

Simply Better Results

Synplicity ASIC Synthesis

5.0 Tutorial

September 2005

Synplicity, Inc. 600 West California Avenue Sunnyvale, CA 94086, USA (U.S.) +1 408 215-6000 direct (U.S.) +1 408 990-0263 fax www.synplicity.com

Copyright and License Agreement

Disclaimer of Warranty

Synplicity, Inc. makes no representations or warranties, either expressed or implied, by or with respect to anything in this material, and shall not be liable for any implied warranties of merchantability or fitness for a particular purpose of for any indirect, special or consequential damages.

Copyright Notice

Copyright © 1994-2005 Synplicity, Inc. All Rights Reserved.

Synplicity software products contain certain confidential information of Synplicity, Inc. Use of this copyright notice is precautionary and does not imply publication or disclosure. No part of this material may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language in any form by any means without the prior written permission of Synplicity, Inc. While every precaution has been taken in the preparation of this material, Synplicity, Inc. assumes no responsibility for errors or omissions. This material and the features described herein are subject to change without notice.

Trademarks

Synplicity, the Synplicity "S" logo, Amplify, Certify, Synplify, Synplify Pro, Synplify ASIC, Amplify ASIC, RealPower, HDL Analyst, SCOPE, Behavior Extracting Synthesis Technology, Embedded Synthesis, Simply Better Synthesis, Simply Better Results, and Synthesis Constraint Optimization Environment are registered trademarks, and BEST, DST, Direct Synthesis Technology, MultiPoint, Partition-Driven Synthesis, PowerPlanner, Physical Synthesis, Synplify Lite, and Total Optimization Physical Synthesis are trademarks of Synplicity, Inc. Verilog is a registered trademark of Cadence Design Systems, Inc. IBM and PC are registered trademarks of International Business Machines Corporation. Microsoft is a registered trademark of Microsoft Corporation. Sun, SPARC, Solaris, and SunOS are trademarks of Sun Microsystems, Inc. UNIX is a registered trademark of X/Open Corporation. All other product names mentioned herein are the trademarks or registered trademarks of their respective owners.

Restricted Rights Legend

Government Users: Use, reproduction, release, modification, or disclosure of this commercial computer software, or of any related documentation of any kind, is restricted in accordance with FAR 12.212 and DFARS 227.7202, and further restricted by the Synplicity Software License Agreement. Synplicity, Inc., 600 West California Avenue, Sunnyvale, CA 94086, U. S. A.

September 2005

Synplicity Software License Agreement

Important! READ CAREFULLY BEFORE PROCEEDING

BY INDICATING YOUR ACCEPTANCE OF THE TERMS OF THIS AGREEMENT, YOU ("LICENSEE") ARE REPRESENTING THAT YOU HAVE THE RIGHT AND AUTHORITY TO LEGALLY BIND YOUR-SELF OR YOUR COMPANY, AS APPLICABLE, AND CONSENTING TO BE LEGALLY BOUND BY ALL OF THE TERMS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO ALL THESE TERMS DO NOT INSTALL OR USE THE SOFTWARE, AND RETURN THE SOFTWARE TO THE LOCATION OF PURCHASE FOR A REFUND. This is a legal agreement governing use of the software program provided by Synplicity, Inc. ("Synplicity") to you (the "SOFTWARE"). The term "SOFTWARE" also includes related documentation (whether in print or electronic form), any authorization keys, authorization codes, and license files, and any updates or upgrades of the SOFTWARE provided by Synplicity, but does <u>not</u> include certain "open source" software licensed by third party licensors and made available to you by Synplicity under the terms of such third party license is a participant in the University Program or has been granted an Evaluation License or Subscription License, then some of the following terms and conditions may not apply (refer to the sections entitled, respectively, **Evaluation License** and **Subscription License**, below).

License. Synplicity grants to Licensee a non-exclusive right to install the SOFTWARE and to use or authorize use of the SOFTWARE by up to the number of nodes for which Licensee has a license and for which Licensee has the security key(s) or authorization code(s) provided by Synplicity or its agents for the purpose of creating and modifying Designs (as defined below). If Licensee has obtained the SOFTWARE under a node-locked license, then a "node" refers to a specific machine, and the SOFTWARE may be installed only on the number of "nodes" or machines authorized, must be used only on the machine(s) on which it is installed, and may be accessed only by users who are physically present at that node or machine. A node-locked license may only be used by one user at a time running one instance of the software at a time. If Licensee has obtained the SOFT-WARE under a "floating" license, then a "node" refers to a concurrent user or session, and the SOFTWARE may be used concurrently by up to the number of users or sessions indicated. All SOFTWARE must be used within the country for which the systems were licensed and at Licensee's Site (contained within a one kilometer radius); however, if Licensee has a floating license then remote use is permitted by employees who work at the site but are temporarily telecommuting to that same site from less than 50 miles away (for example, an employee who works at a home office on occasion), but the maximum number of concurrent sessions or nodes still applies. In addition, Synplicity grants to Licensee a non-exclusive license to copy and distribute internally the documentation portion of the SOFTWARE in support of its license to use the program portion of the SOFTWARE. For purposes of this Agreement the "Licensee's Site" means the location of the server on which the SOFTWARE resides, or when a server is not required, the location of the client computer for which the license was issued.

Evaluation License. If Licensee has obtained the SOFTWARE pursuant to an evaluation license, then, in addition to all other terms and conditions herein, the following restrictions apply: (a) the license to the SOFT-WARE terminates after 20 days (unless otherwise agreed to in writing by Synplicity); and (b) Licensee may use the SOFTWARE only for the sole purpose of internal testing and evaluation to determine whether Licensee wishes to license the SOFTWARE on a commercial basis. Licensee shall not use the SOFTWARE to design any integrated circuits for production or pre-production purposes or any other commercial use including, but not limited to, for the benefit of Licensee's customers. If Licensee breaches any of the foregoing

restrictions, then Licensee shall pay to Synplicity a license fee equal to Synplicity's perpetual list price plus maintenance for the commercial version of the SOFTWARE.

Subscription (Time-Based) License. If Licensee has obtained a Subscription License to the SOFTWARE, the, in addition to all other terms and conditions herein, the following restrictions apply: (a) Licensee is authorized to use the SOFTWARE only for a limited time (which time is indicated on the quotation or in the purchase confirmation documents); (b) Licensee's right to use the SOFTWARE terminates on the date the subscription term expires as set forth in the quotation or the purchase confirmation documents, unless Licensee has renewed the license by paying the applicable fees.

Project Based License. If Licensee has obtained a Project-Based License to the SOFTWARE, in addition to all other terms and conditions herein, the terms of Exhibit A will apply.

Copy Restrictions. This SOFTWARE is protected by United States copyright laws and international treaty provisions and Licensee may copy the SOFTWARE only as follows: (i) to directly support authorized use under the license, and (ii) in order to make a copy of the SOFTWARE for backup purposes. Copies must include all copyright and trademark notices.

Use Restrictions. This SOFTWARE is licensed to Licensee for internal use only. Licensee shall not (and shall not allow any third party to): (i) decompile, disassemble, reverse engineer or attempt to reconstruct, identify or discover any source code, underlying ideas, underlying user interface techniques or algorithms of the SOFT-WARE by any means whatever, or disclose any of the foregoing; (ii) provide, lease, lend, or use the SOFT-WARE for timesharing or service bureau purposes, on an application service provider basis, or otherwise circumvent the internal use restrictions; (iii) modify, incorporate into or with other software, or create a derivative work of any part of the SOFTWARE; (iv) disclose the results of any benchmarking of the SOFTWARE, or use such results for its own competing software development activities, without the prior written permission of Synplicity; or (v) attempt to circumvent any user limits, maximum gate count limits or other license, timing or use restrictions that are built into the SOFTWARE.

Transfer Restrictions/No Assignment. The SOFTWARE may only be used under this license at the designated locations and designated equipment as set forth in the license grant above, and may not be moved to other locations or equipment or otherwise transferred without the prior written consent of Synplicity. Any permitted transfer of the SOFTWARE will require that Licensee executes a "Software Authorization Transfer Agreement" provided by Synplicity. Further, Licensee shall not sublicense, or assign this Agreement or any of the rights or licenses granted under this Agreement, without the prior written consent of Synplicity.

Security. Licensee agrees to take all appropriate measures to safeguard the SOFTWARE and prevent unauthorized access or use thereof. Suggested ways to accomplish this include: (i) implementation of firewalls and other security applications, (ii) use of FLEXIm options file that restricts access to the SOFTWARE to identified users; (iii) maintaining and storing license information in paper format only; (iv) changing TCP port numbers every three (3) months; and (v) communicating to all authorized users that use of the SOFTWARE is subject to the restrictions set forth in this Agreement.

Ownership of the SOFTWARE. Synplicity retains all right, title, and interest in the SOFTWARE (including all copies), and all worldwide intellectual property rights therein. Synplicity reserves all rights not expressly granted to Licensee. This license is not a sale of the original SOFTWARE or of any copy.

Ownership of Design Techniques. "Design" means the representation of an electronic circuit or device(s), derived or created by Licensee through the use of the SOFTWARE in its various formats, including, but not

limited to, equations, truth tables, schematic diagrams, textual descriptions, hardware description languages, and netlists. "Design Techniques" means the data, circuit and logic elements, libraries, algorithms, search strategies, rule bases, techniques and technical information incorporated in the SOFTWARE and employed in the process of creating Designs. Synplicity retains all right, title and interest in and to Design Techniques incorporated in the SOFTWARE, including all intellectual property rights embodied therein, provided that to the extent any Design Techniques are included as part of or embedded within Licensee's Designs, Synplicity grants Licensee a personal, nonexclusive, nontransferable license to reproduce the Design Techniques and distribute such Design Techniques solely as incorporated into Licensee's Designs and not on a standalone basis. Additionally, Licensee acknowledges that Synplicity has an unrestricted, royalty-free right to incorporate any Design Techniques disclosed by Licensee into its software, documentation and other products, and to sublicense third parties to use those incorporated design techniques.

Protection of Confidential Information. "Confidential Information" means (i) the SOFTWARE, in object and source code form, and any related technology, idea, algorithm or information contained therein, including without limitation Design Techniques, and any trade secrets related to any of the foregoing; (ii) either party's product plans, Designs, costs, prices and names; non-published financial information; marketing plans; business opportunities; personnel; research; development or know-how; (iii) any information designated by the disclosing party as confidential in writing or, if disclosed orally, designated as confidential at the time of disclosure and reduced to writing and designated as confidential in writing within thirty (30) days; and (iv) the terms and conditions of this Agreement; provided, however that "Confidential Information" will not include information that: (a) is or becomes generally known or available by publication, commercial use or otherwise through no fault of the receiving party; (b) is known and has been reduced to tangible form by the receiving party at the time of disclosure and is not subject to restriction; (c) is independently developed by the receiving party without use of the disclosure; and (e) is released for publication by the disclosing party in writing.

Each party will protect the other's Confidential Information from unauthorized dissemination and use with the same degree of care that each such party uses to protect its own like information. Neither party will use the other's Confidential Information for purposes other than those necessary to directly further the purposes of this Agreement. Neither party will disclose to third parties the other's Confidential Information without the prior written consent of the other party.

Third Party Software. Licensee understands and agrees that, although provided to Licensee by Synplicity, Licensee's use of each component library or module comprising the Third Party Software shall be governed by the relevant terms and conditions of the third party's license agreements.

Termination. Synplicity may terminate this Agreement immediately if Licensee breaches any provision, including without limitation, failure by Licensee to implement adequate security measures as set forth above. Upon notice of termination by Synplicity, all rights granted to Licensee under this Agreement will immediately terminate, and Licensee shall cease using the SOFTWARE and return or destroy all copies (and partial copies) of the SOFTWARE and documentation.

Limited Warranty and Disclaimer. Synplicity warrants that the program portion of the SOFTWARE will perform substantially in accordance with the accompanying documentation for a period of 90 days from the date of receipt. Synplicity's entire liability and Licensee's exclusive remedy for a breach of the preceding limited warranty shall be, at Synplicity's option, either (a) return of the license fee, or (b) providing a fix, patch, work-around, or replacement of the SOFTWARE. In either case, Licensee must return the SOFTWARE to Synplicity with a copy of the purchase receipt or similar document. Replacements are warranted for the

remainder of the original warranty period or 30 days, whichever is longer. Some states/jurisdictions do not allow limitations, so the above limitation may not apply. EXCEPT AS EXPRESSLY SET FORTH ABOVE, NO OTHER WARRANTIES OR CONDITIONS, EITHER EXPRESS, IMPLIED, STATUTORY OR OTH-ERWISE, ARE MADE BY SYNPLICITY OR ITS LICENSORS WITH RESPECT TO THE SOFTWARE AND THE ACCOMPANYING DOCUMENTATION, AND SYNPLICITY EXPRESSLY DISCLAIMS ALL WARRANTIES AND CONDITIONS NOT EXPRESSLY STATED HEREIN, INCLUDING BUT NOT LIM-ITED TO THE IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, NONINFRINGE-MENT, AND FITNESS FOR A PARTICULAR PURPOSE. SYNPLICITY AND ITS LICENSORS DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE SOFTWARE WILL MEET LICENSEE'S REQUIREMENTS, BE UNINTERRUPTED OR ERROR FREE, OR THAT ALL DEFECTS IN THE PRO-GRAM WILL BE CORRECTED. Licensee assumes the entire risk as to the results and performance of the SOFTWARE. Some states/jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply.

Limitation of Liability. IN NO EVENT SHALL SYNPLICITY OR ITS LICENSORS OR THEIR AGENTS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL OR INCIDENTAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROF-ITS, BUSINESS INTERRUPTIONS, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE, EVEN IF SYNPLIC-ITY AND/OR ITS LICENSORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. FURTHER, IN NO EVENT SHALL SYNPLICITY'S LICENSORS BE LIABLE FOR ANY DIRECT DAM-AGES ARISING OUT OF LICENSEE'S USE OF THE SOFTWARE. IN NO EVENT WILL SYNPLICITY OR ITS LICENSORS BE LIABLE TO LICENSEE FOR DAMAGES IN AN AMOUNT GREATER THAN THE FEES PAID FOR THE USE OF THE SOFTWARE. Some states/jurisdictions do not allow the limitation or exclusion of incidental or consequential damages, so the above limitations or exclusions may not apply.

Intellectual Property Right Infringement. Synplicity will defend or, at its option, settle any claim or action brought against Licensee to the extent it is based on a third party claim that the SOFTWARE as used within the scope of this Agreement infringes or violates any US patent, copyright, trade secret or trademark of any third party, and Synplicity will indemnify and hold Licensee harmless from and against any damages, costs and fees reasonably incurred that are attributable to such claim or action; provided that Licensee provides Synplicity with (i) prompt written notification of the claim or action; (ii) sole control and authority over the defense or settlement thereof (including all negotiations); and (iii) at Synplicity's expense, all available information, assistance and authority to settle and/or defend any such claim or action. Synplicity's obligations under this subsection do not apply to the extent that (i) such claim or action would have been avoided but for modifications of the SOFTWARE, or portions thereof, other than modifications made by Synplicity after delivery to Licensee; (ii) such claim or action would have been avoided but for the combination or use of the SOFT-WARE, or portions thereof, with other products, processes or materials not supplied or specified in writing by Synplicity; (iii) Licensee continues allegedly infringing activity after being notified thereof or after being informed of modifications that would have avoided the alleged infringement; or (iv) Licensee's use of the SOFTWARE is not strictly in accordance with the terms of this Agreement. Licensee will be liable for all damages, costs, expenses, settlements and attorneys' fees related to any claim of infringement arising as a result of (i)-(iv) above.

If the SOFTWARE becomes or, in the reasonable opinion of Synplicity is likely to become, the subject of an infringement claim or action, Synplicity may, at Synplicity's option and at no charge to Licensee, (a) obtain a license so Licensee may continue use of the SOFTWARE; (b) modify the SOFTWARE to avoid the infringement; (c) replace the SOFTARE with a compatible, functionally equivalent, and non-infringing product, or (d)

refund to Licensee the amount paid for the SOFTWARE, as depreciated on a straight-line 5-year basis, or such other shorter period applicable to Subscription Licenses.

THE FOREGOIN PROVISIONS OF THIS SECTION STATE THE ENTIRE AND SOLE LIABILITY AND OBLIGATIONS OF SYNPLICTY, AND THE EXCLUSIVE REMEDY OF LICENSEE, WITH RESPECT TO ANY ACTUAL OR ALLEGED INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHTS BY THE SOFTWARE (INCLUDING DESIGN TECHNIQUES) AND DOCUMENTATION.

Export. Licensee warrants that it is not prohibited from receiving the SOFTWARE under U.S. export laws; that it is not a national of a country subject to U.S. trade sanctions; that it will not use the SOFTWARE in a location that is the subject of U.S. trade sanctions that would cover the SOFTWARE; and that to its knowledge it is not on the U.S. Department of Commerce's table of deny orders or otherwise prohibited from obtaining goods of this sort from the United States.

Miscellaneous. This Agreement is the entire agreement between Licensee and Synplicity with respect to the license to the SOFTWARE, and supersedes any previous oral or written communications or documents (including, if you are obtaining an update, any agreement that may have been included with the initial version of the Software). This Agreement is governed by the laws of the State of California, USA excluding its conflicts of laws principals. This Agreement will not be governed by the U. N. Convention on Contracts for the International Sale of Goods and will not be governed by any statute based on or derived from the Uniform Computer Information Transactions Act (UCITA). If any provision, or portion thereof, of this Agreement is found to be invalid or unenforceable, it will be enforced to the extent permissible and the remainder of this Agreement will not constitute a waiver of the right to enforce rights with respect to the same or any other breach.

Government Users. If the SOFTWARE is licensed to the United States government or any agency thereof, then the SOFTWARE and any accompanying documentation will be deemed to be "commercial computer software" and "commercial computer software documentation", respectively, pursuant to DFAR Section 227.7202 and FAR Section 12.212, as applicable. Any use, reproduction, release, performance, display or disclosure of the SOFTWARE and accompanying documentation by the U.S. Government will be governed solely by the terms of this Agreement and are prohibited except to the extent expressly permitted by the terms of this Agreement.

June 2005

Contents

Amplify ISSP Pro Tutorial Overview2Tutorial Design2Prerequisites4Required Time4Task Outline4Download the Tutorial Files5Directory Structure6
Physical Synthesis Design Flow
Start the Amplify ISSP Pro GUI
Create Project
Set Implementation Options
Verify Timing Constraints
Run Constraint Check
View the Floorplan
Synthesize the Design
Analyze Results40Results Files41Reports41Physical Synthesis Reports49HDL Analyst Views51Physical Analyst View54
For More Information

Preface

Amplify ISSP Pro Tutorial for ISSP-90

•	Amplify ISSP Pro Tutorial Overview 2
•	Physical Synthesis Design Flow 10
•	Start the Amplify ISSP Pro GUI 12
•	Create Project
•	Set Implementation Options
•	Verify Timing Constraints
•	Run Constraint Check
•	View the Floorplan
•	Synthesize the Design
•	Analyze Results
•	For More Information

Amplify ISSP Pro Tutorial Overview

The goal of this tutorial is to familiarize you with the main features of the Amplify ISSP Pro physical synthesis tool so that you can use it to achieve optimal results for your own designs.

Tutorial Design

This tutorial uses a FIFO design which is configured to support independent read and write clock signals. The depth and width of the FIFO are configured by setting a parameter in the RTL code. The FIFO used in this tutorial is configured and optimized for a 1023×32 bit data buffer. FIFO circuits can be used in the following types of applications: processors, controllers, and various communications circuits.

See Figure 1 on page -3 for a graphical view of the FIFO architecture. For more information about the operation and functionality of the FIFO, see the vendor design specification documentation.

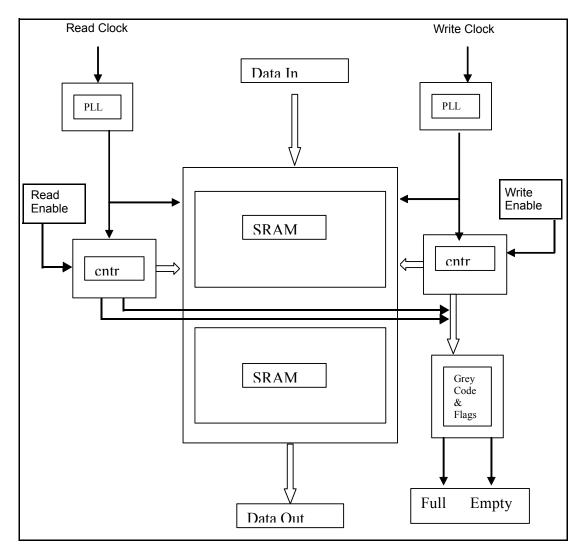


Figure 1: FIFO Design

Prerequisites

To complete the tutorial example in the following sections:

- You need a working knowledge of logic design synthesis and basic window operations.
- You should read Chapter 1, *Getting Started* of the User Guide, which presents a brief introduction to the Synplify ASIC and Amplify ASIC family of tools. Also, read through Chapter 2, *User Interface Overview* of the Reference Manual, which covers features and views of the Amplify ASIC Physical Synthesis UI.
- Create a local copy of the tutorial directory and be sure to delete the packet_buffer.prj file so you can create your own. See the *Directory Structure* on page 2-6 for information.

Required Time

The tutorial should take about 2 to 3 hours to complete.

Task Outline

Tasks in this tutorial include the following:

- Bring up the Amplify ISSP Pro GUI
- Open the project
- Check Implementation Options
- Check Timing Constraints
- Run Constraint Check
- View Floorplan
- Synthesize the Design
- Analyze Results

Download the Tutorial Files

You can download the tutorial design and project files along with the tutorial instructions from the Synplicity Technical Resource Center.

- 1. To access the Synplicity Technical Resource Center:
 - Select WEB->Go to Resource Center from the Project menu. Then, go to the Tutorial section within the Technical Resource Center.
 - Go to http://trc.synplicity.com/tutorials/index.html
- 2. Find the Amplify ISSP Pro tutorial for the applicable release and download these files for the platform you desire.
- 3. Unzip the tutorial files.
 - On a PC, use Winzip to extract the tutorial files.
 - On a UNIX platform, type the following at the command line:

```
gunzip amplify issp.tar.gz
```

Then, to extract the tutorial files, type this at the command line:

```
tar -xvf amplify issp.tar
```

- 4. Copy the amplify_issp/tutorial directory to your working area. Keep the directory structure, because the tutorial is based on this structure. Refer to *Directory Structure* on page 2-6. When you work on your own designs, you can set up the structure as you want.
 - **Note:** The tutorial instructions are contained in the tutorial.pdf, which is also located in this directory. Open this file when you are ready to begin the tutorial.
- 5. Make sure you have read and write privileges for the project files.

Directory Structure

This section contains the directory structure for tutorial, located at:

```
install directory/amplify issp/tutorial
```

where *install_directory* is the path to where your Amplify ISSP Pro tutorial files and instructions are located.

Figure 2 shows the tutorial directory in the Project view of the UI that reflects the virtual directory structure that is automatically created by the tool as you build your design project.

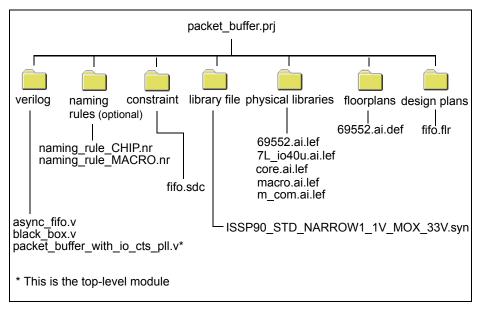


Figure 2: Tutorial Directory Structure

However, with your own designs, you can create any directory structure that works best.

Note: You can delete or rename the packet_buffer.prj and the fifo.sdc files in this directory so you can create your own as part of the tutorial exercises.

Input Files

Here is a brief description of the input files:

- .v contains the HDL source files. packet_buffer_with_io_cts_pll.v is the top level module.
- constraint:fifo.sdc user-specified constraint file, contains the timing constraints.
- packet_buffer.prj packet_buffer project file, contains all the information required to complete a design. This file contains references to source files, and specifications for the target ASIC technology.
- physical libraries:69552.ai.lef, core.ai.lef, 7L_io40u.ai.lef, m_com.ai.lef, macro.ai.lef — physical cell library information that includes process layers, design rules associated with interconnect layers, via cells and parasitics per layer. The libraries also include size, symmetry, and pin characteristics of standard cells and blockages of each macro.
- floorplans:69552.ai.def device floorplan file which contains information such as die size, rows and sites, pin locations, macro locations and orientation, or obstructions.
- design plans:fifo.flr device floorplan editor file which contains floorplan information and allows you to place, move, or change the orientation of macros, draw soft and hard placement obstructions, define regions constraints for instances of RTL modules, and create top-level bounding box constraints.
- library:ISSP90_STD_NARROW1_1V_MOX_33V.syn ISSP-90 technology library in Synplicity internal format. This file contains the merged .sel files, which describe the parameters and specifications for the target technology.
- naming rules:naming_rule_CHIP.nr and naming_rule_MACRO.nr includes a naming rules file for the chip and the macros such that the rules are defined and applied during synthesis.

Output Files

Figure 3 shows the directory structure for the results files, given the default name rev_1. You specify the directory for the results when you define implementation options for the project.

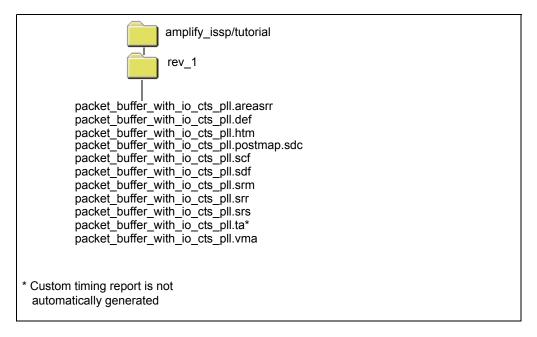


Figure 3: Results Directory Structure

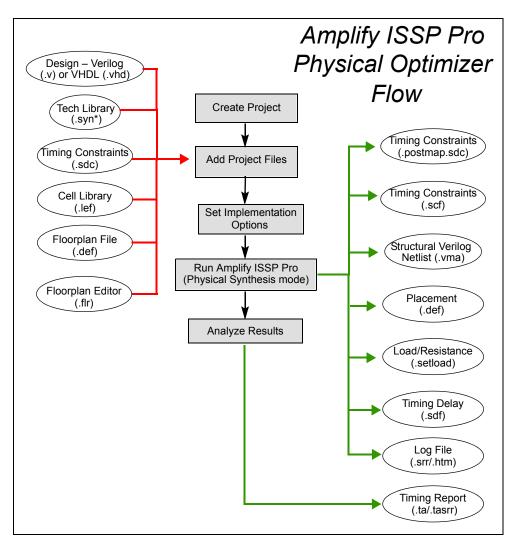
Here is a brief description of the result files that are written to the implementation during synthesis:

- .areasrr a hierarchical report containing area information for each of the modules in the design.
- .def output floorplan file, contains updated information for any of the following: die size, rows and sites, pin locations, macro locations and orientation, or obstructions.
- .htm HTML format of the log file containing the synthesis results. See the .srr file for a description of its contents.
- _ilm_lib.v verilog file containing black box description of all library cells.
- .postmap.sdc timing constraint file that is automatically created for in-place optimization.

- .scf Synopsys-style timing constraint file, used for forward annotation in place-and-route tools. See Appendix C, *Translating Synopsys Constraints* of the *Reference Manual* for information using these constraints in the Amplify ISSP Pro environment.
- .sdf standard delay format file containing pin-to-pin timing delay values for nets and instances and timing requirements for sequential elements.
- .srm output by the mapper stage of the process, contains the actual technology-specific mapped design. This is the representation displayed through the technology view in HDL Analyst and the floorplan view in Physical Analyst.
- .srr log file containing the synthesis results. The *project_name.srr* file contains all warnings and errors encountered during synthesis as well as performance information such as clock frequency, critical paths and run times. There is also information on area, cell usage and FSM extraction. To view this file, click on the View Log button in the Project view.
- .srs output by the compiler stage of the process, contains the RTL level (schematic) view of the design. This is the representation displayed through the RTL view in HDL Analyst and the Floorplan Viewer.
- .ta customized timing report (not generated by default). This file is created when you generate a special timing report using the Timing Report->Generate command.
- .vma structural Verilog design netlist for place and route and verification.

Physical Synthesis Design Flow

The typical flow for Amplify ISSP Pro Physical Optimizer is shown in the following figure.



- 1. Start the Amplify ISSP Pro GUI.
- 2. Create Project:
 - Add source files
 - Specify top-level module/entity
- 3. Set Implementation Options:
 - Choose target technology
 - Specify operating conditions and wire load model
 - Set optimization switches
 - Specify results directory
- 4. Verify Timing Constraints:
 - Compile the design before creating the constraints to initialize SCOPE
 - Set timing constraints through SCOPE
 - Modify timing constraints as appropriate to improve results
- 5. Run Constraint Check.
- 6. View the Floorplan.
- 7. Synthesize the Design.
 - Click Run.
- 8. Analyze Results:
 - Output Files
 - Area Report
 - Timing Report
 - Physical Analyst
 - Log File Warnings
- 9. When design goals are met, go on to place and route and/or verification.

Start the Amplify ISSP Pro GUI

How you start the tool depends on your environment.

- To start the software from a Windows platform, select Programs->Synplicity-> Amplify ISSP Pro from the Start menu.
- To start the software from a UNIX command line, type:

```
install directory/bin/amplify issp pro
```

where: *install_directory* is the path to where your Amplify ISSP Pro software is installed.

For more details on bringing up the tool, installation and licensing, see:

- Installation and Licensing in Chapter 1, Getting Started of the User Guide
- Getting Help in Chapter 1, Get Started of the Reference Manual
- Starting the ASIC Synthesis Tools in Chapter 1, Get Started of the Reference Manual

Create Project

Before synthesizing a design, you need to create a project for it. The project contains all the elements required to run synthesis on a design such as technology library, design files, constraint files, and so on. A project can have one or more associated revisions, also called implementations. Each time you change options, libraries, constraints, or optimization switches to enhance performance, you should create a new implementation for the project. This way you can compare the results of each of your implementations to determine the best combination of files and settings that yield the most optimum results for the design.

To open the project:

- 1. From the Project view, do one of the following:
 - Click on the Open Project button
 - Click on the project button in the toolbar (**P**)
 - Select File->Open Project

Open Project		×
Existing Project New Project New Workspace	Recent Projects	
OK Cancel		

Figure 4: Create Project

2. Click on New Project.

This creates a new project with an associated revision in the project view; default names are proj and rev_1.

C:\software\ampli	ify_issp\tutorial\proj.prj *				_ 🗆 X
Open Project	Bun Amplify [®] ISSP Pro				
Close Project	Tech Lib: <not selected=""></not>				
Add File	C:\software\amplify_issp\tutorial\rev_1 		rev_1	Туре	Modified
Change File	international i				
New Impl					
Impl Options					
View Log					
Physical Synthesis					
			•		Þ

Figure 5: Create Project

Add the design files to the project.

3. Click on the Add File button.

Select Files to Add to Project		? ×
Look in: 🔁 tutorial	- 🔶 🖻) 💣 🎟 -
Constraints		
def_fplan		
ISSP_DEMO_lib		
-		
File name:		
Files of type: Verilog Files (*.v;*.vma;*.vmd)	▼	
VHDL Library:		
Files To Add To Project: 🔽 Use Relat	ive Paths	
		<- Add All
		<- Add
		Remove All ->
		Remove ->
		ОК
		Cancel

- 4. Open the rtl_sims folder. Make sure Files of type field is showing either All Files (*) or Verilog Files (*.v).
- 5. For this exercise, add all the Verilog files in the folder. Click on the <-Add All button, then click OK.

Select Files to Add to Project	? ×
Look in: 🧰 rtl_sims 💌 🗲 🕻	• 🖬 📩
async_fifo.v black_box.v packet_buffer_with_io_cts_pll.v	
File name:	j
Files of type: Verilog Files (*.v;*.vma;*.vmd)]
VHDL Library:	
Files To Add To Project: 🔽 Use Relative Paths	
.\rtl_sims\async_fifo.v .\rtl_sims\black_box.v	<- Add All
. \rtl_sims\packet_buffer_with_io_cts_pll.v	<- Add
	Remove All ->
	Remove ->
	ОК
	Cancel

Figure 6: Add Verilog Files

A virtual design directory structure is created in the Project view. Figure 7 shows the UI after adding the design files.

C:\software\ampli	ify_issp\tutorial\proj.prj *				
Open Project	Bun Amplify [®] IS	SP Pro			
Close Project	Tech Lib: <not selected=""></not>				
Add File	C:\software\amplify_issp\tutorial\rev_1	rev_1	Туре	Modified	
Change File	En terilog En terilog En terilog				
New Impl	i async_nu.v i black_box.v ii packet_buffer_with_io_cts_pll.v				
Impl Options					
View Log					
Physical Synthesis					
,				Þ	
	J				

Figure 7: Add Source Files

6. Specify the top-level module.

C:\software\ampli	fy_issp\tutorial\pro	oj.prj *					_ 🗆 🗙
Open Project	Bun Amplify [®] ISSP Pro						
Close Project	Tech Lib: <not selected=""></not>						
Add File	C:\software\amplify_i	issp\tutorial\rev_1		ev_1	Туре		Modified
Change File	Figure 1 (project) Veriog Seriog Figure 2 (project) Figure 2 (project)			top-level			
New Impl							
Impl Options	• rev_1		ľ		mod	lule	
View Log							
Physical Synthesis							
			•				Þ

Figure 8: Top-Level Module

When you are using a design with only one HDL, you can do this by moving the top-level module to the last position in the list of source files in the Project view. (Click and drag on the module to move the position.) The top-level module is already in the last position for this example. 7. Click on the Add File button again and change to the ISSP_DEMO_lib/lef directory. Make sure Files of type field is showing Physical Library Files (*.lef).

For this exercise, add all the physical library files in the folder. Click on the <-Add All button, then click OK.

Select Files to Add to Project	? ×
Look in: 🗀 lef 💌 🗲 🗈 j	• 🎟
 7L_jo40u.ai.lef 69552.ai.lef core.ai.lef m_com.ai.lef m_com.ai.lef macro.ai.lef 	
File name:	
Files of type: Physical Library Files (*.lef)	
VHDL Library:	
Files To Add To Project: 🔽 Use Relative Paths	
.\ISSP_DEMO_lib\lef\69552.ai.lef .\ISSP_DEMO_lib\lef\7L_io40u.ai.lef	<- Add All
.VISSP_DEMO_libVlefxcore.ai.lef	<- Add
.VISSP_DEMO_libVlef\m_com.ai.lef	Remove All ->
	Remove ->
	ОК
	Cancel

A Physical Libraries directory structure is created in the Project view. The following figure shows the UI after adding the physical library files.

C:\software\ampli	ify_issp\tutorial\proj.prj *			
Open Project	Bun Amplify [®] IS	SP Pro		
Close Project	Tech Lib: <not selected=""></not>			
Add File	C:\software\amplify_issp\tutorial\rev_1	rev_1	Туре	Modified
Change File				
New Impl	En 69552.ai.lef En 7L_io40u.ai.lef			
Impl Options	En core.ai.lef			
View Log	m_com.ai.lef ■			
Physical Synthesis				
		•		Þ

Figure 9: Add Physical Library Files

- 8. Add the technology library files to the project. To do this:
 - Open the ISSP_DEMO_lib/syn folder.
 - Select the ISSP90_STD_NARROW1_1V_MOX_33V.syn file and add it to the project.

Select Files to Add to Project	? ×
Look in: 🔁 syn 💌 🗲 🖻	
▶]ISSP90_STD_NARROW1_1V_MOX_33V.syn	
File name: ISSP90_STD_NARR0W1_1V_M0X_33V.syn	
Files of type: Technology Library Files (*,lib)*,syn;*,sel;*,syr	
VHDL Library:	
Files To Add To Project: 🔽 Use Relative Paths	
.\ISSP_DEMO_lib\syn\ISSP90_STD_NARROW1_1V_MOX_3	<- Add All
	<- Add
	Remove All ->
	Remove ->
	ОК
▼►	Cancel

A Library File folder is created in the project view.

C:\software\ampli	lify_issp\tutorial\proj.prj *				
Open Project	Bun Amplify [®] ISSP Pro				
Close Project	Tech Lib: ISSP90_STD_NARROW1_1V_MOX_33V.syn, WLG: default				
Add File	C:\software\amplify_issp\tutorial\rev_1 rev_1	Type Modified			
Change File	Proj (project) Proj Physical Libraries				
New Impl	Elbrary File ISSP90_STD_NARROW1_1V_M0X_33V.syn → III rev_1				
Impl Options					
View Log					
Physical Synthesis					
	-	Þ			

- 9. Add the floorplan files to the project. To do this:
 - Open the def_fplan/def folder.
 - Select the 69552.ai.def file and add it to the project.

Select Files to Add to Project	<u>? ×</u>
Look in: 🗀 def 📃 🗢 🖻	I 💣 🎟 -
🗐 69552.ai.def	
File name: 69552.ai.def	
Files of type: Floorplan Files (*.def)	
VHDL Library:	
Files To Add To Project: Vise Relative Paths	
.\def_fplan\def\69552.ai.def	<- Add All
	<- Add
	Remove All ->
	Remove ->
	<u> </u>
	Cancel
	//

A Floorplans folder is created in the project view.

C:\software\ampl	lify_issp\tutorial\proj.prj *	
Open Project	Bun Amplify [®] ISSP Pro	
Close Project	Tech Lib: ISSP90_STD_NARROW1_1V_MOX_33V.syn, WLG: default	
Add File	C:\software\amplify_issp\tutorial\rev_1 rev_1 Type proj (project)	Modified
Change File	Veniog Physical Libraries	
New Impl	e- 📴 Library File	
Impl Options	→	
View Log		
Physical Synthesis		
	4	Þ

- 10. Add the design plans floorplan file to the project. To do this:
 - Open the def_fplan/floorplan folder.
 - Select the fifo.flr file and add it to the project.

Select Files to Add to Project	? ×
Look in: 🔁 floorplan 💌 🗲 🔁	💣 🎟 -
國 fifo.ftr	
File name: fifo.flr	
Files of type: Design Plans (*.flr)	
VHDL Library:	
Files To Add To Project: 🔽 Use Relative Paths	
.\def_fplan\floorplan\fifo.flr	<- Add All
	<- Add
	$RemoveAll\! \rightarrow$
	Remove ->
	ОК
	Cancel

A Design Plans folder is created in the project view.

Open Project Close Project	Run Amplify [®] ISS Tech Lib: ISSP90_STD_NAF		syn, WLG: default
Add File	C:\software\amplify_issp\tutorial\rev_1	rev_1	Type Mod
Change File			
New Impl	Library File Floorplans		
Impl Options	⊡		
View Log	▶ Ⅲ rev_1 (fifo.flr)		
Physical Synthesis			

- 11. This example provides optional naming rule files applied to the chip and macros of the design during synthesis. To add these files to the project, do the following:
 - Open the ISSP_DEMO_lib/naming_rule folder.
 - Select the naming_rule_CHIP.nr and naming_rule_MACRO.nr files.

Select Files to Add to Project		? ×
Look in: 🗀 naming_rule 💽 🗢 🖻) 📥 🛙	H •
國 naming_rule_CHIP.nr 國 naming_rule_MACRO.nr		
File name:		
Files of type: Naming Rules Files (*.nr)		
VHDL Library:		
Files To Add To Project: 🔽 Use Relative Paths		
\ISSP_DEMO_lib\naming_rule\naming_rule_CHIP.nr \ISSP_DEMO_lib\naming_rule\naming_rule_MACR0.nr	<-	Add All
	<-	Add
	Rem	ove All ->
	R	emove ->
		OK
•	0	Cancel

A Naming Rules folder is created in the project view.

C:\software\ampl	ify_issp\tutorial\proj.prj *		_ D ×
Open Project	Bun Amplify [®] ISS	P Pro	
Close Project	Tech Lib: ISSP90_STD_NAR	ROW1_1∨_MOX_33∨.:	syn, WLG: default
Add File	C:\software\amplify_issp\tutorial\rev_1	rev_1	Type Modified
Change File	Verilog Errica Physical Libraries		
New Impl	E Dibrary File		
Impl Options			
View Log	┣_ naming_rule_CHIP.nr □ ┣_ naming_rule_MACRO.nr		
Physical Synthesis	rev_1 (fifo.flr)		
		•	

12. Save the project file as packet_buffer.prj in the tutorial directory (File-> Save As).

C:\software\ampli	lify_issp\tutorial\packet_buffer.prj *	_ 🗆 🗵
Open Project	Bun Amplify [®] ISSP Pro	
Close Project	Tech Lib: ISSP90_STD_NARROW1_1V_MOX_33V.syn, WLG:	default
Add File	C:\software\amplify_issp\tutorial\rev_1 rev_1	Туре
Change File	⊕ <mark>⊡</mark> Naming Rules ⊕ <mark>⊡</mark> Verilog	
New Impl	⊕- 📴 Physical Libraries ⊕- 🛅 Floorplans	
Impl Options	⊕- 🚍 Design Plans ⊕- 🚍 Library File	
View Log	rev_1 (fifo.flr)	
Physical Synthesis		
		F

Set Implementation Options

The options you set for a project revision (implementation) determine the optimization settings and inputs such as the technology library, constraint files, operating conditions, and output directory for the synthesis run.

The exercise in this section presents a summary of the implementation options as they relate to the tutorial exercise. For more details on setting implementation options, see *Set Implementation Options* in Chapter 2, *Project Set-up* of the *User Guide*.

- 1. You can bring up the Options for Implementation dialog box in the Project view with one of the following:
 - Impl Options button
 - Select Project->Implementation Options
 - New Impl button (for creating a new implementation only)
- 2. Click on the Device tab.

Options for implementation: packet_buffer : rev_1	×
Device Options Constraints Floorplans Implementation Results Verilog	
	Implementations:
Technology Library Files C:\software\amplify issp\tutorial\ISSP DEMO_lib\syn\ISSP90_STD_NARR(Change	rev_1
ISSP Master Array: 69552	
Wire Load Selection and Mode Operating Conditions	
C Top C Enclosed	I
Vire Load Model Max (None) VISSP90 STD NARROW1 1V MC	
Wire Load Selection Group Min (None) (None)	Syn <i>plicity</i>
	Synplicity
Set Cell Usage Set Design Rules	
OK Cancel Help	

Figure 10: Set Device Options

In summary, this exercise uses the following Device options:

- technology library files = ISSP90_STD_NARROW1_1V_MOX_33V.syn
- issp master array = 69552
- wire load selection and mode = Enclosed
- wire load model = (None)
- wire load selection group = (None)
- operating conditions (max) = ISSP90_STD_NARROW1_1V_MOX_33V_MAX_ABDLSL333IV10.BEST_TREE
- operating conditions (min) = (None)
- 3. Click on the Options tab.

vice Options Constraints Floorplans Implementation Results Verilog	Implementations: rev_1
Resource Sharing	
Disable I/O Insertion Always Resynthesize Compile Points	
Disable Gated Clock Timing Checks	
Preserve Multiplier Boundaries	
Auto Infer Black Box	
Use Scan Cells for Test	
🗆 Stitch Scan Chains	
Option Description	
Performs physical synthesis based on physical libraries	
	Synplicity
,	Synplicity
OK Cancel Help	

Figure 11: Set Option Switches

The following switches are enabled for this exercise:

- Resource Sharing
- Disable I/O Insertion
- Prune Unused Registers
- Map Tristate Cells to Muxes

- Physical Synthesis
- SNAP
- Optimize Instantiated Gates
- Disable Sequential Optimization
- 4. Click on the Constraints tab.

Options for imp	lementation: packet_buffer : rev_1	×
Device Option Frequency	ns Constraints Floorplans Implementation Results Verilog Mhz)	Implementations: rev_1
Wire Resista	ance Scale Factor: 1.0 nce Scale Factor: 1.0 iles "Sel" column for files that apply to this implementation	
	File Module	Syn <i>plicity</i>
	OK Cancel Help	

Figure 12: Set Constraints

For this exercise, the target default frequency is set to 10 MHz. Also, the fifo.sdc constraint file is enabled to use for synthesis. The capacitance/resistance scaling factors that allow some flexibility in placement-based wire estimates is set to the following values:

- Wire Capacitance Scale Factor to 1.0.
- Wire Resistance Scale Factor to 1.0.

5. Click on the Floorplans tab.

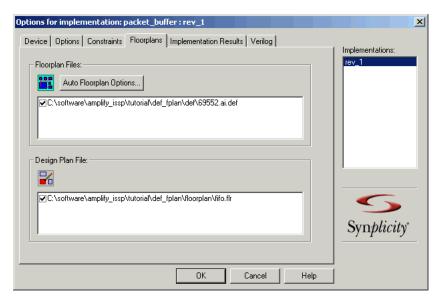


Figure 13: Set Floorplan Options

From this panel select the input floorplan files for your design. For this exercise, make sure the following floorplan files are enabled:

- 69552.ai.def
- fifo.flr

Options for implementation: packet_buffer : rev_1	×
Device Options Constraints Floorplans Implementation Results Verilog Implementation Name: [rev_1 Results Directory: [C:\software\amplify_issp\tutorial\rev_1 Browse Devit File Manage	Implementations: rev_1
Result File Name: Result Format: packet_buffer_with_io_cts_pll.vma vma ✓ Write SDF and Setload Files ✓ Write Mapped VHDL Netlist ✓ Write Timing Constraint File	Synplicity
OK Help	Synplicity

Figure 14: Set Implementation Results Options

6. Click on the Implementation Results tab.

From this panel, you specify the directory in which the results are written. The Amplify ISSP Pro tool automatically outputs:

- Structural Verilog netlist (.vma) that you can use for place and route and/or gate-level simulation.
- Constraints file for forward annotation (.scf)

7. Click on the Verilog tab.

For this exercise, the top-level module specified in the project is packet_buffer_with_io_cts_pll, so it is optionally not specified on this panel.

Dptions for implementation: packet_buffer : rev_1	×
Device Options Constraints Floorplans Implementation Results Verilog Top Level Module: Compiler Directives and Design Parameters Compiler Directives and Design Parameters Compiler Directives and Design Parameters	Implementations: rev_1
Verilog Language	
System Verilog Extract Parameters Compiler Directives: e.g. SIZE=8 Allow Duplicate Modules	
Include Path Order: (Relative to Project File)	5
	Synplicity
OK Cancel Help	

Figure 15: Set Verilog Options

- 8. Click OK to save the implementation options for rev_1 .
- 9. Save the project file (File-> Save).

Verify Timing Constraints

Amplify ISSP Pro synthesizes a design from top-down, so you only need to specify chip-level timing constraints before running synthesis. You can specify these constraints through the SCOPE (Synthesis Constraints Optimization Environment) UI which is a spread-sheet that creates timing constraints in Tcl command format. Through the SCOPE UI, you can define the following types of constraints:

- Clock definition
- I/O delays
- Input drive strength
- I/O loads on top-level ports
- Multi-cycle paths
- False paths
- Min/max paths
- Attributes
- Compile points
- Physical attributes
- Define clock delays

These constraints are defined in Chapter 3, *Timing Constraints Set-up*. See *Defining Constraints Using the SCOPE GUI* of the User Guide for details.

To specify constraints for the tutorial design:

1. Compile the design using Run->Compile Only or F7 so that the pertinent design information (such as net names, clocks, and so on) is available when you create timing constraints in SCOPE.

The job completes with some warnings, which can be ignored for now.

2. Open a constraints file. To do this, click on the SCOPE button (). An initialization dialog box displays.

📰 Create a New SCOPE File	×
Initialize Constraints Select File Type	
Initialize Constraints Click on the constraints you want SCOPE to initialize automatically Clocks All Input/Output Delays I/O Drivers and Loads None	
	OK Cancel Help

3. Leave all constraint types enabled and click OK.

This initializes the SCOPE file with pertinent design-object information. This way, you can select design object names from pull-down menus in certain fields of the SCOPE panels making it easier to define the constraints.

4. For this exercise, specify the clock definition, delays, multi-cycle paths, false paths, and attributes as shown in the following figures:

	Enabled	Clock Object	Clock Name	Period(ns)	Clock Group	Uncertainty	Rise	Fall	Ref Rise
1	P	READ_CLOCK_IN	READ_CLOCK_IN	9	default_clkgr	0.050	0.000	4.500	
2	R	pll_rd.CLKOA	pll_rd.CLKOA	9	default_clkgr	0.050	0.000	4.500	
3	R	WRITE_CLOCK_IN	WRITE_CLOCK_IN	11	default_clkgr	0.050	0.000	5.500	
4	R	pll_wr.CLKOA	pll_wr.CLKOA	11	default_clkgr	0.050	0.000	5.500	
5	R								
< P \	Clocks /								

Figure 16: Clocks

Figure 16 shows the clock definition for the fifo design. The design contains the following:

The READ and WRITE clocks from the pads:

- READ_CLOCK_IN

– WRITE_CLOCK_IN

The READ and WRITE clocks from the PLL:

- pll_rd.CLKOA
- pll_wr.CLKOA

	Enabled	Туре	Port	Clock Edge	Max Rise	Max Fall	Min Rise	Min Fall
1	N	input_delay	DATA_IN[31:0]	WRITE_CLOCK_IN:r	2.500	2.500		
2		input_delay	DATA_IN[31:0]	WRITE_CLOCK_IN:r			0.000	0.000
3		output_delay	DATA_OUT[31:0]	READ_CLOCK_IN:r	3.000	3.000		
4		output_delay	DATA_OUT[31:0]	READ_CLOCK_IN:r			1.000	1.000
5		output_delay	FIFO_EMPTY	READ_CLOCK_IN:r	3.000	3.000		
6		output_delay	FIFO_EMPTY	READ_CLOCK_IN:r			1.000	1.000
7		output_delay	FIFO_FULL	WRITE_CLOCK_IN:r	2.500	2.500		
8		output_delay	FIFO_FULL	WRITE_CLOCK_IN:r			1.500	1.500
9		input_delay	READ_ENABLE	READ_CLOCK_IN:r	3.000	3.000		
10		input_delay	READ_ENABLE	READ_CLOCK_IN:r			0.000	0.000
11		input_delay	WRITE_ENABLE	WRITE_CLOCK_IN:r	2.500	2.500		
12		input_delay	WRITE_ENABLE	WRITE_CLOCK_IN:r			0.000	0.000
<u>тр</u>)	l/O Delays	F			<u></u>	<u></u>	Ī	

Figure 17: I/O Delays

Figure 17 shows the I/O delays to set for the design.

	Enabled	From	То	Through	Start/End	Cycles	Comment	-
1	N		p:DATA_OUT[31:0]			2		
2			p:FIFO_EMPTY			2		-
Multi-Cycle Paths								

Figure 18: Multi-Cycle Paths

Figure 18 shows the multi-cycle path allowed to the ports DATA_OUT[31:0]. This constraint specifies 2 additional clock cycles to these paths for timing analysis and optimization.

	Enabled	From	То	Through	Comment	
1		p:SYS_RESET				1
2						
3						T
4 F				\False	Paths /	

Figure 19: False Paths

A false path is set for the paths originating from port SYS_RESET. See Figure 19. All of these paths are ignored during optimization. You might want to disable this false path constraint to find out if other critical paths exist from input ports.

5. Click on the Attributes tab and specify the syn_ideal_net on the nets as shown in the following figure. First, select the attribute from the pull-down.

	Enabled	Object Type	Object	Attribute	Value	Val Type	Description
1	N	net	n:pad_read_clock_in	syn_ideal_network	1	string	Do not buffer this network during optimization
2		net	n:pll_read_clock_out	syn_ideal_network	1	string	Do not buffer this network during optimization
3		net	n:read_clock_tree	syn_ideal_network	1	string	Do not buffer this network during optimization
4		net	n:pad_write_clock_in	syn_ideal_network	1	string	Do not buffer this network during optimization
5		net	n:pll_write_clock_out	syn_ideal_network	1	string	Do not buffer this network during optimization
6		net	n:write_clock_tree	syn_ideal_network	1	string	Do not buffer this network during optimization
7		net	n:pad_sys_reset	syn_ideal_network	1	string	Do not buffer this network during optimization
8		net	n:reset_tree	syn_ideal_network	1	string	Do not buffer this network during optimization
• •	<u></u>	<u>.</u>			Attrit	utes /	

Figure 20: Attribute Specification

The software treats any net with the syn_ideal_net attribute as if it has infinite drive, therefore it is not buffered during optimization. (See the description column Figure 20). Normally, a buffer tree is inserted during place and route for a reset net, so it does not need to be optimized.

	Enabled	Command	Arguments	Comment
1	ব	define_clock_delay	-rise WRITE_CLOCK_IN -rise READ_CLOCK_IN -false	
2	V	define_clock_delay	-fall WRITE_CLOCK_IN -rise READ_CLOCK_IN -false	
3	V	define_clock_delay	-rise WRITE_CLOCK_IN -fall READ_CLOCK_IN -false	*
4	V	define_clock_delay	-fall WRITE_CLOCK_IN -fall READ_CLOCK_IN -false	
5	V	define_clock_delay	-rise pll_wr.CLKOA -rise pll_rd.CLKOA -false	
6	V	define_clock_delay	-fall pll_wr.CLKOA -rise pll_rd.CLKOA -false	*
7	V	define_clock_delay	-rise pll_wr.CLKOA -fall pll_rd.CLKOA -false	
8	V	define_clock_delay	-fall pll_wr.CLKOA -fall pll_rd.CLKOA -false	
9	V	define_clock_delay	-rise READ_CLOCK_IN -rise WRITE_CLOCK_IN -false	
10	V	define_clock_delay	-fall READ_CLOCK_IN -rise WRITE_CLOCK_IN -false	
11	V	define_clock_delay	-rise READ_CLOCK_IN -fall WRITE_CLOCK_IN -false	
12	V	define_clock_delay	-fall READ_CLOCK_IN -fall WRITE_CLOCK_IN -false	*
13	V	define_clock_delay	-rise pll_rd.CLKOA -rise pll_wr.CLKOA -false	***************************************
14		define_clock_delay	-fall pll_rd.CLKOA -rise pll_wr.CLKOA -false	*
15	V	define_clock_delay	-rise pll_rd.CLKOA -fall pll_wr.CLKOA -false	*
16	V	define_clock_delay	-fall pll_rd.CLKOA -fall pll_wr.CLKOA -false	
►	False Path		Other /	

Figure 21: Other

Defines edge-to-edge delay for clocks in the design. The delay values specified in Figure 21 for this constraint override calculations made by the software.

- 6. Save the SCOPE file in the constraint directory as fifo.sdc.
- 7. Click Yes when you are prompted to add the constraint file to the project.
- 8. Close or minimize the SCOPE UI.
- 9. Save the project file.

For complete descriptions of the constraints and details on applying them, see Chapter 3, *Defining Constraints Using the SCOPE GUI* of the *User Guide*. See Appendix A, *Attributes and Directives* in the *Reference Manual* for information on attributes.

Run Constraint Check

Use the Run->Constraint Check command to generate a report that checks the syntax and applicability of the timing constraints in the .sdc file(s) and/or .scn (test constraints) file in your project. The report is written to the project name.cck file and contains information on the following items:

- · Constraints that are not applied
- · Constraints that are valid and applicable to the design
- · Wildcard expansion on the constraints
- · Constraints on objects that do not exist

Check the Constraint Check Report

When the constraint check command completes, the constraint check report is automatically opened. To open the file when it is closed:

- Double-click on the packet_buffer_with_io_cts_pll.cck file in the Implementation Results directory.
- Click on the Constraint Checker Report in the HTML log file.

The following figures display a sample of the report. The report starts with design information and a summary of the applicable and inapplicable constraints.

```
# Symplicity Constraint Checker, version 5.0, Build 168R, built Jul 22 2005
# Copyright (C) 2003-2005, Symplicity Inc. All Rights Reserved
# Written on Tue Aug 30 10:10:04 2005
Top View:
                  "packet buffer"
Constraint File(s):
                "C:\software\amplify_issp\tutorial\fifo.sdc"
Unconstrained Start/End Points:
                                     108
Inapplicable constraints:
                                     n.
Applicable constraints:
                                     42
Constraints with non-existent objects:
                                     Ω
Constraints with matching wildcard expressions:
                                     0
```

The following sections include details for each constraint check from the summary list provided above.

DETAIL3
JCIAIC3
Unconstrained Start/End Points
p:PLL_BYPASS p:PLL_LOCK_RD p:PLL_LOCK_WR p:PLL_LPS_RD[0] p:PLL_LPS_RD[1] p:PLL_LPS_RD[2] p:PLL_LPS_RD[3]
p:PLL_LPS_WR[0] p:PLL_LPS_WR[1] p:PLL_LPS_WR[2] p:PLL_LPS_WR[3]
Inapplicable constraints ******************
(none)
Applicable constraints ****************
define_attribute n:pad_read_clock_in syn_ideal_network l define_attribute n:pad_sys_reset syn_ideal_network l define_attribute n:pad_write_clock_in syn_ideal_network l define_attribute n:pll_read_clock_out syn_ideal_network l define_attribute n:pll_write_clock_tree syn_ideal_network l
<pre>define_attribute n:reset_tree syn_ideal_network 1 define_attribute n:write_clock_tree syn_ideal_network 1 define_clock READ_CLOCK_IN -name READ_CLOCK_IN -period 9.000 -clockgroup default_clkgroup -uncertainty 0.050 -rise 0.000 -fal define_clock WRITE_CLOCK_IN -name WRITE_CLOCK_IN -period 11.000 -clockgroup default_clkgroup -uncertainty 0.050 -rise 0.000 - define_clock pll_rd.CLKOA -name pll_rd.CLKOA -period 9.000 -clockgroup default_clkgroup -uncertainty 0.050 -rise 0.000 -fal define_clock pll_rd.CLKOA -name pll_wr.CLKOA -period 10.000 -clockgroup default_clkgroup -uncertainty 0.050 -rise 0.000 -fall define_clock delay -fall READ_CLOCK_IN -fall WRITE_CLOCK_IN -false define_clock_delay -fall WRITE_CLOCK_IN -rise WRITE_CLOCK_IN -false</pre>
define_clock_delay -fall WRITE_CLOCK_IN -rise READ_CLOCK_IN -false define_clock_delay -fall pll_rd.CLKOA -fall pll_wr.CLKOA -false define_clock_delay -fall pll_rd.CLKOA -rise pll_wr.CLKOA -false define_clock_delay -fall pll_wr.CLKOA -fall pll_rd.CLKOA -false define_clock_delay -fall pll_wr.CLKOA -rise pll_rd.CLKOA -false
define_output_delay FIFO_EMPTY -min_rise 1.000 -min_fall 1.000 -ref READ_CLOCK_IN:r define_output_delay FIFO_EMPTY -rise 3.000 -fall 3.000 -ref READ_CLOCK_IN:r define_output_delay FIFO_FULL -min_rise 1.500 -min_fall 1.500 -ref WRITE_CLOCK_IN:r define_output_delay FIFO_FULL -rise 2.500 -fall 2.500 -ref WRITE_CLOCK_IN:r define_output_delay FIFO_TULL -rise 2.500 -fall 2.500 -ref WRITE_CLOCK_IN:r define_output_delay (DATA_OUT[31:0]) -min_rise 1.000 -min_fall 1.000 -ref READ_CLOCK_IN:r define_output_delay (DATA_OUT[31:0]) -rise 3.000 -fall 3.000 -ref READ_CLOCK_IN:r
Constraints with non-existent objects
(none)
Constraints with matching wildcard expressions
(none)

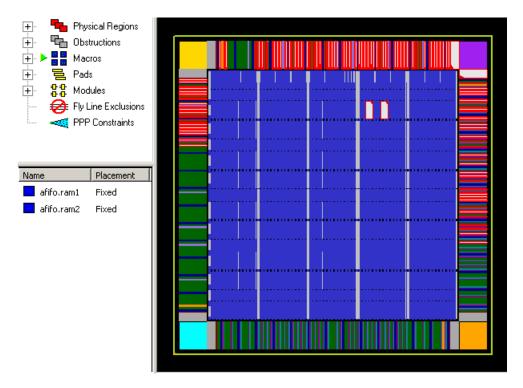
The final section contains a library report for the cell libraries.

```
Library Report
Library: ISSP90_CONFIG
   Cells:
        (no cells found with dont_use or dont_touch set)
Library: ISSP90_STD_NARROW1_1V_MOX_33V_MAX_ABDLSL333IV10
   Cells:
       ABDLSL333IV10
           dont_use: TRUE
dont_touch: TRUE
Library: ISSP90_STD_NARROW1_1V_MOX_33V_MAX_ABDLMA333IV10
   Cells:
       ABDLMA333IV10
           dont_use: TRUE
           dont_touch: TRUE
                      .
                      .
Library: ISSP90_STD_NARROW1_1V_MOX_33V_MIN_SPECIAL
   Cells:
        (no cells found with dont_use or dont_touch set)
Library: ISSP90 STD NARROW1 1V-M COM MIN
   Cells:
       OWSRAM072W512B32C3
           dont_use: TRUE
            dont_touch: TRUE
 End of Constraint Checker Report
```

View the Floorplan

The Floorplan Editor is a graphical analysis tool to display the floorplan. Since a good floorplan is an important phase of physical synthesis, use the Floorplan Editor to determine the current state and quality of the floorplan. The Floorplan Editor can be displayed anytime after the design has been compiled.

You can display device floorplan information, such as, die size, pin locations, macro locations and orientation, rows, or obstructions. The Floorplan Editor also provides additional features for graphically editing the floorplan. To bring up the Floorplan Editor, click on the Floorplan Editor icon (\bowtie). The following figure displays the floorplan for the tutorial design.



Synthesize the Design

To synthesize the design, click on the Run button in the Project view (or select Run->Synthesize). The Amplify ISSP Pro tool goes through the Compiling and Mapping phases. When the job is complete, Done! displays in the Project view. However, in this case there are some warnings to check.

The following figure shows the UI after synthesis is complete. A list of the output results files is shown in the right pane of the Project view. Double-click on the files to display them.

C:\software\ampli	fy_issp\tutorial\packet_buffer.prj Run Run			
	Tech Lib: ISSP90	_STD_NARROW1_1V_MOX_33V.syn, Master: 	69552, OpCond:	ISSP90_S
Add File	C:\software\amplify_issp\tutorial\rev_1	rev_1	Туре	Modified
Change File New Impl Impl Options View Log		jacket_buffer_with_io_cts_pll.sts packet_buffer_with_io_cts_pll.cck.str packet_buffer_with_io_cts_pll.cck.str packet_buffer_with_io_cts_pll.stm packet_buffer_with_io_cts_pll.ma packet_buffer_with_io_cts_pll.def packet_buffer_with_io_cts_pll.areastr packet_buffer_with_io_cts_pll.areastr packet_buffer_with_io_cts_pll.areastr packet_buffer_with_io_cts_pll.stf packet_buffer_with_io_cts_pll.str packet_buffer_with_io_cts_pll.str packet_buffer_with_io_cts_pll.str packet_buffer_with_io_cts_pll.str packet_buffer_with_io_cts_pll.str packet_buffer_with_io_cts_pll.str packet_buffer_with_io_cts_pll.str	RTL Netlist file Gate Netlist Structural verilog constraint floorplans file verilog models log file file	11:37:39 14:39:15 14:39:15 15:27:39 15:27:39 15:27:39 15:27:40 15:27:41 15:27:41 15:27:43 15:27:45
		•		Þ

Analyze Results

After a synthesis run you need to determine if your design has met the goals for area and performance before going on to place and route and/or verification. You can verify results through the reports generated during optimization and visually through the features of the HDL Analyst schematic views or the Physical Analyst floorplan view. When analyzing reports, here are the most important areas to cover:

- Warnings-determine which should be addressed and those that are for information only and can be ignored
- Timing results-determine if the target frequency for the design has been met.
- Area report-determine if the cell usage is acceptable for your design.

You can visually inspect areas of the design through the hierarchical RTL-level and technology-primitive level schematics generated by the HDL Analyst and Physical Analyst features. You can also view and isolate the critical paths, search for and highlight design objects and crossprobe between the schematics and source files. This section describes the various reports generated and presents a brief overview of the HDL Analyst and Physical Analyst views.

Results Files

After you compile or synthesize your design, the output files are created and are displayed in the right pane of the Project view. Double-click on the files to display them. See *Output Files* on page 2-7 for information on these files.

The log file is written (or overwritten) each time you compile or synthesize the project in two file formats: text (project_name.srr) and HTML with an interactive table of contents (project_name.htm). To view the log file in HTML, be sure to enable the View Log File in HTML option on the Options->Project View Options dialog box. This is the default.

Reports

After synthesis, results on timing and area are available in the log file located in the directory that you specified through the implementation options dialog box earlier in the process (in this case, rev_1). The contents of the log file for this exercise are in the text packet_buffer_with_io_cts_pll.srr or the HTML packet_buffer_with_io_cts_pll.htm file.

To display the log file, you can do any of the following:

- Double-click on the file in the Project view
- Click on the View Log button in the Project view

• Select View->Log File

The output is divided into sections that contain the following information:

- Compiler Report—lists the compiled modules for synthesis
- Mapper Report—contains a running history of the processes required to map your design
- DRC Report—lists the types and number of DRC errors

The reports mentioned above can output:

- Syntax or synthesis warnings, errors, and notes
- A running history of the process steps required during each phase of synthesis, respectively
- State machine extraction information, including a list of reachable states if the symbolic FSM compiler is turned on during synthesis
- Timing Reports

Additionally, the HTML log file separates output for the contents of the following sections:

- Session Log—displays a running history of the set-up, operating conditions, and processes used by the tool during synthesis
- Run Option—lists the user options set for synthesis
- Optional Reports—For example, Constraint Checker Report is displayed if you run the Constraint Check command.

Click on the sections listed in the table of contents of the HTML log file to take you to the top of the section you selected and display its contents on the right-side of the viewer.

The file contains these sections on timing information:

- Overall timing information on the design
- Performance summary for the design
- Clock relationships
- Information on top-level ports
- Five starting and ending points with worst slack
- Critical path

Delay numbers are computed based on the wire-load model, operating conditions and the constraints that you specify.

These are the most important areas to analyze in the report:

- Warnings-determine those that need resolution and those that are for information and can be ignored.
- Timing results-determine if the timing constraints for the design have been met.
- Area report-determine if the area is acceptable for your design.

The remaining sections provide examples of these reports.

Warnings

Warnings are displayed in the log file (packet_buffer_with_io_cts_pll.srr or packet_buffer_with_io_cts_pll.htm). Scroll through the file to check the warnings issued to determine which warnings, if any, require action. Figure 22 shows the results for packet_buffer. Your results might vary.

Int roy	1\packet_buffer_with_io_cts_pll.srr (log)
0078	Process took 0h:00m:00s realtime, 0h:00m:00s cputime Symplicity ASIC Technology Mapper, version 5.0. Build 168R, built Jul 22 2005
0079	Copyright (C) 1994-2005, Symplicity Inc. All Rights Reserved
0081	Reading naming rule file : C:software\anplify issp\tutorial\ISSP DEMO lib\naming rule\naming rule CHIP.nr
0082	Reading naming rule file : C:\software\anplify issp\utorial\ISSP DEMO lib\naming rule\naming rule\naming rule NACRO.nr
0083	@W:"C:\software\amplify issp\tutorial\ISSP DEMO lib\naming rule\naming rule MACRO.nr":12:26:12:26 found more than
0084	@W: "C:\software\amplify issp\tutorial\ISSF DEMO lib\naming rule\naming rule MACRO.nr":19:22:19:22 found more than
0085	@W: "C:\software\amplify issp\tutorial\ISSP DEMO lib\naming rule\naming rule MACRO.nr": 36:21:36:21 found more than
0086	@W:"C:\software\amplify_issp\tutorial\ISSP_DEMO_lib\naming_rule\naming_rule_MACRO.nr":50:32:50:32 found_more_than (
0087	Reading constraint file: C:\software\amplify_issp\tutorial\fifo.sdc
0088	Adding property syn_reference_clock1, value "READ_CLOCK_IN,r=0.000,f=4.500,u=0.050,p=9.000,clockgroup=default_clkg:
0089	Adding property syn_reference_clock2, value "pll_rd.CLKOA,r=0.000,f=4.500,u=0.050,p=9.000,clockgroup=default_clkgroup=
0090	Adding property syn_reference_clock3, value "WRITE_CLOCK_IN,r=0.000,f=5.500,u=0.050,p=11.000,clockgroup=default_cl
0091	Adding property syn_reference_clock4, value "pll_wr.CLKOA,r=0.000,f=5.500,u=0.050,p=11.000,clockgroup=default_clkg
0092	Adding property syn_multicycle_path1000, value "2 cycles to p:DATA_OUT[31:0]" to view:work.packet_buffer(verilog)
0093	Adding property syn_false_path1001, value "from p:SYS_RESET" to view work.packet buffer(verilog)
0094	Adding property syn_clock_2_clock_delay, value "-rise WRITE_CLOCK_IN -rise READ_CLOCK_IN -false" to view.work.pack Adding property syn_clock_2_clock_delay, value "-fall WRITE_CLOCK IN -rise READ_CLOCK_IN -false" to view.work.pack
0095	adding property syn_clock_c_clock_delay, value -rais write_COCK_IN -rise REAL_CLOCK_IN -raise to view.work.pack Adding property syn_clock_c_clock_delay, value -rise write_COCK_IN -fall READ_CLOCK_IN -false" to view.work.pack
0096	adding property syn_clock_c_clock_delay, value -rise write_COCK_IN -fall READ_CLOCK_IN -false to view.work.pack
0097	Adding property syn_clock 2 clock delay, value "-rise pll wr CIKOA -rise pll rd CIKOA -raise" to view work packet
0099	Adding property syn_clock 2_clock_delay, value "-fall pll wr.CLK0A -rise pll rd.CLK0A -false" to view work packet
0100	Adding property syn clock 2 clock delay, value "-rise pll wr CLKOA -fall pll rd CLKOA -false" to view work packet
0101	Adding property syn clock 2 clock delay, value "-fall pll wr CLKOA -fall pll rd CLKOA -false" to view:work.packet
0102	Adding property syn clock 2 clock delay, value "-rise READ CLOCK IN -rise WRITE CLOCK IN -false" to view:work.pack
0103	Adding property syn_clock_2_clock_delay, value "-fall READ_CLOCK_IN -rise WRITE_CLOCK_IN -false" to view:work.pack
0104	Adding property syn_clock_2_clock_delay, value "-rise READ_CLOCK_IN -fall WRITE_CLOCK_IN -false" to view:work.pack
0105	Adding property syn_clock_2_clock_delay, value "-fall READ_CLOCK_IN -fall WRITE_CLOCK_IN -false" to view:work.pack
0106	Adding property syn_clock_2_clock_delay, value "-rise pll_rd.CLKOA -rise pll_wr.CLKOA -false" to view:work.packet_
0107	Adding property syn_clock_2_clock_delay, value "-fall pll_rd.CLKOA -rise pll_wr.CLKOA -false" to view:work.packet_
0108	Adding property syn_clock_2_clock_delay, value "-rise pll_rd.CLK0A -fall pll_wr.CLK0A -false" to view:work.packet_
0109	Adding property syn_clock_2_clock_delay, value "-fall pll_rd.CLK0A -fall pll_wr.CLK0A -false" to view:work.packet_
0110	0%: MF231 [Running in 32-bit mode. Reading lib file: C:>software>anplify issp>tutorial>ISSP DEMO lib>syn>ISSP90 STD NARROW1 1V MOX 33V.syn
0111	weading fip file: c:software/amplify_issp/tutoria/lissp_uemo_fip/syn/issr/u_siD_MarkOw1_1v_MoA_33V.syn @W: BN248 [Using default thresholds for library 'ISSP90 CONFIG'
0112	W: BK246 Joshig default thresholias for library isstration complete for library ISSP90 CONFIG - scaling ignored for this libr.
	ev. Dicto princip scaring, nominar for are not complete for findary 155.0_contro - scaring ignored for ones find
	Line 1. Col 1

Figure 22: Log File Warnings

You can also check for synthesis messages in the Tcl window of the Project view by clicking on the Messages tab of the message viewer. Use the live links to bring up the online help for the message ID if it is documented or go the location in the HDL source file or log file associated with this message.

Туре	ID	Message	Source Location	Log Location	Time	Report
E 🚺 6				packet buffer with io cts pll.srr	10:15:21	Mapper Report
Ð \Lambda 5				packet buffer with io cts pll.srr	10:15:21	
- 🚺 2	AI254	Layer M5 is available in this design/technology for signal routing		packet buffer with io cts pll.srr	10:15:21	Mapper Report
	AI258	Calculated wire capacitance (from LEF) for horizontal routing: 0.266 fF/um		packet buffer with io cts pll.srr (163)	10:15:21	Mapper Report
	AI259	Calculated wire resistance (from LEF) for horizontal routing: 0.985 ohm/um		packet buffer with io cts pll.srr (164)	10:15:21	Mapper Report
	AI260	Calculated wire capacitance (from LEF) for vertical routing: 0.255 fF/um		packet buffer with io cts pll.srr (165)	10:15:21	Mapper Report
()	AI261	Calculated wire resistance (from LEF) for vertical routing: 0.985 ohm/um		packet buffer with io cts pll.srr (166)	10:15:21	Mapper Report
- <u>A</u>	AI335	Pin YB of TB7NAND2XB lies outside cell bounding box	-	packet buffer with io cts pll.srr (154)	10:15:21	Mapper Report
- 🔼 7	AI355	Embedded CTS switch is turned off. ECTS will not be performed		packet buffer with io cts pll.srr	10:15:21	Mapper Report
	AM192	Synthesizing using 'enclosed' mode wire load evaluation.		packet buffer with io cts pll.srr (133)	10:15:21	Mapper Report
	AM193	Using wire load selection table 'area_lookup_table_69552' in library 'ISSP	-	packet buffer with io cts pll.srr (134)	10:15:21	Mapper Report
	AM196	Loading Technology specific configuration file C:\Program Files\synplicity		packet buffer with io cts pll.srr (135)	10:15:21	Mapper Report
O	AM212	TNS optimization is OFF		packet buffer with io gts pll.srr (191)	10:15:21	Mapper Report

The log file contains the following types of warnings:

@W| RtMaxNetLength. Skip large-fanout(139)net(name=reset_tree_i_0)
for routing.

@W:"C:\software\amplify_issp\tutorial\ISSP_DEMO_lib\naming_rule\ naming_rule_MACRO.nr":12:26:12:26|found more than one max_length rule. The tightest will be used.

@W: AI335 |Pin YB of TB7NAND2XB lies outside cell bounding box

@W: AI355 |Array statement missing in the floorplan file. Master name from physical library(69552) will be used

@W: AI355 |Embedded CTS switch is turned off. ECTS will not be performed

@W: AI355:"C:\software\amplify_issp\tutorial\ISSP_DEMO_lib\lef\
7L_io40u.ai.lef":555546:0:555546:0|Parser warning message: SITE is
defined before ORIGIN.. Last read token was <;>

@W: AM240 |Versions of Library ISSP90_CONFIG and the ASIC mapper differ. The library version: 3.3.0N, Build 013R and Mapper version: 5.0, Build 168R

@W: BN248 |Using default thresholds for library 'ISSP90 CONFIG'

@W: BN216 |Library scaling: nominal PVT are not complete for library ISSP90_CONFIG - scaling ignored for this library; data left as-is

```
@W: MF215 :|Ignoring syn_ideal_net work on the clock
write_clock_tree_keep
@W: CG370 :"C:\software\amplify_issp\tutorial\rtl_sims\
async fifo.v":47:19:47:23|No assignment to wire ecomp
```

Unused wire inputs/outputs are ignored. You should assign connections to these wires if you want to keep these unused wires in the design. You can ignore the warning about the different versions of the library files and the mapper. The technology library file (.syn) was generated from a different version of the mapper software.

You would also need to fix design rule violations or any other warnings in the design that are not acceptable. However, for the tutorial exercise, you can ignore these warnings.

Timing Report

With each synthesis run, default timing information is reported that consists of the following sections, first for maximum delay, then for minimum delay:

- Performance summary, including the worst max/min slack for the design as well as the worst max/min slack for each clock in the design.
- Arrival and required time as well as slack for all the top-level ports of the design (interface information section)
- Five worst starting points in the design (ordered by slack) (a starting point is either a primary input or a register output or the output of a black-box).
- Five worst ending points in the design (ordered by slack) (an ending point is either a primary output or a register input or the input of a black-box).
- Critical path in the design, including the starting and ending points as well as each point in between.

Delay numbers are computed based on the wire-load model and operating conditions specified through the Implementation Options dialog box, and the constraints specified in the .sdc file.

Figure 23 shows the beginning of the Max Analysis timing report for the fifo design.

packet_buffer_with_io_cts_pl	Lhtm
	##### START OF MAX TIMING REPORT #####
rev 1 (packet buffer with	# Timing Report written on Tue Aug 30 10:15:20 2005
	#
Compiler Report	
Mapper Report	
High Fanout Net Report	Top view: packet_buffer Library name: ISSP90 STD NARROW1 1V MOX 33V MAX ABDLSL333IV10
DRC Report Summary	Operating conditions: ISSP90 STD NARROW1 IV MOX 33V MAX ABDLSL333IV10.BEST TREE
Max Timing Report	Horizontal wire capacitance: 0.266 fF/um(scale factor = 1.00)
Performance Summary	Horizontal wire resistance: 0.985 ohm/um(scale factor = 1.00)
	Vertical wire capacitance: 0.255 fF/um(scale factor = 1.00)
Interface Information	Vertical wire resistance: 0.985 ohm/um(scale factor = 1.00)
Input Ports	Slew propagation multiplier: 1.000000
Output Ports	Slew propagation mode: worst
	Paths requested: 1
	Constraint File(s): C:\software\amplify_issp\tutorial\fifo.sdc
Ending Points with V	
Max Worst Path Inf	
<u> </u>	
Log File Links:	Performance Summary
Session Log	**********
Session Log	
rev 1	The mouth slamb in designs 0,000
Constraint Checker Report (Max worst slack in design: 0.993
Constraint Checker Log (10: 🖵	Total negative slack in design: 0.000 (0/281 endpoints)

Figure 23: Timing Report

Check the Performance Summary section for the worst slack for this design. Depending on the version of the software that you are using and the platform you are running on, your results may vary.

Check all sections of the Max Analysis timing report, then the Min Analysis timing report to ensure the design has met the timing requirements. Scroll through and use Edit->Find to search the report.

You might need more detailed information than what is provided in the default report, such as information on a specific path or you might want to see more than just the worst path. For these requirements, you can customize a report using Analysis->Timing Options.

Report Type • Setup Time Report (Max Timing) • Hold Time Report (Min Timing) • Fitters • From: • To: • Limit Number Of Paths To: • Enable Slack Margin (ns): • • •	Ok Cancel Generate
C Hold Time Report (Min Timing) Filters From: Through To: Limit Number Of Paths To: 5	Cancel
Filters From: Through To: Limit Number Of Paths To: 5	
From: Through To: 5	Generate
Through To:	Generate
To:	Generate
Limit Number Of Paths To: 5	Generate
SDF Backannotation Use SDF Backannotation Data Input Verilog File:	
Input Verilog Library File:	
Input SDF File:	
-Output Files	
TA File (in implementation directory): packet_buffer.ta	
SRM File: C:\software\amplify_issp\tutorial\rev_1\packet_buffer_with_io	_cts
SDF File: C:\software\amplify_issp\tutorial\rev_1\packet_buffer_with_io	o_cts

Figure 24: Timing Report Generation

The Timing Report Generation dialog box is shown in Figure 24. You can also generate this report using back annotated SDF data.

- 1. Select Analysis->Timing Options.
- 2. Change the Limit Number of Paths to 5.
- 3. Make sure the name of the report is packet_buffer.ta, then click Generate.

This step generates and displays the report. You can see the file packet buffer.ta in the right pane of the Project view.

4. Double-click on this file and scroll through the results.

For complete details on reports, see *Generating Timing Reports* and *Analyzing Timing Reports* in Chapter 6, *Analyzing Synthesis Reports* of the User Guide.

Area Report

In the packet_buffer_with_io_cts_pll.srr (log file), scroll down to the Report for cell ... section which begins just after the timing report section. In the packet_buffer_with_io_cts_pll.htm file, click on Cell Usage in the HTML table of contents. Figure 25 shows a sample. Library cells are listed in order of quantity of use. The total area for the design is given at the end of the section.

Report for cell packet_buff Cell usage: cell TB71NVXC TD71C33NUU TB71NVXC TD71C33NUU TB71NVXC TD71C33NUU TB7NMX12XA TC7STDFFQREXHU TB7NAND2EXA TB7NAND2EXA TB7NAND2EXA TB7NAND2EXA TB7NAND2EXA TB7NAND2EXA TB7NAND2EXA TC7STDFFREXLU TC7STDFFREXLU TC7STDFFREXLU TC7STDFFREXLU TC7STDFFREXLU TC7STDFFREXLU TC7STDFFSEXA TB7NOR2XA TB7NOR2XA TB7NOR2XA TB7NOR2XA TB7NOR2XC TB7NOR2XC TB7NOR2XC TB7NOR2XC TB7NOR2XA TB7NOR2XA TB7NOR2XC TB7NOR2XH TB7NOR2XH TB7NOR2XH TB7NOR2XH TB7NOR2XH TD7STECKC33NUU TB7CTSFS	count 585 208 103 103 52 40 36 33 26 23 18 16 13 11 10 55 5 4 4 32 22 22 22 22 22 22 22 22 22 22 22 22	area 8 8 000 3 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 14 000 9 16 94 102 9 102 9 102 9 9 9 9 102 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9	Count *area 4680.000 624.000 927.000 208.000 208.000 360.000 297.000 297.000 297.000 234.000 322.000 64.000 64.000 612.000 612.000 64.000 612.000 64.000 21.000 64.000 21.000 352260.000 352260.000 352260.000 188.000 24.000 24.000 24.000 16.000 0.000 13.000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.00000 0.000000	<pre>% of area 1.25% 0.17% 0.00% 2.59% 0.06% 0.06% 0.00% 0.08% 0.08% 0.08% 0.08% 0.08% 0.08% 0.08% 0.02% 0.33% 0.02% 0.33% 0.02% 0.33% 0.02% 0.01% 0.00% 0.0</pre>	M1/L1L2/D1 0/2/0 0/1/0 0/0/0 1/0/0 0/0/1 0/1/0 0/0/0 1/0/0 0/4/0 0/4/0 0/2/1 1/2/0 1/2/0 1/2/0 1/2/0 0/0/1 0/2/1 1/1/0 0/2/1 1/1/0 0/2/1 1/2/0 0/0/0 1/2/0 0/0/0 0/0/0 1/2/0 0/0/	M1 0 0 103 0 40 336 0 10 10 0 10 10 0 0 5 5 4 4 0 0 0 2 2 2 0 1 0 0 0 1 0 0 1 0 1 0 0 1 0 1	L1 0 208 0 0 46 10 0 46 11 0 0 5 0 4 0 0 2 2 2 0 1 0 0 0 2 2 2 0 1 0 0 0 1 0 0 0 0	$\begin{array}{c} 12\\ 1170\\ 0\\ 0\\ 0\\ 52\\ 0\\ 0\\ 0\\ 46\\ 18\\ 16\\ 0\\ 11\\ 0\\ 0\\ 10\\ 0\\ 4\\ 0\\ 0\\ 4\\ 0\\ 0\\ 0\\ 4\\ 0\\ 0\\ 0\\ 4\\ 1\\ 0\\ 0\\ 0\\ 4\\ 1\\ 0\\ 0\\ 0\\ 1\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\$
TOTAL	1478	Total	373969.000	Desian		269	304	1357

Figure 25: Cell Usage Report

The cell usage section shows the following columns:

- cell-name of the library cell
- count-number of cells used in the design
- area-size of one cell in the units defined in the technology library
- count * area-total area used in the design all instances of the library cell

Figure 25 shows that total area for this design is 373969.0. Depending on the version of the software that you are using and the platform you are running on, your results may vary slightly. The area units in the report are determined by the units specified in the cell library.

Physical Synthesis Reports

In addition to the standard timing and area reports displayed in the log file (.srr/.htm), the physical synthesis reports include:

- Floorplan Summary
- Pin Pair and Fanout Reports
- Congestion Report
- Design Summary

Floorplan Summary

This information is provided as the floorplan file is loaded, before the Mapping Phase of synthesis begins. Search for the Floorplan Summary string to view the summary.

```
Floorplan summarv
Die Area:
                                 ((-5835.00 -5835.00) (5835.00 5835.00)) microns
Rows:
                                0
I/O Pins:
                                178
Placed Standard-cell Instances: 0
Fixed Standard-cell Instances:
                                0
Placed Macro Instances:
                                Π.
Fixed Macro Instances:
                                182
Physical-only Instances:
                                0
Soft placement blockages:
                                3
                                0
Placement blockages:
Routing blockages:
                                Ο
Soft Regions:
                                Ω
Logic Assignments:
                                0
```

Pin Pair and Fanout Reports

This report provides information on the pin pairs (includes ideal and clock nets) and fanouts (excludes ideal and clock nets) in the design. Search for the Pin Pair Report or the Fanout Report string.

```
Pin pair report (includes ideal and clock nets):
Net count: 1677
Total pin pairs: 2460
Average pin pairs: 1.466905
Max pin pair: 162 (reset_tree_i / YB)
Fanout report - (excludes ideal and clock nets):
Net count: 1666
Total fanout: 2170
Average fanout: 1.302521
Max fanout: 24 (afifo.write_allow_0_a2_0 / YB)
```

Congestion Report

This report identifies hot spots for congestion and the percentage of track utilization on the device. Congestion usage is computed by comparing the routing demands through the bins to the available routing resources in both the horizontal (x) and vertical (y) directions on the device. Search for the Congestion Report string. The congestion report should be used in conjunction with the congestion map displayed in the Physical Analyst.

See the Amplify ASIC Physical Optimizer Design Flow and the Physical Analyst chapters of the Synplicity ASIC User Guide for more information.

Design Summary

In addition to the floorplan summary presented earlier in the log file, this summary includes row and area utilization. These numbers show how much of the placeable areas are utilized by the design. Row utilization consists of cell information only. Area utilization includes cells and macros. Search for the Design Summary string.

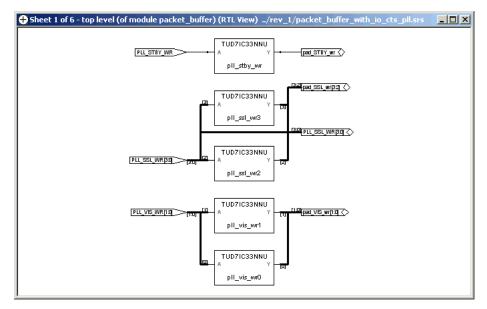
Design Summary	
*********** Top viev: Physical Library(s): Floorplan file(s):	0 11670.00 X 11670.00 microns ² 178 182 1296
COREFF site utilization: COREIVC site utilization: COREIVS site utilization:	0% 0% 1770
COREIVS site utilization: COREMX site utilization:	0% 0%

HDL Analyst Views

HDL Analyst is a graphical productivity tool that helps you visualize the synthesis results. HDL Analyst generates hierarchical RTL-level and technology-primitive level schematics from VHDL and Verilog designs, and lets you crossprobe between the RTL-level and technology-level views and your HDL source code. HDL Analyst also highlights and isolates critical paths within your design so you can analyze problem areas, add timing constraints and resynthesize.

- 1. Generate the RTL view for your design with the corresponding button in the UI (+) or through HDL Analyst->RTL->Hierarchical View.
- 2. Generate the Technology view using the corresponding button (()) or through HDL Analyst->Technology->Hierarchical View.

Figure 26 and Figure 27 show the RTL and Technology views for the fifo design.





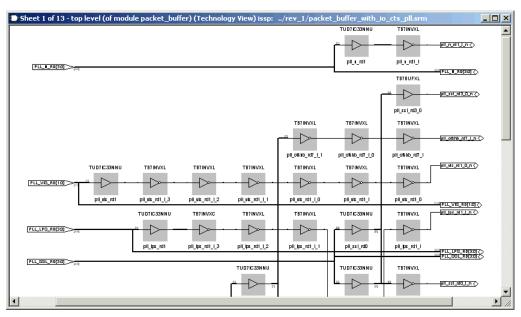


Figure 27: Technology View (Zoomed in to a Portion of the View)

View Critical Path

View the critical path for the design in the Technology view. To do this:

- 1. Flatten the Technology view schematic using HDL Analyst->Flatten current schematic (or HDL Analyst->Technology ->Flattened View).
- 2. Click on the Show Critical Path button in the UI (^(*)) or select HDL Analyst->Show Critical Path from the menus.

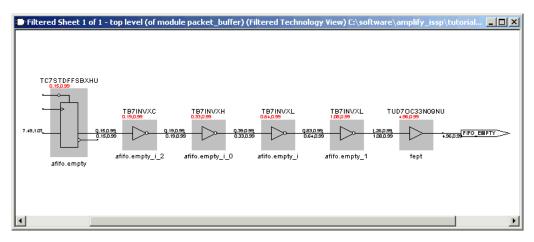


Figure 28: Show Critical Path

Figure 28 shows the critical path for the afifo.empty. You can also show this view selecting HDL Analyst -> Technology -> Flattened Critical Path.

3. Zoom into the blocks on the right side of the view.

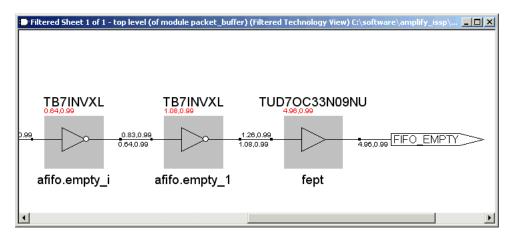


Figure 29: Slack and Cumulative Delay

Timing numbers are annotated on the top-left corner of the instances, showing the cumulative delay to the instance and slack time of the path that goes through the instance. As shown in Figure 29, slack is 4.96 ns and delay is 0.99 ns to instance: fept.

Depending on the version of the software that you are using and the platform you are running on, your results may vary slightly.

Physical Analyst View

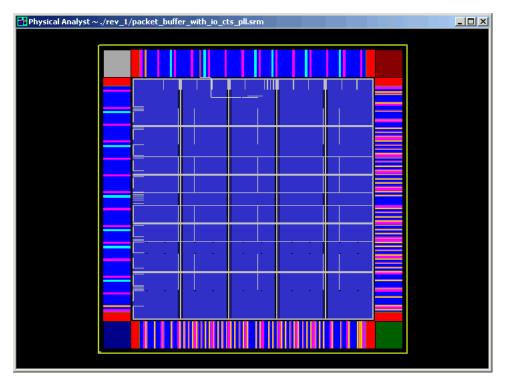
The physical analyst provides a visual and analytical display of the floorplan, placement, and global routing of the design after physical synthesis. The Physical Analyst view shows the device, device ports, instances, nets, obstructions, and congestion and row utilization maps. The view displayed is flat, although the hierarchy of the instance name is retained.

To bring up the Physical Analyst viewer, click on the Physical Analyst icon (E) from the Physical Analyst toolbar.

View Critical Path

The Physical Analyst tool makes it simple to find and examine critical paths and the relevant source code. The following procedure shows you how to filter and analyze a critical path.

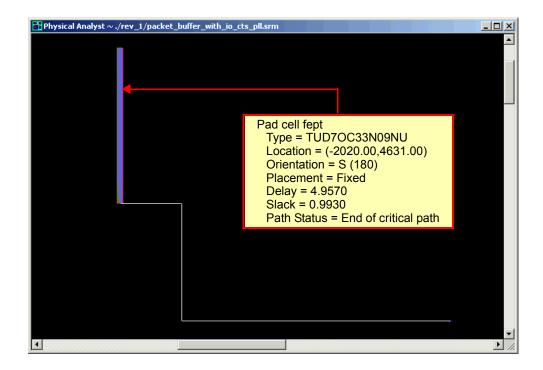
- 1. Display the critical path using one of the following methods. The Physical Analyst view highlights the instances and nets in the most critical path of your design. To generate a view of the critical path:
 - Click on the Show Critical Path icon (stopwatch icon 📷).
 - Right-click and select Critical Path->Show Critical Path from the popup menu.
- 2. To view only the critical timing path, first filter all objects. To do this:
 - Click on the Filter on Selected Gates icon (*).
 - Right-click and select Filter->Show Selected from the popup menu.



- 3. You can also filter site locations from the device so that you have a better view of the critical path in the Physical Analyst view. To do this:
 - Click on the Physical Analyst Control Panel icon (
) to display the Control Panel.
 - On the Objects tab of the Control Panel toggle visibility of sites off.

👪 Physical Analyst ~ ./rev	/_1/packet_buffer_w	ith_io_cts_pll.srm		
Obstructions				
Enhance 💻)	
Status Coloring 🗖				
			L	
Signals 🗖 🗖				
Signal Flow				
Prune Signals 🗖				
Power 🖾 🖾				
Ports 🛛				
Enhance 🖂				
Names 🖂				
Regions X				
DEF Placement				
DEF Routing				
FLR Placement 🖾 🖾				
Sites				
Macro 🖂				
Pad 🖂				
Core 🖂				
Cannot Occupy 🖂	-			
Objects (Maps) Layers)	No. of the No. of the No. of the			

- 4. To find the critical path end point:
 - Right-click and select Find from the popup menu.
 - On the Object Query dialog box of the Instances tab, select the Critical path end command from the Filter Search option. See *Object Query* on page 2-59.
 - Use the Find All button to locate all instances of this search. For this example, move the selected instance dout0 from the Unhighlighted to the Highlighted pane.
 - Close the dialog box. The critical path end point is highlighted in the Physical Analyst view.
 - Zoom in on this object.
 - Move the cursor over the critical path end point instance. A tool tip displays the cumulative delay to the instance and slack time of the path that goes through the instance.



Display the Congestion Map

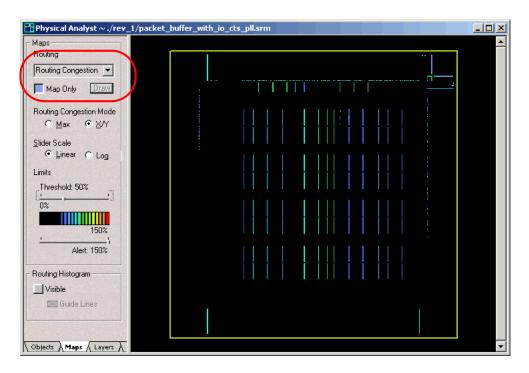
The routing congestion map visually displays routing congestion. The routable area is divided into bins. As nets are routed, their routing demands (design and bins) are compared to the routing resources (technology, layers, pitch, and obstructions) for each bin.

A graduated color bar displays a progression of up to 20 colors from the threshold percentage value up to the alert percentage value. Congestion hot spots are displayed using the burn or alert colors. Congestion utilization is based on the maximum percentage value in either the horizontal (X) or vertical (Y) direction for each bin. A tool tip displays these X and Y percentage utilizations. The higher the percentage, the higher the routing density.

To display the congestion map in the Physical Analyst view:

1. Click on the Physical Analyst Control Panel icon (
) to display the Control Panel.

- 2. On the Maps tab of the Control Panel, select Routing Congestion from the drop-down menu for the Maps Routing option. Then click Draw.
- 3. In the Physical Analyst view, use tool tips to display the X and Y percentage utilizations on the device with the threshold set to 50%.



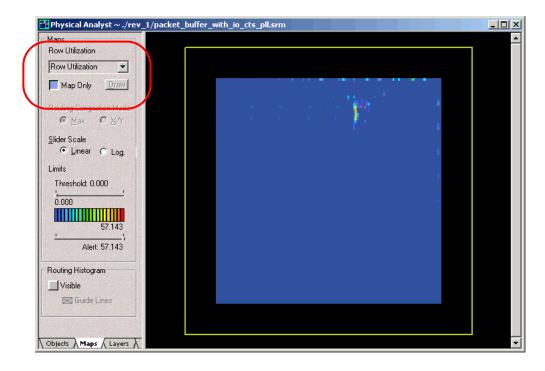
Display the Row Utilization Map

The row utilization map provides visual and analytical display of the percentage of site utilization for each bin in the placeable area. The row utilization data is available after the design is placed.

A graduated color bar displays a progression of up to 20 colors that ranges from the threshold value up to the alert value. Bins with high site utilization are displayed using the burn or alert colors. Bins below the threshold level are displayed in the cool (blue) color; tool tips are available for displaying the percentage of utilization.

To display the row utilization map in the Physical Analyst view:

- 1. Select Row Utilization from the drop-down menu on the Maps tab of the Control Panel. Then click the Draw button.
- 2. When Utilization is enabled, the row utilization map is displayed and tool tips reporting the percentage of utilization for each bin are available.



Object Query

You can highlight various design objects in the schematics and cross probe between the views, the source code and the log files. To bring up the Object Query dialog box, in the RTL, Technology, or Physical Analyst view, right-click and select Find in the popup menu. Go through the tabs and highlight different objects in the schematics or floorplan.

Object Query			×
Instances Symbols Nets Search © Entire Design	Ports	& Below O Current Level Only	
UnHighlighted: 0 of 241 afifo afifo.almostemptyg afifo.almostfullg afifo.data_mux[31:0] afifo.data_out_r[31:0] afifo.empty afifo.empty_3		Highlighted: 0 of 0	
afifo.emptyg afifo.full Highlight Search (*?): *	 	Un-Highlight Selection (*?):	•
Done	Find All	Tech View Dump to loc Netlist	ation
		Close	Help

Figure 30: Object Query–Instances

Object Query				×
Instances Symbols Nets	Ports			
Search				
Entire Design	C Current L	evel & Below	C Current Level 0	nly
UnHighlighted: 0 of 200	•	→ Hi	ighlighted: 0 of 0	
DATA_IN[31:0] DATA_OUT_1[31:0] FIFO_EMPTY FIFO_FULL		<-		
pad_data_in[31:0] pad_data_out[31:0] pad_fifo_empty		All ->		
pad_fifo_full pad_LOCK_rd	-	<- All		
Highlight Search (*?):		Un	-Highlight Selection (*?):	
×	•			•
200 More	Find All		🔽 Jump	to location
			Close	Help

Figure 31: Object Query-Nets

For More Information

For more information on all the features covered in the tutorial design flow, see Chapter 3, *Timing Constraints Set-up*, Chapter 6, *Analyzing Synthesis Reports*, and Chapter 2, *Project Set-up* in the User Guide. Chapter 7 and Chapter 8 also provide complete details on using HDL Analyst and Physical Analyst for the following:

- Flattening Hierarchy
- Filtering Schematics
- Extending Selected Logic
- Viewing Critical Paths in HDL Analyst
- Viewing Critical Paths in Physical Analyst
- Handling Negative Slack