

# Lattice Synthesis Engine for ispLEVER Classic Tutorial



May 2015

---

## Copyright

Copyright © 2015 Lattice Semiconductor Corporation. All rights reserved. This document may not, in whole or part, be reproduced, modified, distributed, or publicly displayed without prior written consent from Lattice Semiconductor Corporation (“Lattice”).

## Trademarks

All Lattice trademarks are as listed at [www.latticesemi.com/legal](http://www.latticesemi.com/legal). Synopsys and Synplify Pro are trademarks of Synopsys, Inc. Aldec and Active-HDL are trademarks of Aldec, Inc. All other trademarks are the property of their respective owners.

## Disclaimers

NO WARRANTIES: THE INFORMATION PROVIDED IN THIS DOCUMENT IS “AS IS” WITHOUT ANY EXPRESS OR IMPLIED WARRANTY OF ANY KIND INCLUDING WARRANTIES OF ACCURACY, COMPLETENESS, MERCHANTABILITY, NONINFRINGEMENT OF INTELLECTUAL PROPERTY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL LATTICE OR ITS SUPPLIERS BE LIABLE FOR ANY DAMAGES WHATSOEVER (WHETHER DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL, INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OF OR INABILITY TO USE THE INFORMATION PROVIDED IN THIS DOCUMENT, EVEN IF LATTICE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME JURISDICTIONS PROHIBIT THE EXCLUSION OR LIMITATION OF CERTAIN LIABILITY, SOME OF THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU.

Lattice may make changes to these materials, specifications, or information, or to the products described herein, at any time without notice. Lattice makes no commitment to update this documentation. Lattice reserves the right to discontinue any product or service without notice and assumes no obligation to correct any errors contained herein or to advise any user of this document of any correction if such be made. Lattice recommends its customers obtain the latest version of the relevant information to establish that the information being relied upon is current and before ordering any products.

---

## Type Conventions Used in This Document

Convention	Meaning or Use
<b>Bold</b>	Items in the user interface that you select or click. Text that you type into the user interface.
<i>&lt;Italic&gt;</i>	Variables in commands, code syntax, and path names.
<b>Ctrl+L</b>	Press the two keys at the same time.
<code>Courier</code>	Code examples. Messages, reports, and prompts from the software.
<code>...</code>	Omitted material in a line of code.
<code>.</code> <code>.</code> <code>.</code>	Omitted lines in code and report examples.
[ ]	Optional items in syntax descriptions. In bus specifications, the brackets are required.
( )	Grouped items in syntax descriptions.
{ }	Repeatable items in syntax descriptions.
	A choice between items in syntax descriptions.

---

# Contents

<b>Lattice Synthesis Engine for ispLEVER Classic Tutorial</b>	<b>1</b>
Learning Objectives	1
Time to Complete This Tutorial	1
System Requirements	1
Accessing Online Help	2
About the Tutorial Design	2
About the Tutorial Data Flow	2
Task 1: Create a New Project	3
Task 2: Target a Device	4
Task 3: Import the VHDL Source Files	6
Task 4: Fit the Design and View the Report	8
Task 5: Perform Static Timing Analysis	10
Summary	12
Glossary	13
Recommended Reference Materials	13



# Lattice Synthesis Engine for ispLEVER Classic Tutorial

This tutorial shows you how to use Lattice Synthesis Engine (LSE) from within ispLEVER® Classic to synthesize a VHDL design for a Lattice CPLD device.

## Note

---

If you want to learn more about LSE, see the *Lattice Synthesis Engine for ispLEVER Classic User Guide* and the ispLEVER Classic online Help.

---

## Learning Objectives

When you have completed this tutorial, you should be able to do the following:

- ▶ Create a new VHDL project in ispLEVER Classic and target a device.
- ▶ Fit the design, generate a JEDEC file, and view the Fitter report.
- ▶ Run static timing analysis using the Performance Analyst and view the results.

## Time to Complete This Tutorial

The time to complete this tutorial is about 20 minutes.

## System Requirements

- ▶ ispLEVER Classic

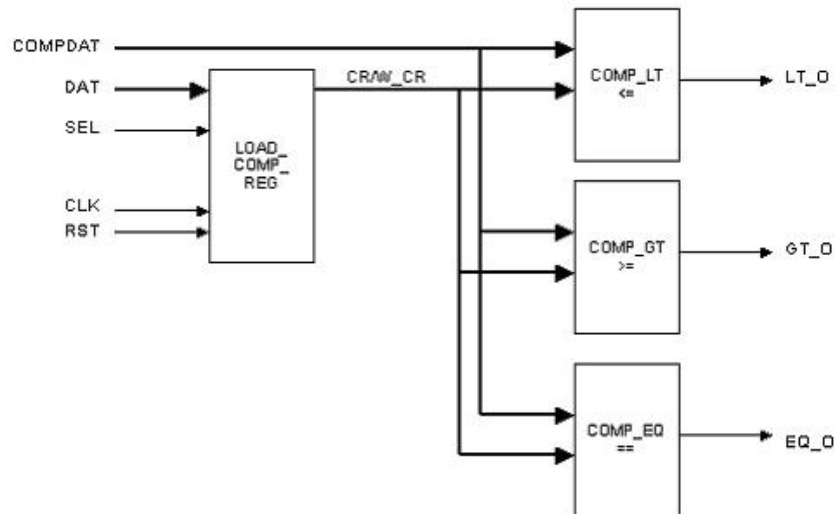
## Accessing Online Help

You can find online Help information on any tool included in the tutorial at any time by pressing the F1 key.

## About the Tutorial Design

The tutorial design consists of a simple set of equal-to, greater-than, and less-than data comparators, as shown in Figure 1:

Figure 1: Tutorial Design

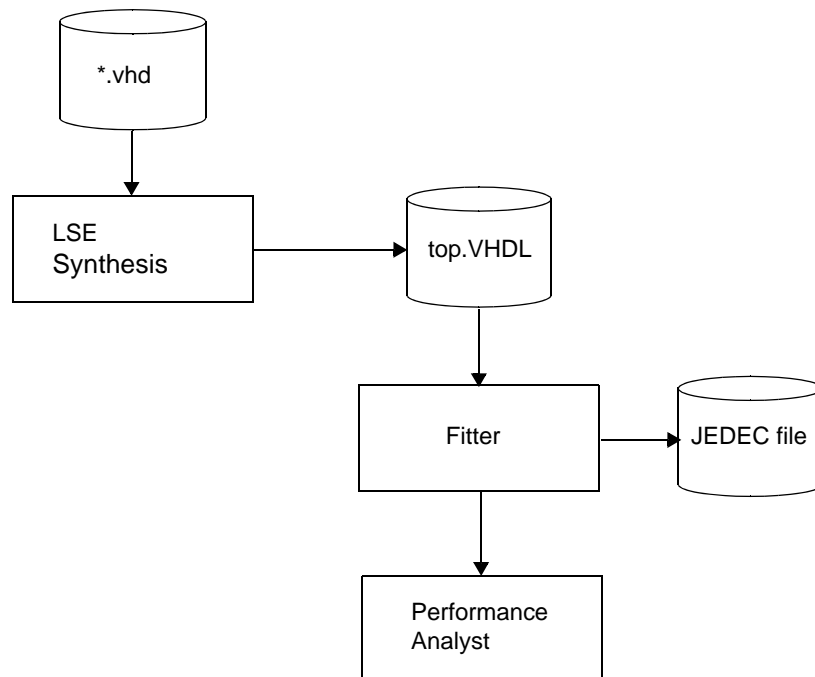


This tutorial first directs you to create a VHDL project in the Project Navigator, then select the target device in which the design will be implemented. It assumes that functional simulation has already been performed. You then import the VHDL files into the Project Navigator project and fit the design. Finally, you perform a static timing analysis and examine the results.

## About the Tutorial Data Flow

Figure 2 illustrates the design flow that the tutorial takes. You may find it helpful to refer to this diagram as you move through the tutorial tasks.



**Figure 2: Tutorial Design Flow**

## Task 1: Create a New Project

To begin a new project in the Project Navigator, you must create a project directory. Then give the project file a name (.syn) and declare the project type (VHDL).

The ispLEVER Classic software saves an initial design file with the .syn file extension in the directory that you specify. All project files are copied to or created in this directory. The project type specifies that all design sources will be of this type.

### To create a new project:

1. Start the ispLEVER Classic system, if it is not already running.
2. In the Project Navigator, choose **File > New Project** to open the Project Wizard dialog box.
3. In the dialog box, do the following:
  - a. In the Project Name box, type **compare**.

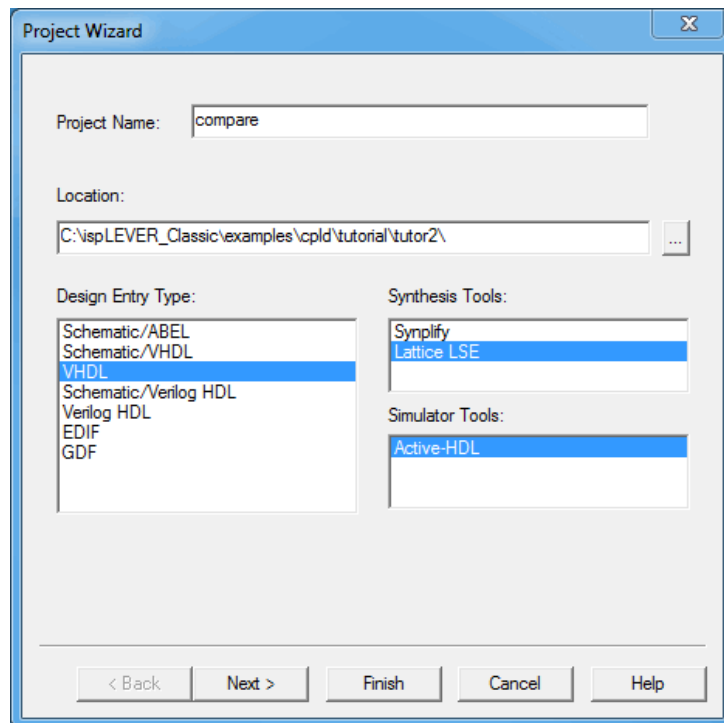
- b. In the Location box, change to the following directory:  
 <install\_path>\examples\cpld\tutorial\_lse\tutor2

**Note**

If you want to preserve the original tutorial design files, save the tutor2 directory to another location on your computer before proceeding.


- c. In the Design Entry Type box, select **VHDL**.  
 d. In the Synthesis Tools box, select **Lattice LSE** as shown in Figure 3.

**Figure 3: Project Wizard Dialog Box**



- e. Click **Next** to activate the Project Wizard – Select Device dialog box.

## Task 2: Target a Device

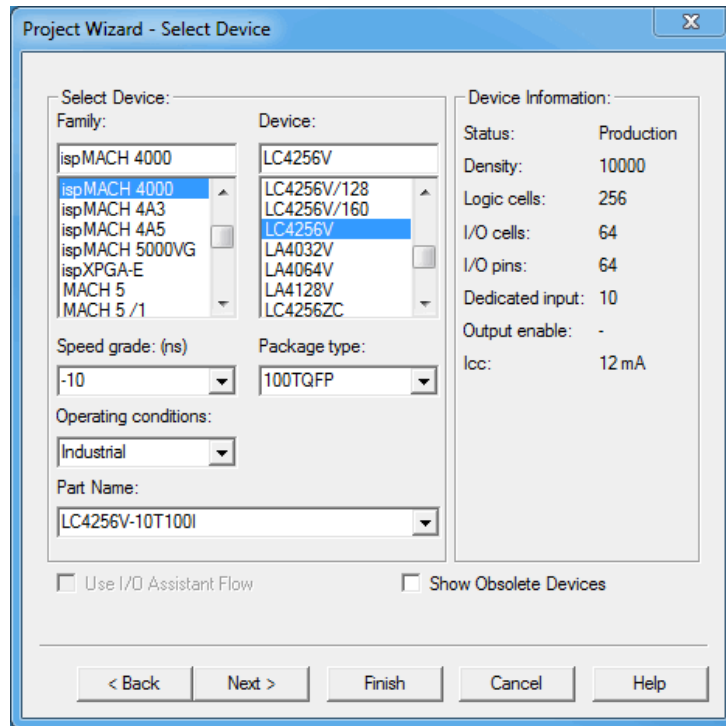
In the Project Navigator Sources in project window, the device icon  appears next to the target device for the project. The Project Navigator enables you to target a design to a specific Lattice device at any time during the design process.

**To view the list of available devices and to change the target device:**

1. In the Project Wizard – Select Device dialog box, do the following:
  - a. In the Family box, choose **ispMACH 4000**.
  - b. In the Device box, choose **LC4256V** from the list.

- c. Accept the default settings as shown in Figure 4, and click **Next** to activate the Project Wizard - Add Source dialog box.

**Figure 4: Project Wizard - Select Device Dialog Box**



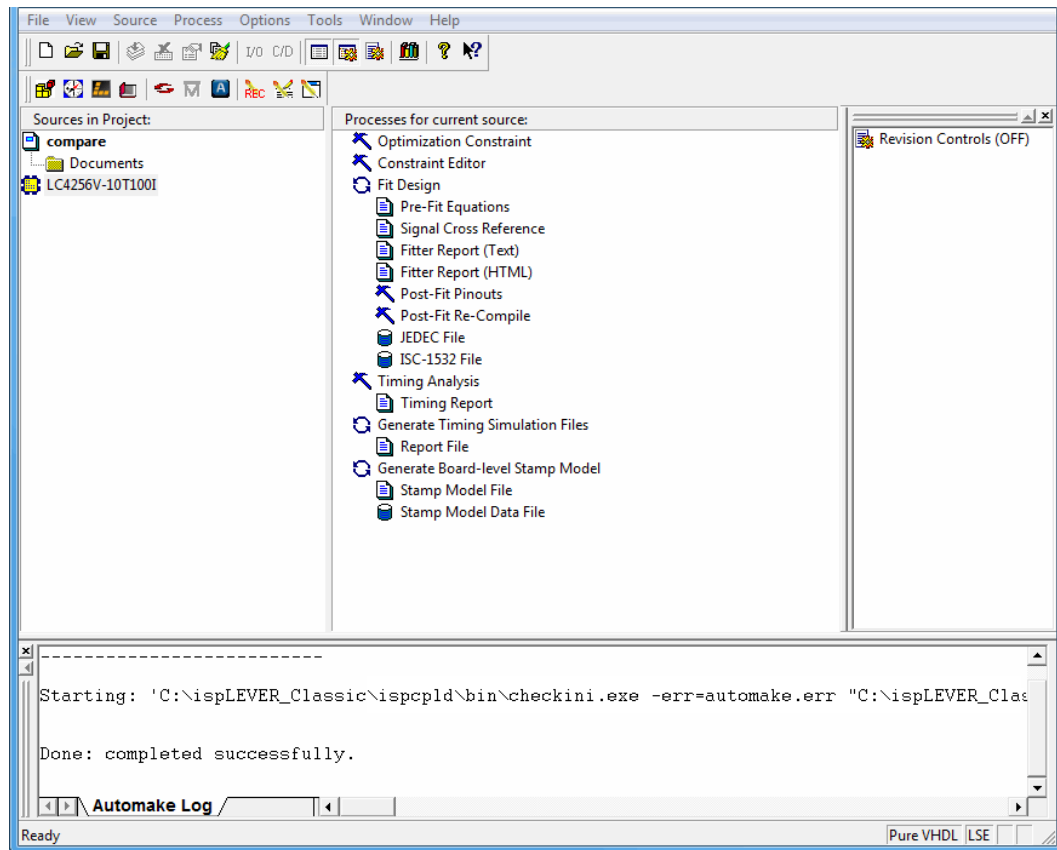
2. In the Project Wizard – Add Source dialog box, click **Next**, then **Finish**.

Your Project Navigator should look like Figure 5.

#### **Note**

Click on the part name to see the contents of the Processes for current source window.


Figure 5: Project Navigator Window Showing New Project

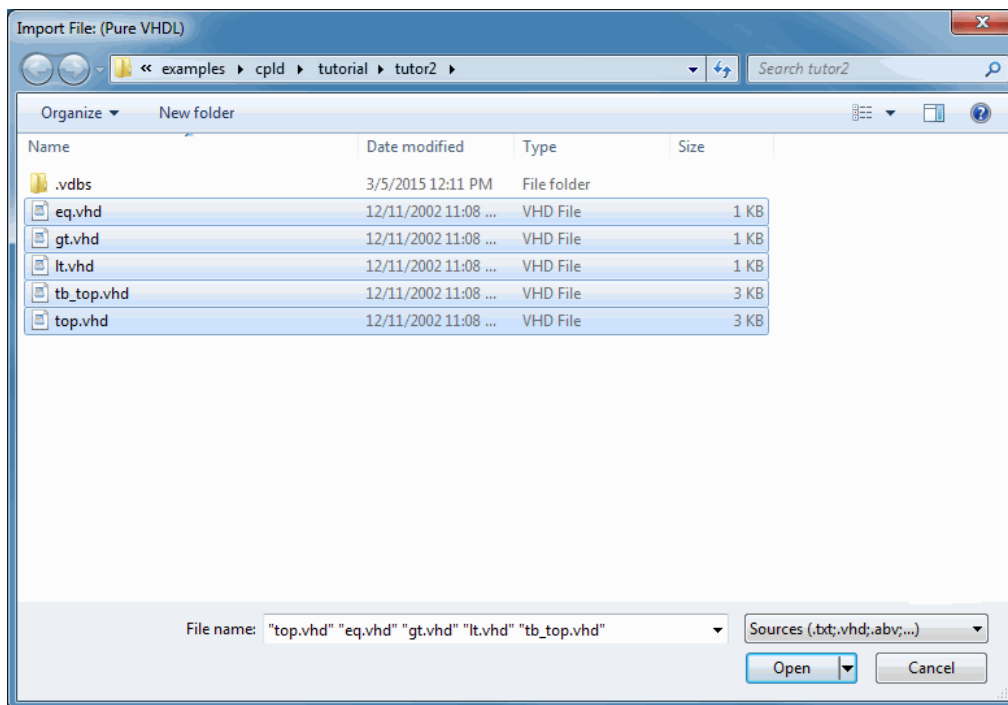


## Task 3: Import the VHDL Source Files

In this task, you will import VHDL source files to your ispLEVER Classic project.

### To add source files to the project:

1. Right-click on the device icon  and choose **Import**.
2. Navigate to the <path>\ispLEVER\_Classic<version>\examples\cpld\tutorial\_lse\tutor2 directory.
3. In the Import File (Pure VHDL), Select the following five VHDL files, as shown in Figure 6.
  - ▶ eq.vhd
  - ▶ gt.vhd
  - ▶ lt.vhd
  - ▶ tb\_top.vhd
  - ▶ top.vhd

**Figure 6: Import File: (Pure VHDL) Dialog Box**

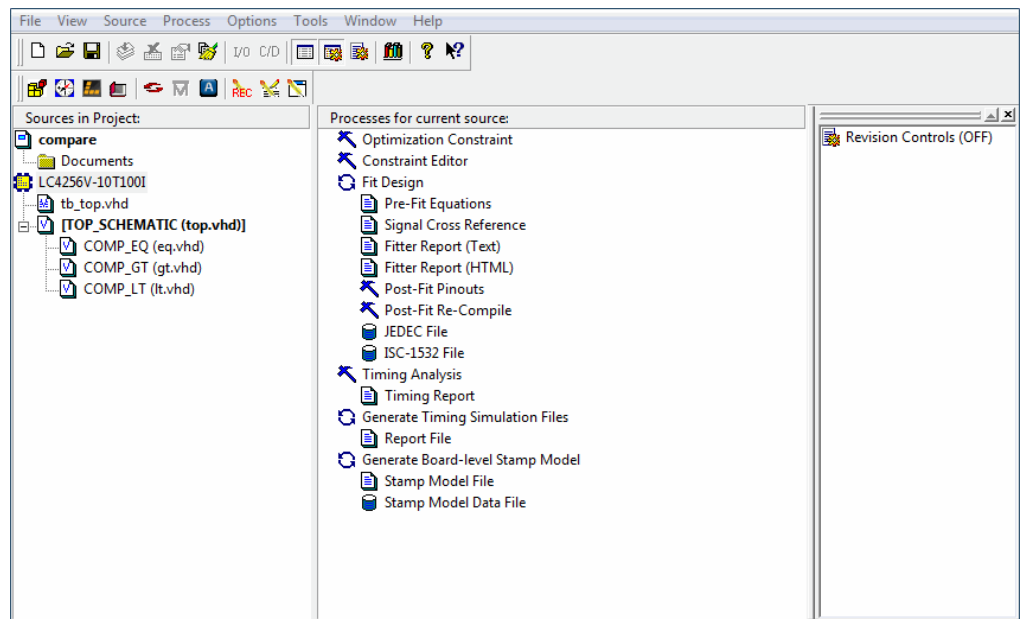
4. Click **Open**. The Import Source Type dialog will open. You will need to select the Type of Source for each file.
5. In the Import Source Type dialog box, for each file, select Type of Source as shown in the table below, and click **OK** for each file, until all files are imported.

**Table 1:**

VHDL File	Type of Source
eq.vhd	VHDL Module
gt.vhd	VHDL Module
lt.vhd	VHDL Module
tb_top.vhd	VHDL Test Bench
top.vhd	VHDL Module

6. The ispLEVER Classic Project Navigator should look like the illustration in Figure 7.

Figure 7: ispLEVER Classic Project Navigator



## Task 4: Fit the Design and View the Report

The ispLEVER Classic software has a single user interface with all options preset to deliver the highest possible push-button performance for most devices. When you double-click a process, all the processes prior to that process run automatically. Therefore, all you have to do is double-click the final process. However, here you will run one process at a time and view the results as you go.

At the end of a successful fitter run, the ispLEVER Classic software generates a JEDEC file, as well as a fitter report so that you can see how the ispLEVER Classic software has utilized and routed the part.

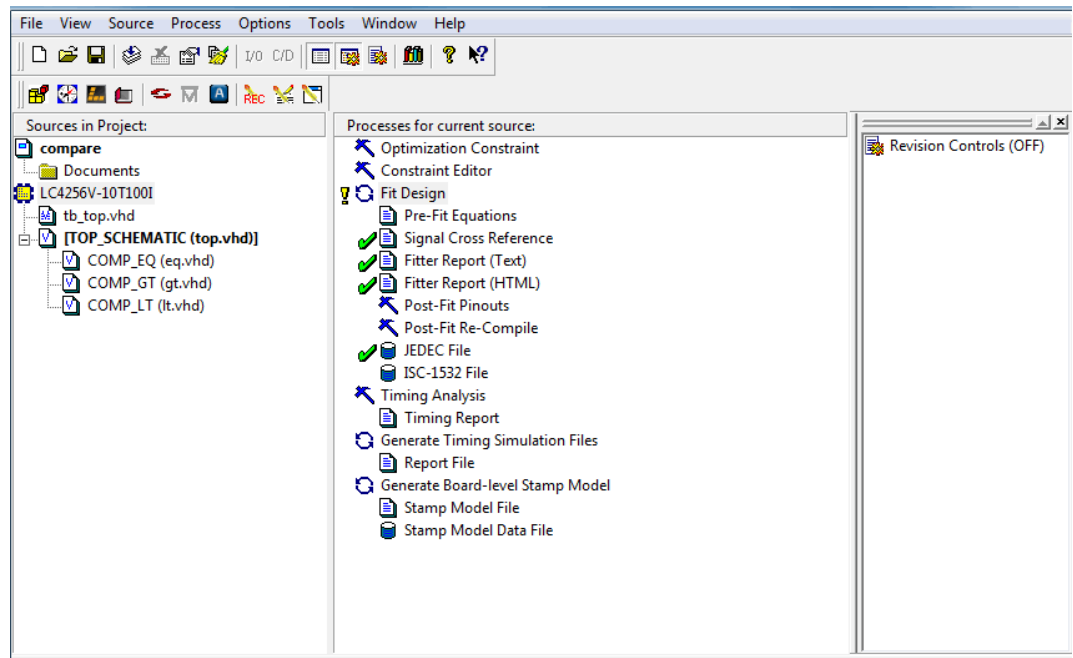
### To run the Fitter and view the report:

1. With the target device selected in the Sources in project window, double-click **Fit Design** in the Processes for current source window to run the Fitter.

The ispLEVER Classic software successfully fits the design in the specified device and generates a JEDEC file. The results are shown in Figure 8.

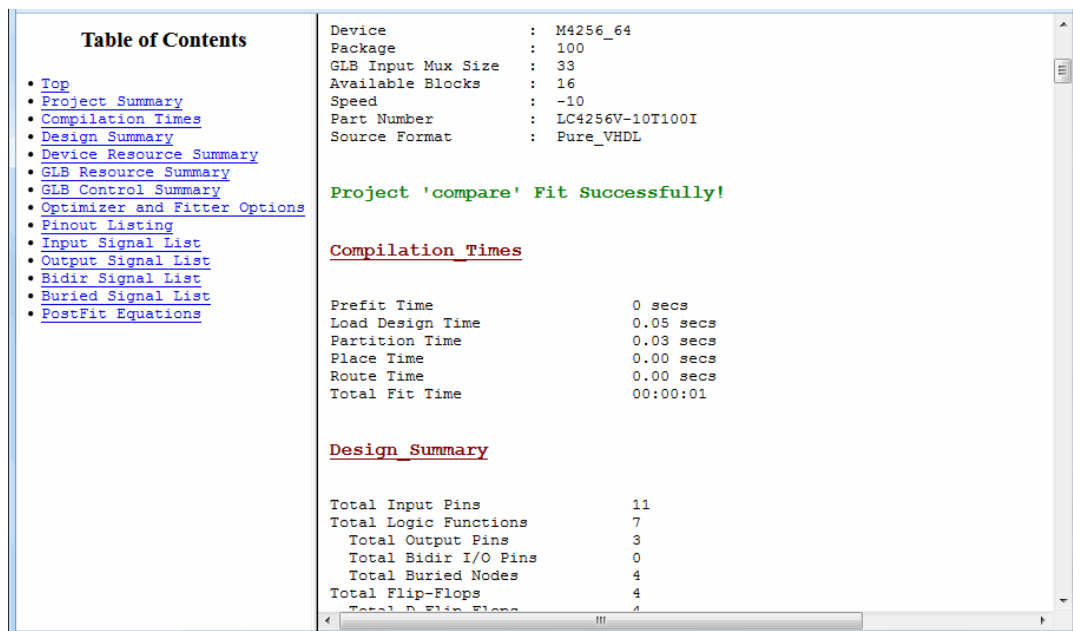
**Optional:** If you like, you can right-click on the **JEDEC File** process and select **View** to look at the contents of the JEDEC file. Close the file when you are through.

Figure 8: Project Navigator Showing Fitted Design



2. Double-click on the **Fitter Report (HTML)** process to open the report in your browser, as shown in Figure 9. View the contents and then close the report.

Figure 9: HTML Fitter Report



## Task 5: Perform Static Timing Analysis

Static timing analysis is the process of verifying circuit timing by totaling the propagation delays along paths between clocked or combinational elements in a circuit. The analysis can determine and report timing data such as the critical path, setup and hold-time requirements, and the maximum frequency.

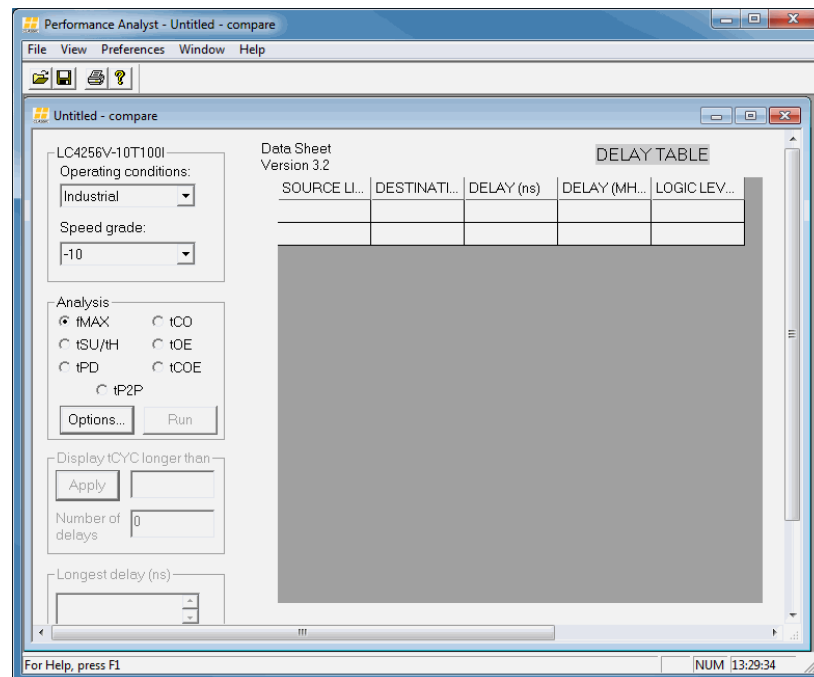
The Performance Analyst traces each logical path in the design and calculates the path delays using the device's timing model and worst-case AC specifications supplied in the device data sheet.

The timing analysis results are displayed in a graphical spreadsheet with source signals displayed on the vertical axis and destination signals displayed on the horizontal axis. The worst-case delay value is displayed in a spreadsheet cell if there is at least one delay path between the source and destination. To more easily identify performance bottlenecks, you can double-click a cell to view the path delay details.

### To perform timing analysis:

1. In the Project Navigator Sources in project window, select the target device.
2. In the Processes for current source window, double-click the **Timing Analysis** process to run the timing analysis and open the Performance Analyst, shown in Figure 10.

**Figure 10: Performance Analyst Window**



The Performance Analyst performs seven distinct analysis types: fMAX, tSU/tH, tPD, tCO, tOE, tCOE and tP2P. The first type, fMAX, is an internal



register-to-register delay analysis. fMAX measures the maximum clock operating frequency, limited by worst-case register-to-register delay. The tP2P type is the path between any two user-specified pins. The remaining five types are external pin-to-pin delay analysis. Timing threshold filters, source and destination filters, and path filters can be used to independently fine-tune each analysis.

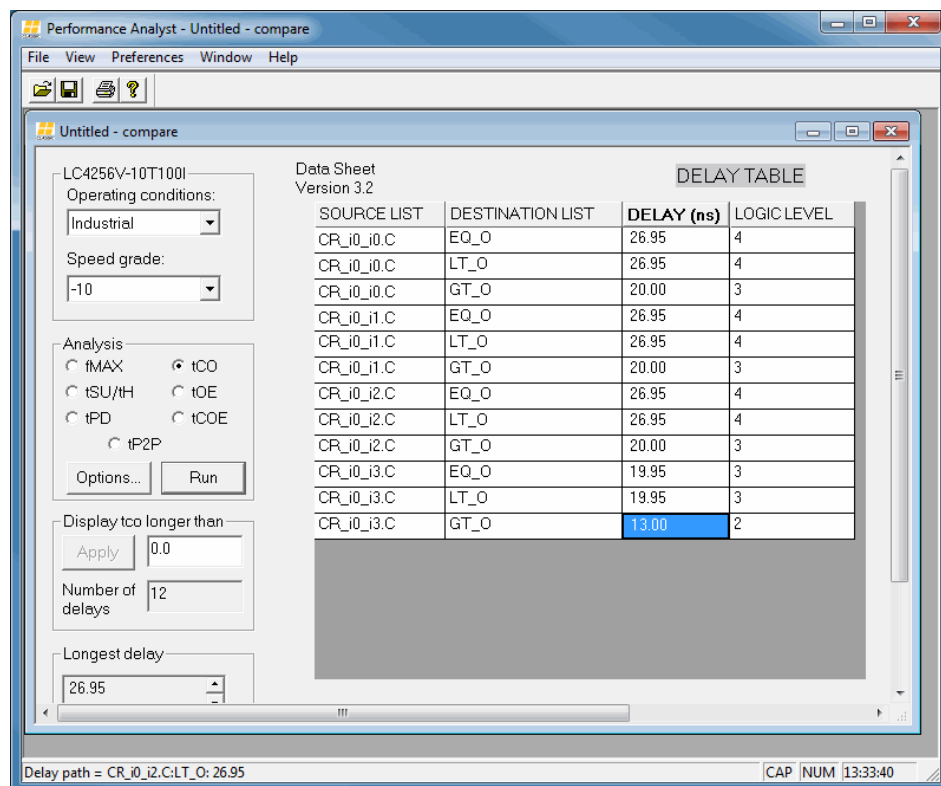
- In the Analysis field, select **tCO** and then click **Run**.

The tCO path trace analysis reports clock-to-out delay starting from the primary input, going through the clock of flip-flops or gate of latches, and ending at the primary output. In this case it is 13.00 ns. Figure 11 shows the results of the analysis.

### Note

Your timing numbers may differ slightly.

**Figure 11: Performance Analyst Window Showing Clock-to-Out Delay**



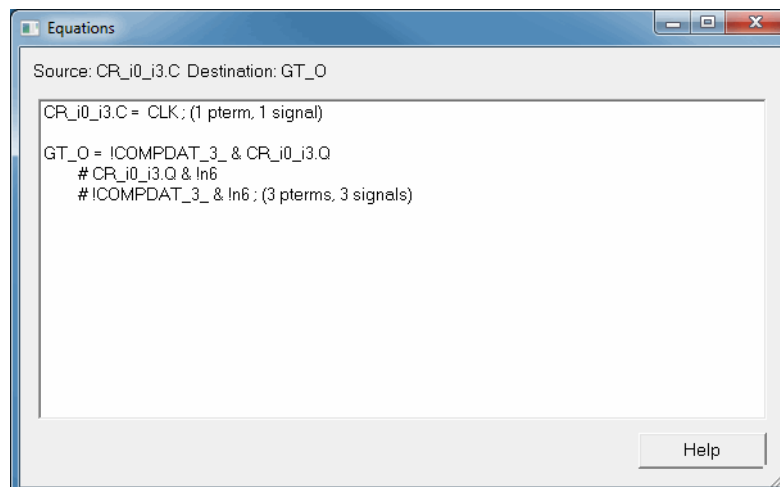
- Click the highlighted cell (**13.00**) in the spreadsheet window to open the Expanded Path dialog box, shown in Figure 12.

This dialog box enables you to analyze individual timing components used to calculate the timing path. It shows a source pin (From) and a destination pin (To). It also shows the delay type, the delay of that path (value ns), and the cumulative delay of all the signals.

**Figure 12: Expanded Path Dialog Box**

delay path	From	Loc	To	Loc	Delay Type	Value (ns)	Total
	CLK	p89	CR_i0_i3.C	A1	tGCLK_IN+IO(LV)	3.28	3.28
	CR_i0_i3.C	A1	CR_i0_i3.Q	A1	tCOi	1.17	4.45
	CR_i0_i3.Q	A1	GT_0	F2	tFBK+ROUTE+IBI	5.26	9.71
	GT_0	F2	GT_0	F2	tPDi	1.74	11.45
	GT_0	F2	GT_0	p19	tORP+IBUF+IOO	1.55	13.00

5. Click **Equations** to open the Equations dialog box, shown in Figure 13, which shows the functional relationship between the selected source and destination.

**Figure 13: Equations Dialog Box**

6. Close the Performance Analyst without saving.
7. Close ispLEVER Classic without saving.

## Summary

You have completed the HDL Synthesis Design with LSE: CPLD Flow tutorial. In this tutorial you have learned how to do the following:

- ▶ Create a new VHDL project in the ispLEVER Classic system and target a device.
- ▶ Fit the design, generate a JEDEC file, and view the Fitter report.
- ▶ Perform static timing analysis using the Performance Analyst and view the results.

## Glossary

Following are the terms and concepts that you should understand to use this tutorial effectively.

**EDIF.** EDIF (Electronic Design Interchange Format) is a format used to exchange design data between different electronic computer-aided design systems. It is designed to be written and read by computer programs that are constituent parts of EDA systems or tools. Its syntax has been designed for easy machine parsing and is similar to LISP. The ispLEVER Classic software supports EDIF Version 2.0.0.

**HDL.** An HDL is a hardware description language, which describes the structure and function of integrated circuits.

**static timing analysis.** Static timing analysis is the process of verifying circuit timing by totaling the propagation delays along paths between clocked or combinational elements in a circuit. The analysis can determine and report timing data such as the critical path, setup and hold-time requirements, and the maximum frequency.

**synthesis.** Synthesis is the process of translating a high-level design (RTL) description consisting of state machines, truth tables, Boolean equations, or all three into a process-specific gate-level logic implementation.

**Verilog.** Verilog is a language for describing the structure and function of integrated circuits.

**VHDL.** VHDL (or VHSIC (Very High-Speed Integrated Circuits) Hardware Description Language) is a language for describing the structure and function of integrated circuits.

## Recommended Reference Materials

You can find additional information on the subjects covered by this tutorial from the following recommended sources:

- ▶ *Lattice Synthesis Engine for ispLEVER Classic User Guide*
- ▶ Lattice Semiconductor ispLEVER Classic online Help
- ▶ Data sheets, technical notes, and other information on CPLDs on the Lattice Web site at [www.latticesemi.com](http://www.latticesemi.com).

