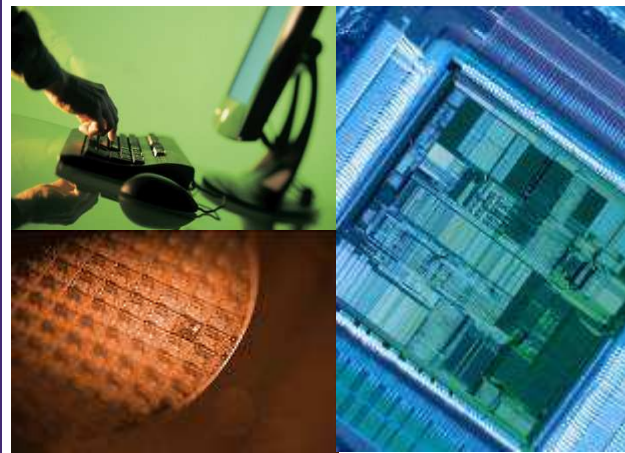


Formality® Customer Update Training E-2010.12

December 2010



SYNOPSYS®
Predictable Success

CONFIDENTIAL INFORMATION

The following material is being disclosed to you pursuant to a non-disclosure agreement between you or your employer and Synopsys. Information disclosed in this presentation may be used only as permitted under such an agreement.

LEGAL NOTICE

Information contained in this presentation reflects Synopsys plans as of the date of this presentation. Such plans are subject to completion and are subject to change. Products may be offered and purchased only pursuant to an authorized quote and purchase order. Synopsys is not obligated to develop the software with the features and functionality discussed in the materials.

2010.12 Enhancements Overview

Design Read/Write

- System Verilog Language Support
- VHDL Language Support
- Enhancement for reading & writing design data

GUI Enhancements

- Double Design Browser
- Queued Setup Commands
- Hierarchical Verification Results Browser
- Formality On-Line Help using a Standard Browser

Verification and Debugging Improvements

- New Clock Gating Solution
- Gate to Gate Verification Runtime Improvements
- Failing and Hard Verification Analysis

2010.12 Enhancements Overview

Automated Setup Enhancements

Formality script Generator

Low Power Support Updates

Support for Supply Sets

Support for non-UPF command `set_related_supply_net`

Changes in Formality variables and commands

Design Read and Write Enhancements

Language Support Improvements

- The following enhancements have been made to keep the Formality readers in lock-step with Presto.
- System Verilog
 - Module Port Initializations
 - Instance port Coercion
- VHDL
 - Impure function
 - Deferred constants
 - 'Boolean' ranged array
 - Interpreting don't cares in CASE condition (VHDL 2008 compliance)
 - Identifying 'all' in sensitivity lists (VHDL 2008 compliance)

Enhancement for Reading & Writing design data

Pre set_top Containers:

- Formality now allows the user to write out a container prior to elaboration (set_top) of the design
 - Containers can now be written prior to a successful set_top using

```
setup> write_container -pre_set_top -r cont.fsc
```
 - The size of a pre_set_top container will be larger
 - The report generated by **report_hdlin_mismatches** will be preserved for containers written prior to set_top

Enhancement for Reading & Writing design data

Pre set_top Containers:

This will not work ...

write_container -r pre-set-top

Error: Top design has not been successfully linked in container 'r' (FM-236)

0

set_top top_ver

Setting top design to 'r:/WORK/top_ver'

This is how it works ...

write_container -r -pre_set_top pre-set-top

Warning: This container will be compatible with current Formality Version.

Info: Wrote file 'pre-set-top.fsc'.

1

set_top top_ver

Setting top design to 'r:/WORK/top_ver'

Enhancement for Reading & Writing design data

Multiple `set_top` Attempts:

- Formality now allows the user to perform `set_top` multiple times till successful elaboration
 - Previous version of Formality allowed a single `set_top` per container
 - If an error occurred due to missing/incorrect source files, then the user was required to start over
 - Formality now allows the user to add the missing/incorrect files and/or variables and continue the elaboration

Enhancement for Reading & Writing design data

```
#-----  
#       Read in Reference Design  
#-----  
read_verilog -r bottom1.v  
No target library specified, default is WORK  
Initializing DesignWare ...  
Can't open file /u/formal/dwroot/dw/dw_lib_info.ctle  
Initialization Complete  
Loading verilog file '/remote/ped61/karthika/fm_verilog_examples/work_pad/bottom1.v'  
Current container set to 'r'  
1  
#kram DEMO read_verilog -r bottom2.v  
read_verilog -r top_fm.v  
No target library specified, default is WORK  
Loading verilog file '/remote/ped61/karthika/fm_verilog_examples/work_pad/top_fm.v'  
1  
set_top -auto  
Setting top design to 'r:/WORK/top_ver'  
Status:   Elaborating design top_ver ...  
Warning: Cannot link cell '/WORK/top_ver/u1' to its reference design 'bottom2'. (FE-LINK-2)  
Status:   Elaborating design bottom1 ...  
Error: Unresolved references detected during link. (FM-234)  
Error: Failed to set top design to 'r:/WORK/top_ver' (FM-156)  
0  
read_verilog -r bottom2.v  
No target library specified, default is WORK  
Loading verilog file '/remote/ped61/karthika/fm_verilog_examples/work_pad/bottom2.v'  
1  
set_top -auto  
Setting top design to 'r:/WORK/top_ver'  
Status:   Elaborating design bottom2 ...  
Status: Implementing inferred operators...  
Top design successfully set to 'r:/WORK/top_ver'  
Reference design set to 'r:/WORK/top_ver'  
1
```

Enhancement for Reading & Writing design data

Maintaining Shared Technology Libraries for Reference & Implementation:

- Allow `read_verilog` and `read_vhdl` into shared libraries space to avoid duplicate reading of shared library
- Previous version of Formality required libraries to be read using
 - `read_verilog -tech -[r | i] OR`
 - `read_vhdl -tech -[r | i]`
- Formality now allows these libraries to be shared between containers
 - `read_verilog -tech`
 - `read_vhdl -tech`

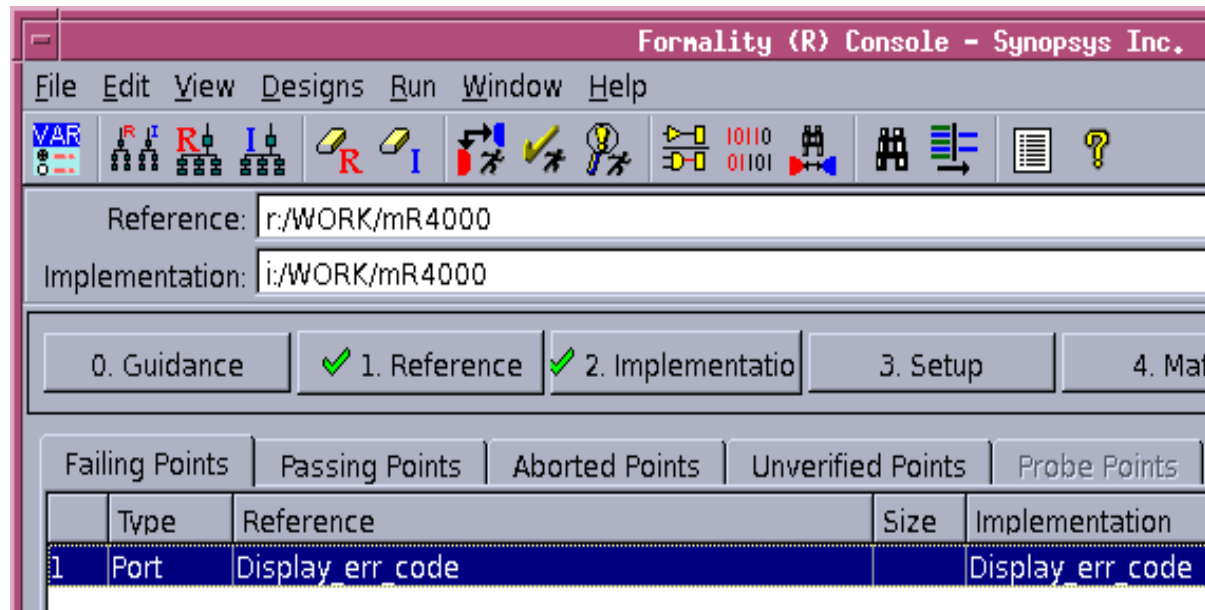
Summary

- System Verilog and VHDL Language support enhancements make Formality readers in sync with Presto
- Containers can now be saved prior to `set_top` using the switch `-pre_set_top`
- Previously Formality would allow only one attempt at `set_top`, Formality now allows multiple attempts at setting top
- The simulation libraries can now be read into a single shared library

Graphical User Interface Enhancements

Double Design Browser

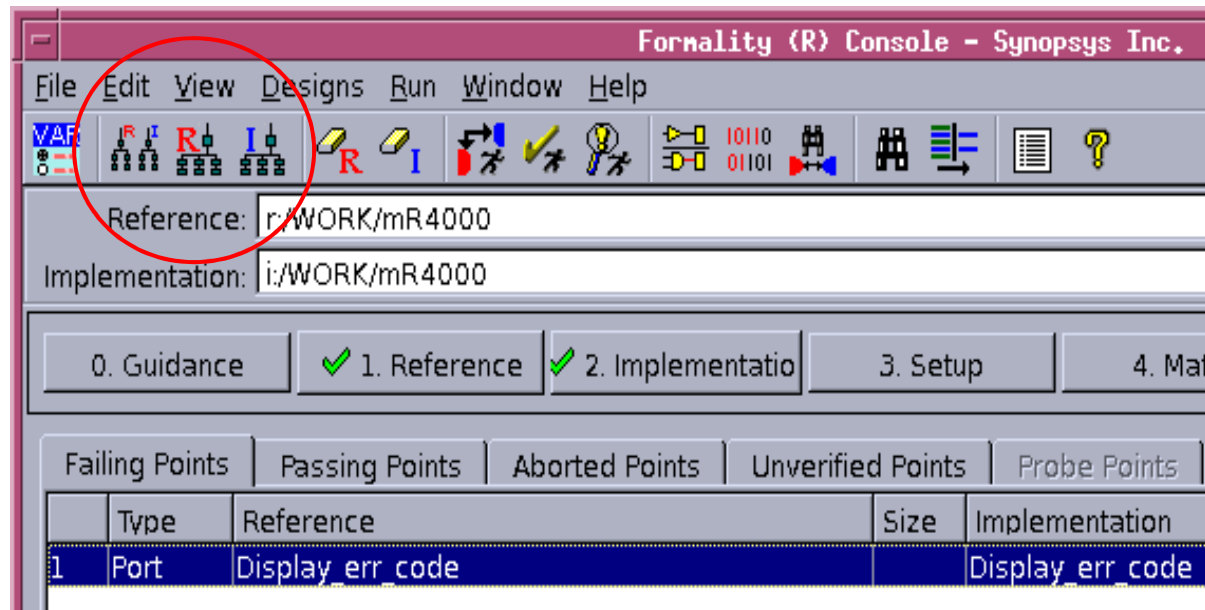
This new GUI feature allows the reference and implementation design hierarchies to be viewed alongside each other in the same window, called the “Double Design Browser”



You can select one or the other type of design browser

Double Design Browser

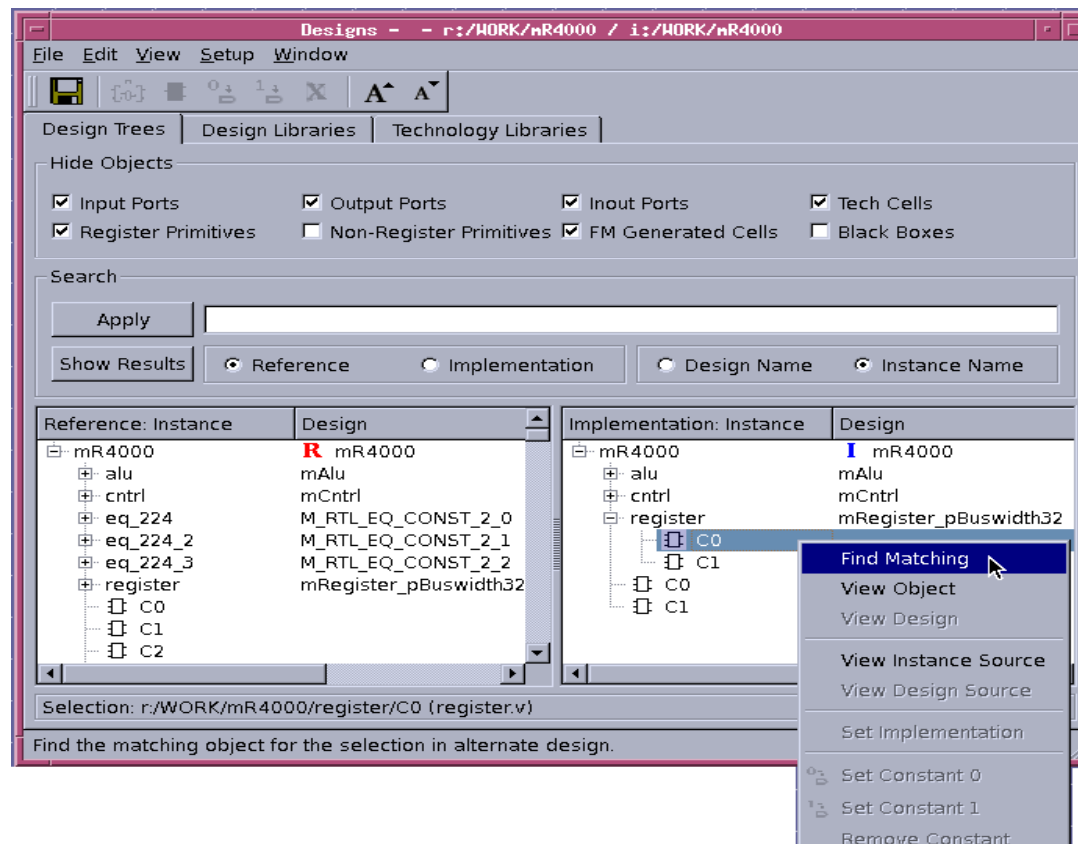
This new GUI feature allows the reference and implementation design hierarchies to be viewed alongside each other in the same window, called the “Double Design Browser”



You can select one or the other type of design browser

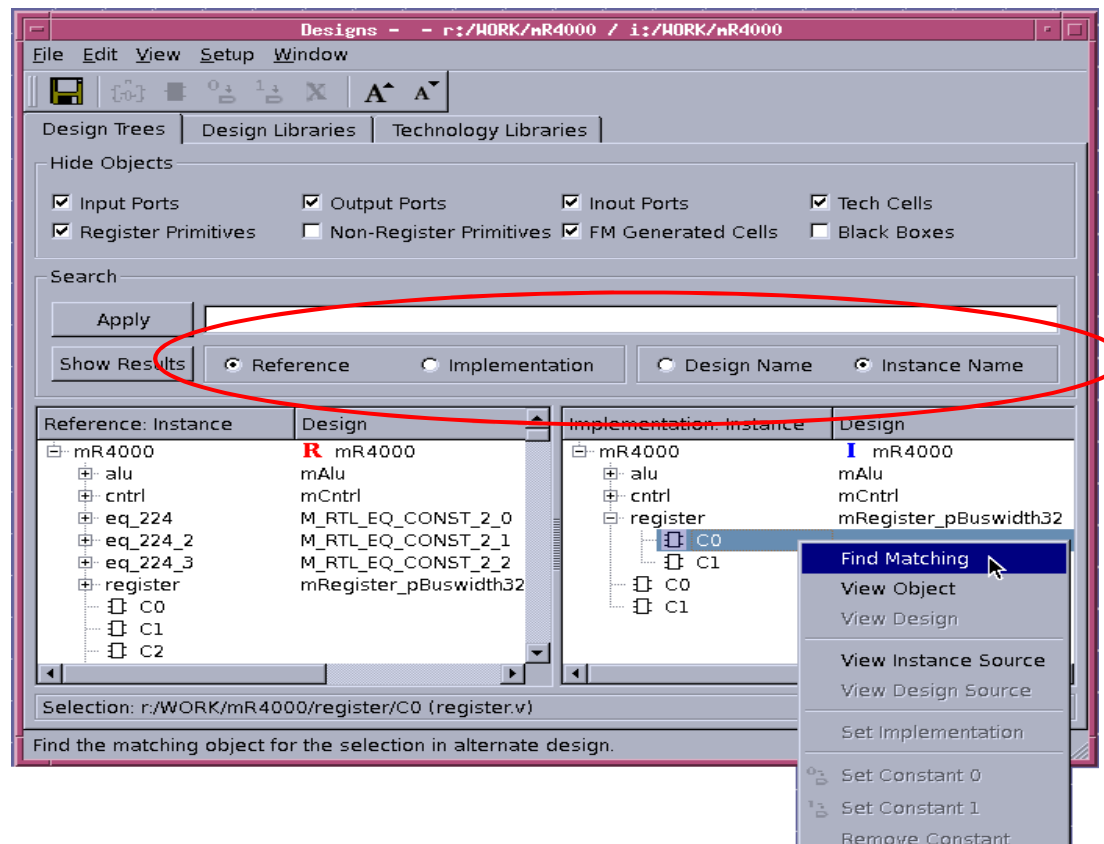
Double Design Browser

The Double Design Browser integrates the reference and implementation design browsers,



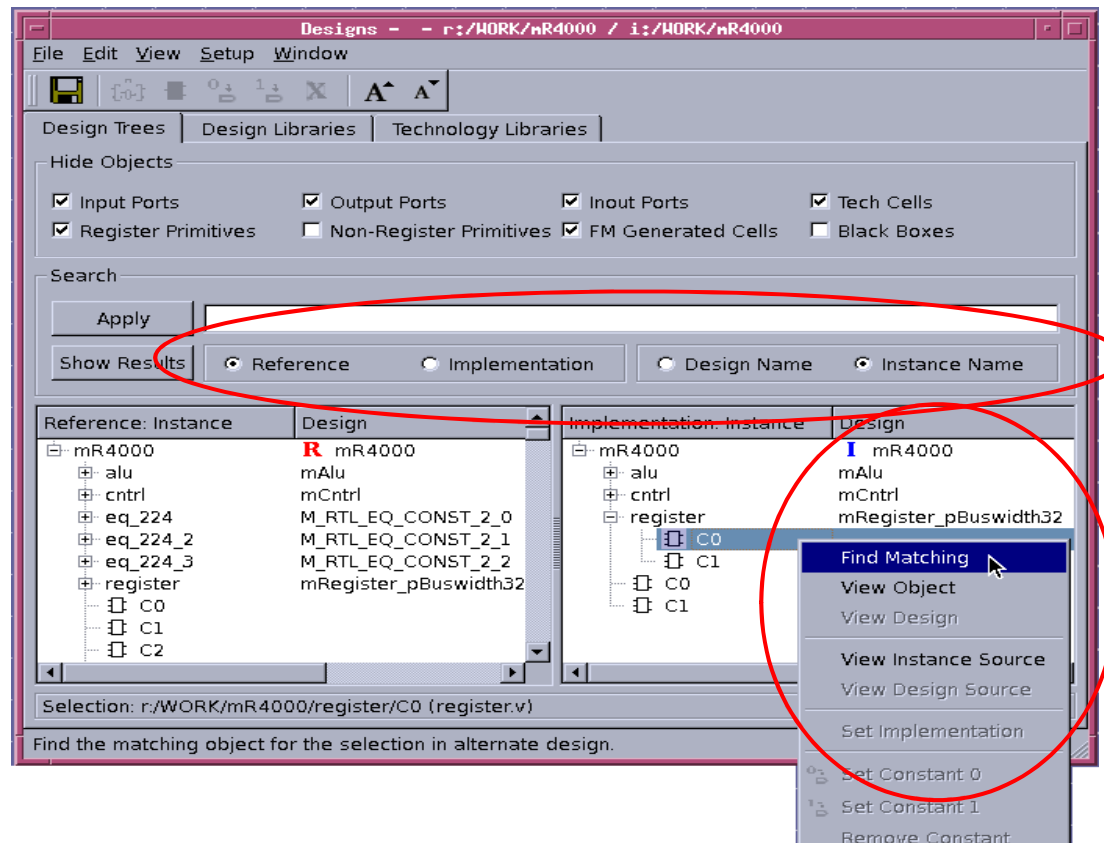
Double Design Browser

The Double Design Browser integrates the reference and implementation design browsers, into a single browser.



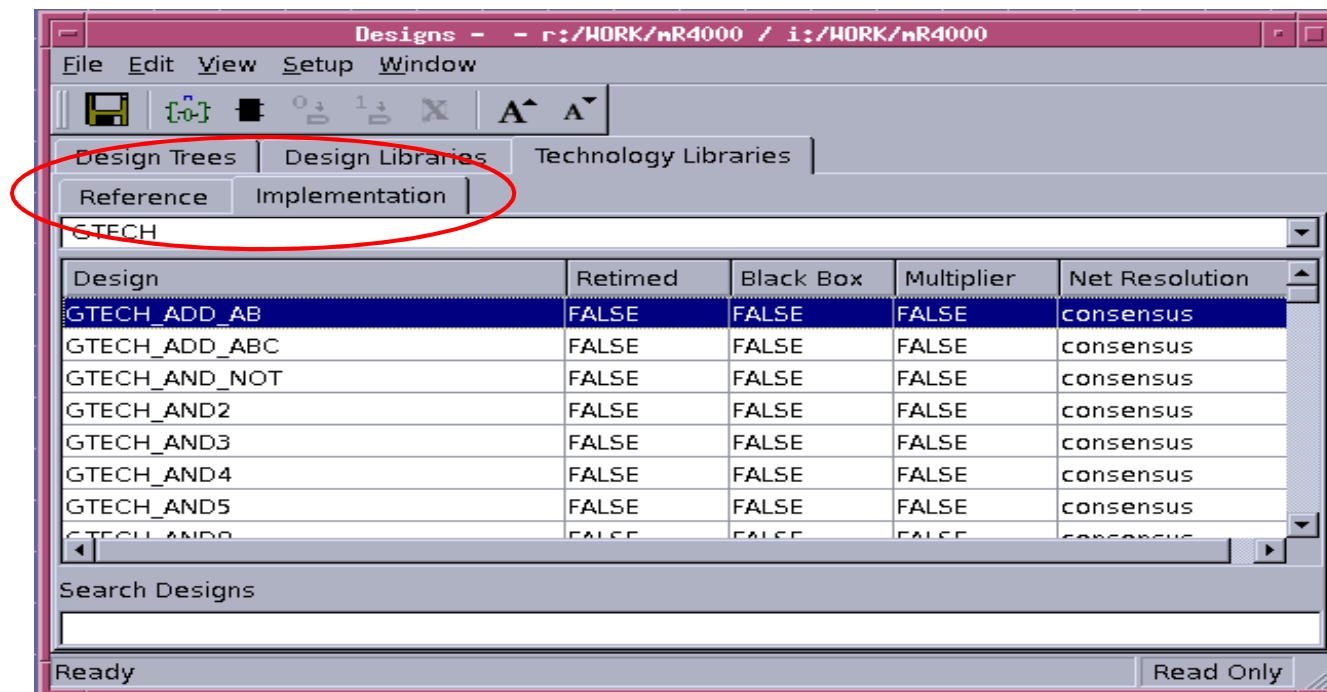
Double Design Browser

A “Find Matching” feature has also been added to allow the user to select an object in one design, and find the corresponding matched object alongside the other design.



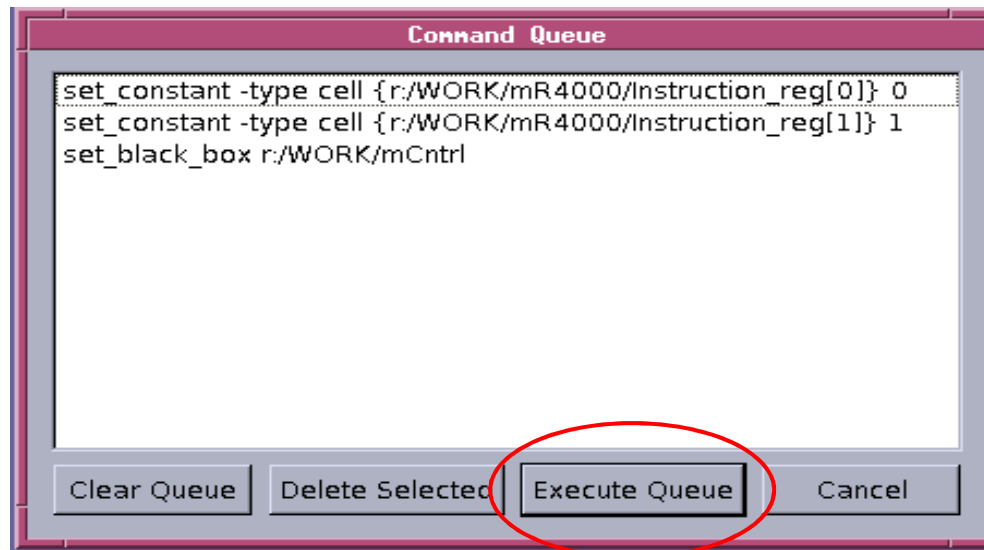
Double Design Browser

The Design Library and Technology Library tabs now have both the reference and implementation libraries available through a tabbed interface



Queued Setup Commands

- When in `MATCH` or `VERIFY` mode, Formality can now store commands that can only be done in `SETUP` mode, and execute them at once, when you are ready to revert back to `SETUP` mode
- A new GUI window appears when executing setup commands from the Design Browser window

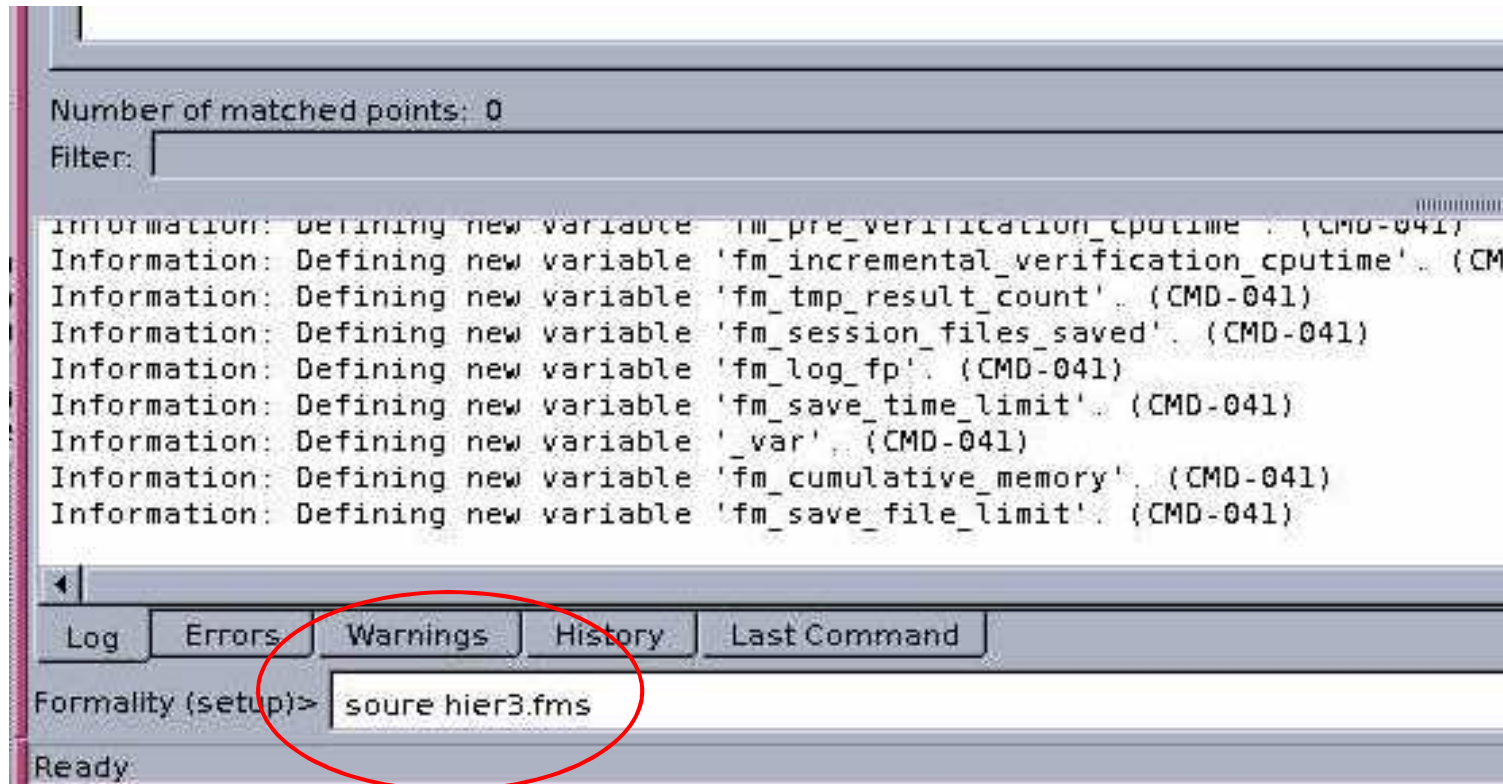


Hierarchical Verification Results Browser

- Formality now allows the user to view the results of a hierarchical verification within the GUI
- In previous releases of Formality the user had to
 - Run hierarchical verification
 - Formality would write out session files for the failed blocks
 - We would have re-start Formality and restore these session for debug
- The user can now debug the failing blocks in the same Formality where the hierarchical verification ran

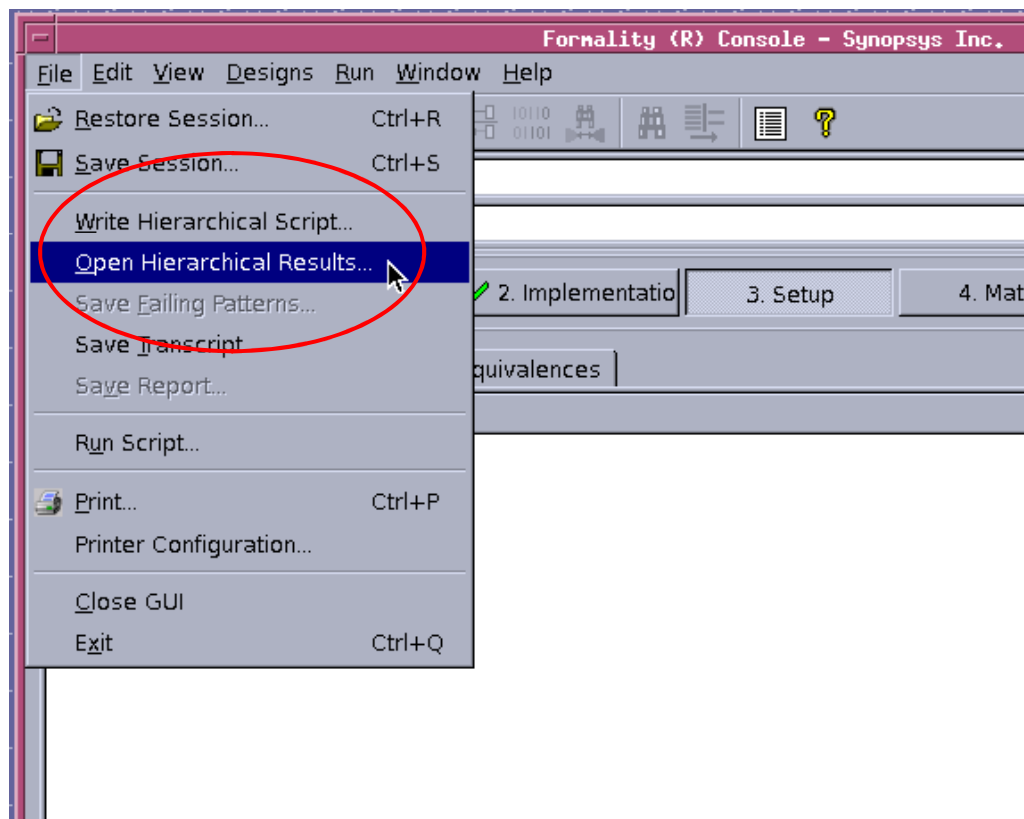
Hierarchical Verification Results Browser

After writing out the hierarchical verification script, source the Formality created hierarchical verification script from within the GUI



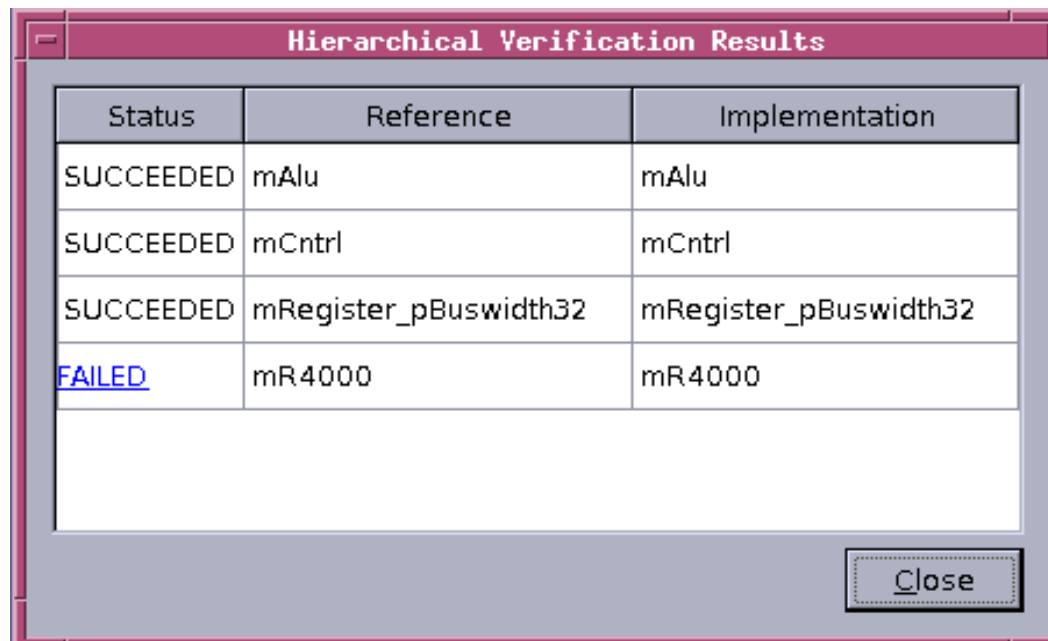
Hierarchical Verification Results Browser

Start the graphical interface using the “Open Hierarchical Results” link in the FILE menu in Formality’s main Window



Hierarchical Verification Results Browser

It also allows the user to open a separate GUI session for each failing hierarchical verification that has been saved.



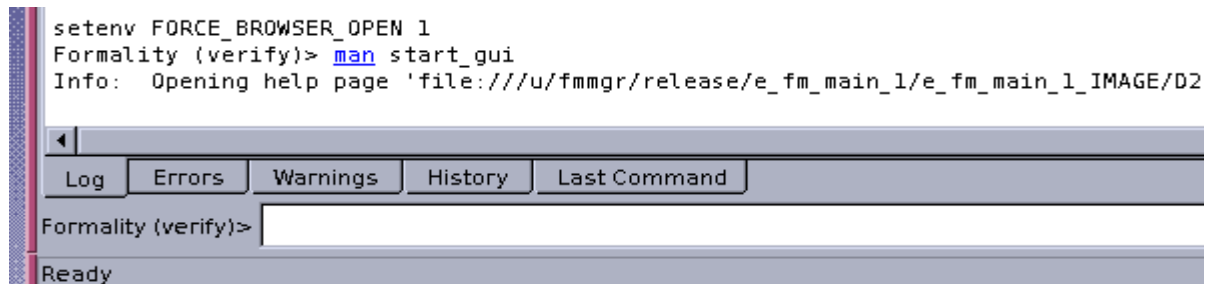
Status	Reference	Implementation
SUCCEEDED	mAlu	mAlu
SUCCEEDED	mCntrl	mCntrl
SUCCEEDED	mRegister_pBuswidth32	mRegister_pBuswidth32
FAILED	mR4000	mR4000

Close

Formality On-Line Help

- In the GUI mode, executing a 'man' command will trigger a Web Browser to open up the Formality on-line help

```
setenv FORCE_BROWSER_OPEN 1
Formality (verify)> man start_gui
Info: Opening help page 'file:///u/fmmgr/release/e_fm_main_1/e_fm_main_1_IMAGE/D2
```

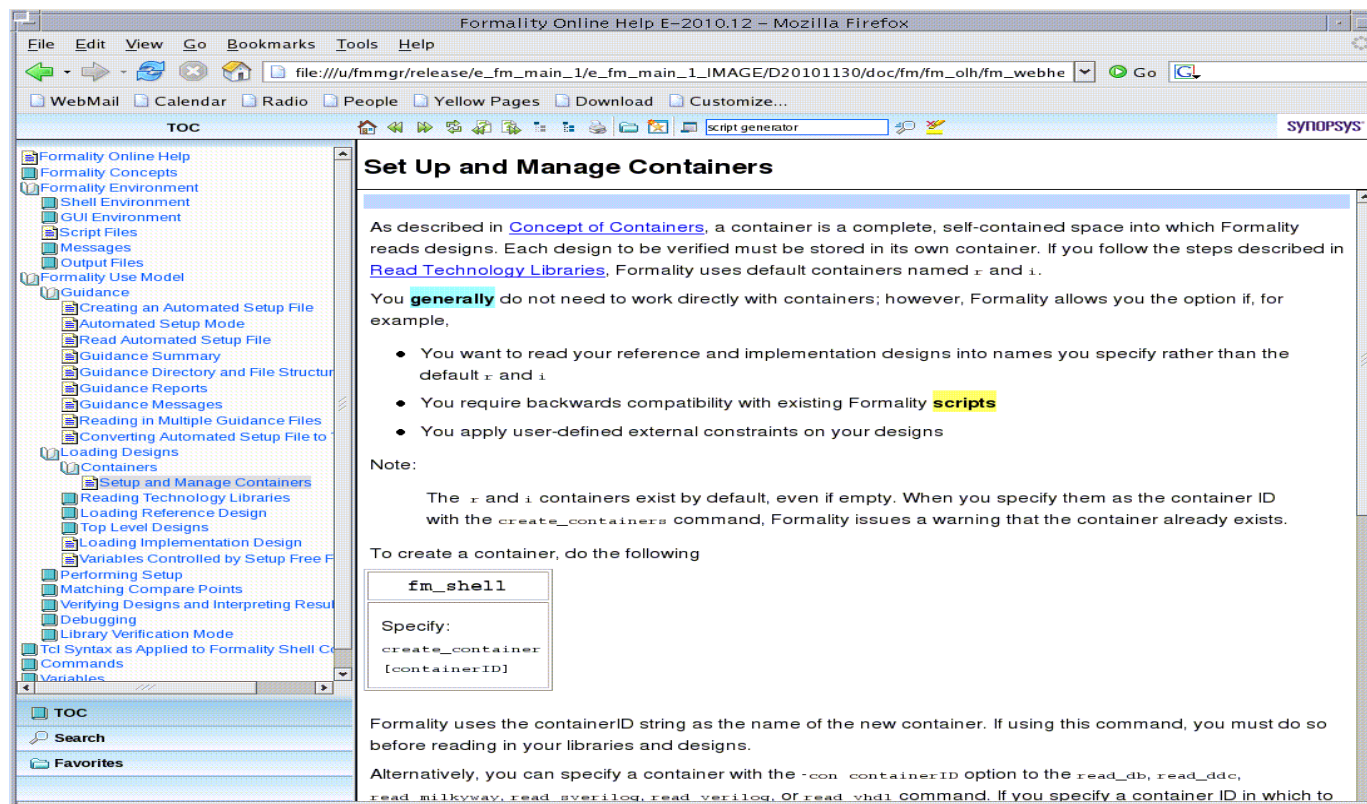


The screenshot shows a terminal window with a light blue background. The text in the terminal is as follows: 'setenv FORCE_BROWSER_OPEN 1', 'Formality (verify)> [man](#) start_gui', and 'Info: Opening help page 'file:///u/fmmgr/release/e_fm_main_1/e_fm_main_1_IMAGE/D2'. Below the terminal text, there is a horizontal bar with a left-pointing arrow on the left. Underneath this bar is a row of five buttons: 'Log', 'Errors', 'Warnings', 'History', and 'Last Command'. Below the buttons is a text input field with the prompt 'Formality (verify)>'. At the bottom of the window, the word 'Ready' is displayed.

- In the fm_shell mode, the usual ASCII man page is displayed
- You can control the way that you would like to view the man pages using the variable **sh_man_browser_mode**. Its possible values are
 - GUI, Shell, Both and None.

Formality On-Line Help

- Users will have the capability to bring up the Online help GUI, search and browse for manual pages of all Formality commands, variables, error and warning messages as was previously available in the Formality command line.



Formality On-Line Help

- Opening the On-Line Help page on a Web Browser in VNC server on a Sun OS is known to CRASH the VNC Server
- Hence you will see this message displayed by Formality if you attempt to do so ...

```
Formality (verify)> man start_gui
Warning: Web browser invocation has been suppressed as invoking the
web browser in a vncserver on a Sun OS machine is known to crash the
vncserver. This is irrespective of the vncviewer used to connect to
the vncserver. This can be worked around by setting your display de
to 8 when running [ running versus launching a] vncserver.
You can force the web browser open by setting an environment variat
"FORCE_BROWSER_OPEN" to 1 in the application if you are sure this
will not crash your vncserver. Please be aware of the risk of crash
vncserver (and hence all applications displaying on the vncserver a
with it).
The help page url is: file:///u/fmmgr/release/e_fm_main_1/e_fm_mair
```

- If you still wish to force the Browser open
 - Set environment variable **FORCE_BROWSER_OPEN = 1**

Summary

- Formality now has a compact Double Design Browser in addition to the classic design browsers for Reference and Implementation Designs.
- If it is attempted to run `setup` commands when either in `match` or `verify` mode, Formality brings up a GUI which queues up these `setup` commands. Pressing '`Execute`' will switch to setup mode and execute all the queued commands.
- Formality now has a hierarchical verification results viewer in the GUI.
- Formality now has a fully functional On-Line Help. Executing '`man`' in the GUI mode brings this up by default. In the `fm_shell` mode the ASCII man pages are displayed by default

Verification and Debugging Improvements

New Clock Gating Solution

- The current clock solution in Formality is controlled by the user variable `verification_clock_gate_hold_mode` to identify clock gating latches (LATCG).
- If LATCGs are not correctly identified, they are treated as regular latches and may cause false verification failures. The new clock gating solution addresses these problems.
- Enable the new clock gating feature by using the following setting:
`set verification_clock_gate_edge_analysis true`
- Using this setting will disable the legacy clock gating software, rendering the legacy variable `verification_clock_gate_hold_mode` *ineffective*.
- The Formality GUI will include special rising (r) and falling (f) notations as well as other transition notations on failing patterns and cone schematics representing edge values on signals.

Gate to Gate Verification Runtime Improvements

- Formality has been enhanced to exhibit (up to 2X) better runtime performance in verifications between two versions of gate level designs.
 - Measured using the Formality D-2010.03 as a benchmark

Failing and Hard Verification Analysis

- The `analyze_points` command will now look for two additional failure causes in the E-2010.12 release.

Failing and Hard Verification Analysis

- The **analyze_points** command will now look for two additional failure causes in the E-2010.12 release.
- Matched black box nets
 - Previously, the **analyze_points** command would only look for unmatched black box nets in the cone of a failing point
 - Now it will look for both matched and unmatched black box nets as the possible cause of a verification failure
- Unmatched Inputs
 - Previously, **analyze_points** would report that unmatched input as a possible cause of the failure only if the unmatched input appeared to be a constant
 - In the E-2010.12 release, **analyze_points** will now examine all unmatched inputs in the both the reference and implementation design as a possible failure cause

Failing and Hard Verification Analysis

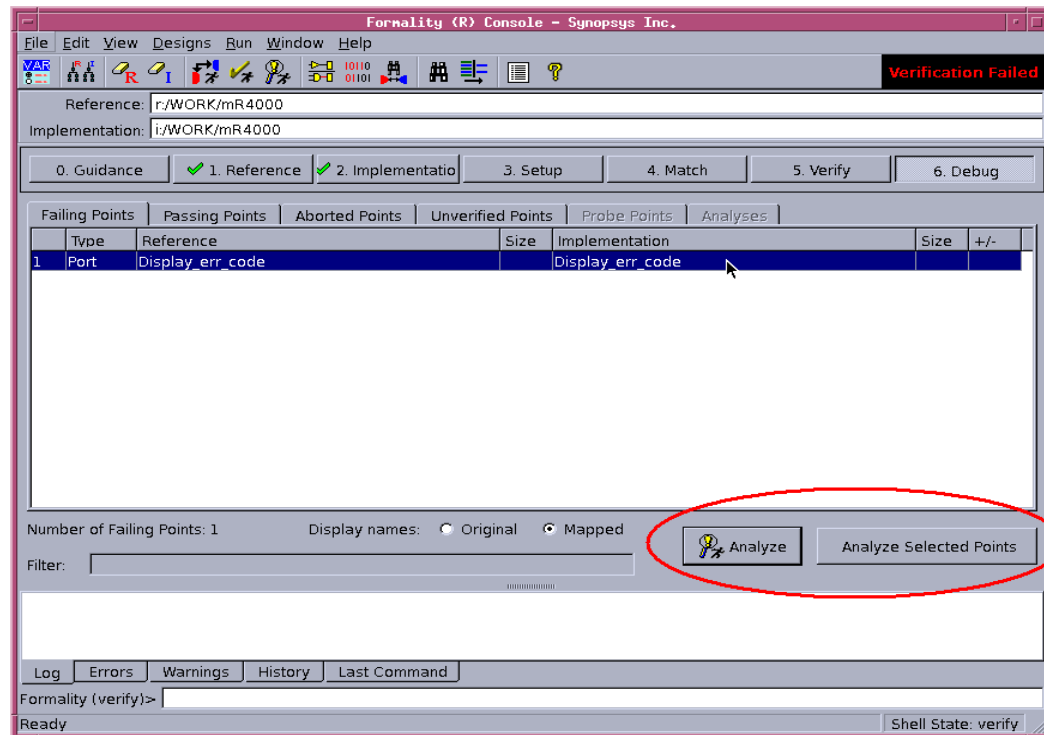
- You may use `analyze_points` in the 'Formality shell' in the following manner

- | | |
|--|---|
| - <code>analyze_points -all</code> | <i>To analyze all failing/aborted points</i> |
| - <code>analyze_points -failing</code> | <i>To analyze all failing points</i> |
| - <code>analyze_points -aborted</code> | <i>To analyze all aborted points</i> |
| - <code>analyze_points [failing point(s)]</code> | <i>To analyze a single failing point, or list of failing points</i> |

- You may use `analyze_points` in the 'Formality GUI' in the following manner
 - In the Formality console window, under the failing point tab in debug mode, there are two buttons: Analyze and Analyze Selected Points

Failing and Hard Verification Analysis

- To run the `analyze_points` command on all failing points, click on the Analyze button
- To run `analyze_points` on a single failing point or a selected set of failing points, first select one or more failing points in the window and then click the Analyze Selected Points button



Summary

- Formality has a brand new solution for identifying clock gating latches. It is a new and improved solution and can be turned on by setting the variable **verification_clock_gate_edge_analysis** to true. It is also capable of displaying differences in reference and implementation design because of edge differences
- Formality runs up to two times faster on gate to gate verification flows
- The enhanced **analyze_points** command can now report failing points occurring due to matched black box nets and unmatched inputs in the design

Automated Setup Enhancements

Formality Script Generator

The Formality script generator can now pass information about the,

- Design files
- Libraries used
- Verification Information

automatically through the SVF file

Syntax:

fm_mk_script <svf...> [-o[utput] <file>]

<svf...> : List of SVF files/directory produced during the synthesis.

-o[utput] : Specifies the output script to be created.

If there is no output name specified the default name will be fm_mk_script.tcl

Formality Script Generator

Example:

`fm_mk_script default.svf` *Creates fm_mk_script.tcl*

`fm_mk_script default.svf -output fm.tcl` *Creates fm.tcl*

The purpose is to support the standard DC synthesis flow, defined by the Reference Methodology.

- This feature works only when the SVF file is written out using **DC 2010.03.SP1** or later

Summary

- Formality can now automatically generate a fully usable Formality run script using the `fm_mk_script` utility. It uses the enhanced information in the SVF file to write out such this script.
- The script may often need manual intervention because of differences in the run directories and other user aware changes

Low Power Support Updates

Low Power Support Updates

Support for source/sink and diff_supply_only for isolation

```
set_isolation  
[ -source source_supply_ref  
| -sink sink_supply_ref  
| -source source_supply_ref -sink sink_supply_ref ] [ -  
diff_supply_only <TRUE|FALSE> ]
```

As with other tools in the Synopsys low power flow, Formality will now accept these options to the set_isolation command. Formality will use them to insert isolation in the RTL the same as simulation so that synthesized designs can be correctly verified.

Low Power Support Updates

Support for set_design_attributes command

```
set_design_attributes  
    < -elements element_list >  
    [ -attribute name value ]*
```

- To support a hierarchical synthesis flow, the UPF commands set_design_attributes is read by Formality.
- These attributes are used in the hierarchical synthesis flow, and allow Design Compiler to convey information needed to characterize and merge power domains during synthesis.
- The design attributes are read by Formality but are not used in verification.

Low Power Support Updates

Support for set_port_attributes command

```
set_port_attributes  
    [ -ports {port_list} ]  
    [ -attribute name value]
```

- To support a hierarchical synthesis flow, the UPF commands set_port_attributes is read by Formality.
- The attribute value for the attributes named iso_source and iso_sink are used to determine the source and sink supply signals for top level input and output isolation. The attribute value must be a previously defined supply set name.
- These attributes are used in conjunction with the new set_isolation -source/-sink options.

Low Power Support Updates

Support for non-UPF command `set_related_supply_net`

`set_related_supply_net`

- This is a non-UPF command that allows users to tell the tools in the Synopsys flow additional information about which UPF supplies specific nets, ports and pins are associated with.
- Formality will use the `set_related_supply_net` commands in the UPF files to help it accurately verify the design based on the available supply nets specified in the command.
- It is only accepted in UPF files during `load_upf`.

Summary

- Formality now supports the following UPF commands. They can be used in conjunction with supply sets in the UPF flow
 - `set_isolation`
 - `set_design_attributes`
 - `set_port_attributes`
- Formality also now supports the non-UPF command
 - `set_related_supply_net`

Changes in Formality variables and commands

- VARIABLE: `verification_verify_unread_bbox_inputs`

The Formality variable `verification_verify_unread_bbox_inputs` now has a default value of `true`

- COMMAND: `read_simulation_library`

Starting with the E-2010.12 release, the `read_simulation_library` command will no longer be available. The `read_simulation_library` command is now completely replaced by the command `"read_verilog -technology_library"`.



Thank You

SYNOPSYS[®]
Predictable Success