Liberty Syntax for
Switch Cell Modeling
Application Note
Version 1.0

Copyright Notice and Proprietary Information

Copyright Notice and Proprietary Information

Copyright © 2007 Synopsys, Inc. All rights reserved. This software and documentation contain confidential and proprietary information that is the property of Synopsys, Inc. The software and documentation are furnished under a license agreement and may be used or copied only in accordance with the terms of the license agreement. No part of the software and documentation may be reproduced, transmitted, or translated, in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without prior written permission of Synopsys, Inc., or as expressly provided by the license agreement.

Right to Copy Documentation

The license agreement with Synopsys permits licensee to make copies of the documentation for its internal use only. Each copy shall include all copyrights, trademarks, service marks, and proprietary rights notices, if any. Licensee must assign sequential numbers to all copies. These copies shall contain the following legend on the cover page:

“This document is duplicated with the permission of Synopsys, Inc., for the exclusive use of __________________________ and its employees. This is copy number _______."

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader’s responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Registered Trademarks (®)

Synopsys, AMPS, Cadabra, CATS, CRITIC, CSim, Design Compiler, DesignPower, DesignWare, EPIC, Formality, HSIM, HSPICE, iN-Phase, i-Sync, Leda, MAST, ModelTools, NanoSim, OpenVera, PathMill, Photolynx, Physical Compiler, PrimeTime, SiVL, SNUG, SolvNet, System Compiler, TetraMAX, VCS, Vera, and YIELDirector are registered trademarks of Synopsys, Inc.

 Trademarks (™)

AFGen, Apollo, Astro, Astro-Rail, Astro-Xtalk, Aurora, AvanWaves, Columbia, Columbia-CE, Cosmos, CosmosEnterprise, CosmosLE, CosmosScope, CosmosSE, DC Expert, DC Professional, DC Ultra, Design Analyzer, Design Vision, DesignerHDL, Direct Silicon Access, Discovery, Encore, Galaxy, HANEX, HDL Compiler, Hercules, Hierarchical Optimization Technology, HSIM™, HSPICE-Link, i-Virtual Stepper, iN-Tandem, Jupiter, Jupiter-OS, JupiterXT, JupiterXT-ASIC, Liberty, Libra-Passport, Library Compiler, Magellan, Mars, Mars-Xtalk, Milkyway, ModelSource, Module Compiler, Planet, Planet-PL, Polaris, Power Compiler, Raphael, Raphael-NEC, Saturn, Scirocco, Scirocco-i, Star-RCXT, Star-SimXT, Taurus, TSUPREM-4, VCS Express, VCSI, VHDL Compiler, VirSim, and VMC are trademarks of Synopsys, Inc.

Service Marks (℠)

MAP-in, SVP Café, and TAP-in are service marks of Synopsys, Inc.

SystemC is a trademark of the Open SystemC Initiative and is used under license. ARM and AMBA are registered trademarks of ARM Limited.

Saber is a registered trademark of SabreMark Limited Partnership and is used under license. All other product or company names may be trademarks of their respective owners.
1 Introduction

Advanced low-power design methodologies such as multivoltage and power gating require new library cells that need additional modeling attributes in order to drive implementation tools during the library cell selection. Library Compiler is continually enhancing Liberty syntax with attributes and statements to support tool features.

When design partitions are shut down, the standard cells are disconnected from their power or ground sources, thus reducing leakage current flow. This is achieved by adding power switches between the cells and the power or ground network. Turning off the power supply to inactive logic by using an internal switch cells is an effective technique for leakage reduction.

Power switches can be organized throughout the design in either a coarse-grain or a fine-grain switch configuration. This document highlights the Liberty syntax that applies only to coarse-grain switch cells.

In the future, all switch syntax modeling enhancements will be based on the Liberty power and ground pin syntax. For more information about power and ground pin syntax, see the Liberty Syntax for Power and Ground (PG) Pin Standard Cells Application Note, version A-2007.12.
2 Switch Cell Modeling

With coarse-grain designs, you place one or more power switch cells in each block or domain that need to be turned off. The power switches connect the main power rails of the chip to a set of virtual rails. When the power switches are turned off, the connected virtual power rail is turned off along with the related standard cells.

2.1 Coarse-Grain Switch Cells

There are two types of coarse-grain switch cells: header switch cells, which control power nets based on a PMOS transistor, and footer switch cells, which control ground nets based on a NMOS transistor.

Coarse-grain switch cells have the following characteristics:

- At least one output virtual power or virtual ground, one regular input power and ground pin, and one switch input pin is required. There is no limit to the number of pins on a coarse-grain switch cell.
- The logic condition under which the cell turns off. This is modeled with a switching function based on signal pins and a power-down function based on power pins.
- Acknowledge output pins, which are output pins whose signal is used to propagate the switch signal. Timing is also propagated from the input switch pin to the acknowledge output pins.
- Multiple input signal pins and multiple acknowledge output pins.
- Steady state current (I/V) information to determine the resistance value when the switch is on.
2.2 Syntax
The following example shows the coarse-grain switch cell syntax:

library(<coarse_grain_library_name>) {
...

lu_table _template ( template_name ) /* Please refer to section 2.3.1 */
  variable_1 : input_voltage;
  variable_2 : output_voltage;
  index_1 ( <float>, ... );
  index_2 ( <float>, ... );
}
...

cell(<cell_name>) {
  switch_cell_type : coarse_grain; /* Please refer to section 2.4.1 */
...
  pg_pin ( <VDD/VSS pin name> ) {
    pg_type : primary_power;
    direction : input;
    ...
  }
  /* virtual power and ground pins use "switch_function" to describe the logic
to shut off the attached design partition */
  pg_pin ( <virtual VDD/VSS pin name>) {
    pg_type : internal_power | internal_ground;
    direction: output;
    ...
    switch_function : "<function_string>"; /* Please refer to section 2.5.1 */
    pg_function : "<function_string>"; /* Please refer to section 2.5.2 */
  }
  dc_current ( <dc_current_name> ) { /* Please refer to section 2.4.2 */
    related_switch_pin : <input_pin_name>; /* Please refer to section 2.4.3 */
    related_pg_pin : <VDD pin name>; /* Please refer to section 2.4.4 */
    related_internal_pg_pin : <Virtual VDD>; /* Please refer to section 2.4.5 */
    values("<float>, ...");
  }
  pin (<input_pin_name>) { 
    direction : input;
    switch_pin : true; /* Please refer to section 2.5.3 */
    ...
  }
  /* acknowledge output pin uses "function" to represent the propagated
switching signal */
  pin(<acknowledge_output_pin_name>) { 
    function : "<function_string>";
    power_down_function : "function_string"; /* Please refer to section 2.5.5 */
    direction : output;
    ...
  } /* end pin group */
} /* end cell group */
2.3 Library-Level Groups

2.3.1 lu_table_template Group
The library level lu_table_template group is used to model the templates for the steady state current information later modeled inside the dc_current group. The input_voltage variable specifies the input voltage values of the switch pin and the output_voltage variable specifies the voltage difference between the related_pg_pin and the related_internal_pg_pin. The dc_current table used for steady state current modeling must be defined at the cell level.

2.4 Cell Level Groups and Attributes
The following attribute groups and attributes can be applied at the cell level.

2.4.1 switch_cell_type Simple Attribute
The switch_cell_type attribute provides a complete description of the switch cell so that Library Compiler does not need to infer the switch cell type through the cell information. The only type currently supported enumerated type value is coarse_grain.

2.4.2 dc_current group
The cell level dc_current group is used to model the steady state current information with variables specified using the lu_table_template group specified at the library level. The table is used to specify the DC current through the output pin (generally the related_internal_pg_pin) of the cell in the current units specified at the library level using the current_unit attribute. The table includes other simple attributes namely the related_switch_pin, related_pg_pin, and related_internal_pg_pin attributes, which are described in the next sections.

2.4.3 related_switch_pin Simple Attribute
The related_switch_pin string attribute is used to specify the name of the related switch pin for the coarse-grain switch cell. It is a simple attribute defined inside the dc_current group.

2.4.4 related_pg_pin Simple Attribute
The related_pg_pin string attribute is used to specify the name of the power and ground pin that represents the VDD or VSS power source. It is a simple attribute defined inside the dc_current group.

2.4.5 related_internal_pg_pin Simple Attribute
The related_internal_pg_pin string attribute is used to specify the name of the power and ground pin that represents the virtual VDD or virtual VSS power source. It is a simple attribute defined inside the dc_current group.

2.5 Pin Level attributes
The following attributes are pin-level attributes for switch cells.
2.5.1 switch_function simple Attribute
The switch_function string attribute identifies the condition when the switch is turned off by the input switch_pin. For a coarse-grain switch cell, the switch_function attribute is defined on power and ground pins and identifies the signal pins which can turn the power pin ON.

2.5.2 pg_function Simple Attribute
The pg_function attribute is used for the switch cells internal and virtual power output pins to represent the propagated power level through the switch as a function of the input power and ground pins. In coarse grain switch cells this attribute is specified inside the pg_pin group.

2.5.3 Switch_pin Simple Attribute
The switch_pin attribute is a pin level boolean attribute. When it is set to true, it is used to identify the pin as the switch pin of a switch cell.

2.5.4 function Simple Attribute
The function attribute in a pin group defines the value of an output pin or inout pin in terms of the input pins or inout pins in the cell group or model group. The function attribute describes the boolean function of only non sleep input signal pins.

2.5.5 power_down_function Simple Attribute
The power_down_function string attribute is used to identify the boolean condition under which the cell’s signal output pin is switched off (when the cell is in off mode due to the external power pin states). If the power_down_function is set to 1 then X is assumed on the pin.

Note: In the absence of certain signal pins being associated to a PG-pin using the related_power_pin and the related_ground_pin attributes, Library Compiler will associate those signal pins to the primary power rails defined on that cell and inform you of such associations with the following warning:

Warning: Line 1641, Connect pin 'XYZ' to the default power pg_pin 'VDD'. (LBDB-725)

2.6 Library Compiler Checking Rules
Library Compiler performs the following checks for switch cells and quits with an error if the conditions are not satisfied:

- The switch_function can only contain the switch pins where the switch pin is the input pin (set by the switch_pin attribute) set to true. (Error)
- The function attribute may also contain switch pins for a coarse grain switch cell.

Note: Library Compiler does not check whether the timing is modeled with Composite Current Source (CCS) timing or Nonlinear Delay Model (NLDM) timing, and it does not check whether the power should be modeled with CCS power or NLDM power.
A cell is identified as a coarse grain switch cell when the following conditions for the model information inside the body of the cell are met:

- It must define `switch_cell_type` with the `coarse_grain` enumerated value to be identified as a switch cell. Any other value will generate LC Error.
- It must define the `pg_pin` level `switch_function` attribute to identify the control logic of its switch pins.
- It must have at least one switch pin.
- It must have at least one internal power or ground pin and one regular power and ground pin (that is, a virtual VSS `pg_pin` and a VSS `pg_pin` or a virtual VDD `pg_pin` and a VDD `pg_pin`). Each of these internal output power pins must have a `pg_function` definition containing parameters from the input power pins.
- The `pg_function` must contain only the input power and ground pins.
- The `related_switch_pin` value can be either an internal pin or a switch pin.
3 Examples

Switch cells can be described in different levels of detail. The following sections provide examples of switch cells.

3.1 Simple Coarse-Grain Header Switch Cell Example

Figure 1 Simple Coarse-Grain Switch Cell

```plaintext
library (simple_coarse_grain_lib) {
    ...
    current_unit : 1mA;
    ...
    voltage_map (VDD, 1.0);
    voltage_map (VVDD, 0.8);
    voltage_map (VSS, 0.0);
    operating_conditions (XYZ) {
        process : 1.0;
        voltage : 1.0;
        temperature : 25.0;
    }
    default_operating_conditions : XYZ;
    lu_table_template (ivt1) {
        /* index_1 specifies the input_voltage value specified at the switch pin
        with respect to the ground */
        /* index_2 specifies the voltage value specified at the
        related_internal_pg_pin with respect to the ground */
        variable_1 : input_voltage;
        variable_2 : output_voltage;
        index_1 ("0.1, 0.2, 0.4, 0.8, 1.0");
        index_2 ("0.1, 0.2, 0.4, 0.8, 1.0");
    }
    ...
    cell (Simple_CG_Switch) {
        ...
        switch_cell_type : coarse_grain;
    }
    pg_pin (VDD) {
        pg_type : primary_power;
        direction : input;
    }
    ...
}
```
voltage_name : VDD;
}

pg_pin ( VVDD ) {
  pg_type : internal_power;
  voltage_name : VVDD;
  direction : output;
  switch_function : "SLEEP";
  pg_function : "VDD";
}

pg_pin ( VSS ) {
  pg_type : primary_ground;
  direction : input;
  voltage_name : VSS;
}

/* I/V curve information */
dc_current ( ivt1 ) {
  related_switch_pin : SLEEP; /* control pin */
  related_pg_pin : VDD;      /* source */
  related_internal_pg_pin : VVDD;   /* drain */
  values( "0.010, 0.020, 0.030, 0.030, 0.030",
           "0.011, 0.021, 0.031, 0.041, 0.051",
           "0.012, 0.022, 0.032, 0.042, 0.052",
           "0.013, 0.023, 0.033, 0.043, 0.053",
           "0.014, 0.024, 0.034, 0.044, 0.054" );
}

pin ( SLEEP ) {
  switch_pin : true;
  capacitance: 1.0;
  related_power_pin : VDD;
  related_ground_pin : VSS;
} /* end pin */
} /* end cell */
} /* end library */
3.2 Complex Coarse-Grain Header Switch Cell Example

Figure 2. Complex Coarse-Grain Switch Cell

library (Complex_CG_lib) {
  …
  current_unit : 1mA;
  …
  voltage_map (VDD, 1.0);
  voltage_map (VnVDD, 0.8);
  voltage_map (VSS, 0.0);
  …
  operating_conditions (XYZ) {
    process : 1.0;
    voltage : 1.0;
    temperature : 25.0;
  }
  default_operating_conditions : XYZ;
  …
  lu_table_template (ivt1) {
    /* index_1 specifies the input_voltage value specified at the switch pin
       with respect to the ground */
    /* index_2 specifies the voltage value specified at the
       related_internal_pg_pin with respect to the ground */
    variable_1 : input_voltage;
    variable_2 : output_voltage;
    index_1 ("0.1, 0.2, 0.4, 0.8, 1.0");
    index_2 ("0.1, 0.2, 0.4, 0.8, 1.0");
  }
  …
  cell (Complex_CG_Switch) {
    …
    switch_cell_type : coarse_grain;
    pg_pin (VDD) {
      pg_type : primary_power;
      voltage_name : VDD;
      direction : input;
    }
pg_pin ( VVDD ) {
    pg_type : internal_power;
    direction : output;
    voltage_name : VVDD;
    switch_function : "SLEEP";
    pg_function : "VDD";
}
pg_pin ( VSS ) {
    pg_type : primary_ground;
    voltage_name : VSS;
    direction : input;
}

/* I/V curve information */
dc_current ( ivt1 ) {
    related_switch_pin : SLEEP; /* control pin */
    related_pg_pin : VDD; /* source power pin */
    related_internal_pg_pin : VVDD; /* drain internal power pin*/
    values(
        "0.010, 0.020, 0.030, 0.040, 0.050", \
        "0.011, 0.021, 0.031, 0.041, 0.051", \
        "0.012, 0.022, 0.032, 0.042, 0.052", \
        "0.013, 0.023, 0.033, 0.043, 0.053", \
        "0.014, 0.024, 0.034, 0.044, 0.054");
}

pin ( SLEEP ) {
    direction : input;
    switch_pin : true;
    capacitance: 1.0;
    related_power_pin : VDD;
    related_ground_pin : VSS;
    ...
}

pin (Y) {
    direction : output;
    function : "SLEEP";
    related_power_pin : VDD;
    related_ground_pin : VSS;
    power_down_function : "!VDD + VSS";
    timing() {
        related_pin : SLEEP;
        ...
    }
} /* end pin group */
} /* end cell group*/
} /* end library group*/
library (Complex_CG_lib) {
  current_unit : 1mA;

  voltage_map (VDD, 1.0);
  voltage_map (VVDD, 0.8);
  voltage_map (VSS, 0.0);

  operating_conditions (XYZ) {
    process : 1.0;
    voltage : 1.0;
    temperature : 25.0;
  }
  default_operating_conditions : XYZ;

  lu_table_template (ivt1) {
    /* index_1 specifies the input_voltage value specified at the switch pin
    with respect to the ground */
    /* index_2 specifies the voltage value specified at the
    related_internal_pg_pin with respect to the ground */
    variable_1 : input_voltage;
    variable_2 : output_voltage;
    index_1 ("0.1, 0.2, 0.4, 0.8, 1.0");
    index_2 ("0.1, 0.2, 0.4, 0.8, 1.0");
  }

  cell (COMPLEX_HEADER_WITH_INTERNAL_SWITCH_PIN) {
    cell_footprint : complex_mtpmos;
    area : 1.0;
    switch_cell_type : coarse_grain;

    pg_pin (VDD) {
      voltage_name : VDD;
      pg_type : primary_power;
      direction : input;
    }
pg_pin(VVDD) {
    voltage_name : VVDD;
    pg_type : internal_power;
    direction : output;
    switch_function : "SLEEP";
}

pg_pin(VSS) {
    voltage_name : VSS;
    pg_type : primary_ground;
    direction : input;
}

dc_current(ivt) {
    related_switch_pin : internal;
    related_pg_pin : VDD;
    related_internal_pg_pin : VVDD;
    values( "0.010, 0.020, 0.030, 0.040, 0.050", \n           "0.011, 0.021, 0.031, 0.041, 0.051", \n           "0.012, 0.022, 0.032, 0.042, 0.052", \n           "0.013, 0.023, 0.033, 0.043, 0.053", \n           "0.014, 0.024, 0.034, 0.044, 0.054");
}

pin(SLEEP) {
    switch_pin : true;
    direction : input;
    capacitance : 1.0;
    related_power_pin : VDD;
    related_ground_pin : VSS;
    ...
}

pin(Y) {
    direction : output;
    function : "SLEEP";
    related_power_pin : VDD;
    related_ground_pin : VSS;
    power_down_function : "!VDD + VSS"
    timing() {
        related_pin : "SLEEP"
    }
}

pin(internal) {
    direction : internal;
    timing() {
        related_pin : "SLEEP"
    }
}

/* end pin group */

/* end cell group */

/* end library group */
3.3 Complex Coarse-Grain Switch Cell with Two Switches in Parallel

![Complex Coarse-Grain Switch Cell with Two Switches in Parallel](image)

**Figure 4.** Complex Coarse-Grain Switch Cell with Two Parallel Switches

```plaintext
library (Complex_CG_lib) {
  ...
  current_unit : 1mA;
  ...
  voltage_map(VDD, 1.0);
  voltage_map(VVDD, 0.8);
  voltage_map(VSS, 0.0);

  operating_conditions(XYZ) {
    process : 1.0;
    voltage : 1.0;
    temperature : 25.0;
  }
  default_operating_conditions : XYZ;

  lu_table_template ( ivt1 ) {
    /* index_1 specifies the input_voltage value specified at the switch pin
    with respect to the ground */
    /* index_2 specifies the voltage value specified at the
    related_internal_pg_pin with respect to the ground */
    variable_1 : input_voltage;
    variable_2 : output_voltage;
    index_1 ( "0.1, 0.2, 0.4, 0.8, 1.0" );
    index_2 ( "0.1, 0.2, 0.4, 0.8, 1.0" );
  }
  ...

  cell (COMPLEX_HEADER_WITH_TWO_PARALLEL_SWITCHES) {
    cell_footprint : complex_mtpmos;
    area : 1.0;
    switch_cell_type : coarse_grain;
  }
}
```
pg_pin(VDD) {
    voltage_name : VDD;
    pg_type : primary_power;
    direction : input;
}

pg_pin(VVDD) {
    voltage_name : VVDD;
    pg_type : internal_power;
    direction : output;
    switch_function : "CTL1 + CTL2";
}

pg_pin(VSS) {
    voltage_name : VSS;
    pg_type : primary_ground;
    direction : input;
}

dc_current(ivt1) {
    related_switch_pin : CTL1;
    related_pg_pin : VDD;
    related_internal_pg_pin : VVDD;
    values("0.010, 0.020, 0.030, 0.040, 0.050",
        "0.011, 0.021, 0.031, 0.041, 0.051",
        "0.012, 0.022, 0.032, 0.042, 0.052",
        "0.013, 0.023, 0.033, 0.043, 0.053",
        "0.014, 0.024, 0.034, 0.044, 0.054");
}

dc_current(ivt1) {
    related_switch_pin : CTL2;
    related_pg_pin : VDD;
    related_internal_pg_pin : VVDD;
    values("0.010, 0.020, 0.030, 0.040, 0.050",
        "0.011, 0.021, 0.031, 0.041, 0.051",
        "0.012, 0.022, 0.032, 0.042, 0.052",
        "0.013, 0.023, 0.033, 0.043, 0.053",
        "0.014, 0.024, 0.034, 0.044, 0.054");
}

pin(CTL1) {
    switch_pin : true;
    direction : input;
    capacitance : 1.0;
    related_power_pin : VDD;
    related_ground_pin : VSS;
...}

pin(CTL2) {
    switch_pin : true;
    direction : input;
    capacitance : 1.0;
    related_power_pin : VDD;
    related_ground_pin : VSS;
...}

pin(CTL1OUT) {
    direction : output;
    function : "CTL1";
    related_power_pin : VDD;
    related_ground_pin : VSS;
    power_down_function : "!VDD + VSS";
    timing() {
        related_pin : "CTL1";
        ...
    }
} /* end pin group */
pin(CTL2OUT) {
  direction : output;
  function : "CTL1";
  related_power_pin : VDD;
  related_ground_pin : VSS;
  power_down_function : "!VDD + VSS";
  timing() {
    related_pin : "CTL2"
  }
}

/* end library group */
4 References


