

Insight Analyzer v5.00 Release Notes

5.00-12024
2021 August 25

Overview

This major version update of Insight Analyzer has significant updates over prior versions in the v4x series. There are new features, behavior changes, and migration aspects.

Contents

Overview	1	Load Netlist.....	10
Updates & Changes	2	Define Power Modes.....	10
Graphical User Interface	2	Define Testbench.....	10
General Look.....	2	Define UPF Controls.....	11
Netlist Loading.....	2	Run Checks.....	11
Devices Setup.....	2	Plugin Applications	11
Power & States Editing.....	3	Analog Floats.....	11
Power / Ground Rails.....	3	Config Options.....	12
Boundary / IO.....	4	Contention.....	12
Clearing Trouble.....	4	Config Options.....	12
State-Based Sweeps.....	4	Domain Crossings.....	12
Topology Review.....	6	Reporting.....	13
Reports & Waiving.....	6	Legacy Compatibility.....	13
Finder Tool.....	6	Domain Leakage.....	14
Schematic Cross Probe.....	7	New Cases.....	14
Setup.....	7	Legacy Compatibility.....	14
Cross Probe.....	7	Fanout.....	14
Working Examples & Search.....	8	Config Options.....	14
Workflows	8	Power Connections.....	14
System with Controls / UPF.....	9	Config Options.....	14
Appropriate Use.....	9	Reference Numbers.....	15
Analog States.....	9	API	15
Appropriate Use.....	9	Licensing	15
How it Works.....	9	Scenario Editing Feature Set.....	16
Workflow Outline.....	10	License File Migration.....	16
		Compatibility	16

Updates & Changes

Graphical User Interface

General Look

The GUI has been re-designed for an improved workflow, and tighter coupling with the internal processing engine. The GUI is now able to show enhanced detail and increased circuit information.

Real time status is available at various points along the workflow, whether the internal engine is idle or running. Pop-up windows have been reduced to only those cases where a transient editing function is being performed.

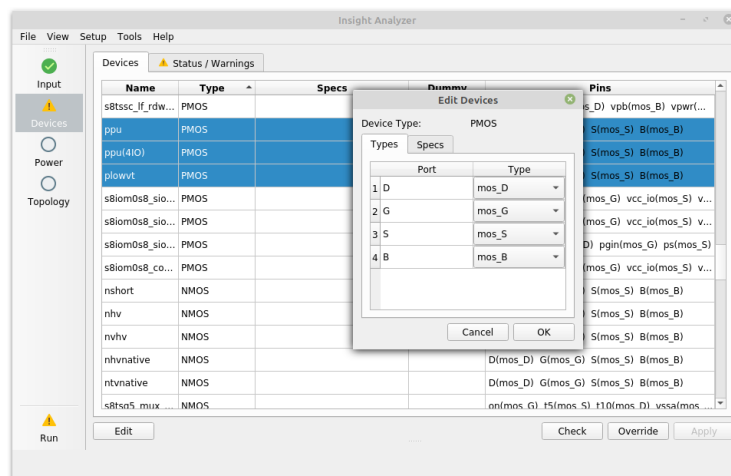
Netlist Loading

The “Netlist” workspace shows file path, library path, options, and a final review after the netlist has been loaded.

The “Supplements” tab has controls for loading parasitic overlay files such as SPF, DSPF, SPEF.

Devices Setup

The “Devices” workview now combines device specs along with useful info such as pin type assignments, in a single table.



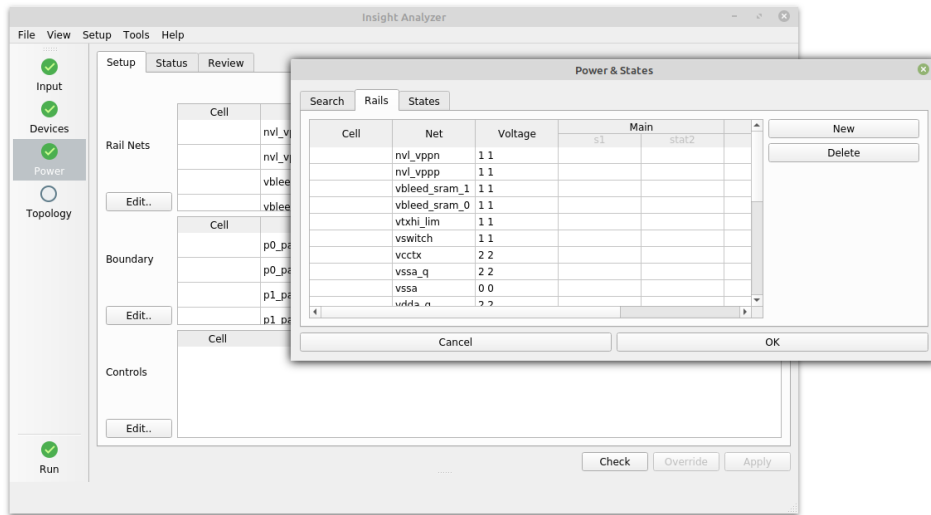
You edit this device table through a modal dialog that supports device spec and pin type editing, together.

If there is trouble detected within the device settings, a second tab will indicate warning status. Click the warning tab, and inside there, find the button for “Override”. This function will override

a snapshot of the current warnings. You would then press “Apply” to apply the settings into the circuit.

Power & States Editing

The GUI provides a complete state editing workspace, where you define relationships between power supplies, boundary signals, and controls.



This workspace combines many aspects of state together in a set of overview tables. Editing is done with pop-up modal dialogs, each dedicated to a part of the overview.

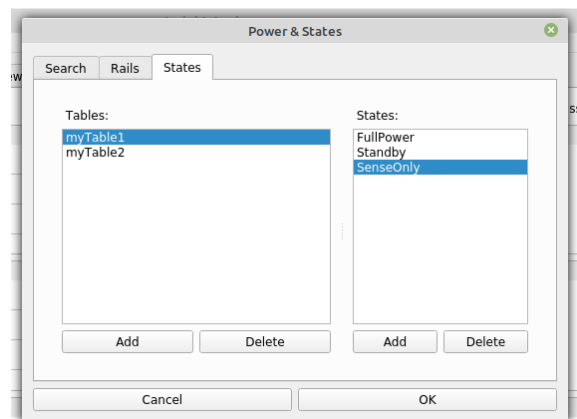
Power / Ground Rails

To begin, you would first work on the top part, the power and ground definitions. In the section “Rail Nets”, press “Edit”, and you will open a dialog for power search, rail values, and creating power states.

In the “Power & States” dialog, the first tab is “Search”. You will use this to find suspected power rails, or power rails by name pattern if needed. From there, you will “Accept” rail names, causing them to appear in the second tab.

The second tab, “Rails”, is a list of the rail names that have been accepted from the previous step. You may add a new rail manually in this list. Here, you will define voltage values for these rails. If needed, you would create a power state table in the third tab.

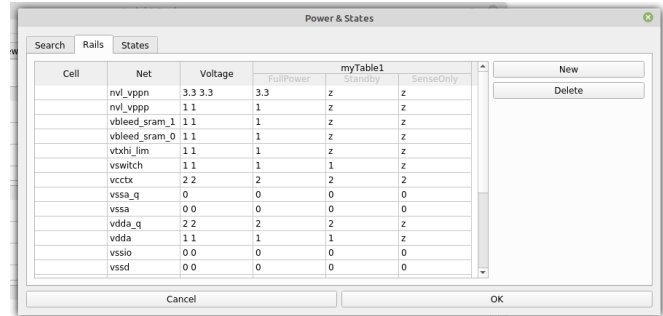
The third tab is “States”. In this tab, under the “Tables” list, use the “Add” button to create an empty table name. Type into the tables list, the name of a table. Then next, in the “States” list, you would press



“Add” to create an empty line, and type a state name into that line.

When done with creating tables and states in the “States” tab, return to the “Rails” tab. There, you will see the rails, voltages, and tables with editing cells for each state.

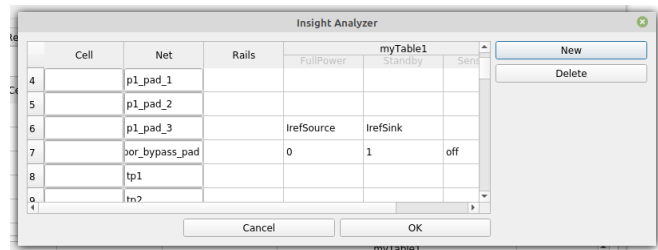
If you are using power state tables (above steps), you will provide specific values in the “Rails” tab. (Known issue: The states must have values entered, otherwise the table will not be saved.) Press “OK” when done editing these values, to apply the edits into the main workview.



Cell	Net	Voltage	FullPower	myTable1 Standby	SenseOnly
nvf_vppn	3.3	3.3	3.3	z	z
nvf_vppp	1.1	1	1	z	z
vbleed_sram_1	1.1	1	1	z	z
vbleed_sram_0	1.1	1	1	z	z
vtxhi_lim	1.1	1	1	z	z
vswitch	1.1	1	1	z	z
vcctx	2.2	2	2	z	z
vssa_q	0	0	0	0	0
vssa	0.0	0	0	0	0
vdda_q	2.2	2	2	z	z
vdda	1.1	1	1	z	z
vssio	0.0	0	0	0	0
vssd	0.0	0	0	0	0

Boundary / IO

Then continuing, you would work on the middle part, the boundary settings. These boundary settings are made in terms of associated rails, voltage values, and / or function such as IrefSink, IrefSource, Vref, or “off”.



Cell	Net	Rails	myTable1 FullPower	myTable1 Standby	Sens
4	p1_pad_1				
5	p1_pad_2				
6	p1_pad_3		IrefSource	IrefSink	
7	por_bypass_pad		0	1	off
8	tp1				
9	ln?				

In the middle section of the main workview, for Boundary, press the “Edit” button. This will open an edit dialog for all top level IO ports.

In the editing table, you may define IO in terms of rail names, hard values, or values and functions per state. The states are derived from the power state table states of the previous steps.

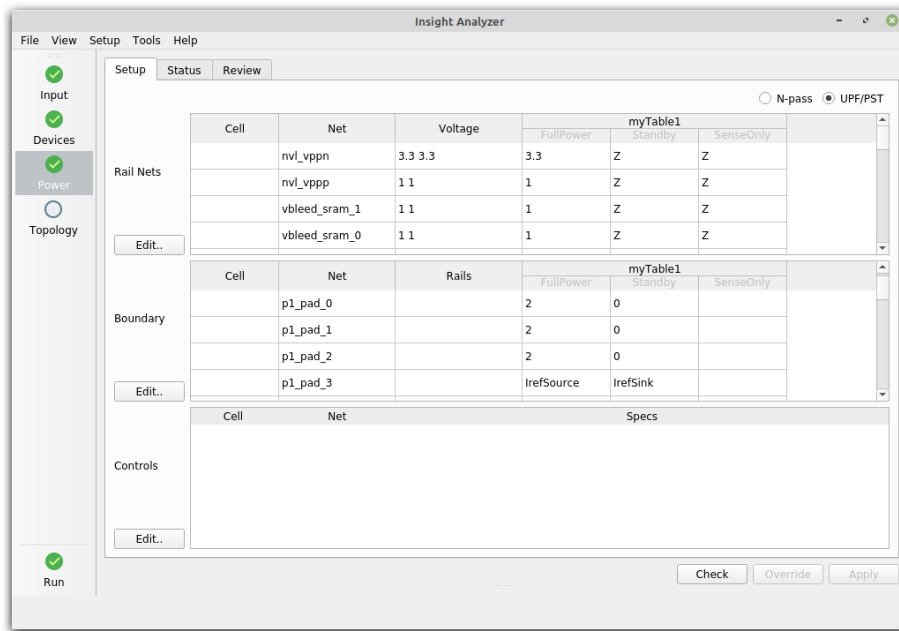
Clearing Trouble

When all power and boundary editing is finished, you can press “Check” to see if there is any obvious trouble with the definitions. The “Status” tab will show a trouble indication, including descriptive text. To clear this trouble, you would either make edits (previous steps), or press “Override”. The override function takes a snapshot of the current trouble, to prevent it from a future recurrence in the current session.

Finally, you would press “Apply” to set all definitions into the circuit database.

State-Based Sweeps

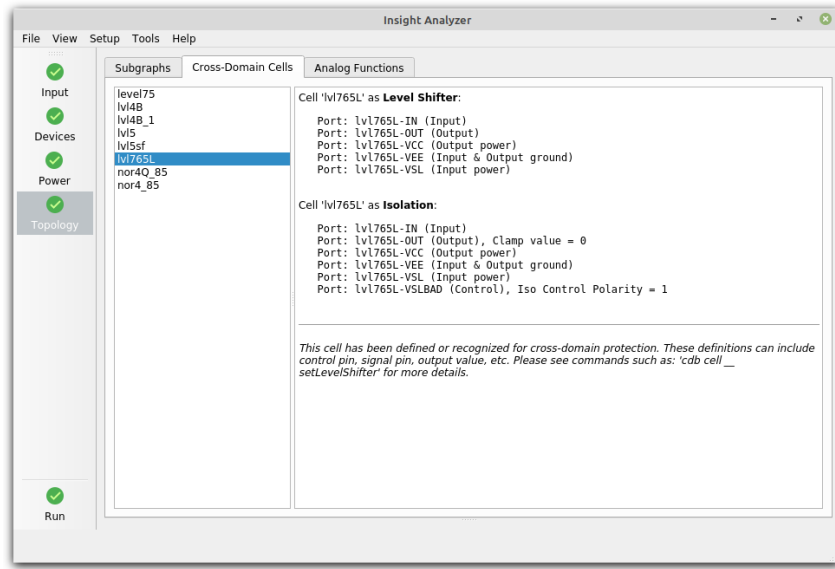
One key enhancement in v5 software is the association of power states along with boundary states. This is a big topic that has impact on a number of checks and applications. In essence, the boundary states described above (such as “V-ref”, “off”, etc) are combined with specific power states (PST, or power state table).



A notable application of this states association is in the Analog Floats checking. In all prior Insight Analyzer versions, floats would be detected in a kind of wildcard fashion, where it was not possible to lock down specific input combinations that were known to be prevented externally. Now, in v5, you define the valid external state combinations, such as an I-ref-source that will be provided as long as analog power is valid.

For more information, please see the script command '[cdb states pstLinks](#)', or the checks that use these states, including "Analog Floats" and "Power Connections".

Topology Review



The workflow icons (left hand side) include a step for “Topology”. Within this step, you would review the topology findings (automatic recognition) for the following:

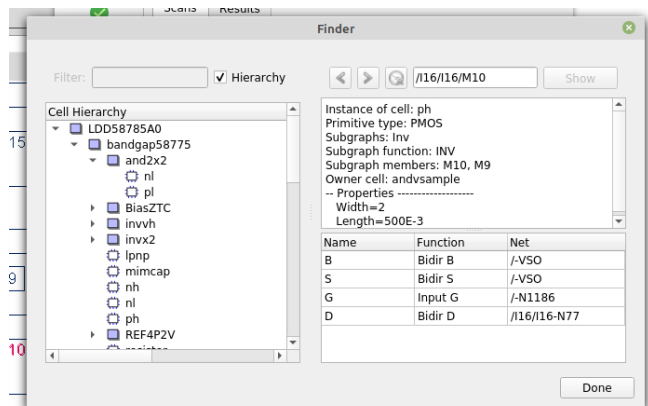
- Subgraphs, such as CMOS structures, cross coupling, etc.
- Cross-domain functions, such as level shifters and isolation cells.
- Analog functions, such as current mirrors, amplifiers, etc.

Reports & Waiving

Finder Tool

To support multiple modes of investigation, the Finder tool has been implemented as a floating window. This can be invoked separately, or from a violations report, to explore further information about circuit objects.

To begin, use menu “Tools > Finder”. A destination text entry is available on the top right, where you can enter text by typing. As you make an entry, a form of command completion will show the valid choices, following a hierarchy path. Press “Show” to show the selected destination net or instance.



In an alternate mode, you can open the finder while viewing a report. In this mode, by clicking in

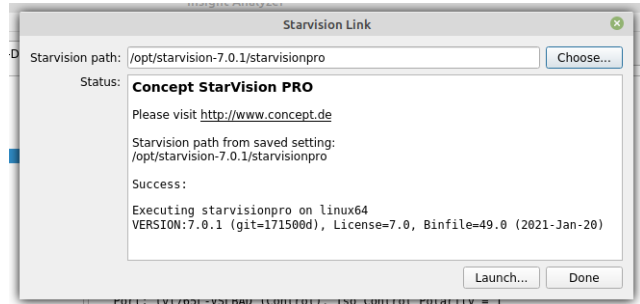
the report view, the Finder will be updated to track your selections and display further details.

This updated Finder reduces the steps needed in older versions of Insight Analyzer, and makes the Finder more useful for probing reports and navigating the circuit.

Schematic Cross Probe

Insight Analyzer is linked with Concept StarvisionPRO, for cross probing outward from the Insight GUI. Clicking on objects from the Finder tool, and from violation reports, invoke corresponding probes in StarvisionPRO schematics.

See also www.concept.de.



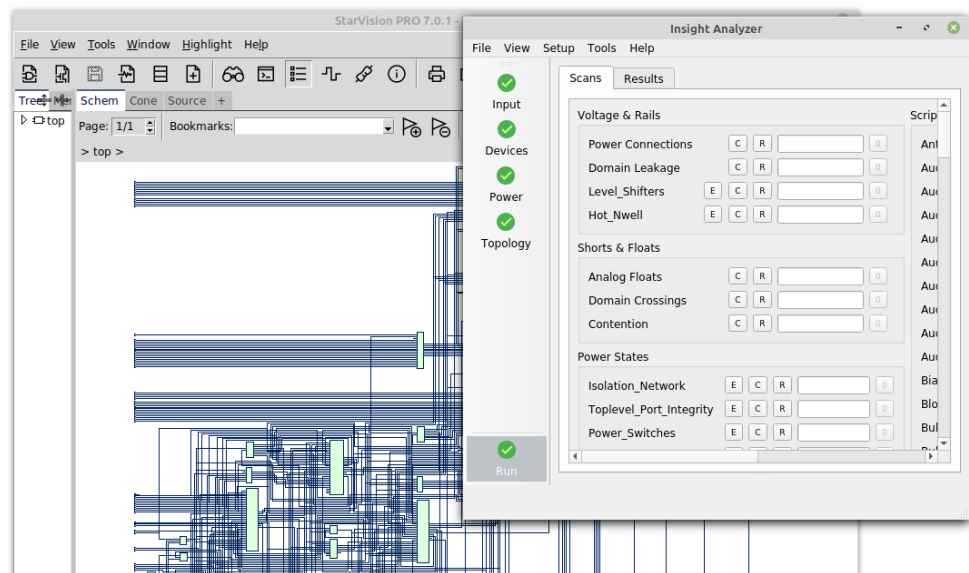
Setup

To initialize the link with StarvisionPRO, use the menu “Tools > StarvisionPRO”, and open the configuration dialog.

In the dialog, use the “Choose” button to navigate to the StarvisionPRO binary. This will run a quick internal test, to make sure the StarvisionPRO path is valid and binary can be run. On success, you should see a confirmation message in the dialog.

Cross Probe

Using the menu “Tools > StarvisionPRO”, open the configuration dialog, and press “Launch”. This will open StarvisionPRO and automatically load the current netlist file.



Please note that internally, the power / ground, and the device models and types, have been parsed from your existing settings within Insight Analyzer. (Known issue: the transfer of settings from Insight to Concept is currently undergoing updates, and some device or power settings may not be completely transferred.)

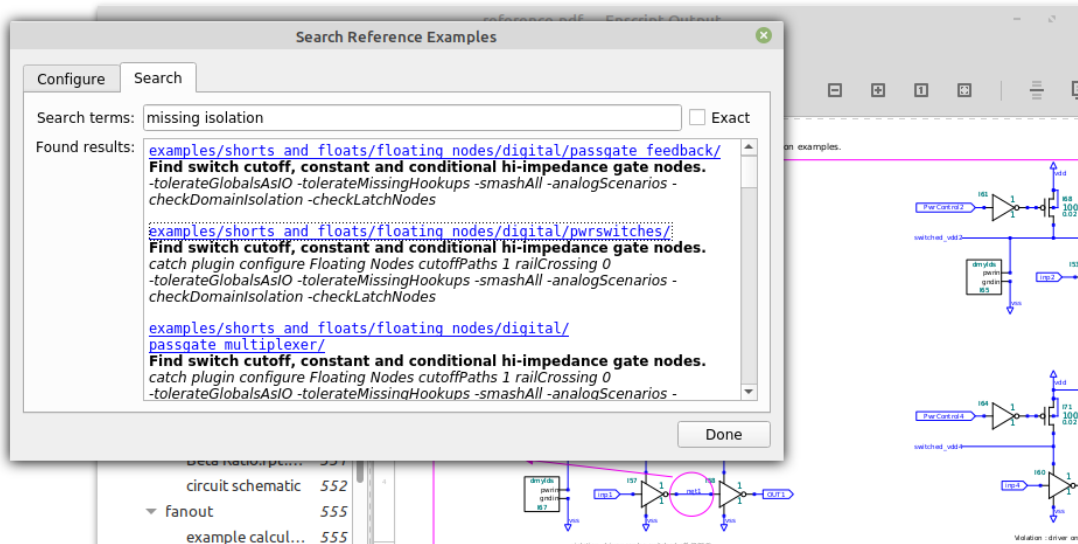
To cross probe, you would click a circuit object in most Insight Analyzer windows such as checking report, report details, or the “Tools > Finder” (above).

Working Examples & Search

To help find examples, there is a searching tool available from menu "Help > Reference Examples". This is a powerful utility that can quickly find answers to complex questions. The searching covers all relevant parts of the example data, including:

- Tcl run script.
- Tcl custom scan.
- UPF setup file.
- Baseline results, report files.

From the search results, you click to open a PDF compilation of example data files.



Workflows

As system checking continues to grow more complex, it has been necessary to break up some of the problem solving into separate parts. This version of Insight Analyzer supports two major modes of state based circuit checking.

System with Controls / UPF

This is the older mode of state aware checking with Insight Analyzer. A circuit is treated as a system, with states and controls defined with a UPF mindset. In this mode, all power states are available during a single pass of analysis. Checks such as Domain Crossings, Power Connections, and Domain Leakage will consider UPF states. In addition, the Domain Crossings check will also consider defined isolation controls.

Appropriate Use

This method is most appropriate for use at the system level, where multiple power modes are defined with UPF or power state tables. Cross-domain isolation is controlled by specific signals, defined in UPF.

This method is not appropriate for checking analog blocks that might require a classic external testbench.

Analog States

This is a new mode of checking in Insight Analyzer v5. In contrast to older versions, this new mode overcomes a prior limitation: Checks such as Analog Floats could produce “wildcard combinations” as proposed causes of failure, without knowing that the proposal would be prevented by external factors. For example, an external current reference input might be removed, while an external disable control has not yet been asserted. While true that this would cause an internal float, the user might know that these external factors would never be presented to the circuit.

In this new mode, a power state table has been provided (UPF, above), but further detail is desired. In particular, you can now provide state based boundary conditions such as current reference, enable controls, etc.

Appropriate Use

This method is most appropriate when there are multiple power modes applied to analog circuits.

How it Works

Analog bias circuits do not conform well to the terms of PST / UPF power states and control signals. Bias is a complex fabric of inter-dependent states, which can not be supported by the “lookup table” approach of UPF based checking. Further, analog circuits also depend on the presence or absence of voltage or current references. These are factors commonly needed to sustain valid analog state, but simply not feasible to check in the single pass, PST based methods described above (system of controls).

Instead, it is necessary to consider aspects such as current reference being provided in one state, then shut off (high impedance) in another state. There are other analog aspects such as regulated

voltage values or bias fabric that is held by a control signal. The old “wildcard” handling of variables can not support checking of analog blocks that have multiple operating modes.

The solution implemented in v5 is based on:

1. Making definitions in the Power workview. PST states are combined with contemporary analog states in a boundary table (testbench).
2. State sweeping: The relevant checks, such as Analog Floats and Power Connections (both of which depend on analog bias) scan the circuit in multiple iterations. Before each iteration, a specific state is selected from the PST. All boundary and power settings are made as within that one specific state. (Example: Analog power is switched on, and external current reference is provided.)
3. Collating results: As the check is running through the state sweeping, violations are issued according to the particular state at hand. For example, an internal float during power down state is reported with the state name “power down” (assigned in the state definitions, in Power & States).

See also ‘[cdb states pstLink setGlobalState](#)’.

Workflow Outline

Whether you choose the UPF or State Sweep mode of checking, the required steps are integrated together in the Insight GUI.

Load Netlist

The circuit must be loaded, and devices defined, as normally done for any workflow.

Define Power Modes

Open the “Power” main workspace. Use the “Edit” button to open the dialog “Edit Rails & States”. Use the “Create States” tab to define a state table with at least two states.

Ultimately, the Power workspace should have power / ground rails displayed, along with at least one power state table on the right hand side. All further definitions will depend on the power state names.

Define Testbench

In the Power main workspace, below the power / ground table, you should see a table of the “Boundary” nets (IO). The columns of this table correspond to the states named in the power state table, above.

Use the “Edit” button to open the dialog “Edit Boundary”.

You may start by making boundary definitions only for the important nets, such as inputs that control circuit behavior. (It is usually not necessary to define anything on outputs). You will define boundary conditions as follows:

- General conditions, applicable at all times, such as power / ground rails for a particular IO. This would be done in the column “Settings”, adjacent to the IO net name.
- State conditions, applicable for non-UPF controls that change according to states. This is done in the column for each state, driven by the power state table (above). These settings would include input voltage value that might change from state to state, or a current reference that might exist in one state but be removed in another state.

Define UPF Controls

If you are using isolation controls in a UPF workflow, they are defined by script command. Currently, the Insight GUI does not support the interactive definition of these controls, as their parameters can be complex and are more appropriate when driven by script / UPF.

Run Checks

When you run checking applications, you will make the selection for either UPF / system mode, or Analog States mode. The individual checks have handling for these modes that is tailored to each application.

For example: The Power Connections application has a selector to run either in System mode (single iteration, wildcard assumptions), or Analog States mode (multiple iterations, driven by power and boundary state definitions). When you choose to run Power Connections in Analog States mode, the generated report will show state names under the “States” column.

Plugin Applications

Analog Floats

The application formerly named “Floating Nodes” has been renamed to “Analog Floats”. This has been done to focus on one core area of specialty, while moving unrelated checks aside for system users (described elsewhere). In brief:

- The “Analog Floats” application is meant to be run on lower cells of analog nature.
- This application drops the checking of generally all cross-domain issues related to supply rail differences (isolation cells, etc). Only one aspect of cross-domain remains in this application: A sending supply that is off indirectly leads to a high impedance node within an analog circuit (Insight reference R192).
- A compatibility flag supports comparing against legacy data, where domain crossings are

reported along with analog types. See “Compatibility” section.

- This application adds support for state-based testbench definitions. A signal level, a voltage reference, and a current reference may all be defined. These boundary conditions may be defined in conjunction with power / ground states.
- A “State” column has been added to the report, indicating which particular testbench state a violation corresponds with.

Config Options

Option flags for this application have been updated as follows:

- `-statesMultiSweep 1/0`: If 1, make multiple iterations, per the defined power and testbench states. If 0, make a single iteration (legacy v4 mode).
- `-statesRepeatViolations 1/0`: Used in conjunction with `-statesMultiSweep`. If 1, allow repeating violations per each state, if applicable. If 0, report a violation once, in the first state where found.

Contention

Config Options

Option flags for this application have been updated as follows:

- `-checkLatchNodes 1/0`
- `-doRailContention 1/0`
- `-doSignalContention 1/0`
- `-showUnsafeCell 1/0`
- `-strictPassgateCorners 1/0`
- `-effort`: This option has been removed, but continues to take effect for compatibility on a read-only basis.

Domain Crossings

This new application has been added. It inherits the cross-domain checking that was formerly covered by “Floating Nodes”, but is now able to focus more specifically on the complexities of this type of checking. This application is meant to be run on upper cells where power supply switching or domain isolation is implemented.

Reporting

With respect to the former float checks in v4, the report columns have been changed to be more appropriate for system level checking. The new columns are:

- Violation, Cell: Same as in prior versions.
- Z Net: The float net being reported. In complicated cases, this may be converted to a familiar signal such as the input pin of a faulty isolation cell.
- Control: The name of the associated isolation control, or undefined (suspected) control, where appropriate.
- Instance: Cross-probe link to the most relevant device or cell instance. In complicated cases, this may be converted from where a detail float is first detected.
- Crossing: Summary of the domain crossing rail names being blamed for the violation.

Violation names have been expanded to be more descriptive than in v4. They are as follows:

- "Missing isolation" (also v4)
- "Faulty isolation" (appears less frequently now, in lieu of the following detail violations)
- "Iso ctrl undefined" (new)
- "Iso ctrl wrong state" (new)
- "Wrong ctrl for domain" (new)
- "Iso ctrl interrupted" (was "Faulty iso ctrl" in v4)
- "External float" (also v4)
- "Float, LS internal" (new, Insight reference R253)
- "Float, switch cutoff" (moved from "Analog Floats")

Legacy Compatibility

By setting a flag, the following changes take effect:

- Call to run "Domain Crossings" plugin will run "Analog Floats" instead. (v5 run script for will issue results matching v4 results.)
- The "Analog Floats" plugin will issue violations that would normally be issued by "Domain Crossings". This is the legacy mode of reporting all issues in one report. (v4 run script will work for v5, issuing results matching v4 results.)

- The domain crossing violations will be formatted into the columns of “Analog Floats” check, rather than using the new column definitions.
- The domain crossing violations will use the old wording of v4, rather than the updated violation wording.

See “Compatibility” section for details on this control.

Domain Leakage

New Cases

The following new victim profile has been added:

- MOSFET functional diode connected directly between rails (R200).

Legacy Compatibility

By setting a flag, the following changes take effect:

- Suppress display of R reference numbers in report details.
- Avoid handling forward MOSFET diodes directly between rails (R200).

Fanout

Config Options

Option flags for this application have been updated as follows:

- `-mergeParallelDrivers`
- `-excludeStdcellHeaderFooter`

Power Connections

Config Options

Option flags for this application have been updated as follows:

- `-statesMultiSweep 1/0`: If 1, make multiple iterations, per the defined power and testbench states. If 0, make a single iteration (legacy v4 mode).
- `-statesRepeatViolations 1/0`: Used in conjunction with `-statesMultiSweep`. If 1, allow repeating violations per each state, if applicable. If 0, report a violation once, in the first state where found.

Reference Numbers

Some new checks (such as the Faulty_Loops plugin) will include Insight Reference Numbers in a report with each violation. As a new plugin, there is no compatibility concern (comparing against saved reports), so inclusion of the reference numbers is adopted going forward. Legacy checks, for the near future, will continue to issue violations without reference numbers for compatibility purposes.

In future versions, Insight Analyzer will issue violations with reference numbers for all checks.

API

`cdb cell _ characterizeAnalogPorts`; create testbench listings for an analog block, to be used with `pstLink`.

`cdb net _ tracePathTo _ -gateNets`; option to return the gating conditions.

`cdb net _ findConditionsHolding up | down`; trace bias fabric to detect gating condition (1st order and further upstream).

`cdb inst _ isCascodeMirrorDiode | ...Slave`

`cdb inst _ isSRAMbit`

The given MOSFET instance is part of an SRAM bit storage function.

`cdb inst _ getSRAMbitInfo`

Return details of the SRAM bit storage that includes the given instance. Example:

`RWfets: /XI1/MM5[0] /XI1/MM5[1] /XI1/MM5[2]...`

`coreFets: /XI1/MM0 /XI1/MM2 /XI1/MM4...`

`dataLines: /-wbl_n0 /-wbl_n1 /-wbl_n2 /-wbl_n3...`

`controls: /-wwl0[15] /-wwl1[15] /-wwl2[15]...`

`cdb net _ findForwardDiodeWith _`; find plain diode, cascode diode, virtual diode / mirror slave. Includes any gating condition that might exist.

`cdb states pstLink setNetStates _ -table _ -states _`; correlate testbench values and roles with PST states.

`cdb states pstLink setGlobalState _`; select a specific PST state for one check iteration.

`cdb states pstLink setStateSequence _`; specify a sequence of global states for multiple check iterations.

Licensing

Insight Analyzer v5 introduces a number of features not available in v4. Most of these features are evolutionary, and included by default for users who are migrating from v4 to v5.

Scenario Editing Feature Set

One feature set is substantially new in v5, and intended for particular workflows, so it is available through an additional license token called “Scenario Editing”. This license covers the following parts, all of which are new in v5:

- Script command ‘`cdb states pstLink ...`’
- Editing power state tables and boundary states in the GUI.
- Running checks in state sweep mode, such as Analog Floats and Power Connections. This mode uses defined power and boundary states.

License File Migration

If you are migrating from Insight Analyzer v4, you will need an updated license file for v5. Legacy license files have been generated with “4.0” level, which must be updated to “5.0”. Please contact your sales representative or support contact for an updated license file.

To enable the Scenario Editing feature set, you will need a new license token (`scenario_editing` in the license file). If applicable to your workflow, please include this in your request for an updated license file.

Here is a summary of the license file codes:

Code	v4.x	v5.x
<code>erc_platform</code>	4.0	5.0
<code>esd_emp</code>	4.0	5.0
<code>power_conns</code>	4.0	5.0
<code>floating_nodes</code>	4.0	5.0
<code>power_states</code>	4.0	5.0
<code>scenario_editing</code>	(not available)	5.0
<code>fanout</code>	4.0	5.0
<code>developer</code>	4.0	5.0

Compatibility

The configuration (inputs) and report results (outputs) have changed in many ways from v4 to v5. For QA and acceptance purposes, we have invested substantial effort in minimizing these changes. A compatibility setting allows you to switch between v4 and v5 behavior, and between v4 and v5 run scripts, where feasible and appropriate. Any of the following options will apply this setting (please choose only one):

- Option 1: In shell, prior to startup: ‘`setenv INSIGHT_BACK_COMPAT_V4 1`’
- Option 2: In command line at startup, add ‘`-backCompatibilityV4`’
- Option 3: In Tcl, after startup: ‘`runenv setHiddenFeatures backCompatibilityV4`’

Has impact on the following:

- General reporting: Suppress display of R reference numbers in report details, both in saved report files and in GUI view.
- Power Connections report, restore old wording for “FET Vb wrong”.
- Power Connections additional pins: In v5 we are now checking all substrate pins for forward bias, whereas in v4 these pins were checked only to the extent of the first encountered bias violation for the given device. In other words, v5 can report multiple bias violations per device instance, if the additional pins exist.
- Floating Nodes: Run all float checking under the umbrella of Analog Floats check, including cross domain / isolation cases, and issue violations in the old format, in a single report, as had been done in v4.
- Floating Nodes: Remove the “states” column from the report (for report file diff).
- Printing decimal values: v5 uses 3 significant figures, where v4 had formerly used 2. Example: v5 may print “1.25” where v4 would print “1.2”. With the compatibility flag set, the older 2 figure mode is restored.