k_volt_skew_rise`	`: float ;`
`k_volt_slope_fall`	`: float ;`
`k_volt_slope_rise`	`: float ;`
`k_volt_wire_cap`	`: float ;`
`k_volt_wire_res`	`: float ;`

Ignored Attributes	k_temp_rise_propogation k_temp_rise_wire_resistance

Refer to the *Legend Table* for the attribute types and their values.

DomainGroup

```
domain( complex_string ){
```

calc_mode	: complex_string ;
variables(complex_string_list);	
mapping(simple_string, simple_string);	
variable_1_range(float, float);	
variable_2_range(float, float);	
variable_3_range(float, float);	
variable_4_range(float, float);	
variable_5_range(float, float);	
variable_6_range(float, float);	
variable_7_range(float, float);	
coefs (float_list) ;	
orders (float_list) ;	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

Ignored Attributes	index_1 index_2 index_3 variable_1 variable_2 variable_3 variable_n_range

ScaledCellGroup

```
scaled_cell( simple_string, simple_string ){
```

timing_model_type	: simple_string ;
user_function_class	: simple_string ;
retention_cell	: simple_string ;
CellData	
BusGroup	
BundleGroup	
FFGroup	
FFBankGroup	
GeneratedClockGroup	
InternalPowerGroup	
LatchGroup	
LatchBankGroup	
LeakagePowerGroup	
LuTableTemplateGroup	
MemoryGroup	
ModeDefinitionGroup	
PinGroup	
PGPinGroup	
RoutingTrackGroup	
SeqGroup	
SeqBankGroup	
StateGroup	
StatetableGroup	
TestCellGroup	
TypeGroup	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

Ignored Attributes	failure_rate scan_group

ModelGroup

```
model( complex_string ){
```

`cell_name`	`: complex_string ;`
`short( complex_string_list );`	
CellData	
BusGroup	
BundleGroup	
FFGroup	
FFBankGroup	
GeneratedClockGroup	
InternalPowerGroup	
LatchGroup	
LatchBankGroup	
LeakagePowerGroup	
LutGroup	
MemoryGroup	
ModeDefinitionGroup	
PinGroup	
PGPinGroup	
RoutingTrackGroup	
SeqGroup	
SeqBankGroup	
StateGroup	
StatetableGroup	
TestCellGroup	
TypeGroup	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

Ignored Attributes	em_temp_degradation_factor failure_rate timing_model_type power_cell_type auxiliary_pad_cell auxiliary_pad_cell base_name drive_type fpga_cell_type io_type power_cell_type resource_usage scan_group slew_type
Ignored Groups	edif_name fpga_condition fpga_domain_style fpga_isd

CellDegradationGroup

```
cell_degradation( complex_string ){
  TimingData
}
```

CellFallGroup

```
cell_fall( complex_string ){
    TimingData
}
```

CellRiseGroup

```
cell_fall( complex_string ){
    TimingData
}
```

DefaultData

default_cell_leakage_power	: float ;
default_connection_class	: simple_string_list ;
default_edge_rate_breakpoint_f0	: float ;
default_edge_rate_breakpoint_f1	: float ;
default_edge_rate_breakpoint_r0	: float ;
default_edge_rate_breakpoint_r1	: float ;
default_fall_delay_intercept	: float ;
default_fall_pin_resistance	: float ;
default_fanout_load	: float ;
default_hold_coefficient	: float ;
default_inout_pin_cap	: float ;
default_inout_pin_fall_res	: float ;
default_inout_pin_rise_res	: float ;
default_input_pin_cap	: float ;
default_intrinsic_fall	: float ;
default_intrinsic_rise	: float ;
default_leakage_power_density	: float ;
default_max_capacitance	: float ;
default_max_fanout	: float ;
default_max_transition	: float ;
default_max_utilization	: float ;
default_min_porosity	: float ;
default_operating_conditions	: complex_string ;
default_output_pin_cap	: float ;
default_output_pin_fall_res	: float ;
default_output_pin_rise_res	: float ;
default_rc_fall_coefficient	: float ;
default_rc_rise_coefficient	: float ;
default_reference_capacitance	: float ;
default_rise_delay_intercept	: float ;

default_rise_pin_resistance	: float ;
default_setup_coefficient	: float ;
default_slope_fall	: float ;
default_slope_rise	: float ;
default_wire_load	: complex_string ;
default_wire_load_area	: float ;
default_wire_load_capacitance	: float ;
default_wire_load_mode	: enum_string ;
default_wire_load_resistance	: float ;
default_wire_load_selection	: complex_string ;
em_temp_degradation_factor	: float ;
default_threshold_voltage_group	: complex_string ;

Refer to the *Legend Table* for the attribute types and their values.

ElectroMigrationGroup

```
electromigration( [ complex_string ] ) {
   related_pin      : simple_string_list ;
   related_bus_pins : simple_string_list ;
   EmMaxToggleRateGroup
}
```

Ignored Attributes	when index_1 index_2 values

Refer to the *Legend Table* for the attribute types and their values.

EmLutTemplateGroup

```
em_lut_template( complex_string ) {
     variable_1 : enum_string ;
     variable_2 : enum_string ;
     index_1( float_list );
     index_2( float_list );
}
```

Ignored Attributes	variable_3

Refer to the *Legend Table* for the attribute types and their values.

EmMaxToggleRateGroup

```
em_max_toggle_rate( complex_string ){
    TimingData
}
```

Ignored Attributes	poly_convert threshold

FallConstraintGroup

```
fall_constraint( complex_string ){
    TimingData
}
```

FallNetDelayGroup

```
fall_net_delay( complex_string ){
  TimingData
}
```

FallPowerGroup

```
fall_power( complex_string ){
    TimingData
}
```

Ignored Attributes	poly_convert

FallPropagationGroup

```
fall_propagation( complex_string ){
    TimingData
}
```

FallTransitionDegradationGroup

```
fall_transition_degradation( complex_string ){
   TimingData
}
```

FallTransitionGroup

```
fall_transition( complex_string ){
   TimingData
}
```

InputVoltageGroup

```
input_voltage ( complex_string ){
```

vih	: float ;
vil	: float ;
vimax	: float ;
vimin	: float ;
	: float ;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

InterconnectDelayGroup

```
interconnect_delay( complex_string ){
   TimingData
}
```

Ignored Attributes	poly_convert

InternalPowerGroup

```
internal_power ( [ complex_string ] ){
```

equal_or_opposite_output	: simple_string_list ;
related_pin	: simple_string_list ;
related_bus_pins	: simple_string_list ;
when	: boolexpr ;
related_falling_pin	: simple_string_list ;
related_rising_pin	: simple_string_list ;
related_inputs	: simple_string_list ;
related_outputs	: simple_string_list ;
falling_together_group	: simple_string_list ;
rising_together_group	: simple_string_list ;
switching_together_group	: simple_string_list ;
switching_interval	: float ;
values(float_iter);	
index_1(float_list);	
index_2(float_list);	
index_3(float_list);	
power_level	: complex_string ;
related_pg_pin	: simple_string_list ;
FallPowerGroup	
PowerGroup	
RisePowerGroup	

```
}
```

Ignored Attributes	equal_or_opposite_output_net_capacitance related_input

Ignored Groups	domain

Refer to the *Legend Table* for the attribute types and their values.

LeakagePowerGroup

```
leakage_power ( [ complex_string ] ){
```

`when`	`: boolexpr ;`
`value`	`: double ;`
`power_level`	`: complex_string ;`
`related_pg_pin`	`: simple_string_list ;`
PowerGroup	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

LuTableTemplateGroup

```
lu_table_template( complex_string ){
```

`variable_1`	`: enum_string ;`
`variable_2`	`: enum_string ;`
`variable_3`	`: enum_string ;`
`index_1( float_list );`	
`index_2( float_list );`	
`index_3( float_list );`	

```
}
```

Ignored Attributes	index_4 variable_4
Ignored Groups	domain

Refer to the *Legend Table* for the attribute types and their values.

LutGroup

```
lut( complex_string ){
```

`input_pins`	`: boolexpr ;`

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

MemoryGroup

```
memory( [ complex_string ] ){
```

`address_width`	`: float ;`
`column_address`	`: "list of int` **or** `int:int"`
`row_address`	`: "list of int` **or** `int:int"`
`type`	`: enum_string ;`
`word_width`	`: float ;`

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

MemoryReadGroup

```
memory_read( [ complex_string ] ){
```

address	: simple_string_list ;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

MemoryWriteGroup

```
memory_write( [ complex_string ] ){
```

address	: simple_string_list ;
clocked_on	: boolexpr ;
enable	: boolexpr ;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

MinimumPeriodGroup

```
minimum_period( [ complex_string ] ){
```

constraint	: float ;
sdf_cond	: complex_string ;
when	: boolexpr ;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

MinPulseWidthGroup

```
min_pulse_width( [ complex_string ] ){
```

constraint_high	: float ;
constraint_low	: float ;
sdf_cond	: complex_string ;
when	: boolexpr ;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

ModeDefinitionGroup

```
mode_definition( complex_string ){
    mode_value( complex_string )   {
```

when	: boolexpr ;
sdf_cond	: complex_string ;

```
    }
}
```

Refer to the *Legend Table* for the attribute types and their values.

OperatingConditionsGroup

```
operating_conditions( complex_string ){
```

`process`	`: float ;`
`temperature`	`: float ;`
`tree_type`	`: enum_string ;`
`voltage`	`: float ;`
`power_rail( complex_string, float );`	
`calc_mode`	`: complex_string ;`

```
}
```

Ignored Attributes	parameter1/2/3/4/5

Refer to the *Legend Table* for the attribute types and their values.

OutputVoltageGroup

```
output_voltage( complex_string ){
```

vol	: float ;
voh	: float ;
vomin	: float ;
vomax	: float ;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

PolyTemplateGroup

```
poly_template( complex_string ){
```

`variables( enum_string_list );`
`mapping( simple_string, simple_string );`
`variable_1_range( float, float );`
`variable_2_range( float, float );`
`variable_3_range( float, float );`
`variable_4_range( float, float );`
`variable_5_range( float, float );`
`variable_6_range( float, float );`
`variable_7_range( float, float );`
DomainGroup

```
}
```

Ignored Attributes	orders input_noise_height input_noise_width input_peak_time_ratio variable_1_range variable_n_range

Refer to the *Legend Table* for the attribute types and their values.

PowerGroup

```
power( complex_string ){
```

related_pin	: simple_string_list ;
when	: boolexpr ;
TimingData	

```
}
```

Ignored Attributes	poly_convert

Refer to the *Legend Table* for the attribute types and their values.

PowerLutTemplateGroup

```
power_lut_template( complex_string ){
```

`variable_1`	`: complex_string ;`
`variable_2`	`: complex_string ;`
`variable_3`	`: complex_string ;`
`index_1( float_list );`	
`index_2( float_list );`	
`index_3( float_list );`	

```
}
```

Ignored Groups	domain

Refer to the *Legend Table* for the attribute types and their values.

PowerPolyTemplateGroup

```
power_poly_template( complex_string ){
```

`variables( enum_string_list );`
`mapping( simple_string, simple_string );`
`variable_1_range( float, float );`
`variable_2_range( float, float );`
`variable_3_range( float, float );`
`variable_4_range( float, float );`
`variable_5_range( float, float );`
`variable_6_range( float, float );`
`variable_7_range( float, float );`
DomainGroup

```
}
```

Ignored Attributes	variable_n_range

Refer to the *Legend Table* for the attribute types and their values.

PowerSupplyGroup

```
power_supply( [ complex_string ] ){
```

`default_power_rail`	`: complex_string ;`
`power_rail( complex_string, float );`	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

RetainFallSlewGroup

```
retain_fall_slew ( complex_string ){
   TimingData
}
```

RetainingFallGroup

```
retaining_fall ( complex_string ){
  TimingData
}
```

RetainingRiseGroup

```
retaining_rise ( complex_string ){
    TimingData
}
```

RetainRiseSlewGroup

```
retain_rise_slew ( complex_string ){
  TimingData
}
```

RiseConstraintGroup

```
rise_constraint ( complex_string ){
  TimingData
}
```

RiseNetDelayGroup

```
rise_net_delay ( complex_string ){
  TimingData
}
```

RisePowerGroup

```
rise_power ( complex_string ){
  TimingData
}
```

RisePropagationGroup

```
rise_propagation ( complex_string ){
  TimingData
}
```

RiseTransitionDegradationGroup

```
rise_transition_degradation ( complex_string ){
  TimingData
}
```

RiseTransitionGroup

```
rise_transition ( complex_string ){
  TimingData
}
```

RoutingTrackGroup

```
routing_track ( complex_string ){
```

tracks	: float ;
total_track_area	: float ;

```
}
```

Ignored Attributes	short

Refer to the *Legend Table* for the attribute types and their values.

ScalingFactorGroup

```
scaling_factors ( complex_string ){
  ScalingFactorData
}
```

TimingRangeGroup

```
tming_range ( complex_string ){
```

`faster_factor`	`: float ;`
`slower_factor`	`: float ;`

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

TLatchGroup

```
tlatch ( complex_string ){
```

`edge_type`	`: enum_string ;`
`tdisable`	`: Boolean ;`

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

TypeGroup

```
type ( complex_string ){
```

base_type	: complex_string ; //array type is only supported
bit_from	: float ;
bit_to	: float ;
bit_width	: float ;
data_type	: complex_string ; //bit type is only supported
downto	: Boolean ;

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

WireLoadGroup

```
wire_load ( complex_string ){
```

`area`	`: float ;`
`capacitance`	`: float ;`
`resistance`	`: float ;`
`slope`	`: float ;`
`fanout_length (float, float, float, float, float) ;` or `fanout_length (float, float) ;`	
InterconnectDelayGroup	

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

WireLoadSelectionGroup

```
wire_load_selection ( [ complex_string ]  ){
```

`wire_load_from_area( float, float, complex_string );`

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

WireLoadTableGroup

```
wire_load_table ( complex_string ){
```

`fanout_area( float, float ) ;`
`fanout_capacitance( float, float ) ;`
`fanout_length( float , float ) ;`
`fanout_resistance( float, float ) ;`

```
}
```

Refer to the *Legend Table* for the attribute types and their values.

Appendix_IA

The following table displays the list of attributes that are ignored by the SpyGlass Library Compiler:

Scope	**Ignored Attributes**
BundleGroup	is_three_state power_gating_pin
BusGroup	is_three_state
CellGroup	driver_waveform driver_waveform_fall driver_waveform_rise em_temp_degradation_factor failure_rate pin_name_map sensitization_master antenna_diode_type auxiliary_pad_cell base_name drive_type fpga_cell_type fpga_domain_style io_type is_decap_cell is_filler_cell is_pll_cell is_tap_cell ocv_arc_depth ocv_derate_distance_group ocv_derate_group physical_connection power_cell_type resource_usage scan_group slew_type
DomainGroup	index_1 index_2 index_3 variable_1 variable_2 variable_3 variable_n_range

ElectroMigrationGroup	when index_1 index_2 values
EmLutTemplateGroup	variable_3
EmMaxToggleRateGroup	poly_convert threshold
FallPowerGroup	poly_convert
InterconnectDelayGroup	poly_convert
InternalPowerGroup	equal_or_opposite_output_net_capacitance related_input
LibraryGroup	preferred_input_pad_voltage resistance_unit base_curve_type capacitance_conversion_factor curve_x curve_y default_fpga_isd default_ocv_derate_distance_group default_ocv_derate_group default_part define_group dist_con_factor dist_conversion_factor distance_unit fpga_domain_style fpga_isd fpga_technology k_temp_rise_propogation k_temp_rise_wire_resistance ocv_arc_depth ocv_derate scan_group va_parameters
LuTableTemplateGroup	index_4 variable_4

ModelGroup	em_temp_degradation_factor failure_rate timing_model_type auxiliary_pad_cell base_name drive_type fpga_cell_type io_type power_cell_type resource_usage scan_group slew_type
OperatingConditionsGroup	parameter
PGPinGroup	bias_connection bus_hold_function has_pass_gate open_drain_function open_source_function physical_connection pull_down_function pull_up_function resistive_0_function resistive_1_function resistive_function

PinGroup	bus_hold_function is_three_state open_drain_function open_source_function power_gating_pin pull_down_function pull_up_function resistive_0_function resistive_1_function resistive_function alive_during_partial_power_down antenna_diode_related_ground_pins antenna_diode_related_power_pins bias_connection data_in_type has_pass_gate is_pll_feedback_pin is_pll_output_pin is_pll_reference_pin is_unbuffered physical_connection pulse_clock
PolyTemplateGroup	orders input_noise_height input_noise_width input_peak_time_ratio variable_1_range variable_n_range
PowerPolyTemplateGroup	variable_n_range
PowerGroup	poly_convert
ScaledCellGroup	failure_rate
TimingGroup	interdependence_id pin_name_map sensitization_master steady_state_resistance_above_high steady_state_resistance_below_low steady_state_resistance_high steady_state_resistance_low tied_off wave_fall wave_rise cv_arc_depth

RoutingTrackGroup	short
ScaledCellGroup	scan_group
ScalingFactorData	k_temp_rise_propogation k_temp_rise_wire_resistance
capacitance	coefs orders
fall_capacitance	coefs orders
fpga_condition_value	fpga_arc_condition
fpga_isd	drivepability io_type slew
hyperbolic_noise_above_high	area_coefficient height_coefficient width_coefficient area_coefficient height_coefficient width_coefficient area_coefficient height_coefficient width_coefficient area_coefficient height_coefficient width_coefficient
iv_lut_template	index_1 variable_1
lower	coefs orders variable_1_range variable_2_range variable_n_range
max_trans	coefs orders variable_1_range variable_2_range variable_n_range
maxcap_lut_template	index_1 index_2 variable_1 variable_2

maxtrans_lut_template	index_1 index_2 index_3 variable_1 variable_2 variable_3
noise_lut_template	index_1 index_2 variable_1 variable_2
output_current_template	index_1 index_2 index_3 variable_1 variable_2 variable_3
part	default_step_level max_count num_blockrams num_cols num_ffs num_luts num_rows pin_count valid_speed_grade valid_step_levels
propagation_lut_template	index_1 index_2 index_3 variable_1 variable_2 variable_3
receiver_capacitance1_fall	index_1 ndex_2 values
receiver_capacitance1_rise	index_1 index_2 values
receiver_capacitance2_fall	index_1 index_2 values

receiver_capacitance2_rise	index_1 index_2 values
rise_capacitance	coefs orders
speed_grade	fpga_isd step_level
upper	coefs orders variable_1_range variable_2_range variable_n_range
user_parameters	parameteri
vector	index_1 index_2 index_3 reference_time values

Appendix_IG

The following table displays the list of groups that are ignored by the SpyGlass Library Compiler:

Scope	Ignored Groups
BundleGroup	ccsn_last_stage fall_capacitance_range hyperbolic_noise_above_high hyperbolic_noise_below_low hyperbolic_noise_high hyperbolic_noise_low max_cap max_trans pin_capacitance receiver_capacitance rise_capacitance_range tlatch
BusGroup	ccsn_last_stage hyperbolic_noise_above_high hyperbolic_noise_below_low hyperbolic_noise_high hyperbolic_noise_low max_cap max_trans pin_capacitance receiver_capacitance tlatch
CellGroup	dc_current dynamic_current intrinsic_parasitic leakage_current cell_based_variation critical_area_table edif_name fpga_condition fpga_isd functional_yield_metric gate_leakage ocv_derate retention_condition
InternalPowerGroup	domain

LibraryGroup	base_curves ccs_lu_table_template ccs_timing_base_curve ccs_timing_base_curve_template compact_lut_template iv_lut_template maxcap_lut_template maxtrans_lut_template noise_lut_template normalized_driver_waveform output_current_template pg_current_template propagation_lut_template sensitization user_parameters cont_layer critical_area_lut_template dc_current_template device_layer faults_lut_template model ocv_table_template output_current part pg_current_template poly_layer routing_layer timing
LuTableTemplateGroup	domain

PinGroup	ccsn_first_stage ccsn_last_stage fall_capacitance_range hyperbolic_noise_above_high hyperbolic_noise_below_low hyperbolic_noise_high hyperbolic_noise_low input_signal_swing max_cap max_trans output_signal_swing pin_based_variation pin_capacitance receiver_capacitance rise_capacitance_range electromigration pin_based_variation
PowerLutTemplateGroup	domain

TimingGroup	cell_fall_to_pct cell_rise_to_pct compact_ccs_fall compact_ccs_rise compressed_ccs_timing_rise noise_immunity_above_high noise_immunity_below_low noise_immunity_high noise_immunity_low output_current_fall output_current_rise propagated_noise_height_above_high propagated_noise_height_below_low propagated_noise_height_high propagated_noise_height_low propagated_noise_peak_time_ratio_above_high propagated_noise_peak_time_ratio_below_low propagated_noise_peak_time_ratio_high propagated_noise_peak_time_ratio_low propagated_noise_width_above_high propagated_noise_width_below_low propagated_noise_width_high propagated_noise_width_low receiver_capacitance1_fall receiver_capacitance1_rise receiver_capacitance2_fall receiver_capacitance2_rise steady_state_current_high steady_state_current_low steady_state_current_tristate timing_based_variation compact_ccs_retain_fall compact_ccs_retain_rise compact_ccs_rise ocv_sigma_cell_fall ocv_sigma_cell_rise
ModelGroup	edif_name fpga_condition fpga_domain_style fpga_isd
fall_capacitance_range	lower upper
fpga_condition	fpga_condition_value

output_current_fall	vector
output_current_rise	vector
part	fpga_isd speed_grade
pin_capacitance	capacitance fall_capacitance fall_capacitance_range rise_capacitance rise_capacitance_range
receiver_capacitance	receiver_capacitance1_fall receiver_capacitance1_rise receiver_capacitance2_fall receiver_capacitance2_rise
rise_capacitance_range	lower upper

List of Topics