© Cadence Design Systems, Inc. Do not distribute.

Genus™ Synthesis Solution

Course Version 16.2

Lab Manual **Revision 1.0**

cadence®

© Cadence Design Systems, Inc. Do not distribute.

© 1990-2016 Cadence Design Systems, Inc. All rights reserved.

Printed in the United States of America.

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Trademarks: Trademarks and service marks of Cadence Design Systems, Inc. (Cadence) contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence trademarks, contact the corporate legal department at the address shown above or call 1-800-862-4522.

All other trademarks are the property of their respective holders.

Restricted Print Permission: This publication is protected by copyright and any unauthorized use of this publication may violate copyright, trademark, and other laws. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. This statement grants you permission to print one (1) hard copy of this publication subject to the following conditions:

- The publication may be used solely for personal, informational, and noncommercial purposes;
- The publication may not be modified in any way;
- Any copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement; and
- Cadence reserves the right to revoke this authorization at any time, and any such use shall be discontinued immediately upon written notice from Cadence.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. The information contained herein is the proprietary and confidential information of Cadence or its licensors, and is supplied subject to, and may be used only by Cadence customers in accordance with, a written agreement between Cadence and the customer.

Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

Restricted Rights: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

© Cadence Design Systems, Inc. Do not distribute.

Table of Contents

Genus Synthesis Solution

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Module 1: About This Course

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

There are no labs for this module

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Module 2: Overview of Genus Synthesis Solution

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

There are no labs for this module

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Module 3: Genus Synthesis Solution Fundamentals

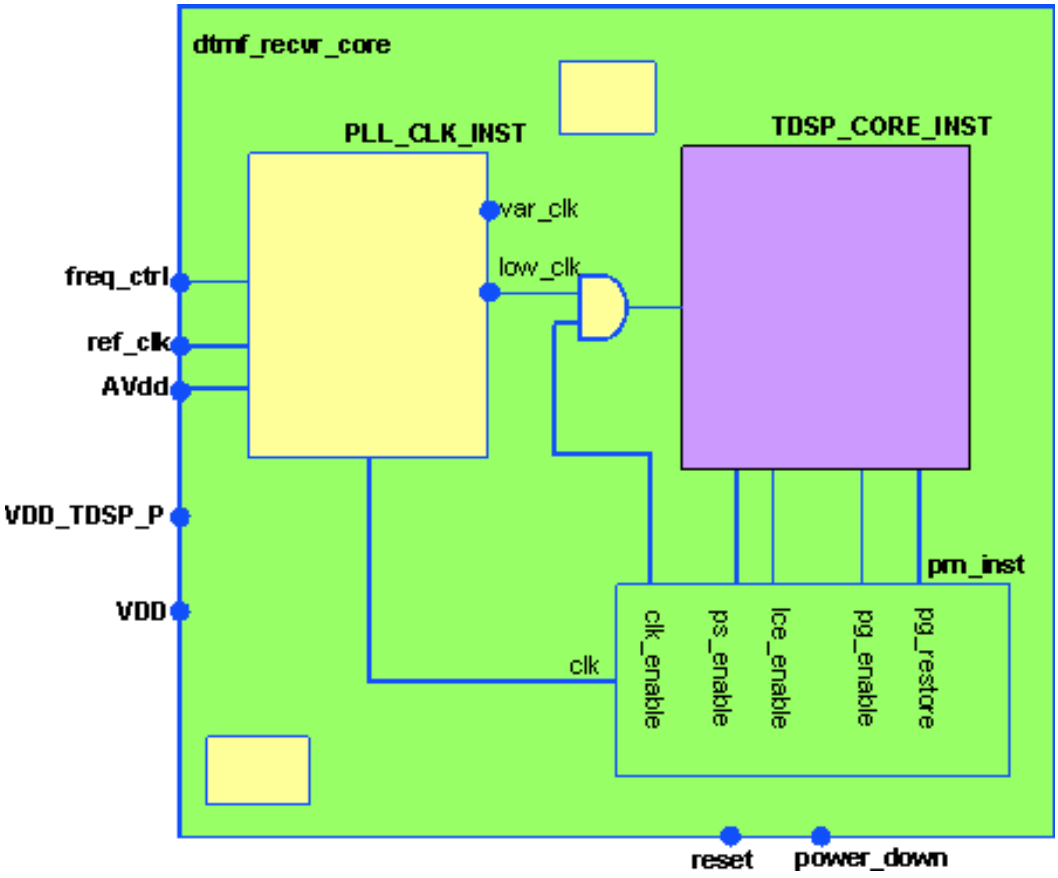
© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Lab 3-1 Running the Basic Synthesis Flow

Objective: To run the basic synthesis flow for a design.

The sample design provided with this module is a Verilog description of a *dual-tone multifrequency* (DTMF) receiver. In a telephone network, DTMF is a common in-band signaling technique used for transmitting information between network entities. DTMF signals are commonly generated by touch-tone telephones.



DTMF Chip

You need the following files to run the synthesis:

- Synthesis library
- Verilog or VHDL netlist, preferably at the RTL level
- Constraints

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab Directory Structure

The lab directory structure is as shown here:

```
ls genus_labs/
```

- captable/ //Contains lab captable file for PLE
- constraints/ //Contains SDC (constraint) files
- def/ //Contains the floorplan DEF file
- etc/ //Contains additional information
- libraries/ //Contains the libraries
- power/ //Contains CPF files
- rtl/ //Contains HDL files
- sim/ //Contains simulation files and TCF files from simulation
- tcl/ //Contains the backup of all scripts
- work/ //This is where you run your labs

Starting Genus™

1. Change to the ***work*** directory by entering this command:

   ```
   cd genus_labs/work
   ```

2. Start the software in Legacy mode by entering this command:

   ```
   genus -legacy_ui -legacy_gui
   ```

   You can type commands interactively at the legacy_genus*:/*> shell prompt.

   The command shell that starts the Genus is dedicated to the Genus (legacy_genus) Legacy mode shell and Legacy GUI. You must view files in a separate terminal window and not in the Genus shell.

3. Write a template file.

   ```
   write_template -split -outfile run.tcl
   ```

   This creates a *run.tcl* and a *setup_run.tcl* file. These two files are used to run the synthesis and optimization of this design.

4. Open a separate terminal window and change to the *work* directory.

   ```
   cd genus_labs/work
   vi ../tcl/setup_run.tcl
   ```

   You can use your favorite editor to view the *../tcl/setup_run.tcl* file.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

The *setup_run.tcl* contains all the variables needed for the design. The following list shows a few of the important variables from the *setup.tcl* file:

- The *DESIGN* variable is set to *dtmf_recvr_core*.
- The *RTL_LIST variable is set to the list of RTL/HDL files.*

You can use these and other variables from the *setup.tcl* file to complete the commands in the *run.tcl* file that you generate later in this section.

The search paths to point to the directories are as follows:

```
set_attribute init_lib_search_path {../libraries/…} /
set_attribute script_search_path {../tcl} /
set_attribute init_hdl_search_path {../rtl} /
```

Tip: When writing Tcl, multiple directories must be separated by a space and enclosed in double quotes or flower braces. (Cases where variables must be used, use double quotes.)

Loading Libraries and Designs

1. Source the setup file.

```
source ../tcl/setup_run.tcl
```

- This file has the attribute that sets the library:

```
set_attribute library $LIB_LIST /
```

The *library* attribute is set at the root level.

Tip: You can get help on all the library attributes by entering:

```
get_attribute -help *library* *
```

Tip: You can use the *get_attribute* command to get help on any attribute.

List the libraries that are loaded into Genus.

Answer: _____

Use the ***ls*** command.

Example: *ls /libraries*

For this lab, you can ignore the warnings printed by the tool.

Are all the libraries specified by the $LIB_LIST from setup.tcl read in correctly?

Answer: _____

- This file has the attribute that sets the LEF library to turn PLE on:

```
set_attribute lef_library $LEF_LIST /
set_attribute cap_table_file $CAP_TABLE_FILE /
```

The physical layout estimation (PLE) is automatically set to *ple* when you read an LEF file. Use *get_attribute interconnect_mode* to check the wire delay estimation method used by the tool.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Hereafter, instead of sourcing the *run.tcl*, execute, use the individual commands to identify and understand them.

2. Read the RTL files using the following command:

   ```
   read_hdl $RTL_LIST
   ```

   You can ignore the warning messages.

3. Elaborate the design.

   ```
   elaborate $DESIGN
   ```

   When you elaborate the design, it displays some warning messages. In a real design scenario, you typically attend to the messages in detail.

   Also, the elaboration must finish with the "Done elaborating...." message.

   In this lab, you look for unresolved instances.

   Most messages you see are CDFG-#. You can find these messages later in the /*messages* directory of the *genus* shell.

   List the designs that are loaded into Genus.

   Answer: _____

   Is there only one top-level design?

   Answer: _____

   Are there any unresolved references or blackboxes in the design?

   Answer: _____

Resolving all the design references is very important before proceeding to synthesis, so that your synthesis results are accurate. Also, you will not be wasting your time doing hours of synthesis without having all the proper design files.

4. Check for **unresolved** references.

   ```
   check_design -unresolved
   ```

   Are there any unresolved instances?

   Answer: _____

5. Identify the missing module.

   What is the name of the module that is missing?

   Answer: _____

   Do you remember the difference between a module and an instance?

6. Locate the RTL file containing the missing module.

   Which UNIX directory are the RTL files located in?

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Answer: _____

Which file contains the module definition for the missing module?

Answer: _____

7. Modify the *setup_run.tcl* to include the missing RTL file in the $*RTL_LIST* variable.

8. Resolve the unresolved instances.

   a. Delete the design from the memory of the tool.

   ```
   rm $DESIGN
   ```

   Important: If there are multiple top-level designs in your */designs* directory, then delete all of them before reading the designs.

   b. Source the updated *setup_run.tcl* file, read the RTL, and elaborate the design.

   ```
   source ../tcl/setup_run.tcl (updated)
   read_hdl $RTL_LIST
   elaborate $DESIGN
   ```

   Is the elaboration done and complete?

   Answer: _____

   Do you have one top-level design?

   Answer: _____

   If the answer is **no**, repeat this entire step.

   Note: You can repeat previous commands from the history by using your arrow keys.

9. Do *not* close the software.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

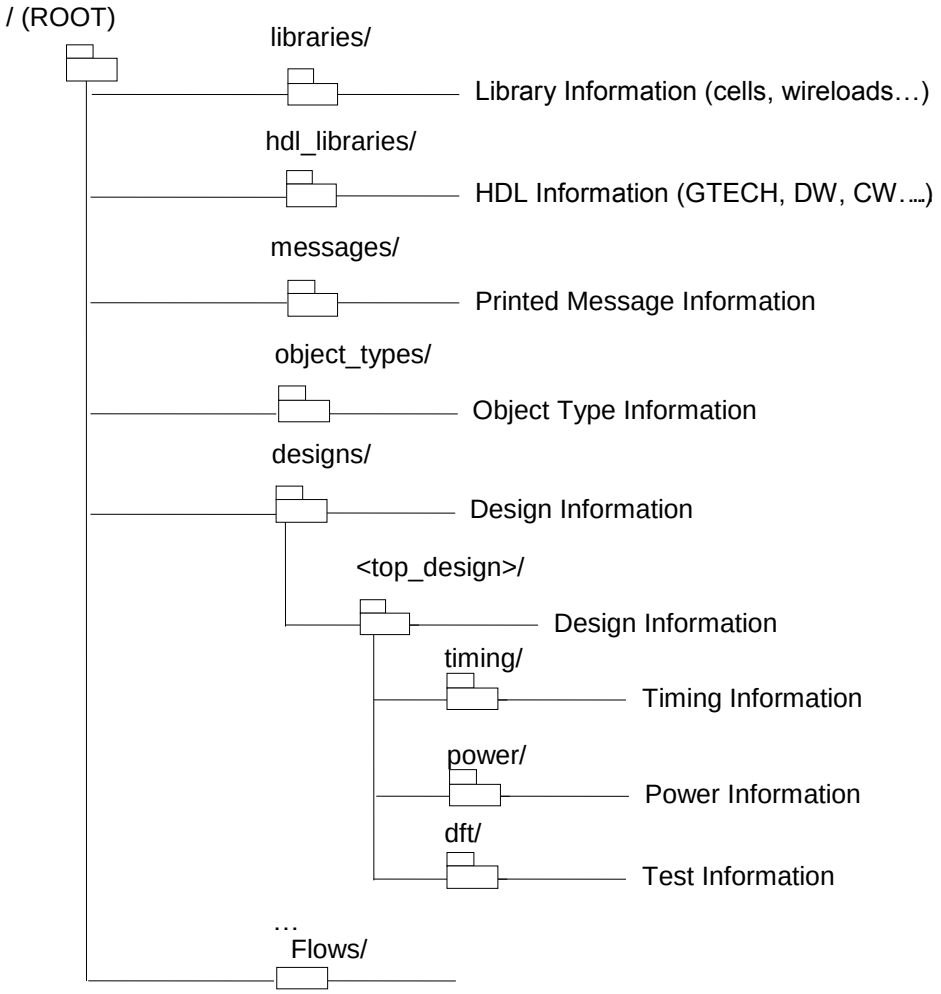
© Cadence Design Systems, Inc. Do not distribute.

Lab 3-2 Navigating the Design Hierarchy

Objective: To use the navigational commands to explore the hierarchy and to view the attributes of design objects.

Design Hierarchy

The following figure illustrates the hierarchy of the design objects.



The directory structure within the Genus Legacy shell is similar to the UNIX directory structure. Make sure you enter your commands in the legacy_genus shell.

Filtering Objects by Type and Name

1. List all the attributes of a design object in the design hierarchy by entering:

   ```
   ls -l -a [find / -design *]
   ```

   Notice that this *ls* command works just like the UNIX command in Genus Legacy mode.

 © 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Also, notice that the wild characters like * are allowed in the command usage.

You can also use the *more* command to break up the output.

```
more ls -la
```

2. As in the previous step, find the attributes of the following design objects:
   - subdesign **tdsp_core**
   - instance EXECUTE_INST
   - port **refclk**
   - port **reset** (top-level port)
   - pin **reset** in the **EXECUTE_INST** instance

Filtering Objects Based on Attribute Value

1. Check for latches in your design by entering:

```
filter latch true [find / -instance *]
```

2. Find the ports that are bidirectional.

```
filter direction inout [find / -port *]
```

   Notice that the *direction* attribute has values other than *true* or *false*.

3. Change to the top-level design directory.

```
cd /designs/dtmf_recvr_core
```

4. Navigate through the design hierarchy and the subdirectories and list the attributes of some of the design objects.

Using Procedures

You can use the Tcl language to write and use command procedures that perform scripting functions.

1. Open a separate window and examine the **list_subdes.*tcl*** file in the *tcl* directory.

   The Tcl-based script finds all the subdesigns in the design.

2. In the Genus shell, change back to the root directory.

```
cd
```

3. Load the **list_subdes.tcl** script into the Genus by entering:

```
include list_subdes.tcl
```

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

4. Enter the following command:

```
list_subdes
```

This command returns a list of all the subdesigns.

What is the difference between using this procedure and a simple find command that lists all the subdesigns in the design?

Answer: _____

Many such procedures are in the *load_etc.tcl* file in the software installation directory. Use *include load_etc.tcl* to load these procedures into the Genus.

5. Do *not* close the software.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 3-3 Reading SDC Constraints

Objective: To read the SDC constraints of your design into the software and to debug any constraint issues and problems.

In this lab, you read and debug SDC constraints.

Reading SDC Constraints

1. Read the constraint file.

   ```
   read_sdc ../constraints/constraints.sdc
   ```

   The SDC file has some errors. The log file reports the errors and the reasons.

   Are there any syntax errors? If so, what are they?

   Answer: _____

   You can also view and search the log file.

   Are there any other failed commands? If so, what are they?

   Answer: _____

Debugging Failed SDC Constraints

1. Open a separate window and examine the ../*constraints*/*constraints.sdc* file.

2. You can write these errors into a file by entering this command:

   ```
   echo $::dc::sdc_failed_commands > failed.sdc
   ```

   The syntax errors are not reported by this command.

   The transcript and the log file reports the errors and the reasons.

3. Optional Step.

   a. Fix the syntax errors and the failed commands.

      If you need help, refer to the Constraints Appendix in your lecture book.

   b. Save the fixed commands into the *constraints.sdc* file.

      Note: To reset the timing of the design, use the *reset_design* command.

   c. To read the constraints again, reset the timing of the design and read the updated SDC file.

      ```
      reset_design
      read_sdc ../constraints/constraints.sdc (updated file)
      ```

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Are there any errors?

Answer: _____

If the answer is yes, repeat this step.

4. If you did not complete the previous optional step, read the constraints using the following commands:

```
reset_design
read_sdc ../constraints/dtmf_recvr_core.sdc
```

Analyzing Missing SDC Constraints

1. Check for any missing constraints by entering:

```
check_timing_intent
```

What types of messages are reported?

Answer: _____

Are there any real missing constraints?

Answer: _____

If any, how would you fix the missing constraints?

Answer: _____

In this lab, you do not have to fix these missing constraints.

2. Do *not* close the software.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 3-4 Synthesizing the Design and Using the Graphical Interface

Objective: To synthesize your design and to use the graphical interface to debug your design.

Synthesizing the Design

1. Set the Generic Synthesis effort using attribute.

   ```
   set_attribute syn_generic_effort medium
   ```

   By default, generic synthesis is run using *medium* effort.

2. Synthesize the design to generic gates. It takes a list of top-level designs and synthesizes the RTL blocks to generic gates using the given constraints and performs RTL optimization.

   ```
   syn_generic
   ```

3. Set the effort level for mapping to technology library and optimization.

   ```
   set_attribute syn_map_effort medium
   set_attribute syn_opt_effort medium
   ```

4. Synthesize the design to technology gates and optimize it.

   ```
   syn_map
   syn_opt
   ```

   This command maps the design to the cells described in the supplied technology library and performs logic optimization.

   When you synthesize the design, the tool shows you informational messages on how your synthesis is being done. You can also view the log file in a separate window to view the same results.

   In a real design scenario, you typically attend to these messages in detail.

5. Report the synthesis results.

   ```
   report_qor
   ```

   Is the design meeting timing?

   Answer: _____

   What is the total area and power of the design?

   Answer: _____, _____

   What is the total run time and memory usage to this point?

   Answer: _____, _____

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

6. Write out the following:
   - The netlist

     ```
     write_hdl > ${DESIGN}_lab1.v
     ```
   - The constraints

     ```
     write_sdc > ${DESIGN}_lab1.sdc
     ```

   Cadence® Genus Synthesis Solution does not use a database. Changes to the netlist and constraints are *not* written out unless explicitly specified.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Viewing Logical Hierarchy, HDL and Schematic

When the synthesis is complete, run the following steps to use the graphical interface.

1. View or unhide the graphical interface.

   ```
   gui_show
   ```

   The interface displays four main windows:

   - Hierarchy window
   - HDL window
   - Schematic window
   - Layout window (This window is enabled after placement with the Genus_*Physical*_opt license or if you read in a DEF file along with the LEF files.)

   These windows are dynamically refreshed to identify the logical hierarchy you are currently in.

2. View the contents of the *Schematic* window.

   You can control the attributes to be displayed by choosing **File – Preferences – Schematic**.

   a. Click and drag towards the **bottom-right corner** in the window.

      This action zooms into the area.

   b. After you zoom in, use the arrow keys on the keyboard to pan across the schematic window.

   c. Use the *Search* icon to find the **RESULTS_CONV_INST** instance. 

   d. Click ***RESULTS_CONV_INST*** to select it.

      Use the **f** key to fit the schematic into the window if you zoom into the wrong area.

   e. **Double**-click ***RESULTS_CONV_INST***.

      The hierarchical instances under *RESULTS_CONV_INST* are shown in the schematic window.

   f. Select any instance, pin, or net. Then, hold the mouse over the design object.

      The name of the design object is displayed.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Note: If you modify the schematic display preference, by choosing **File – Preferences – Schematic**, then the information such as power domains are displayed along with the object's name. You can then choose **File – Preferences –Save Preferences** to save the graphical interface preferences in the ~/*.cadence*/genus.*gui* file. The next time Genus is started, it uses the saved preferences.

g. Click and drag towards the **bottom-left corner** in the window.

   This action zooms out of the area.

3. View the contents of the **Hierarchy** tab.

   a. Click **RESULTS_CONV_INST** *[results_conv]* to select it.

   b. **Right**-click **RESULTS_CONV_INST** *[results_conv]*.

   A pop-up menu appears.

   You can view the HDL or the schematic of the instance in a separate window. You can also view the attributes of the instance from this pop-up menu.

   In order for the cross-probing between the windows and the HDL to work, you must set the *hdl_track_filename_row_col* attribute to *true* before elaborating the design. For this lab, this attribute has been set through the *setup_run.tcl* file.

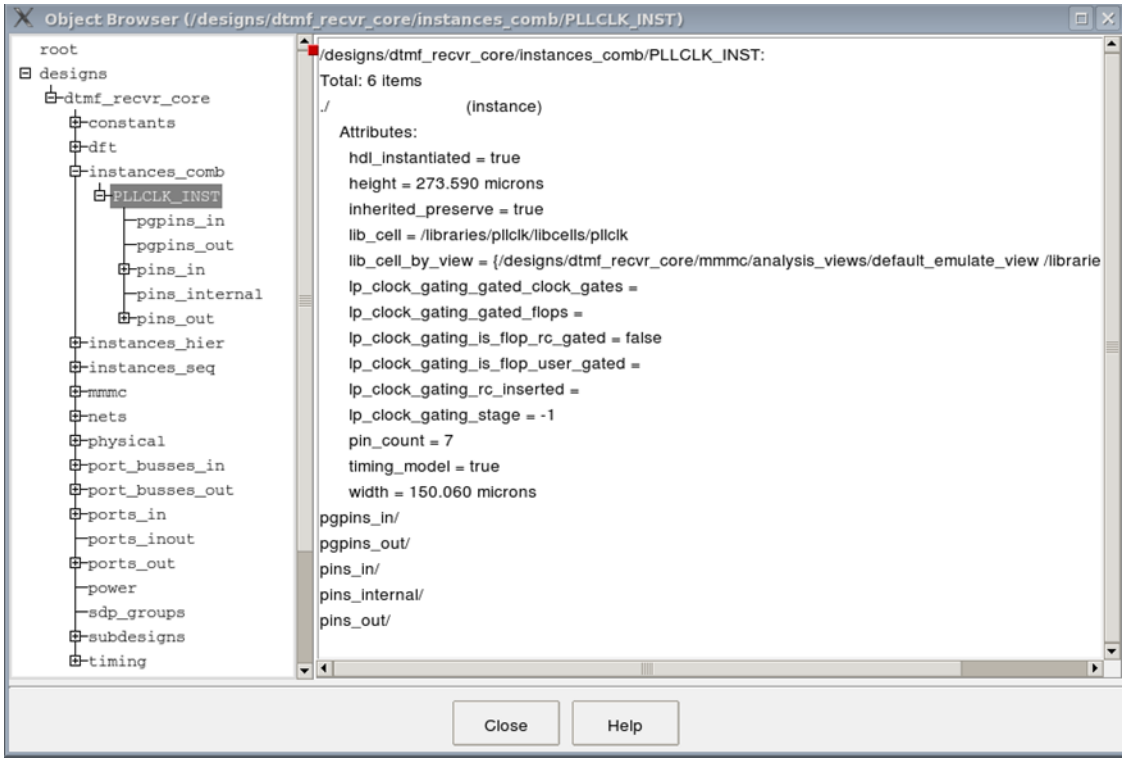
   c. **Right**-click ***dtmf_recvr_core***, select **Open In – Object Browser**.

   - Browse through each category by clicking the **+** symbol.

     This option expands each category to show the design objects under that category. If the category listed is a design object, then the software also shows the set of attributes that are set or computed for that design object.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.



- Choose ***PLLCLK_INST*** under /*designs*/*dtmf_recvr_core*/*instances_comb*.

  The attributes of *PLLCLK_INST* are displayed.

Is this instance a blackbox or a timing model?

Answer: _____

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Generating Reports

1. Choose **Timing – Report – Worst Slack**.

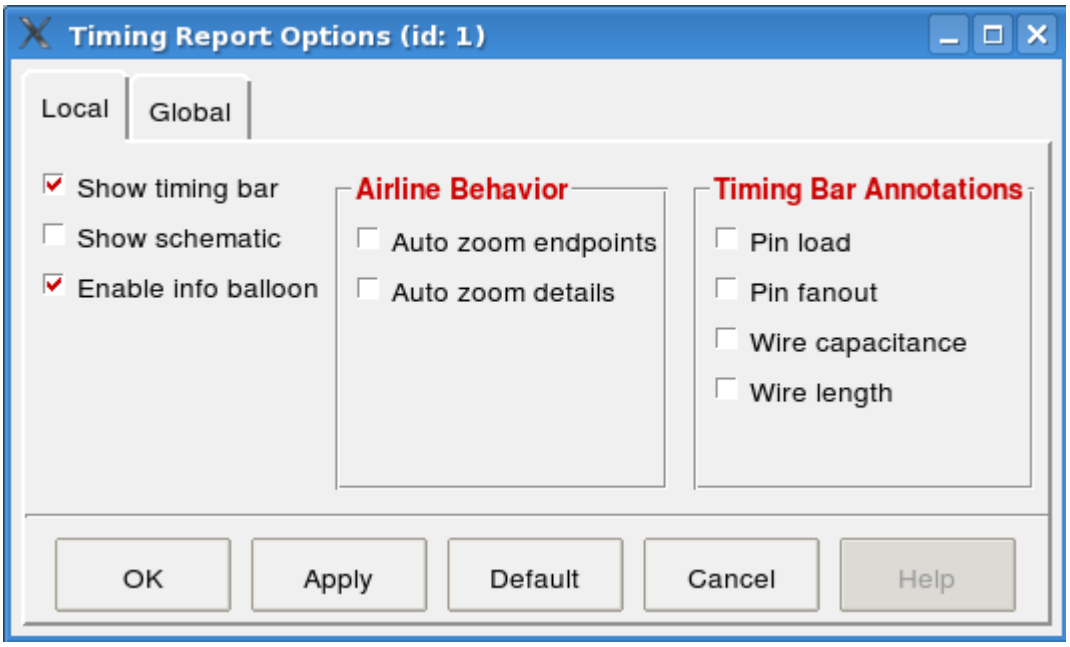
   A window displays the worst path and the schematic of the worst path with all the instances in the critical path.

   a. Explore the timing report.

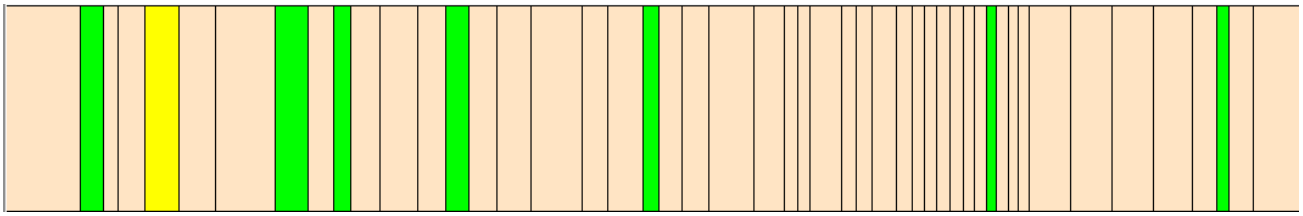
      Notice that the critical path is extracted and the components are shown.

   b. On the left of the timing window, click the **Options** button.

   c. In the *Timing Report Options* window, choose **Show timing bar** and click **OK**.



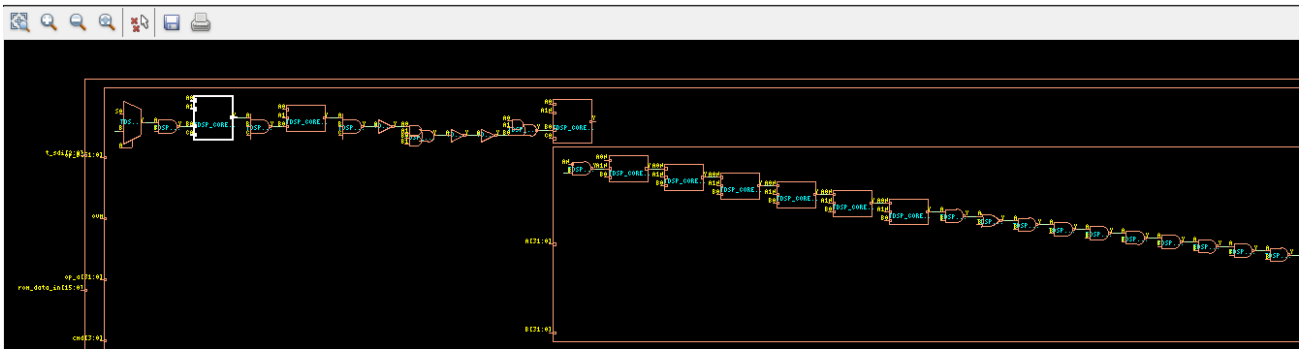
   This shows you a timing bar.



© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

If the **Show schematic** option is also chosen, it displays the corresponding schematic window below timing bar.



d. Using the mouse, hover over each segment in the timing bar. The tool shows the corresponding instance.

e. **Middle**-click to select the objects in the timing bar to view the object's name in the textual report.

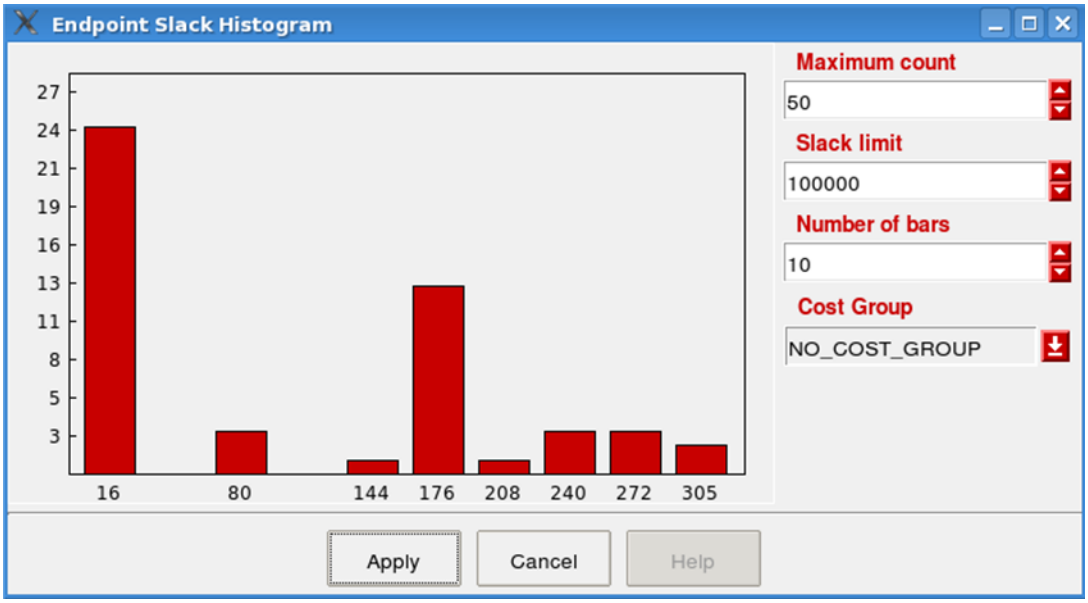
What type of library cell is causing the worst delay in the worst path?

Answer: _____

f. Click **Close** to close the window.

2. Choose **Timing – Report – Endpoint Histogram**.

A window pops up.



a. Set the Maximum Count to **50** and click **Apply**.

b. **Double**-click one of the histograms.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Another window pops up with all the paths. You can view the most critical path for that slack group and the corresponding schematic view.

c. Click the worst path to select it.

d. Go to extreme left and click the **Options** tab.

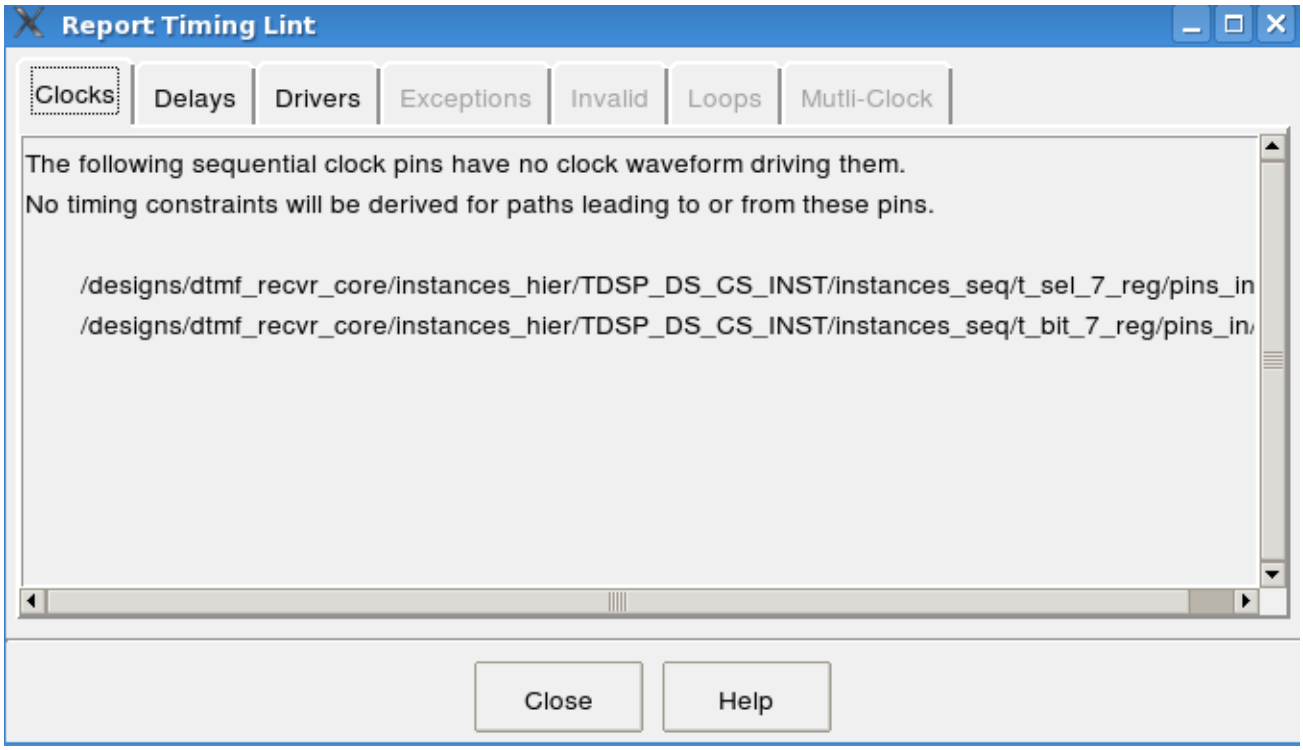
e. A window pops up. Choose **Show timing bar** and click **Apply**.

f. **Middle**-click to select the objects in the timing bar to view the object's name in the textual report.

g. **Middle**-click a different timing endpoint to show the timing report for that endpoint.

3. Choose **Timing – Report – Timing Lint**

A window pops up.



What types of issues are being reported?

Answer: _____

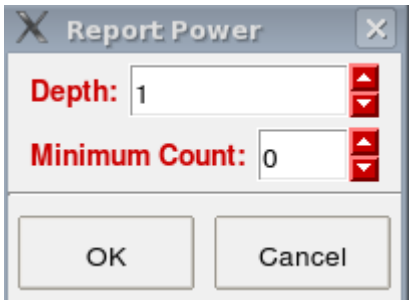
4. Choose **Power – Report – Detailed Report**.

a. A window pops up.

b. Choose a depth of **1** and click **OK**.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.



c. A depth of **1** allows you to view the report for the top level and the blocks directly under the top level.

d. View the report.

Note: Switching power is the combination of internal power and net power. The total power dissipated by the design is the sum of switching power and leakage power of the design.

Which is worse for this design: Leakage Power or Switching (Dynamic power)?

Answer: _____

What is the block that dissipates the most power?

Answer: _____

5. Choose **Power – Report – Instance Power Usage**.

This shows the relative power dissipation of the blocks directly under the top-level design in the form of a pie chart.

What is/are the worst instance(s) for this design?

Answer: _____

What is the percentage power that the block(s) dissipate(s)?

Answer: _____

6. Explore the rest of the menus.

- The **File – Report – Datapath** menu shows reports for area, components and MUXes.
- The DFT menu shows reports for scan synthesis. A scan must be performed for most of these reports to work.
- The **File – Report – Netlist Statistics** menu allows you to view the reports for type, instances, area and area percentage.

7. Close the software by entering:

```
exit
```

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Choose **Control-c** to stop some optimization commands from further execution. Control-c also closes the software when pressed twice within five seconds. Make sure you close the software by using the *exit* command and not using Control-c.

Summary

The labs in this module run the basic synthesis flow. The navigation helps you in redirecting your efforts to the most important aspects of your design.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 3-5 Accessing Common UI GUI Using Legacy Mode

Objective: To explore common UI GUI using Genus Legacy mode.

Genus now supports a new Graphic User Interface which is aligned with the Innovus GUI to provide support for the Common User Interface.

Starting the Genus Common UI GUI Using Legacy Shell

1. Change to the ***work*** directory by entering this command:

   ```
   cd genus_labs/work
   ```

2. Start the software in Legacy mode by entering this command:

   ```
   genus -legacy_ui
   ```

   Important: Do not use ***legacy_gui*** while launching software to use Common UI GUI capabilities.

   You can switch to the legacy GUI by launching Genus with the ***-legacy_gui*** option.

3. Source the script which runs setup and synthesize the design.

   ```
   source ../tcl/cui_gui_run.tcl
   ```

4. Open the GUI using:

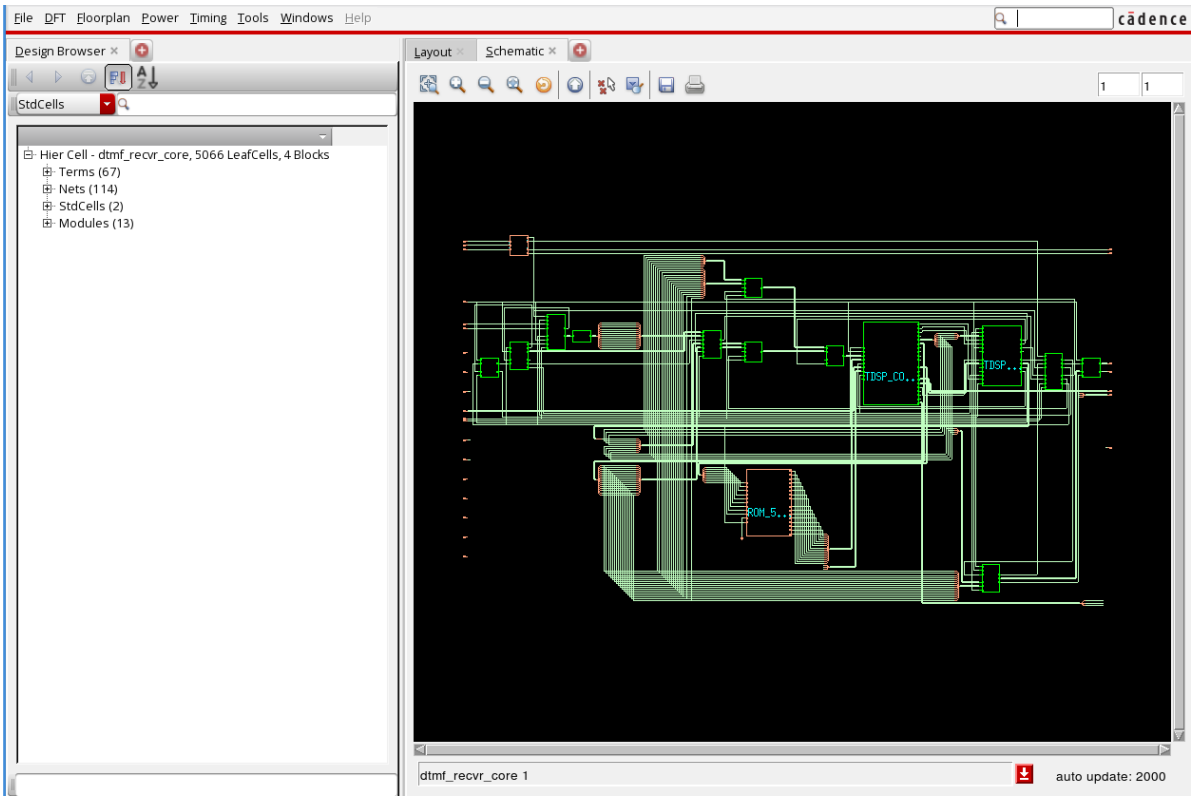
   ```
   gui_show or gui_raise
   ```

   The new GUI contains these main features:

   - Menu bar
   - Viewers
     - Design Browser
     - Layout Viewer
     - Schematic Viewer
     - HDL Viewer
     - Object Attributes
   - Layer Control
   - Toolbar
   - Status Bar

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.



© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

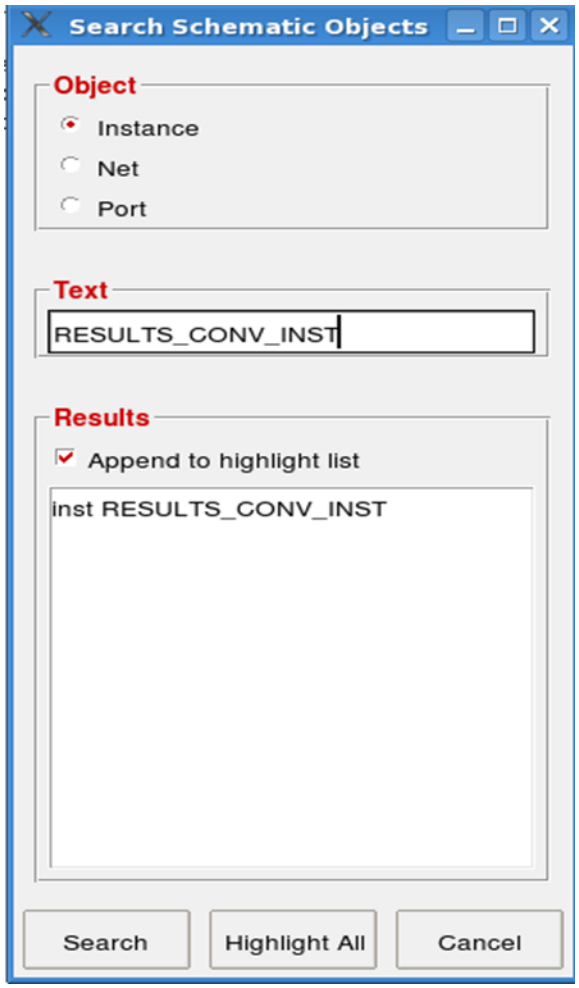
Exploring Different Viewers of GUI

1. View the contents of the *Schematic* Viewer.

   a. Use the *Search* icon to find the **RESULTS_CONV_INST** instance. 

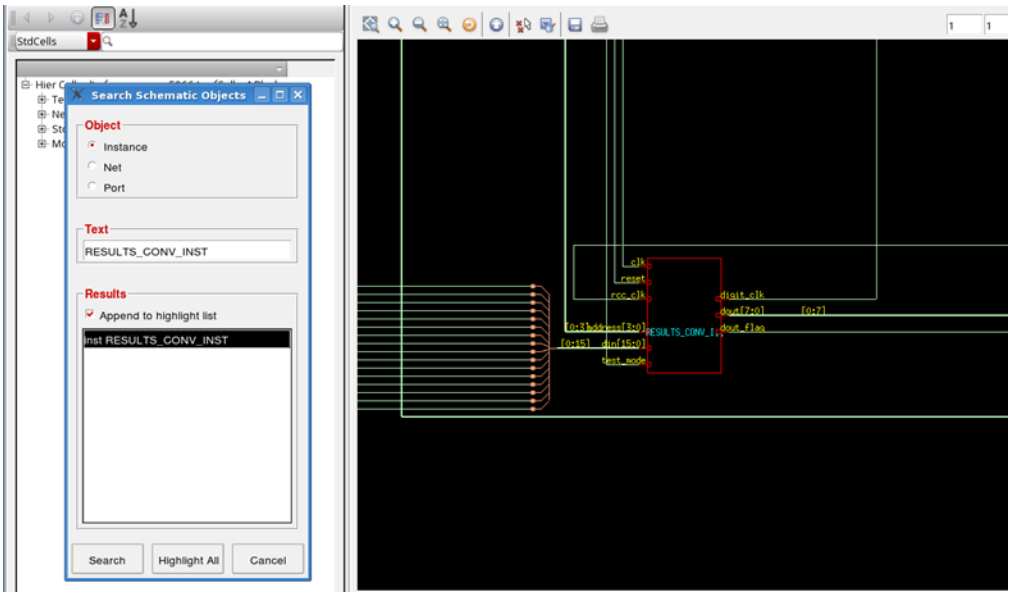
   b. Click ***RESULTS_CONV_INST*** to select it.

   Use the **f** key to fit the schematic into the window if you zoom into the wrong area.



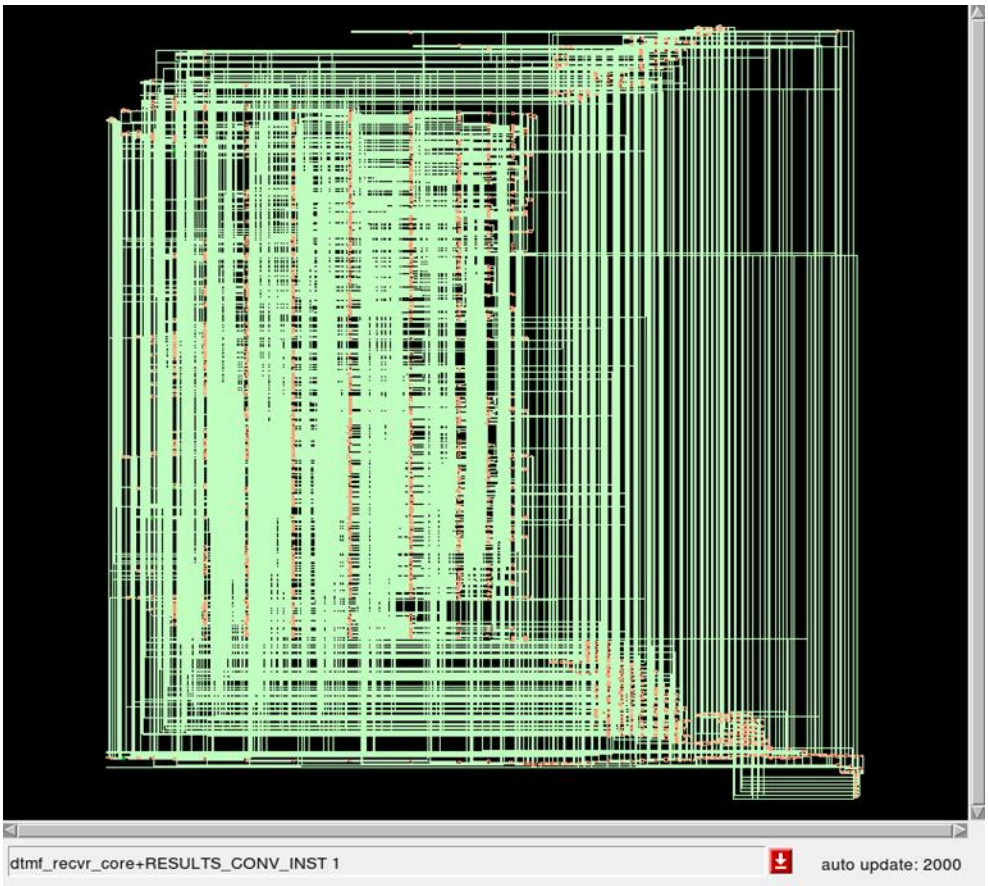
© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.



c. **Double**-click ***RESULTS_CONV_INST***.

The hierarchical instances under *RESULTS_CONV_INST* are shown in the schematic window.



© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

d. Select any instance, pin, or net. Then, hold the mouse over the design object.

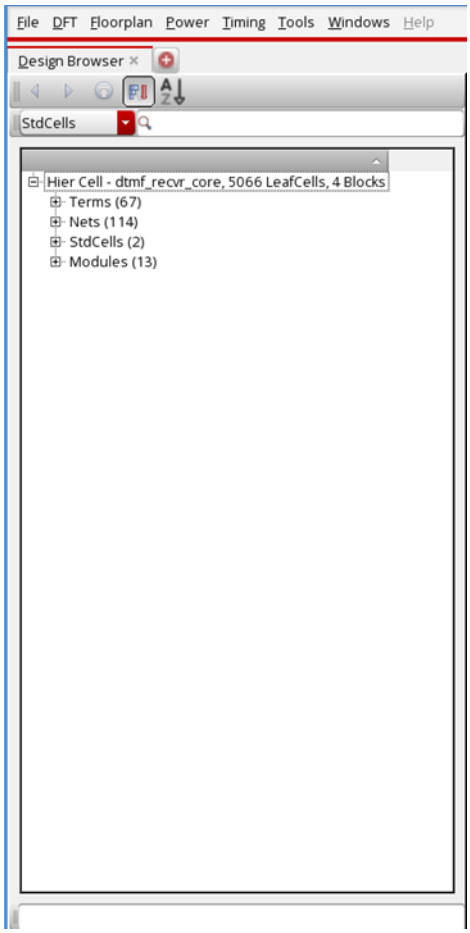
The name of the design object is displayed.

2. View the contents of the Design Browser.

The Design Browser is categorized based on object types. The following are major categories:

- Hier Cell
- Modules
- Terms
- Nets
- StdCells
- Search bar
- Status bar

a. Click on the **+** sign next to different categories and explore the objects.



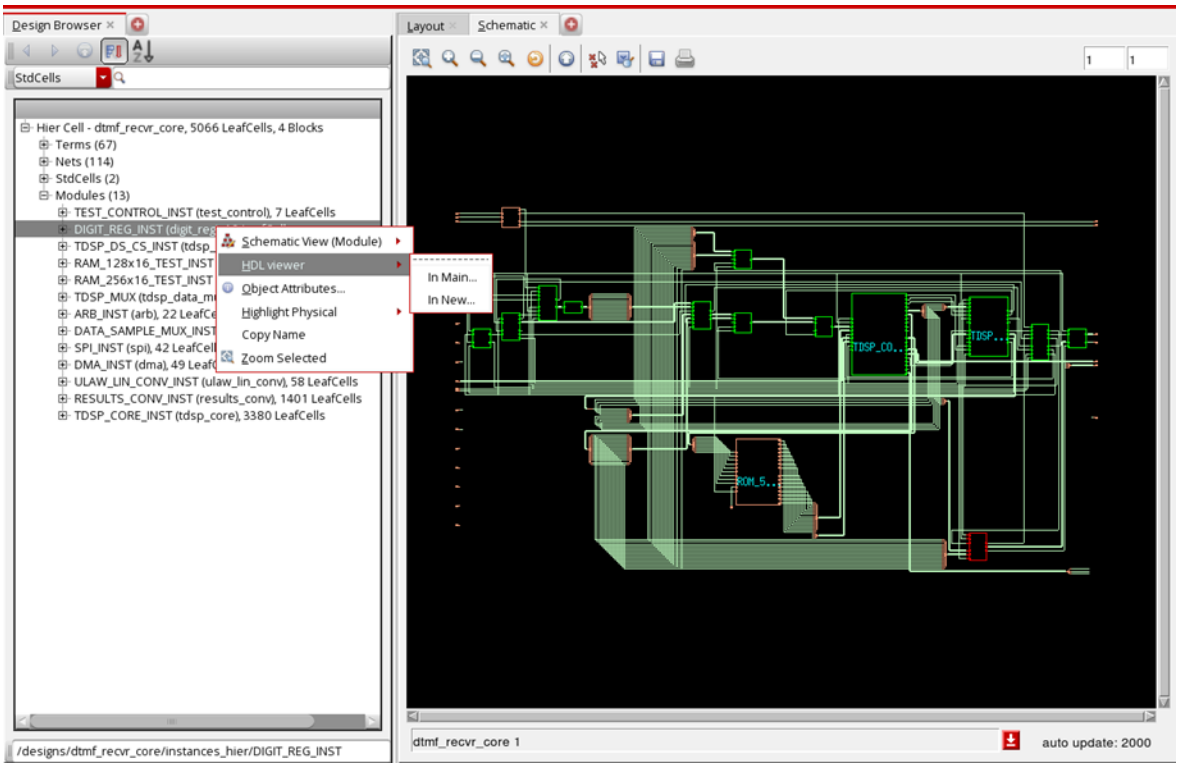
© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

3. Explore cross probing in the Design browser.

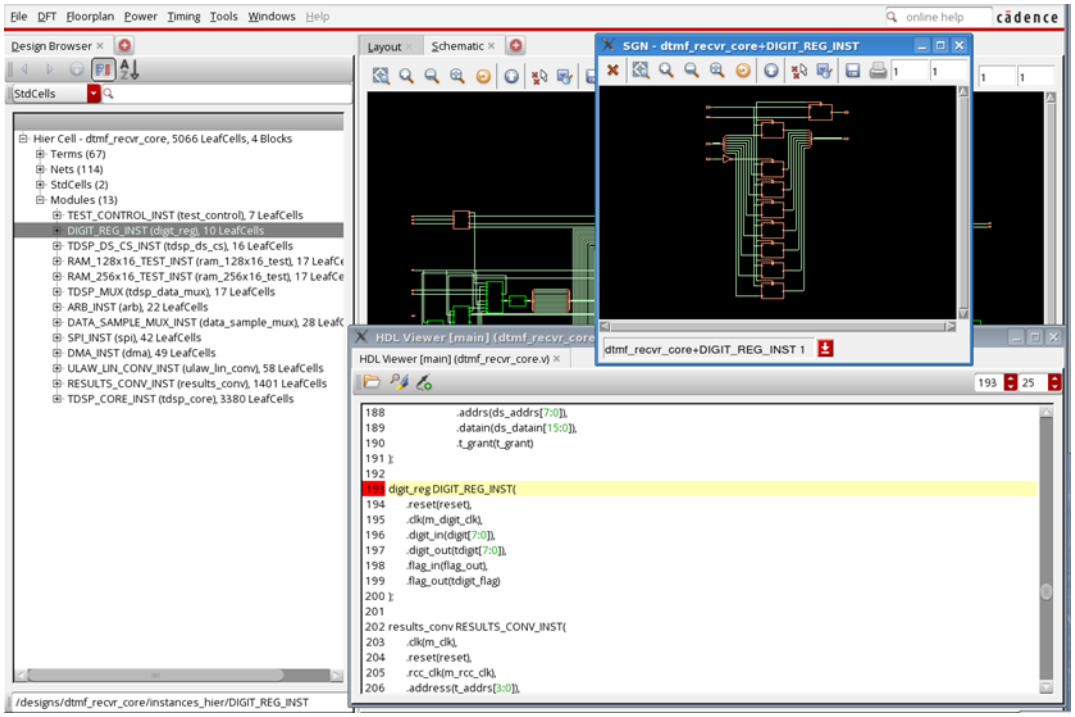
   To cross-probe any object, **right**-click on the object name. This opens the context menu. From the context menu, you can choose to cross probe the design in the HDL Viewer or the Schematic Viewer.

   a. **Right**-click on **DIGIT_REG_INST** under Modules to open a new HDL and Schematic view.



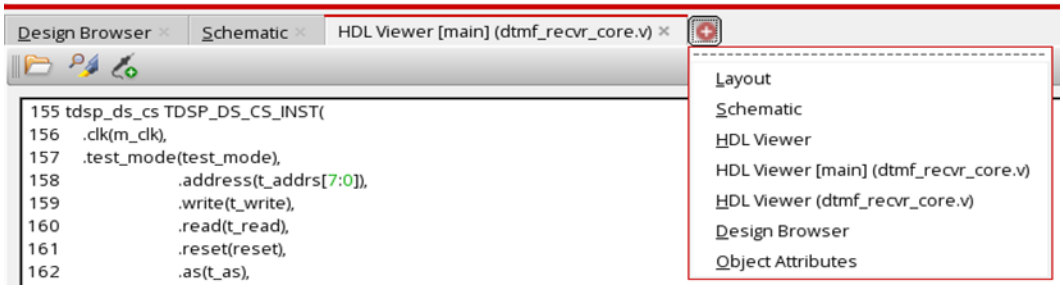
© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.



4. Explore Object Attributes.

   a. Click on the **+** sign to open the Object Attribute viewer.

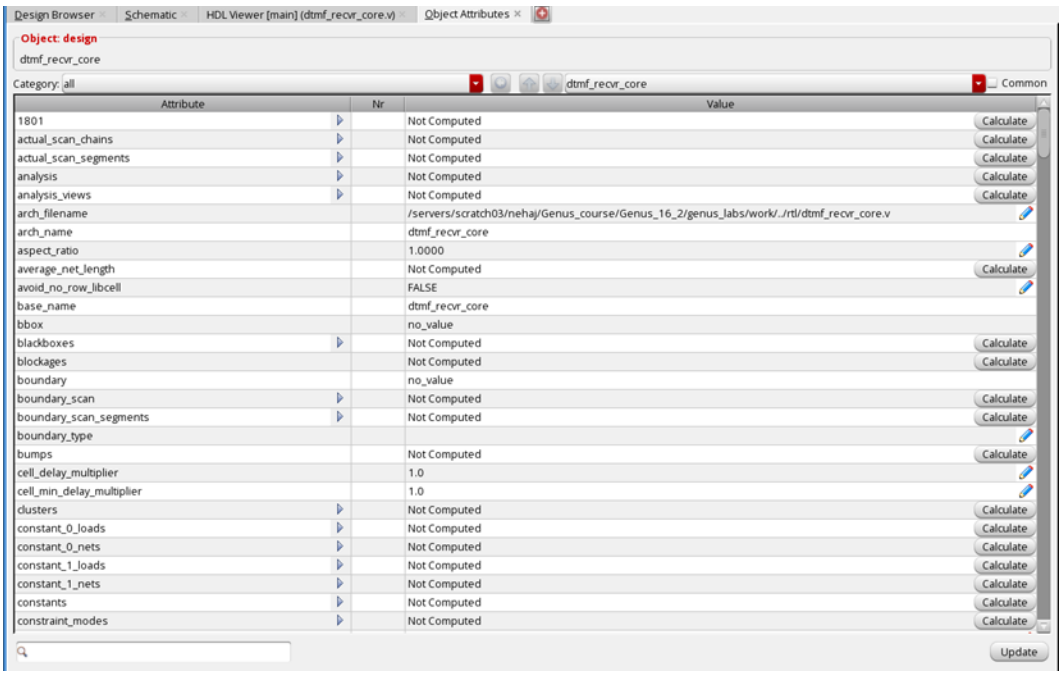


The *Object Attributes* displays the values of the attributes of an object or of the complete design.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

b. Explore different attributes and their values in this view.



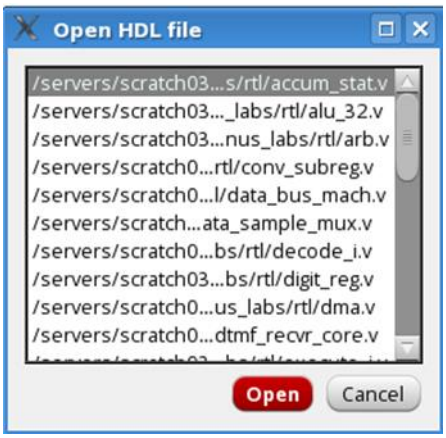
5. Explore the HDL Viewer.

a. Click on the **+** sign to open HDL viewer.

This viewer helps to view the RTL and cross-probe the design across other viewers. To cross probe the design from the HDL Viewer, cross probing needs to be enabled from the toolbar.

b. Open an HDL file by clicking the **Open HDL** file icon in the toolbar

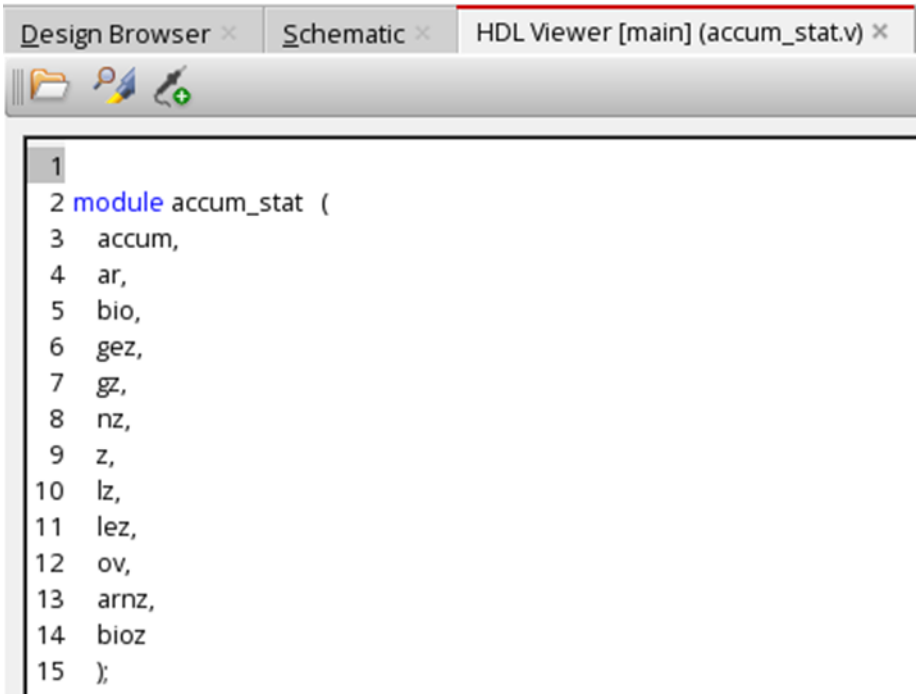
c. Select file *accum_stat.v* to open it.



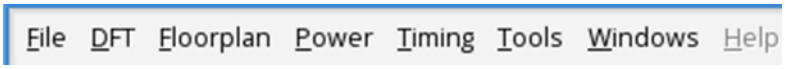
© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

d. Content of the file will get displayed in the HDL viewer.



6. Explore different reporting tabs present in the Menu bar.



7. Exit the software by entering:

```
exit
```

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Module 4: Datapath Synthesis (Optional)

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Lab 4-1 Running Datapath Synthesis (Optional)

Objective: To run data path synthesis on the design to analyze the data path components.

Datapath Synthesis

This design is *with* hierarchically separated, related operators.

1. Change to the lab directory by entering:

   ```
   cd genus_labs/work
   ```

2. Start Genus™ with the following command:

   ```
   genus -legacy_ui -legacy_gui -f ../tcl/template.dp.tcl -log dp.log
   ```

   This step takes about five minutes.

   The *suspend* commands are added in the script to allow reviewing the various stages of the flow. Use *resume* to continue the script execution, according to the given instructions.

3. When run gets suspended for the first time, enter the command:

   ```
   report_dp
   ```

   What data path components can you see?

   Answer: _____

   Are there any external MUXes present?

   Answer: _____

   What is the percentage of data path modules before synthesis?

   Answer: _____

   What is the cell area for data path modules at this stage?

   Answer: _____

4. Continue with the run using the command:

   ```
   resume
   ```

5. When run gets suspended again, generate a data path report after generic synthesis and compare the result with the report after elaboration

   Are there any external MUXes present now?

   Answer: _____

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Is there any change in percentage of data path modules compared to the previous report?

Answer: _____

What is the cell area for data path modules now?

Answer: _____

6. Continue with the run using the command:

```
resume
```

7. Once the run is finished, analyze the synthesis log.

Has the tool done any carrysave optimization?

Answer: _____

How many CSA groups are created during carrysave optimization?

Answer: _____

8. Close the software.

```
exit
```

Summary

In this lab, you have run *carrysave* optimization on the related operators.

The tool automatically overrides the hierarchies and performs ungrouping on smaller modules to improve optimization during *syn_map*. This can be controlled by the *auto_ungroup* attribute.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Module 5: Debug Design Scenarios

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

There are no labs for this module

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Module 6: Genus Physical Synthesis

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Lab 6-1 Exploring Optimization Strategies Using PLE

Objective: To synthesize and optimize the design, to analyze the synthesis results, and to use the PLE mode to generate wire delay.

Setting PLE Mode and Optimization Attributes

In this section, you set the PLE mode to determine wire delays.

1. Change to the ***work*** directory.

   ```
   cd genus_labs/work
   ```

2. View the **setup.opto.tcl** and **ple_run.tcl** files in the *tcl* directory and make sure the following commands are listed and uncommented:

   - Read the LEF library and the capacitance tables into the software:

     ```
     set_attribute lef_library <LEF_LIST> /
     set_attribute cap_table_file <CAP_TABLE_FILE> /
     ```

     These commands automatically set the *interconnect_mode* attribute to *ple*.

3. Start the software by entering this command:

   ```
   genus –legacy_ui -legacy_gui -f ../tcl/ple_run.tcl -log ple.log
   ```

   This command loads the libraries, reads the design, elaborates the design, and reads the constraints.

4. List the cost groups in /*designs*/*dtmf_recvr_core*/*timing*/*cost_groups*.

   Cost groups are automatically generated for each *create_clock* encountered in the SDC file by the *read_sdc* command.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Running Generic Synthesis

1. Report the datapath components in the design.

   ```
   report_dp
   ```

   List a few types of datapath components from the report.
   Are there any merged components?

   Answer: _____, _____

   At this point, the design is in an elaborated stage, and you should not see any merged components.

2. Run a generic optimization of the design using *medium* effort.

   This step takes about five minutes on Linux machines.

   List a couple of the optimizations reported by the tool.

   Answer: _____, _____

3. Report the datapath components in the design again.

   ```
   report_dp
   ```

   List a few types of datapath components from the report.
   Are there any merged components?

   Answer: _____, _____

Mapping to a Target Library

1. Map the design using **medium** effort.

   ```
   set_attribute syn_map_effort medium
   syn_map
   ```

   This step takes about five minutes on Linux machines.

   You can view the several stages in the synthesis process as it runs. You can also check the log file for a log of the synthesis process.

   What is the target slack for the m_clk cost group?

   Answer: _____

   What is the Global Incremental target slack for m_clk cost group? How does it compare to the previous target slack at Global Mapping Target Info (% difference in terms of clock period)?

   Answer: _____, _____

   Note: A target slack difference within 10% can be considered a good target. Otherwise, check the constraints for any unrealistic numbers before proceeding any further.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

What is the total negative slack (Group Total Worst Slacks) after the optimization step?

Answer: _____

Optimizing the Design

1. Run commands.

```
set_attribute syn_opt_effort medium
syn_opt
```

This step takes about five minutes on Linux machines.

What is the worst timing slack reported for the m_clk cost group?

Answer: _____

What is the end point reported for the worst slack?

Answer: _____

What is the total area of the design after optimization?

Answer: _____

What is the total power of the design after optimization?

Answer: _____

Generating and Analyzing Reports

1. Report the timing of the design.

What is the worst delay listed on the report with a fanout listed? What is the name of the instance with this delay? What library cell of this instance?

Answer: _____, _____, _____

How does the delay of this instance compare to instances with similar library cells that have similar or lesser fanout?

Answer: _____

Use the following commands to help determine the delays of cells.

```
filter libcell [find /lib* -libcell BIGFANCELL]  [find /des* -instance *]
report_timing -through [find / -instance BIGFANINST]
```

Note: Here BIGFANCELL is used as a general name for the instance with fanout. Use the actual instance name that you got from the timing report when you run the filter and report_timing commands.

2. Report the timing to the *EXECUTE_INST/acc_reg[6]/D* pin.

What is the timing slack on this report?

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Answer: _____

Use the following command to pipe to the *report* command:

```
find [find / -inst EXECUTE_INST/acc_reg[6]] -pin D
```

Does this path have any timing exceptions?

Answer: _____

3. Report the design rule violations by entering:

```
report_design_rules
```

Are there any design rule violations?

Answer: _____

4. Report the gate count by entering:

```
report_gates
```

What is the total instance count?

Answer: _____

5. Report the power consumption of the design by entering:

```
report_power -depth 1
```

What is the total leakage power consumption of the design?

Answer: _____

What is the total dynamic power consumption of the design?

Answer: _____

What is the block that consumes the most power?

Answer: _____

Note: The design includes some RAM and ROM modules. Because they are custom blocks (hard macros), Genus Synthesis Solution does not modify their structure during the compiles. These blocks are still resolved and timed correctly, and their power consumption and area are included in the various reports.

6. Report the quality of results by entering:

```
report_qor
```

7. View the reports to find the following information:

Cell (leaf instance) count in the design

Answer: _____

Cell area

Answer: _____

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Number of sequential elements

Answer: _____

Is there a timing violation?

Answer: _____

Saving Synthesis Results

1. Save the design files by entering:

```
write_design -base_name outputs/$DESIGN $DESIGN
```

2. Close the software.

Summary

In this lab, you synthesized the design using the PLE mode.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 6-2 Running Physical Synthesis

Objective: To run physical synthesis.

Starting the Lab

1. Start the Genus legacy using the following commands:

```
genus -legacy_ui -legacy_gui -f ../tcl/phys_run.tcl -log phys.log
```

2. Review the **phys_run.tcl** file in detail.

   The following are a few important attributes:

   - innovus_executable
   - invs_temp_dir
   - invs_preload_script
   - invs_postload_script

Loading Libraries and Designs

The initial set of commands load the libraries, LEF files, captable file, read the HDL files, and elaborate them; constraints and DEF files are read. Run is suspended after setting the cost groups.

Important: At this point, if you resume the script by entering *resume* you can skip entering the commands from this section. Just check the results in the log file or the reports directory.

Running Physical Synthesis in Genus

1. Set the directory for the files and database from the Genus-P run. It contains the database before going to Innovus™ and after coming from it.

```
set_attribute invs_temp_dir innovus_tmp_dir /
```

In this lab, you do not modify any Innovus attributes. Therefore, you provide dummy scripts for postload, preload, and pre-export attributes for Innovus.

2. Provide dummy preload and postload scripts for the Innovus run.

```
set_attribute invs_preload_script dummy_pre_load /
```

   This sets the path of the script that is sourced first when Innovus starts.

```
set_attribute invs_postload_script dummy_post_load /
```

   Specifies the script to include in the Implementation setup file after the setup steps (after the libraries, design, user constraints, and user modes are loaded).

 © 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.


```
set_attribute invs_preexport_script dummy_pre_export /
```

Specifies the script to include in an Innovus batch file prior to data export (that is after placement, trial route, and statistics generation are completed, but before leaving Innovus).

3. The following attribute is can be set to invoke Innovus from within Genus (Optional).

```
set_attr innovus_executable /cds/Innovus161/bin/innovus /
```

OR

```
set_attr innovus_executable <Path to Innovus Installation>/bin/innovus /
```

Important: Even if you start with the default license and do not set the Innovus executable or invoke Genus with Physical option, then when the "-physical" option is used during synthesis, Genus checks out the Physical license.

4. Run Generic level physical synthesis to place the design.

```
set_attribute syn_generic_effort medium
syn_generic -physical
```

5. Synthesize the design to gates and optimize it with physical capability.

```
set_attribute syn_map_effort medium
syn_map -physical
set_attribute syn_opt_effort medium
syn_opt -physical
```

6. Generate reports of the placement statistics and summary report table.

```
write_reports -outdir $_REPORTS_PATH -tag final
report_summary -outdir $_REPORTS_PATH
```

7. Write out the design.

```
write_design -innovus -gzip –base_name ${_OUTPUTS_PATH}/final/${DESIGN}
```

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

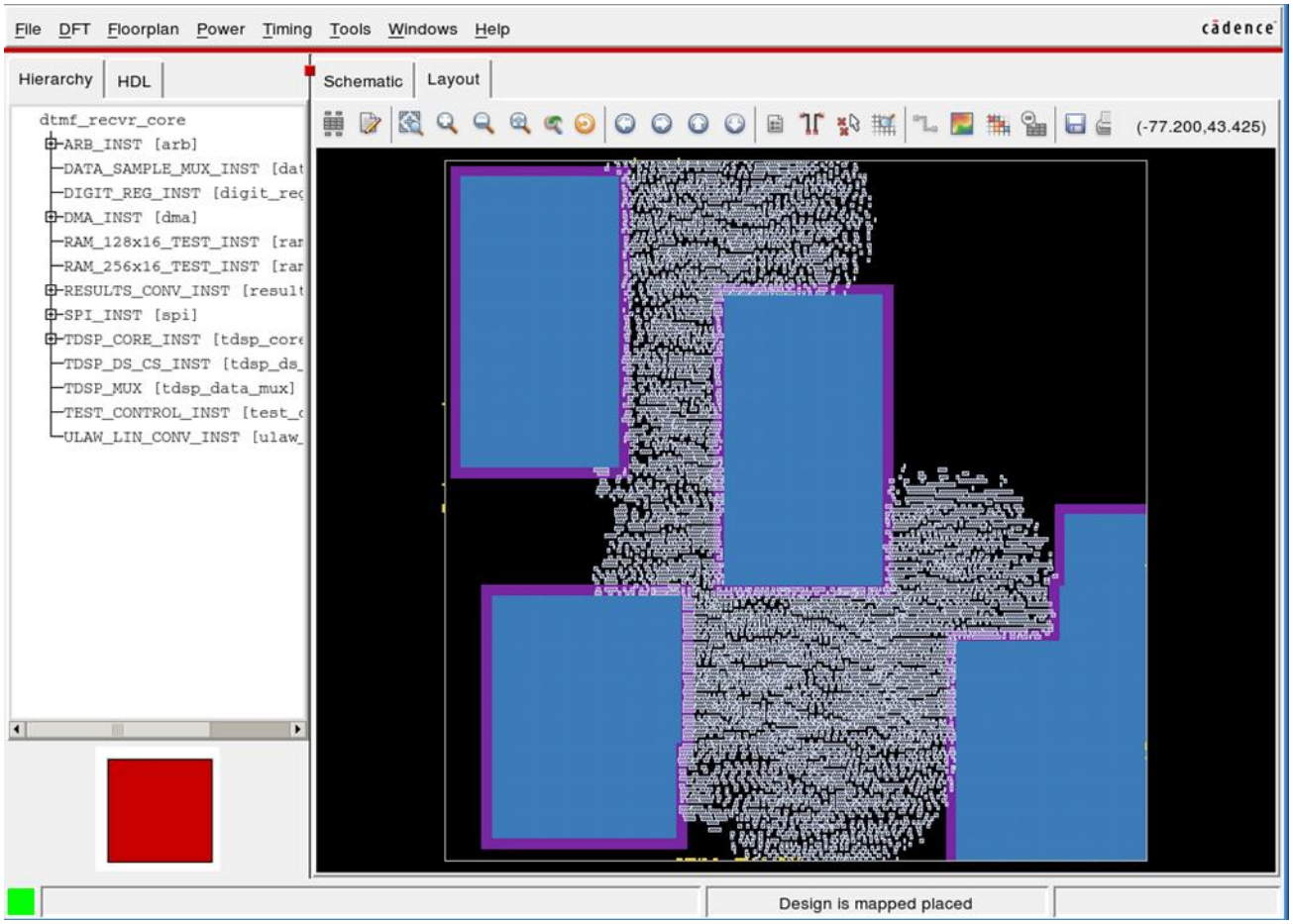
Modifying the Floorplan

1. Open the GUI using the following command:

   ```
   gui_show
   ```

2. Click the **Layout** tab.

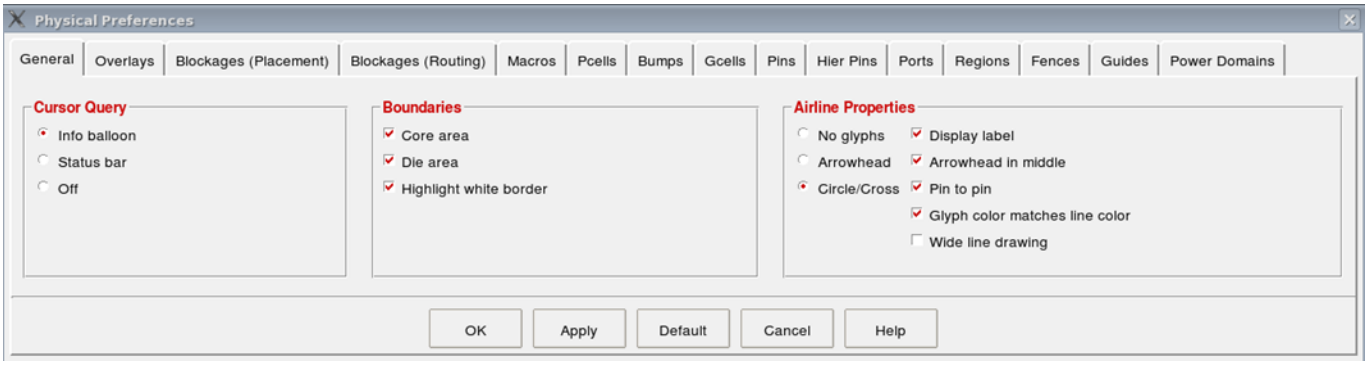
   This gives you the floorplan of the design with the standard cells and macros placed in it.



© 2016 Cadence Design Systems, Inc. All rights reserved.

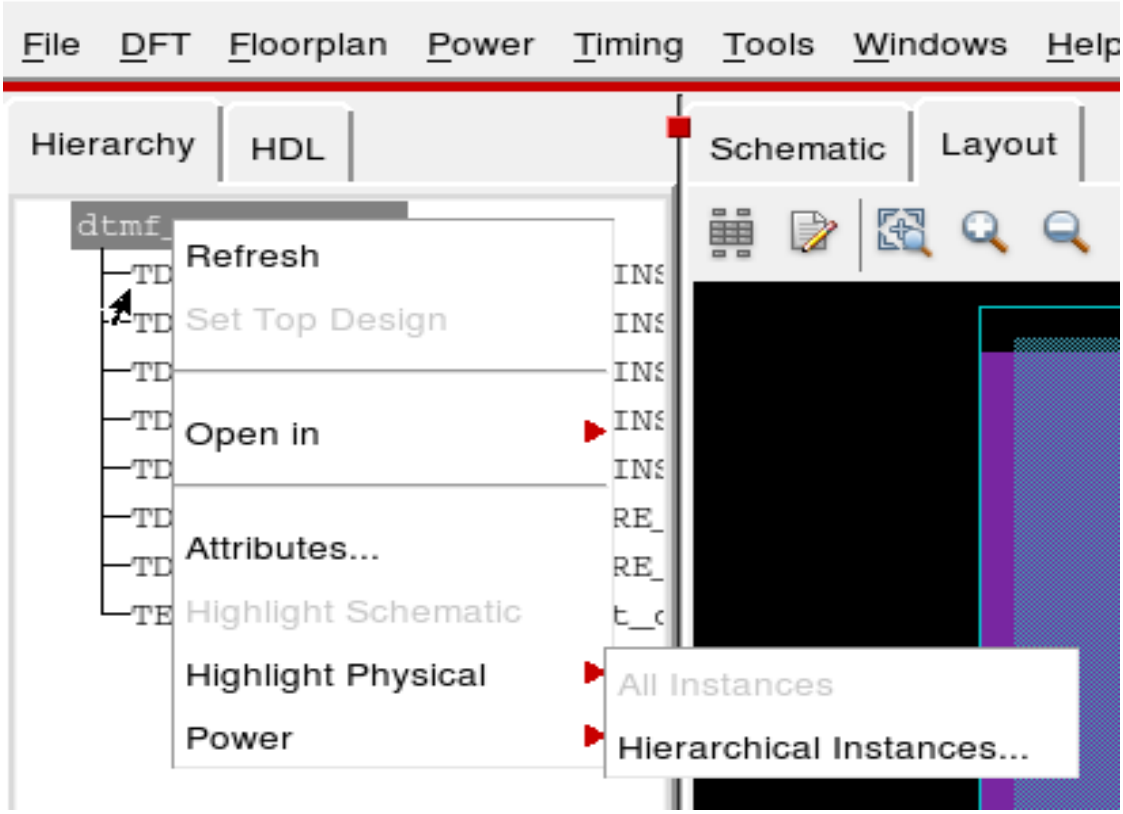
© Cadence Design Systems, Inc. Do not distribute.

Tip: You can modify the physical display preference by choosing **File – Preferences – Layout**. Then choose Physical Preferences and click **Ok/Apply** to save the graphical interface preferences in the *~/.cadence/genus.gui* file.



The next time the Genus is started, it uses the saved preferences.

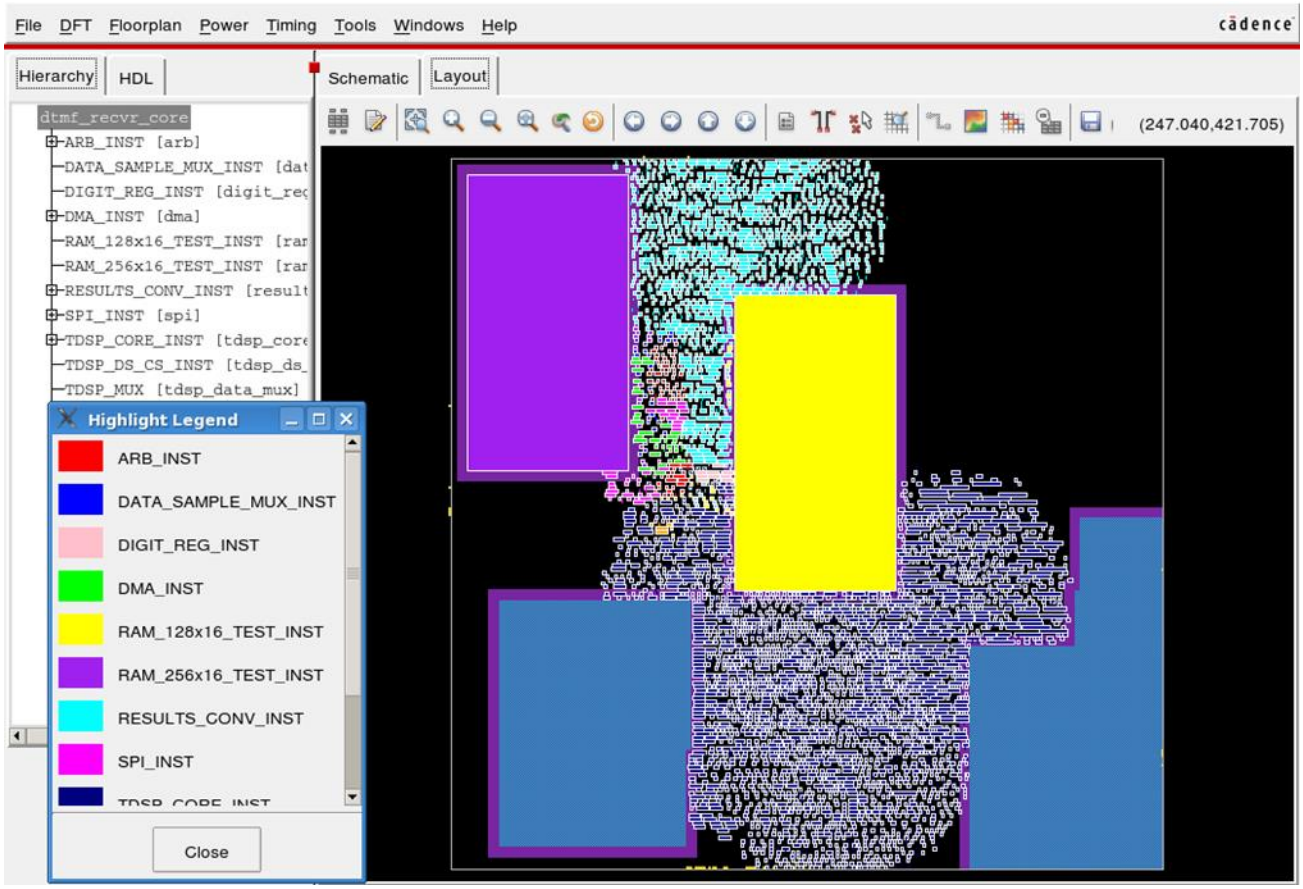
3. In the hierarchical window, **right**-click **dtmf_recvr_core** and select **Highlight Physical → Hierarchical Instances**.



© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

You see the module highlighted in the physical view.



4. In the layout window, click on any instance to select it.

5. Click and drag towards the bottom-right in the window to zoom into the area.

6. Use the arrow keys on the keyboard to pan across the physical window and scroll the view left, right, up, and down.

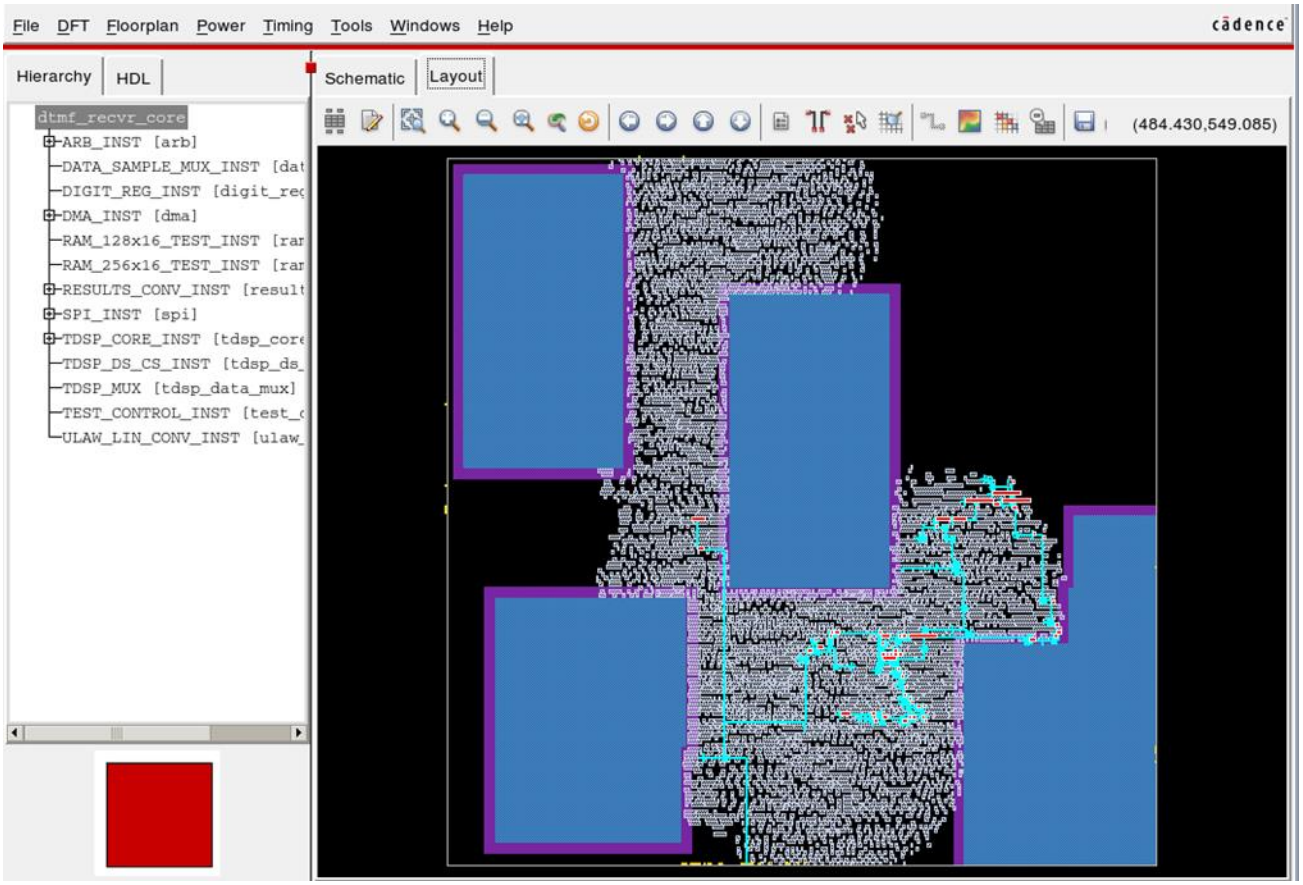
7. Show the critical path in the physical view.

```
report_timing -physical -gui
```


© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

This highlights the critical path in the physical view with airlines connecting all the cells in that path, which helps in finding congestion in the reported path.

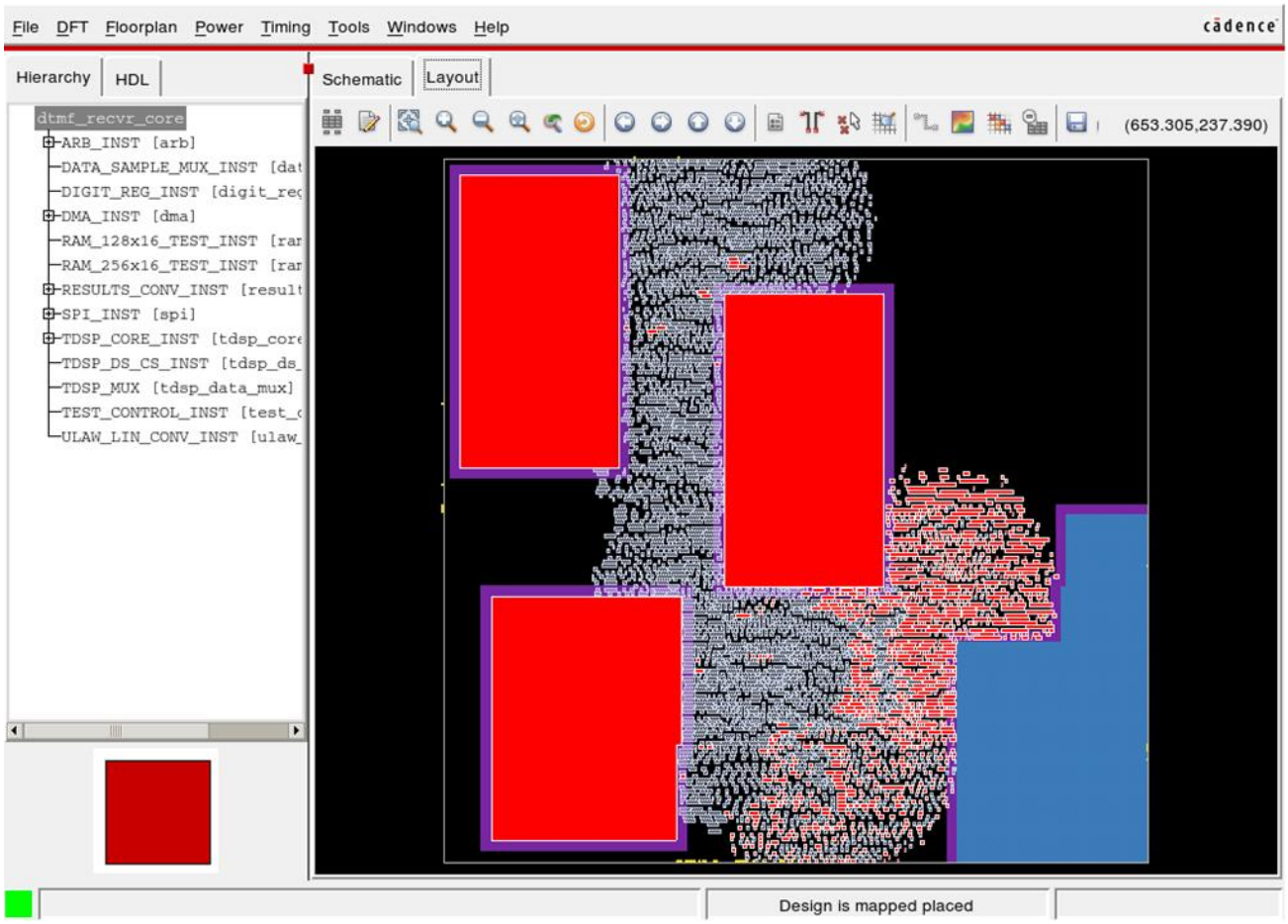


© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

8. View the fanout of any pin or port. (Type the entire name on one line.)

```
fanout -gui [find [find / -inst TDSP_CORE_INST/EXECUTE_INST/acc_reg[3]] -pin Q]
```



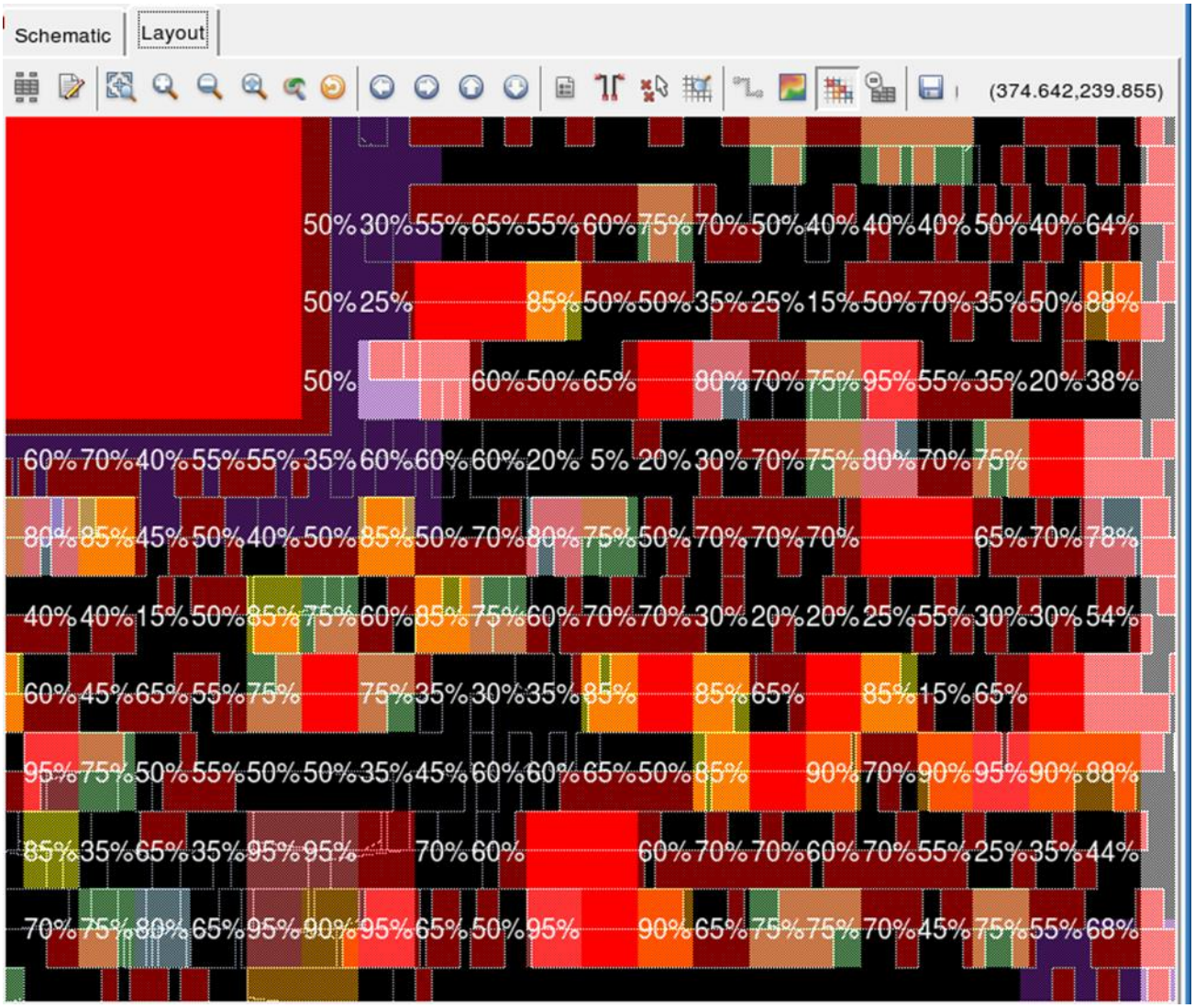
© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

9. Click the **Utilization** icon to check the area utilization map. 

   The utilization map shows the utilization area in a colored format.

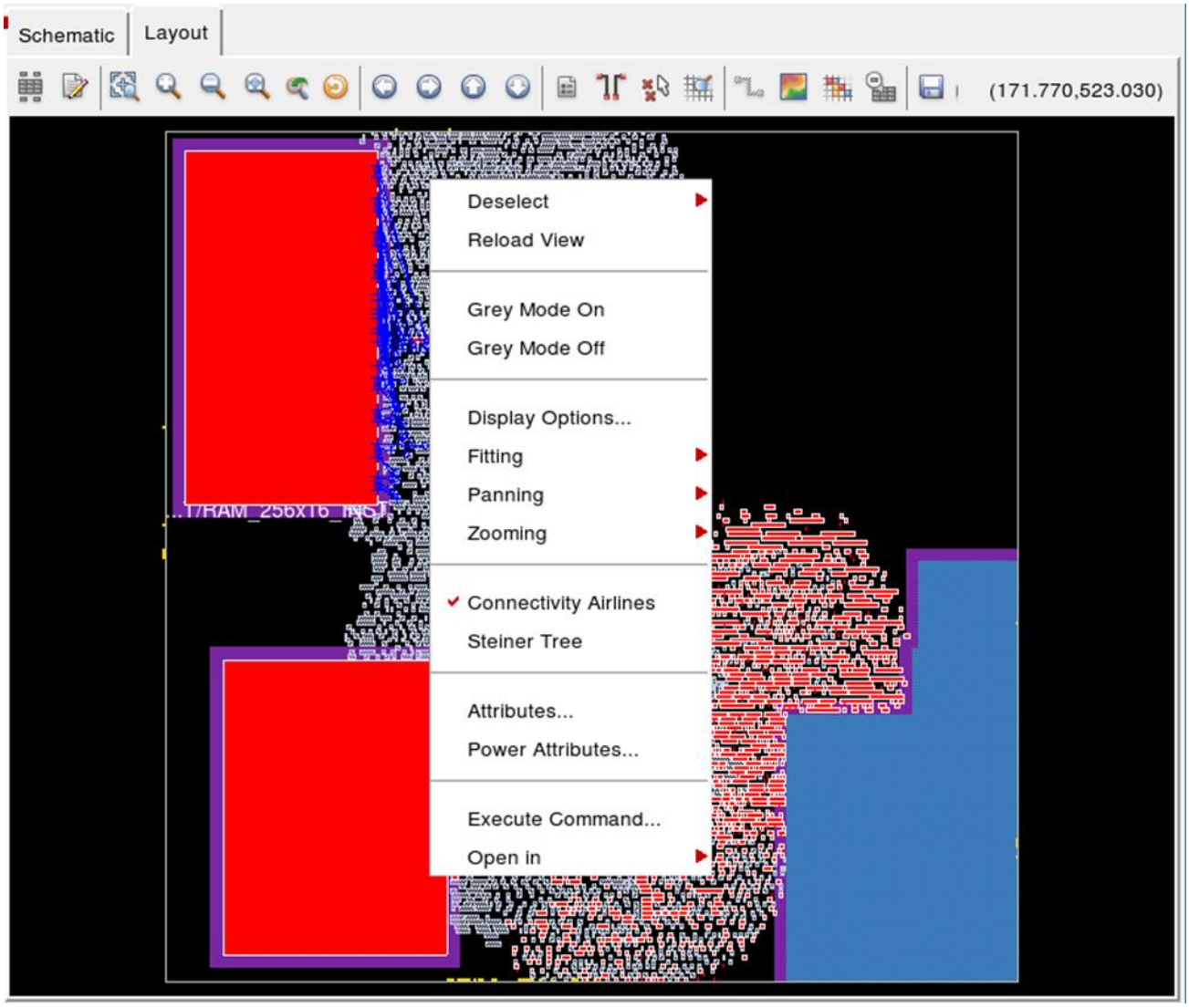
   Zoom in to an area and you will find the percentage number denoting the color.



© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

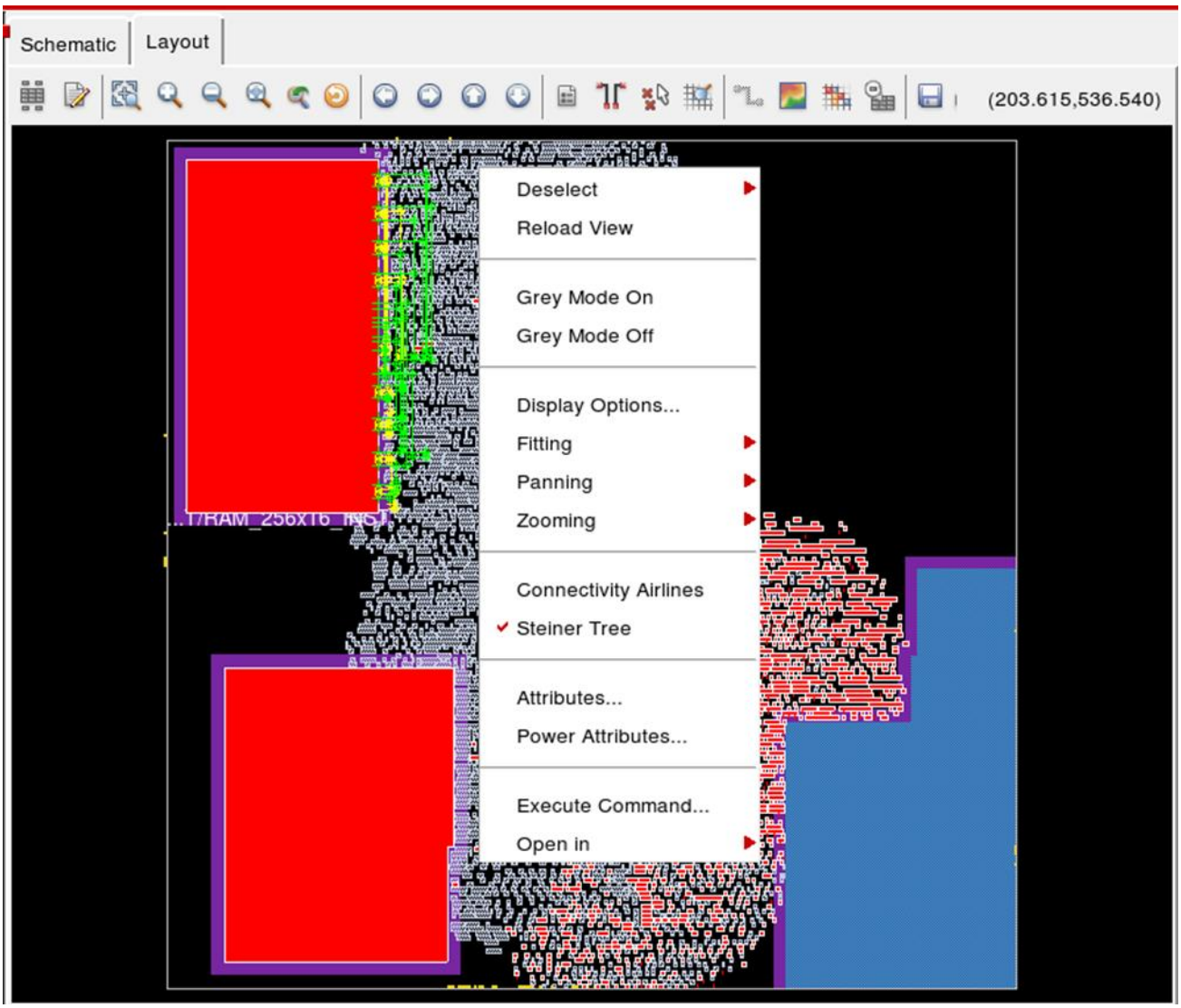
10. **Right**-click a design object and select **Connectivity Airlines** to view its direct logical connections.



© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

11. **Right**-click a design object and select **Steiner Tree** to view routing information.



Floorplan Editing Options

1. In the Layout window, click the **Edit Mode** button. 

   You see new edit options enabled in the physical window.

2. Test the options available in the **Edit Mode** to do the following:
   - Move object
   - Delete object
   - Flip object vertically or horizontally
   - Rotate object clockwise or anticlockwise

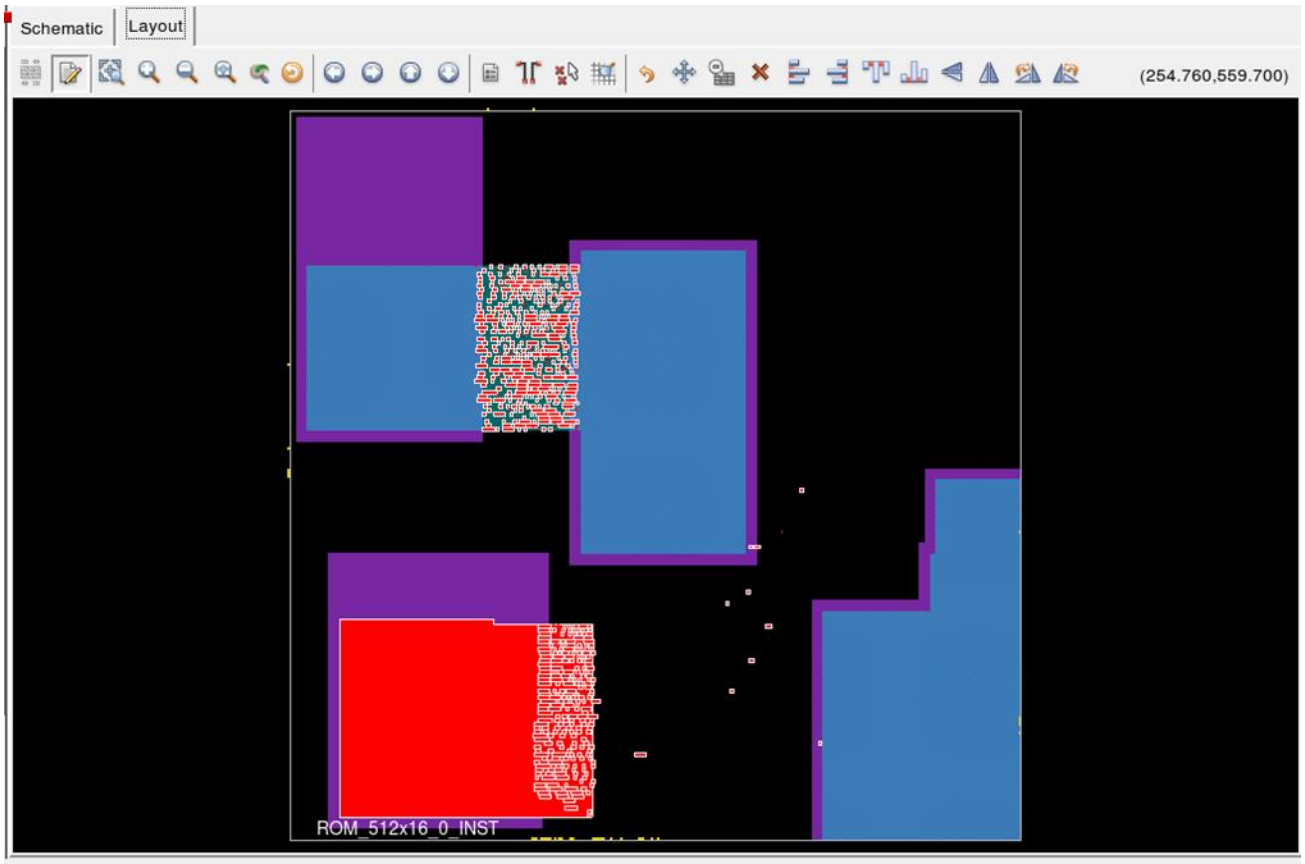
© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

3. Run the following command at the prompt after you complete any of the above tasks:

```
check_placement
```

You should see violations highlighted if you have overlapping placed instances.



4. Exit the tool.

```
exit
```

Summary

In this lab, you ran physical synthesis and applied techniques to view the floorplan in the GUI.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Module 7: Low-Power Optimization

© Cadence Design Systems, Inc. Do not distribute.

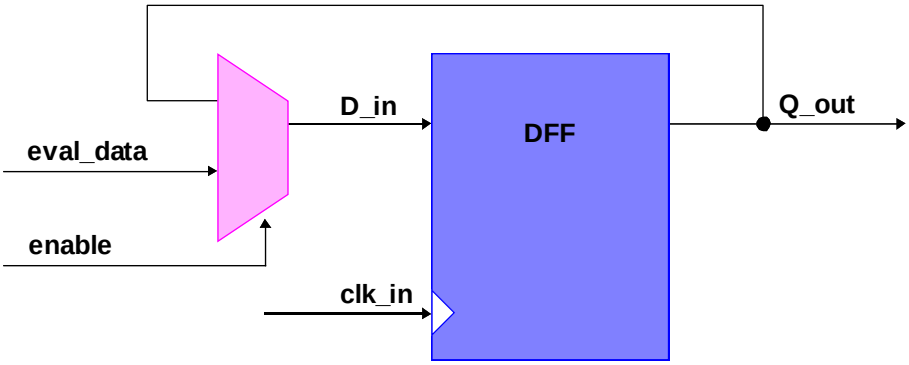
© Cadence Design Systems, Inc. Do not distribute.

Lab 7-1 Running Low-Power Synthesis

Objective: To insert clock gating for power optimization and to optimize the leakage and dynamic power.

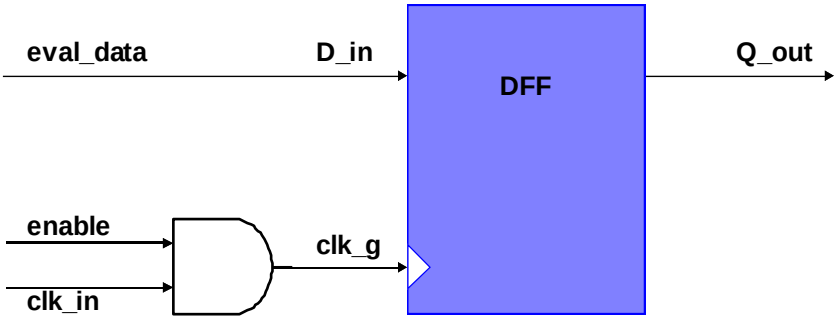
Using Clock Gating

The clocking strategy (following circuit) ensures that it updates the flip-flop every clock cycle even though the *eval_data* signal is valid when the enable line goes high.



Gating Circuit Using MUX for Data

You can optimize the circuit above for power consumption, ensuring that the flip-flop is clocked, only when the eval_data line is validated with the enable signal if the clock is gated like the Clock-Gating Circuit in the following figure.



Clock-Gating Circuit

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

The variable *lp_insert_clock_gating*, if set to one, enables clock gating during elaboration. The tool automatically chooses the library cell for the clock gating if no clock-gating cell is specified.

Using the clock-gating feature requires a different approach at the elaboration stage. Before the additional clock-gating code is brought into the design, the Cadence® Genus™ Synthesis Solution environment has to be modified to enable clock gating.

Setting Up the Environment

In this section, you set the environment of the synthesis session, load the RTL files, and elaborate the top-level design.

1. Change to the ***work*** directory:

   ```
   cd genus_labs/work
   ```

2. Start the software by entering:

   ```
   genus –legacy_ui -legacy_gui -f ../tcl/template.lp.tcl -log lps.log
   ```

Enabling Clock Gating

In this section, you set the variables to enable low-power synthesis and tell the Genus the style of clock gating to use before elaboration.

1. Enable clock-gating features by entering:

   ```
   set_attribute lp_insert_clock_gating true /
   ```

   It is important to enable clock gating before elaborating your design.

Loading Designs

1. Read the design and elaborate by entering:

   ```
   read_hdl $RTL_LIST
   elaborate $DESIGN
   ```

2. Read the constraint file.

   ```
   read_sdc ../constraints/dtmf_recvr_core.sdc
   ```

   Make sure the constraints file has no errors. The log file reports the errors and the reasons.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Setting Power Optimization Attributes

Low-power synthesis is included as part of the synthesis process.

1. Set the leakage power optimization effort.

   ```
   set_attribute leakage_power_effort medium /
   ```

2. Set the maximum dynamic power attribute.

   ```
   set_attribute max_dynamic_power 100 $DESIGN
   ```

3. Specify the optimization weight for dynamic power versus leakage power.

   ```
   set_attribute lp_power_optimization_weight 0.5 $DESIGN
   ```

   You see a warning about skewed total power calculation. At this time, it is best to decide what your final number might look like and calculate the weight based on that. From experience from the previous labs for this design, the leakage power is less than 1% for this design for this technology. So, let's set it to the following:

   ```
   set_attribute lp_power_optimization_weight 0.99 $DESIGN
   ```

   You should not see any warnings now.

4. Set the clock-gating style by entering:

   ```
   set_attribute lp_clock_gating_style latch $DESIGN
   ```

   The *lp_clock_gating_style* variable tells the software to use a clock-gating cell with a *latch*.

Annotating the RTL-Switching Activity

At this stage, you run the simulation of the design and generate the toggle count format (TCF) file using a simulation tool. The *../sim/sim.opt* file and the *../DESIGN/rtl/dtmf_recvr_core_test.v* file are provided as inputs to the Verilog-XL simulator. However, in this case, the TCF file is generated for you. The TCF file determines the exact power consumption of the design.

1. View the *../sim/top.hier.rtl.tcf* file.

   The generated TCF contains two sets of numbers. One represents the probability of the signal, and the other represents the number of times the signal has toggled. The net and instance probabilities are given separately.

2. Read the RTL TCF file into the Genus.

   ```
   read_tcf ../sim/top.hier.rtl.tcf
   ```

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

3. From the log file, gather the following information:

   What is the total number of nets in the design?

   Answer: _____

   What is the coverage of this TCF?

   Answer: _____

Running Generic Synthesis

1. Synthesize the design to generic logic

```
set_attribute syn_generic_effort medium
syn_generic
```

2. Report the power consumption of the design by entering:

```
report_power -depth 0
```

   What is the estimated total power of the design?

   Answer: _____

Running and Reporting Low-Power Synthesis

1. Synthesize the design:

```
set_attribute syn_map_effort medium
syn_map
set_attribute syn_opt_effort medium
syn_opt
```

   Multi-VT optimization and leakage power optimization are run at this stage.

   This step takes about five minutes in Linux. To get the CPU run time, use the *cpu_runtime* root attribute.

2. Report the clock gating.

```
report_clock_gating
```

   How many clock-gating instances are inserted?

   Answer: _____

3. Report the power consumption of the gates by entering:

```
report_gates -power
```

   How many instances are from High-Vt library?

   Answer: _____

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

How many instances are from the normal library?

Answer: _____

4. Report the power consumption of the design by entering:

```
report_power -depth 1
```

What is the total leakage power consumption of the design?

Answer: _____

What is the total dynamic power consumption of the design?

Answer: _____

What is the total power consumption of the design?

Answer: _____

You can change the unit of power by entering:

```
set_attribute lp_power_unit mW /
```

The power annotation during mapping is not affected by the *lp_power_unit* attribute.

5. Report the data path information by entering:

```
report_dp
```

Are there any data path modules? If so, what type?

Answer: _____

6. Report the area and the timing of the design.

What is the area of the design? What is the worst timing slack of the design?

Answer: _____, _____

Annotating Gate-Switching Activity (Optional)

At this stage, you run the gate-level simulation of the design and generate the gate-level TCF file using a simulation tool. The *../sim/sim_gate.opt* and the *../rtl/dtmf_recvr_core_test.v* files are provided as inputs to the Verilog-XL simulator. However, in this case, the TCF file is generated for you. The TCF file determines the exact power consumption of the design.

1. View the *../sim/top.hier.gates.tcf* file.

2. Read the gate-level TCF file into the Genus.

What is the coverage of this TCF?

Answer: _____

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

3. Report the power.

```
report_power -depth 0
```

This is a more accurate estimation of your power considering the gate-level TCF is accurate to your design.

What is the total power consumption of the design after reading the gate-level TCF?

Answer: _____

4. Write out the mapped netlist.

```
write_hdl > dtmf_chip.lps.v
```

5. Close the software.

```
exit
```

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Module 8: Test Synthesis

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Lab 8-1 Running Scan Synthesis

Objective: To insert scan chains and to improve the testability of a design.

Setting Up the Environment

In this section, you set the environment of the synthesis session, load the RTL files, and elaborate the top-level design.

1. Change to the *work* directory:

```
cd genus_labs/work
```

2. Start the software and set the environment by entering:

```
genus –legacy_ui -legacy_gui -f ../tcl/setup_dft.tcl -log dft.log
```

This loads the Verilog files for the design, elaborates the top-level module and reads constraints.

Setting Up for Scan Synthesis

At this stage, you define the DFT constraints, incrementally synthesize the design, and connect the scan chains. Notice in the script that some of attributes have been set for you.

For example, the Tcl script sets the DFT controllability for the PLL for the design.

```
set_attribute dft_controllable "PLLCLK_INST/refclk non_inverting"
   PLLCLK_INST/clk1x
```

1. Set the scan style to MUX scan.

```
set_attribute dft_scan_style muxed_scan /
```

2. Set the DFT signals for the design.

```
define_shift_enable -active high scan_en
define_test_mode -active high test_mode
define_test_clock scan_clk
```

The *shift_enable* and the *test_mode* options are necessary when you define the test constraints.

3. Report the DFT setup by entering:

```
report_scan_setup
```

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Checking DFT Violations

1. Check the DFT rules by entering:

   ```
   check_dft_rules
   ```

   Verify that all the registers have passed the DFT rules.

   Are there any violations? If so, what are the types of the violations?

   Answer: _____

   How many registers pass the DFT rule checks?

   Answer: _____

2. Fix any DFT violations in your design.

   Use the *fix_dft_violations* command.

Inserting Shadow DFT Logic

The RAM and ROM blocks used in this design are untestable logic. You add shadow logic around the untestable modules to complete the scan chain.

1. Synthesize to generic logic.

   ```
   set_attribute syn_generic_effort medium
   syn_generic
   ```

2. Use the shadow DFT logic to build testability around the logic connected to the RAM by entering:

   ```
   add_shadow_logic -auto -test_control test_mode
   ```

   This inserts the shadow logic for the RAM* and the ROM* instances.

3. Rerun the DFT rule checker.

   How many registers pass the DFT rule checks?

   Answer: _____

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Running Scan Synthesis

Mapping to scan is included as part of the synthesis process.

You can set the scan map mode to specify what types of flops to convert to scan flops. When you set the scan map mode to *tdrc_pass*, you convert the flip-flops (DFF*) that pass the DFT rule checker to scan flip-flops (SDFF*).

```
set_attribute dft_scan_map_mode tdrc_pass /des*/dtmf_recvr_core
```

Because this is the default setting, you can skip it for this session.

1. Synthesize the design:

   ```
   set_attribute syn_map_effort medium
   syn_map
   set_attribute syn_opt_effort medium
   syn_opt
   ```

   Mapping to scan is automatically run at this stage.

   This step takes about five minutes in Linux and 15 minutes in Solaris. To get the CPU run time use the *cpu_runtime* root attribute, or use the *timestat* command.

2. Report the status of the DFT registers.

   ```
   report_scan_registers
   ```

   How many flops are scan flops?

   Answer: _____

Configuring the Scan Chains

1. Set the scan input and output ports by entering:

   ```
   define_scan_chain -name chain1 -create_ports -sdi tdi -sdo tdo
   ```

   The *-create_ports* option creates new ports for the scan chains.

2. Set the minimum number of scan chains required by entering:

   ```
   set_attr dft_min_number_of_scan_chains 2 /designs/$DESIGN
   ```

3. Preview your scan chains to check your configuration and make sure they are correct.

   ```
   connect_scan_chains -auto_create_chains -preview
   ```

   How many scan chains are created?

   Answer: _____

   How many flops are in each scan chain?

   Answer: _____

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Are all the scan inputs and outputs defined? What are they?

Answer: _____

Note: The DFT prefix in the scan chain names can be changed by the *dft_prefix* root attribute.

Connecting Scan Chains

After you are satisfied with your scan configuration output, you can connect the scan chains.

1. Connect the scan chains with this command:

   ```
   connect_scan_chains -auto_create_chains
   ```

2. Report the scan chains, the scan registers and check the scan connections.

   ```
   report_scan_chains
   ```

   Are there any lockup latches in the scan chains? Why or why not?

   Answer: _____, _____

3. Run incremental synthesis by entering:

   ```
   syn_opt -incremental
   ```

   Incremental optimization fixes any timing violations caused by the scan insertion and fixes any DRC violations.

4. Report the quality of results.

   ```
   report_qor
   ```

Writing Design DFT Outputs

1. Save the design and constraint files to Innovus™.

   ```
   write_design -innovus -base_name FINAL/$DESIGN
   ```

2. Create the ATPG files for your design.

   ```
   write_dft_atpg_other_vendor_files -stil > $DESIGN.stil.atpg
   ```

3. Save the scan DEF file.

   ```
   write_scandef > $DESIGN.scanDEF
   ```


© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

4. Write the scan abstraction mode.

```
write_dft_abstract_model > $DESIGN.scanAbstract
```

Important: Do not close the software. You continue this session in the next lab.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Module 9: Logic Equivalence Checking

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

There are no labs for this module

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Module 10: Interfacing with Other Tools

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Lab 10-1 Interfacing with Other Tools

Objective: To write a netlist that interfaces with place-and-route tools by changing the names of design objects, eliminating assign statements, and blasting the bits of bus ports.

Continue from the previous lab session and complete the post synthesis processing of the netlist.

Changing the Names of Design Objects

In this section, you change names of all subdesigns using *LAB_* as a prefix to match with the names of the place-and-route tools.

1. Change the names of all the subdesigns to add the *LAB_* prefix:

   ```
   update_names -subdesign -prefix LAB_
   ```

2. Save the mapped netlist with the changed names:

   ```
   write_hdl > LAB_names.v
   ```

3. Open the *LAB_names.v* and confirm that the subdesign names are changed.

Removing Assign Statements from the Netlist

In this section, you remove assign statements that are not recognized by the place-and-route tools.

1. Open *LAB_names.v* and confirm that the assign statements are present.

2. Remove the assign statements and constants in your design by entering:

   ```
   set_attribute remove_assigns true /
   ```

   Ideally, the *remove_assigns* attribute must be specified before optimization.

3. This command allows each constant assignment to be replaced with a tie cell.

   ```
   set_attribute use_tiehilo_for_const duplicate /
   ```

4. This command allows TIEHI and TIELO cells to be used to tie the change constant assignments.

   ```
   set_attribute avoid false [find / -libcell TIE*]
   ```

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

5. This command allows BUFX8MTH to be used for wire assignments.

```
add_assign_buffer_options -buffer_or_inverter \
   [find / -libcell BUFX8MTH]
```

6. Synthesize design incrementally:

```
syn_opt -incremental
```

Note: Normally, you set the *remove_assigns* attribute and corresponding settings before running optimization so that you do not have to run incremental optimization again.

What is the timing slack and the area of the design?

Answer: _____ _____

7. Save the netlist by entering:

```
write_hdl > LAB_noas.v
```

8. Open *LAB_noas.v* and confirm that the assign statements have been removed and replaced with a buffer or a tie cell.

Controlling the Bit-Blasting of Bus Ports

A review of the netlist shows that multibit signals are shown only as vectors in the module port definitions.

1. Open the bit map of the port and show each bit individually rather than as vectors:

```
set_attribute bit_blasted_port_style %s_\{%d\} /
set_attribute write_vlog_bit_blast_constants true /
set_attribute write_vlog_bit_blast_mapped_ports true /
```

Some place-and-route tools require the ports to be bit-blasted, and these are automatically generated through the *write_hdl* command.

2. Save the netlist by entering:

```
write_hdl > LAB_bb.v
```

3. Open *LAB_bb.v* and confirm that the ports in the module definitions have been broken into bits.

 © 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Ungrouping the Hierarchy

1. Create a report summary to view the hierarchy.

```
report_hierarchy
```

2. Prevent the *tdsp_core* and *results_conv* modules from being flattened by entering:

```
set_attribute preserve 1 [find / -subdesign *results_conv]
set_attribute preserve 1 [find / -subdesign *tdsp_core]
```

3. Ungroup the design and report the hierarchy to confirm the ungrouping results by entering:

```
ungroup -all -flatten
report_hierarchy
```

How many top-level hierarchical instances now exist in the design?

Answer: _____

Generating Interface Files for Other Tools

1. Save the design files by entering:

```
write_hdl > $DESIGN.final.v
write_sdc > $DESIGN.final.sdc
```

2. Write the Conformal® LEC software data files by entering:

```
write_lec_script -revised $DESIGN.final.v -log rtl2final.lec.log >
   rtl2final.lec.do
```

3. Write the Innovus™ Implementation System data files.

```
write_design -innovus –base_name final/$DESIGN
```

4. Close the software:

```
exit
```

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Appendix A: Advanced Synthesis Features

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

There are no labs for this appendix

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Appendix B: Genus Synthesis Solution Constraints (Optional)

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Lab B-1 Applying Genus Synthesis Solution Constraints (Optional)

Objective: To set the Cadence® Genus™ Synthesis Solution constraints, such as clocks, external delays, false paths, and multicycle paths, after elaboration and before synthesis.

This lab provides some practice on native Genus Synthesis Solution constraints.

Setting Up the Environment

In this section, you set the environment of the synthesis session, load the RTL files, and elaborate the top level.

1. Change to the ***work*** directory by entering:

   ```
   cd genus_labs/work
   ```

2. Start the software by entering this command:

   ```
   genus –legacy_ui -legacy_gui -f ../tcl/template.opto.tcl -log con.log
   ```

   This command loads the libraries, reads the design, and elaborates it.

3. Run is suspended after elaborating the design. Stop at this point and follow the instruction in the next section for constraints. (If you want to move ahead, you can resume the run and read out sdc, synthesize the design, and write out the output files.)

   Note: The *tcl/constraints.tcl* file contains all the constraints necessary for the labs. You can copy and paste the constraint commands from the *constraints.tcl* file to avoid typing. You must delete the failed constraints using the *rm* command and re-enter all the corresponding failed commands manually.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Setting Clocks

The system clock *refclk* goes to a PLL that produces the generated clocks *clk1x* at the same frequency and *clk2x* at half the frequency of the port clock *refclk*.

These clocks feed the *TEST_CONTROL_INST*, which selects between the scan and regular clock for each of the modules in the design.

1. Set the top-level clock by entering:

   ```
   define_clock -p 6000 -name refclk [find /des* -port refclk]
   ```

   Check the attributes of the clock constraint under */des*/dtmf_recvr_core/timing*.

   Is it set as a clock? Is the constraint set on the right port?

   Answer: _____

   Is the period set correctly? What is the waveform?

   Answer: _____

   What other attributes of the clock can be set here?

   Answer: _____

2. Model a fall transition and rise transition on the *refclk* clock of **20ps** on the rise and fall edges.

3. Define the internal clocks of the design.

   a. Define a clock with a period of **6000ps** on the *m_clk* pin of the *TEST_CONTROL_INST* instance.

   b. Define clocks with a period of **12000ps** on the following pins:

      m_rcc_clk, *m_spi_clk*, *m_dsram_clk*, *m_ram_clk*, and *m_digit_clk* under the *TEST_CONTROL_INST* instance.

   To remove a constraint, use the rm command on the exact constraint.

4. Set the source and network latency of **2000ps** on all the internal clocks by entering:

   ```
   set_attribute clock_source_late_latency 2000 [find / -clock m*clk]
   set_attribute clock_network_late_latency 2000 [find / -clock m*clk]
   ```

5. Model an uncertainty of **250ps** on all clocks.

   ```
   set_attribute clock_setup_uncertainty 250 [find / -clock *]
   ```


© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Applying External Delays

1. Set the input delay of **500ps** on all input ports and an output delay of **500ps** on all output ports with respect to the *refclk* clock.

```
external_delay -input 500 -clock refclk [all_inputs]
external_delay -output 500 -clock refclk [all_outputs]
```

2. Find the external output delay on the *tdigit[0]* port from a suitable report of timing.

   Answer: _____

   Use *report_timing -to [find / -port port_name]*.

Setting Ideal Drivers

1. Set the *refclk* to ideal driver to avoid applying design rule constraints to it.

```
set_attribute ideal_driver true [find /des* -port refclk]
```

   In the Genus Synthesis Solution, the clock ports do not accept any delay setting. Therefore, the delay setting does not need to be specifically deleted.

2. Set the *reset* to ideal driver to avoid applying design rule constraints to it:

```
external_delay -input 0 -clock refclk [find /des* -port reset]
set_attribute ideal_driver true [find /des* -port reset]
```

Applying Path Exceptions

In this section, you set path exceptions, such as false and multicycle paths.

1. Set the false paths in the design so that the software does not waste any optimization cycles on these paths by entering:

```
path_disable -from [find / -port reset]
path_disable -from [find / -port test_mode]
path_disable -from [find / -port scan_en]
path_disable -from [find / -port spi_data]
path_disable -from [find / -port spi_fs]
```

2. Set up the multicycle paths in the design to the corresponding cycles of the data.

```
multi_cycle -to [find [find / -inst EXECUTE_INST] -inst acc_reg* ] \
   -launch_shift 0 -capture_shift 2 -name ACC_REG_SLOW
multi_cycle -to [find [find / -inst EXECUTE_INST] -inst p_reg* ] \
   -launch_shift 0 -capture_shift 2 -name P_REG_SLOW
```

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.


```
multi_cycle -to [find [find / -inst EXECUTE_INST] -inst ov_flag_reg*] \
   -launch_shift 0 -capture_shift 2 -n OVFLAG_REG_SLOW
```

The launch shift and the capture shift define the amount of cycles for the slow logic path. A launch shift of zero and a capture shift of two add one additional cycle.

3. Verify if the exception to the instance *ov_flag_reg** is applied and name the clock that drives the flip-flop from a suitable report of timing.

   Answer: _____

Setting Design Rule Checks

1. Set the capacitance loading on all output ports at the top level to ***two times*** the load capacitance on the PAD pin of the *PDIDGZ* library cell.

```
set cap [get_attr load [find [find /lib* -libcell PDIDGZ] -libpin PAD]]
set_attribute external_pin_cap [expr 2*$cap] [all_outputs]
```

   A nested set of find commands locates the *load* attribute of the pin and assigns it to a variable called *cap*.

   The Tcl expression *[expr x * y]* specifies the capacitance load on the *external_pin_cap* attribute of the output ports.

2. Use *ls -l -a* or *get_attribute* on the *tdigit_flag* output port and get the value of the external pin capacitance.

   Answer: _____

   What is the unit for capacitance?

   Answer: _____

3. The external driver attribute on all input ports point to the *PDO04CDG* library cell.

```
set_attribute external_driver [find [find /lib* -libcell PDO04CDG] \
   -libpin PAD] [all_inputs]
```

4. Remove the driver settings from the *reset* and the *refclk* pins.

```
set_attribute external_driver "" [find /des* -port reset]
set_attribute external_driver "" [find /des* -port refclk]
```

   This is usually done to prevent the buffering of the large nets during design rule fixing. You might want to add clock trees for these nets later in the design cycle.

5. Set ***max_fanout*** to **15** on all input pins by entering:

```
set_attribute max_fanout 15 [all_inputs]
```


© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

6. Remove the ***max_fanout*** setting from *refclk* and *reset* pins by entering:

```
set_attribute max_fanout "" [find /des* -port reset]
set_attribute max_fanout "" [find /des* -port refclk]
```

   The *max_fanout* setting is a design rule check that creates a violation for nets that exceed a specified fanout limit.

7. Check for any missing constraints by entering:

```
check_timing_intent
```

   Identify the missing constraints. Does this problem need fixing? If so, how would you fix the problem?

   Answer: _____

   The syntax errors are not reported by this command. The log file reports the errors and the reasons.

8. Synthesize the design and write out output files.

9. Close the software.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Appendix C: Genus Common UI

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

There are no labs for this appendix

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Appendix D: Solutions to Labs

© Cadence Design Systems, Inc. Do not distribute.

© Cadence Design Systems, Inc. Do not distribute.

Lab Solutions

The solutions for this lab are provided using the 16.2 base version, GENUS Version: 16.20-p004_1 build. The solutions might differ slightly with other versions or builds. Therefore, the answers are listed as approximate (~) values.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 3-1: Running the Basic Synthesis Flow

In the *Loading Libraries and Designs* section:

List the libraries that are loaded into Genus.

Answer: pllclk, rom512x16A_slow, ss_hvt_1v08_125c, ram_256x16_slow, and ss_g_1v08_125c

Are all the libraries specified by the $LIB_LIST from setup.tcl read in correctly?

Answer: Yes. All the libraries specified by the $LIB_LIST variable in the **setup.tcl** file should be loaded correctly.

In the *Elaborating the Design* section:

List the designs that are loaded into Genus.

Answer: dtmf_recvr_core

Is there only one top-level design?

Answer: Yes

Are there any unresolved references or blackboxes in the design?

Answer: Yes, ACCUM_STAT_INST

Are there any unresolved instances?

Answer: Yes

What is the name of the module that is missing?

Answer: accum_stat

Which UNIX directory are the RTL files located in?

Answer: As defined by the *hdl_search_path* they should be in genus_*labs*/rtl.

Which file contains the module definition for the missing module?

Answer: *accum_stat.v*. The *accum_stat*.v file is missing. Add it to the RTL_LIST variable in the setup.tcl file and rerun elaboration.

Is the elaboration done and complete?

Answer: If the steps are followed correctly, the elaboration step should complete by saying, “Done elaborating…” and there should be no missing modules.

Do you have one top-level design?

Answer: If the steps are followed correctly, the elaboration step should result in only one design. You can check this using *ls /designs/*.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 3-2: Navigating the Design Hierarchy

In the *Filtering Objects by Type and Name* section:

As in the previous step, find the attributes of the following design objects:

Answers

- **subdesign tdsp_core**

  ```
  ls -la [find / -subdesign tdsp_core]
  ```

- **instance EXECUTE_INST**

  ```
  ls -la [find / -instance EXECUTE_INST]
  ```

- **port refclk**

  ```
  ls -la [find / -port refclk]
  ```

- **port reset** (top-level port)

  ```
  ls -la [find / -port reset]
  ```

- **pin reset** in the **EXECUTE_INST instance**

  ```
  ls -la [find [find / -instance EXECUTE_INST] -pin reset]
  ```

In the *Using Procedures* section:

What is the difference between using this procedure and a simple find command that lists all the subdesigns in the design?

Answer: The *dirnames* are not displayed. You see a list of subdesign base names and they are shown one on each line.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 3-3: Reading SDC Constraints

Are there any syntax errors? If so, what are they?

Answer: There are 2 syntax errors.

Command failed at line 420: ***set_mx_transition*** must be set_max_transition.

Command failed at line 426: The option ***-lbirary*** should be *-library* for the *set_wire_load_model* command.

In Line 425 of constraint.sdc

The interconnect_mode attribute is automatically set to `ple` when physical information is read, and attribute 'wireload_mode' here is set to 'top' hence Error.

In Line 426 of constraint.sdc

```
set_wire_load_model "TSMC_13k_Conservative" -library "ss_1v08_125c.lib"
```

Could not find any such library with name "ss_1v08_125c.lib" in the present directory structure.

Comment these lines (425 and 426) as interconnect_mode attribute is set to ple.

Are there any other failed commands? If so, what are they?

Answer: The *set_input_delay* and the *set_output_delay* commands should fail because of the missing clock definition. You can fix them using the following command:

```
create_clock -name "m_digit_clk" -add -period 16.0 -waveform {0.0 8.0}
   [get_pins TEST_CONTROL_INST/m_digit_clk]
```

In the *Debugging Failed SDC Constraints* section:

Are there any errors?

Answer: If you read the *dtmf_recvr_core.sdc*, you should have no failed commands or syntax errors. If you are reading your own *constraints.sdc*, make sure there are no further errors before you proceed.

In the *Analyzing Missing SDC Constraints* section:

What types of messages are reported?

Answer: Missing clocks, missing input and output delay definitions (a.k.a. external delays), inputs without external driver/transition and outputs without external load.

Are there any real missing constraints?

Answer: Yes, in a real design scenario these missing constraints could actually be a problem. Clock definition is missing for the *p_clk* port of the PM_INST. This can be ignored for now, because this is a power module which has not been completely designed.

 © 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

TDSP_DS_CS_INST gets a defined clock m_clk, but it is gated with a data signal. You can define these gated clocks in your constraints.

Missing external delays. Some of these ports have no real connections in the RTL, so they can be ignored until placement or routing.

If any, how would you fix the missing constraints?

Answer: You can define a new clock constraint for each of these missing clocks. You can also define the external delays for each port.

But, you do not fix the rest of the constraints in this lab. Let's assume that you delegated this issue to logic design/verification group.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 3-4: Synthesizing the Design and Using the Graphical Interface

In the *Synthesizing the Design* section:

Is the design meeting timing?

Answer: Yes

What is the total area and power of the design?

Answer: ~0.246970 mm^2, ~ 15.888 mW

What is the total run time and memory usage to this point?

Answer: ~198 secs, ~701 MB

In the *Viewing Logical Hierarchy, HDL and Schematic* section:

Is this instance a blackbox or a timing model?

Answer: The PLL_INST instance is a timing model.

In the *Generating Reports* section:

What type of library cell is causing the worst slack in the worst path?

Answer: The library cell is listed by default in the timing report along with the load, delay and fanout of the cell. For this design, it is the TDSP_CORE_INST/EXECUTE_INST/sel_op_a_reg[1]/Q (Library cell listed is DFFQX4M).

What types of issues are being reported?

Answer: Missing external delays, missing clocks to a couple of flops, sequential data pins driven by clock signals. All these issues can be ignored for the purposes of this lab.

Which is worse for this design: Leakage Power or Switching (Dynamic Power)?

Answer: Dynamic

What is the block that dissipates most power?

Answer: TDSP_CORE_INST. You can also use *report_power –depth 1* to determine this.

What is/are the worst instance(s) for this design?

Answer: TDSP_CORE_INST

What is the percentage power that the block(s) dissipate(s)?

Answer: ~17.77%

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 4-1: Running Datapath Synthesis

In the *Datapath Synthesis* section:

What data path components can you see?

Answer: Adders and Multipliers

Are there any external MUXes present?

Answer: Yes

What is the percentage of data path modules before synthesis?

Answer: ~ 92.61%

What is the cell area for data path modules at this stage?

Answer: ~ 0.019208 mm^2

Are there any external MUXes present now?

Answer: No

Is there any change in percentage of data path modules compared to the previous report?

Answer: Yes, but very little

What is the cell area for data path modules now?

Answer: ~ 0.018856 mm^2

Has the tool done any carrysave optimization?

Answer: Yes

How many CSA groups are created during carrysave optimization?

Answer: 3

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 6-1: Exploring Optimization Strategies Using PLE

In the *Running Generic Synthesis* section:

List the cost groups in /designs/dtmf_recvr_core/timing/cost_groups.

Answer: m_clk m_digit_clk m_dsram_clk m_ram_clk
m_rcc_clk m_spi_clk refclk

List a few types of data path components from the report.
Are there any merged components?

Answer: Adders, multipliers, subtractor**,** and so on. No merged components.

List a couple of the optimizations reported by the tool.

Answer: MUX optimization, datapath optimizations such as carrysave, optimizations for constant propagation and redundancy removal, merging of sequential instances.

List a few types of datapath components from the report.
Are there any merged components?

Answer: Adders, multipliers, subtractor, left shifter and so on. No merged components.

In the *Mapping to a Target Library* section:

What is the target slack for the m_clk cost group?

Answer: ~ 224ps

What is the Global Incremental target slack for m_clk cost group? How does it compare to the previous target slack at Global Mapping Target Info (% difference in terms of clock period)?

Answer: ~149ps, ~ 0.9375% (((224-149)/8000)*100)

What is the total negative slack (Group Total Worst Slacks) after the optimization step?

Answer: 0ps

In the *Optimizing the Design* section:

What is the worst timing slack reported for the m_clk cost group?

Answer: ~ 0ps

What is the end point reported for the worst slack?

Answer: End-point: TDSP_CORE_INST/EXECUTE_INST/p_reg[31]/D

What is the total area of the design after optimization?

Answer: ~ 0.2465967 mm^2

What is the total power of the design after optimization?

Answer: ~ 15.809 mW

 © 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

In the *Generating and Analyzing Reports* section:

What is the worst delay listed on the report with a fanout listed? What is the name of the instance with this delay? What library cell of this instance?

Answer: ~ 569 ps, TDSP_CORE_INST/DATA_BUS_MACH_INST/ data_out_reg[13]/Q, SDFFRHQX8MTH

How does the delay of this instance compare to instances with similar library cells that have similar or lesser fanout?

Answer: Larger delay for the cell with larger fanout, load, and slew combination.

```
legacy_genus:/> find -libcell SDFFRHQX8MTH
```

Result is:

```
/libraries/ss_hvt_1v08_125c/libcells/SDFFRHQX8MTH
```

```
legacy_genus:/> filter libcell [find /lib* -libcell
   SDFFRHQX8MTH] [find /des* -instance *]
```

Results in single cell:

```
{/designs/dtmf_recvr_core/instances_hier/TDSP_CORE_INST/instances
   _hier/DATA_BUS_MACH_INST/instances_seq/data_out_reg[13]}


Use: report timing -through [find / -inst]


legacy_genus:/> report_timing -through [find / -instance
   data_out_reg[13]] -summary
Timing slack :      0 ps
```

What is the timing slack on this report?

```
report timing -to [find [find / -inst EXECUTE_INST/acc_reg[6]] -pin D]
```

Answer: ~ 4 ps

Does this path have any timing exceptions?

Answer: No. Timing exceptions can be anything from false paths, disabled timing arcs, and so on. They should be listed in the slack report as timing exceptions.

Are there any design rule violations?

Answer: No.

What is the total instance count?

Answer: ~5051

What is the total leakage power consumption of the design?

Answer: ~ 0.042 mW

What is the total dynamic power consumption of the design?

Answer: ~15.767 mW

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

What is the block that consumes the most power?

Answer: TDSP_CORE_INST, ~5.022 mW

Cell (leaf instance) count in the design:

Answer: ~ 5051

Cell area:

Answer: ~0.218476 mm^2

Number of sequential elements:

Answer: ~ 514

Is there a timing violation?

Answer: No

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 7-1: Running Low-Power Synthesis

In the *Annotating the RTL-Switching Activity* section:

What is the total number of nets in the design?

Answer: ~ 26024

What is the coverage of this TCF?

Answer: ~ 27.77% (Percentage of Nets asserted)

In the *Running Generic Synthesis* section:

What is the estimated total power of the design?

Answer: ~ 17.141 mW

In the *Running and Reporting Low-Power Synthesis* section:

How many clock-gating instances are inserted?

Answer: ~41

How many instances are from the High-Vt library?

Answer: ~ 4088

How many instances are from the normal library?

Answer: ~ 881

What is the total leakage power consumption of the design?

Answer: ~ 0.039 mW

What is the total dynamic power consumption of the design?

Answer: ~ 10.445 mW

What is the total power consumption of the design?

Answer: ~10.483 mW

Are there any data path modules? If so, what type?

Answer: Yes. Multipliers, adders, left-shift, subtractor, and so on.

What is the area of the design? What is the worst timing slack of the design?

Answer: ~0.239041 mm^2, ~0ps

In the *Annotating Gate-Switching Activity (Optional)* section:

What is the coverage of this TCF?

Answer: ~ 17.83%

What is the total power consumption of the design after reading the gate-level TCF?

Answer: ~ 10.486 mW

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 8-1: Running Scan Synthesis

In the *Checking DFT Violations* section:

Are there any violations? If so, what are the types of the violations?

Answer: No. No violations.

How many registers pass the DFT rule checks?

Answer: 565

In the *Inserting Shadow DFT Logic* section:

How many registers pass the DFT rule checks?

Answer: 628

In the *Running Scan Synthesis* section:

How many flops are scan flops?

Answer: 628

In the *Configuring the Scan Chains* section:

How many scan chains are created?

Answer: 2 (chain1 and AutoChain_1)

How many flops are in each scan chain?

Answer: Chain1 (tdi -> tdo) has 129 registers

AutoChain_1 (DFT_sdi_1 -> DFT_sdo_1) has 499 registers

Are all the scan inputs and outputs defined? What are they?

Answer: Yes. Defined for one scan chain as tdi and tdo. The others were automatically created with the DFT_ prefix as DFT_sdi1 and DFT_sdo1.

In the *Connecting Scan Chains* section:

Are there any lockup latches in the scan chains? Why or why not?

Answer: No, merging of clock edges is off. If you specify attribute *dft_mix_clock_edges_in_scan_chains* as true, then scan flip-flops triggered by different active edges of the same test clock can be mixed along the same scan chain.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab 10-1: Interfacing with Other Tools

In the *Removing Assign Statements from the Netlist* section:

What is the timing slack and the area of the design?

Answer: ~0ps and ~0.250248 mm^2

In the *Ungrouping the Hierarchy* section:

How many top-level hierarchical instances now exist in the design?

Answer: Count the numbers of level 1. There should only be two top-level hierarchical instances.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Lab B-1: Applying Genus Synthesis Solution Constraints (Optional)

In the *Setting Clocks* section:

Is it set as a clock? Is the constraint set on the right port?

Answer: Yes. Yes, it is set on refclk.

Is the period set correctly? What is the waveform?

Answer: Yes. The waveform is [0 3000].

What other attributes of the clock can be set here?

Answer: Latency, uncertainty, and so on.

In the *Applying External Delays* section:

Find the external output delay on the tdigit[0] port from a suitable report of timing.

Answer: report_timing -to [find / -port tdigit[0]] slack: -3851ps

In the *Applying Path Exceptions* section:

Verify if the exception to the instance ov_flag_reg is applied and name the clock that drives the flip-flop from a suitable report of timing.*

Answer: report_timing -to [find [find / -inst EXECUTE_INST] -inst ov_flag_reg*]

Exception : 'multi_cycles/OVFLAG_REG_SLOW' launch shift 0, capture shift 2

Timing slack : 6632ps

clock is m_clk

In the *Setting Design Rule Checks* section:

Use ls -l -a or get_attribute on the tdigit_flag output port and get the value of the external pin capacitance.

Answer: *external_pin_cap = 0.0 femtofarads*

What is the unit for *capacitance?*

Answer: fF

Identify the missing constraints. Does this problem need fixing? If so, how would you fix the problem?

Answer: There are missing constraints related to sequential data pins driven by a clock signal, sequential clock pins without clock waveform, inputs without external drivers/transition, and so on.

© 2016 Cadence Design Systems, Inc. All rights reserved.

© Cadence Design Systems, Inc. Do not distribute.

Yes, you need to check and fix the required missing constraints before moving ahead in the design.

All external delays and clocks must be defined. Any overlapping constraints must be checked and removed appropriately. If this is done, then most of your constraint problems are eliminated.

End of Lab

© 2016 Cadence Design Systems, Inc. All rights reserved.