Lattice Radiant Software Design Flow Overview for Intel Quartus Users

User Guide

FPGA-UG-12151-1.1

April 2022

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults and associated risk the responsibility entirely of the Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
CPLD	Complex Programmable Logic Device
DDR	Double Data Rate
EDA	Electronic Design Automation
FPGA	Field Programmable Gate Array
HDL	Hardware Description Language
I/O	Input/Output
IP	Intellectual Property
LSE	Lattice Synthesis Engine
PAR	Place and Route
PLC	Programmable Logic Controller
PLL	Phase Locked Loop
RTL	Register Transfer Level
STA	Static Timing Analysis
TCL	Tool Command Language
VCD	Value Change Dump

1. Introduction

The Lattice Semiconductor Field Programmable Gate Array (FPGA)/Complex Programmable Logic Device (CPLD) design flow is similar in conception and implementation to the Intel®[1] FPGA design flow. At its core, a hardware description language (HDL) code of your register transfer level (RTL) can be imported into one of Lattice's design software and then configured to one of Lattice's FPGA.

This document guides Intel FPGA designers familiar with the Intel® Quartus®[1] Prime software, specifically version 20.1, in migrating existing designs to the Lattice Radiant™ software, specifically version 3.0. It also highlights some of the differences and similarities between the design flows of Quartus and Lattice Radiant software.

This user guide starts with an overview and comparison of the design software and a mapping of the tools and file extensions between the two. The next section compares the design flows of the design software. The succeeding sections provides a step-by-step walkthrough about the following: project creation and management, design entry, compilation flow, simulation, and programming/configuration.

For more details on the Lattice Radiant software design flow, refer to the Lattice Radiant software video tutorials listed in Table 1.1.

Table 1.1. Lattice Radiant Software Video Tutorial Links

Introduction to Lattice Radiant Software	Lattice Radiant Software introductory video featuring key easy design experience features.
Lattice Radiant Software Tool Flow - Part 1	Lattice Radiant Software Tool Flow training series with step-by-step procedure on full design flow and key features.
Lattice Radiant Software Tool Flow - Part 2	
Lattice Radiant Software Tool Flow - Part 3	
Setting up a Floating License Tutorial	Instructional video on how to setup a floating license on a server and how to setup environment variables on a client machine to access the server license.

[1] *Intel, the Intel logo, and Quartus are trademarks of Intel Corporation or its subsidiaries.*
Intel Quartus software is used to show the differences and similarities between the design flows of Quartus and Lattice Radiant software.

2. Design Software Overview

A design software is required to develop and implement FPGA/CPLD designs. The Lattice Radiant software features the following characteristics to reduce the design's time-to-market:

- Full-featured – Able to offer all necessary tools for design development
- High-performance – Able to perform powerful optimizations and analyses
- Intuitive user interface – Able to provide the best user experience with a graphical user interface that is both modular and wizard driven

The Lattice Radiant software is a solution for low-end to mid-range FPGA designs. It features a leading-edge software design environment for cost-effective, low-power Lattice FPGA architectures. The Lattice Radiant software integrated tool environment provides a modern, comprehensive user interface for controlling the Lattice Semiconductor FPGA implementation process.

The Lattice Radiant software uses an expanded project-based design flow and integrated tool views so that design alternatives and what-if scenarios can be created and analyzed. The Implementations and Strategies concepts provide a convenient way to try alternate design structures and manage multiple tool settings. System-level information—including process flow, hierarchy, and file lists—is available, along with integrated HDL code checking and consolidated reporting features. A fast Timing Analysis loop and Programmer provide capabilities in the integrated framework. The cross-probing feature and the shared memory architecture ensure fast performance and better memory utilization. The Lattice Radiant software is highly customizable and provides Tcl scripting capabilities from either its built-in console or from an external shell.

Most of the capabilities available in the Quartus software are available as well in the Lattice Radiant software. However, the terminology of the individual tools, features, and file formats may differ from one software to the other. Table 2.1 lists some of the tools, features, and file formats in Quartus and its corresponding name in the Lattice Radiant software.

Table 2.1. Mapping of Quartus and Lattice Radiant Software Tools, Features, and File Extensions

Description	Quartus Tools	Lattice Radiant Software Tools
Optimization or implementation settings	Compiler settings	Design strategies
Design entry	HDL file IP catalog Schematic entry	HDL file IP catalog
Design constraints	Prime Assignment Editor Timing Analyzer Text Editor	Device/Physical Constraint Editor Timing Constraint Editor
Synthesis tools	Precision Synthesis Synplify® Synplify Pro®	LSE Synplify Pro
Tools comparison	RTL Viewer Technology Map Viewer	Netlist Analyzer Technology Viewer
Place and Route tools	Quartus Fitter (Plan, Early Place, Place, Route, Retime and Finalize)	Place and Route
Supported simulators	Aldec Active-HDL™ Aldec Riviera-PRO™ Cadence® Incisive® Synopsys® VCS® QuestaSim Mentor Graphics® ModelSim	Cadence Xcelium Synopsys VCS Questasim Mentor Graphics ModelSim
Simulation levels	RTL Simulation Gate-Level Simulation	RTL Simulation Post-Synthesis Simulation Post-Route Gate-Level Simulation Post-Route Gate-Level+Timing Simulation
Power analysis	Power Estimator (EPE) and Power Analyzer	Power Calculator

Description	Quartus Tools	Lattice Radiant Software Tools
On-chip debug tool	Signal Tap II Logic Analyzer	Reveal™
Hardware programming tool	Programmer	Programmer
Project file extension	.qpf + .qsf	.rdf
IP configuration file extension	.qip	.ipx
Soft processor development and generator	Platform Designer	Lattice Propel™ Builder
Soft processor	Nios II	RISC-V

3. FPGA Design Flow Overview

When creating designs for Field Programmable Gate Arrays (FPGAs), Lattice and Intel software tools have similarities in terms of concepts, approach, and functionality. The Lattice Radiant software framework technology uses the typical FPGA design flow (see Figure 3.1) that adheres to a sequence of steps, which initially requires setting up the design environment and ends with the generation of programming files that will be used to program the hardware. In cases where Intel designers who are familiar with Quartus Prime would like to convert their existing Quartus designs to the Lattice Radiant software environment, they can simply import their HDL (hardware description language) files to Lattice Radiant Software and begin implementing the project-based methodology (see Figure 3.2).

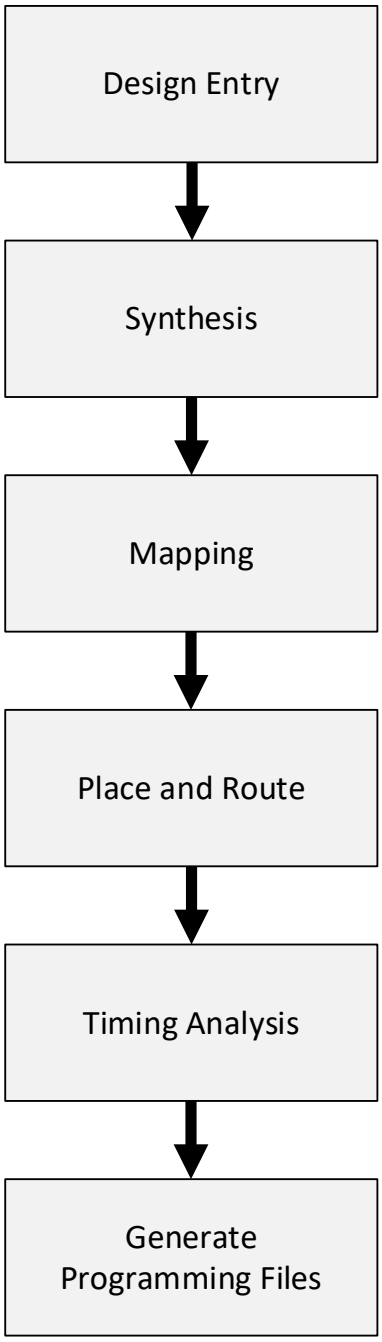


Figure 3.1. Typical FPGA Design Flow Used by Intel Quartus and Lattice Radiant Software

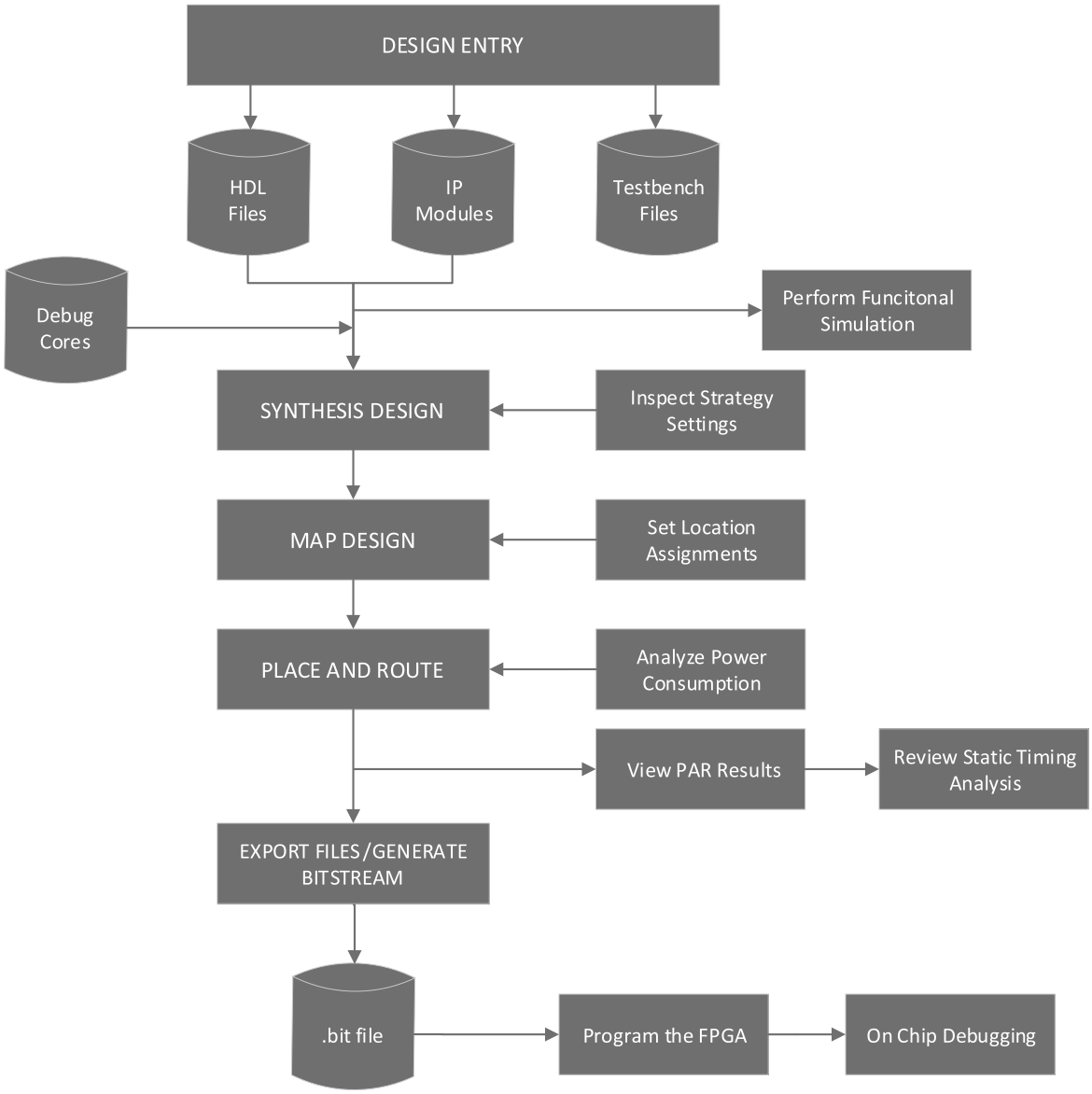


Figure 3.2. Lattice Radiant Software Detailed Design Flow

4. FPGA Design Flow Tools and Views

This section shows how Lattice Radiant and Intel Quartus software handle the FPGA design flow steps with various tools and views.

4.1. Project Creation and Management

In setting up the initial design environment, both Intel Quartus and Lattice Radiant Software provide a new project wizard functionality that guides you through the steps of project creation. The Lattice Radiant software New Project option is shown in Figure 4.1.

The Lattice Radiant software New Project wizard allows you to specify the project name and location, as shown in Figure 4.2, add existing source files as shown in Figure 4.3, select the target device for the design, as shown in Figure 4.4, and choose the synthesis tool, as shown in Figure 4.5.

Note: All project settings may be changed at later stages in the design process.

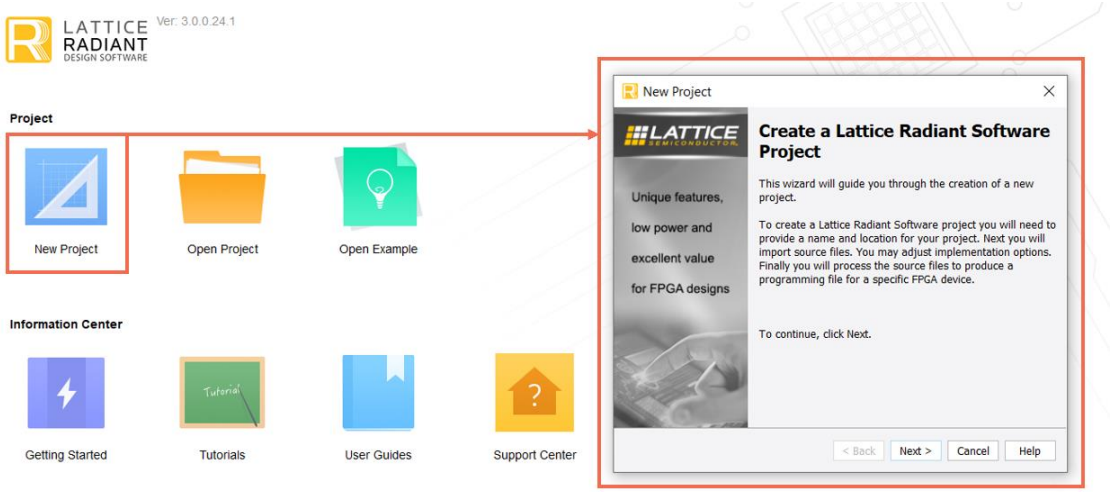


Figure 4.1. New Project Wizard

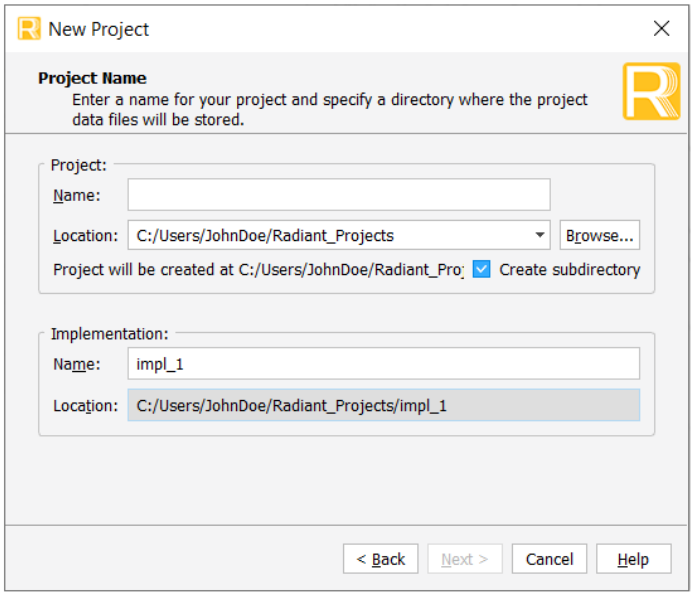


Figure 4.2. Project Settings

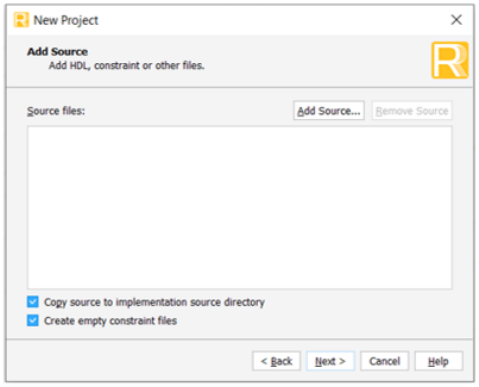
In the Project Name page, the following items are available.

Under the Project group:

- Name – Name of project
- Location – file path where the project is saved
- Create subdirectory – Enables the software to create a subfolder where the project is saved

The following items are available under the Implementation group:

- Name – Implementation name
- Location – File path where the implementation is saved



Radiant Add Source File Types

File Description	File Types
Input Files	*.vhd *.v *.sv *.h *.ipx *.sbx *.vm
Pre-Synthesis Constraint Files	*.sdc
Lattice LSE	*.ldc
Post-Synthesis Constraint Files	*.pdc
Debug Files	*.rvl *.rva
Script Files	*.spf
Analysis Files	*.pcf
Programming Files	*.xcf
Other Files	*.*

Figure 4.3. Adding Existing Source

In the Add Source page, the following items are available.

- Add Source – Adds HDL files, constraint files or testbench in the current project.
- Copy source to implementation source directory – Copies the source files added in the current project directory
- Create empty constraint files – Automatically generates a blank Pre Synthesis (.sdc) and Post Synthesis (.pdc) constraint files

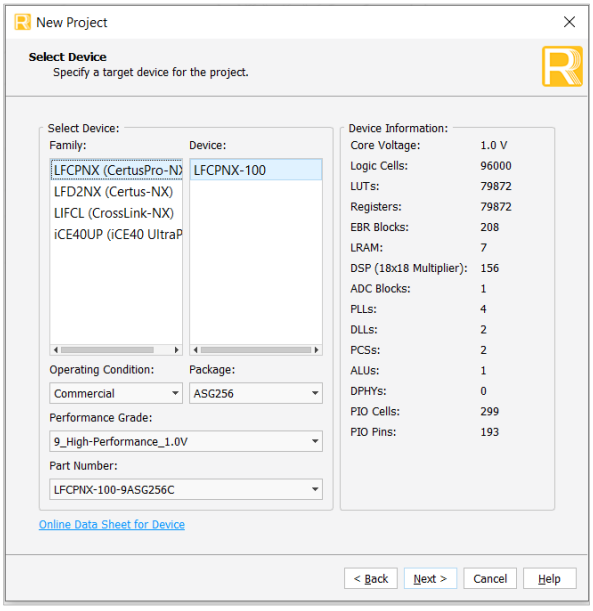


Figure 4.4. Target Device Selection

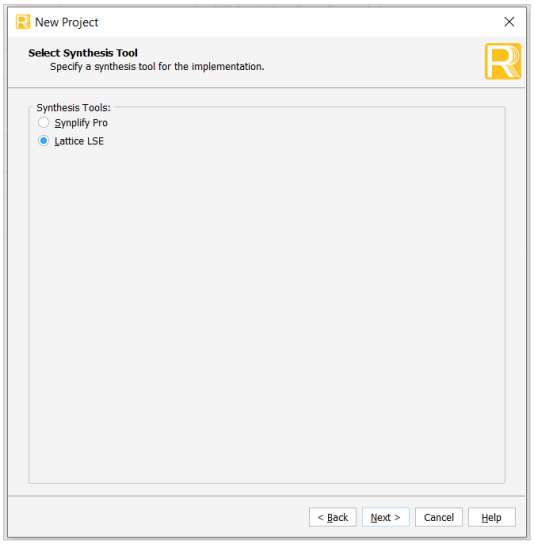


Figure 4.5. Synthesis Tool Selection.

Once the project is created, choosing the logical and consistent way of organizing each source file allows you to locate and modify them as needed. Both Intel Quartus and Lattice Radiant Software support this practice by combining design sources into different categories and listing them in a Files/Files List View. In the Lattice Radiant software, the source files are classified and listed under the following category folders:

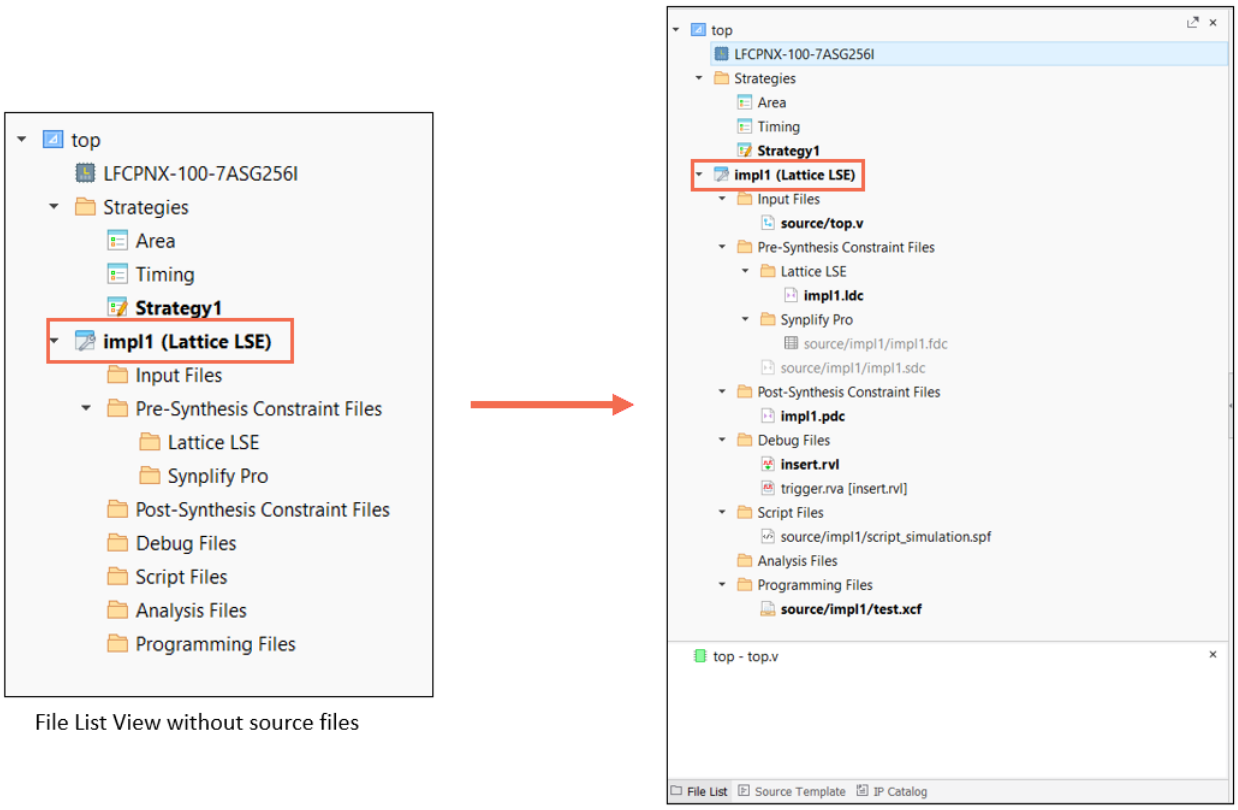
- Strategies
- Input Files
- Synthesis Constraint Files
- Debug Files
- Script Files
- Analysis Files
- Programming Files

Refer to Table 4.1 for the file types and extensions.

Table 4.1. Files Listed in Lattice Radiant Software File List View

File Type	File Extension
Project Title	None
Target Device	None
Predefined Strategy (Area, I/O Assistant, Quick, and Timing)	.sty
Strategy1 (that can be customized)	.sty
Implementation	None
Verilog Files	.v, .veri, .ver, .vo, .h
Structural Verilog	.vm
SystemVerilog	.sv
IP Module Config Files	.ipx
Undefined or incorrect	Any source reference
Synthesis Constraint Files	.sdc, .fdc, .ldc
Reveal Project File	.rvl
Simulation Project File	.spf
Reveal Analyzer Files	.rva
Power Calculator Files	.pcf
Programmer Project File	.xcf

You can also see implementations listed in the File List View, as shown in Figure 4.6.



File List View without source files

File List View with source files

Figure 4.6. Lattice Radiant Software File List View

The Lattice Radiant software also offers different ways of involving a source file in your design. If you right-click on each source file, you are given different options depending on the file type.

These options are:

- Excluding the file from implementation
- Setting the design file as top-level
- Removing the file for either or both synthesis and/or simulation
- Checking for properties

For implementation and strategy, you can create a different version or a clone of each, edit the properties, select the synthesis tool, and select the top level unit.

4.2. Design Strategy

In Intel Quartus, you can modify the optimization settings by going to the Compiler Settings → Advanced Settings (Synthesis) or Advanced Settings (Fitter). Aside from modifying each setting, there are available optimization modes with pre-defined settings depending on the goal of the designer (i.e. power, performance, and area).

In the Lattice Radiant software, this is equivalent to the design strategy. The Lattice Radiant software allows you to choose or create your own custom strategy for your implementation. A strategy is a collection of implementation-related tool settings or recipes for how the design will be implemented.

The Lattice Radiant software provides two pre-defined strategies:

- Area – This strategy is used for area optimization. Its purpose is to minimize the total logic gates used while enabling the tight packing option available in Map.
- Timing– This strategy is used for timing optimization. Its purpose is to achieve timing closure.

It also enables you to create customized strategies, which can be edited, cloned, and removed. All strategies are available to all of the implementations, and any strategy can be set as the active one for an implementation.

You can view the strategies on the File List View. Double-click on the strategy to tool settings for each of the steps of the design flow.

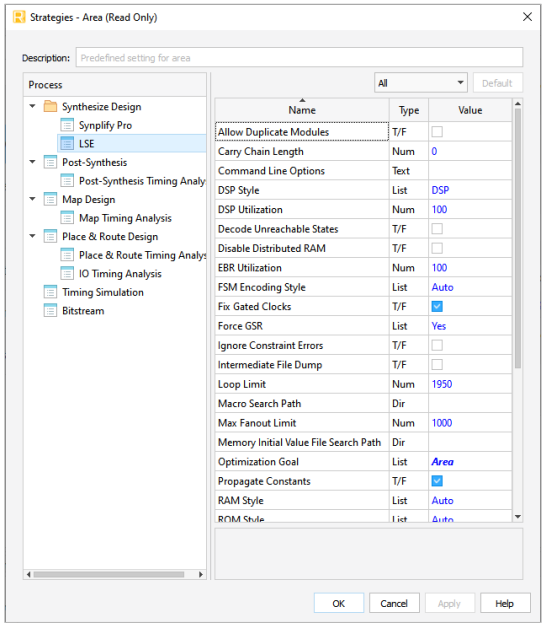


Figure 4.7. Area Pre-defined Strategy Settings

4.3. Design Entry

Lattice Radiant software supports HDL source files and IP catalog as design entry methods.

4.3.1. HDL Source Files

In the Quartus software, you can create a new HDL source file by clicking File > New and selecting the HDL file to add to the design. Similarly, in the Lattice Radiant software, clicking File > New opens the New File dialog box where you can choose which HDL file type to add to the design, as shown in Figure 4.8.

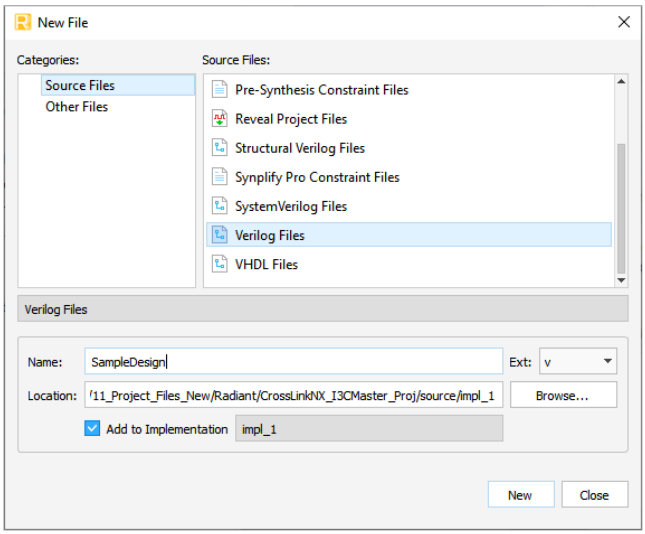


Figure 4.8. New File Dialog Box

Quartus has design templates and constructs that you can insert on your design by right-clicking on the HDL source file and then selecting Insert Template. In the Lattice Radiant software, you can access these templates by clicking the Source Template tab.

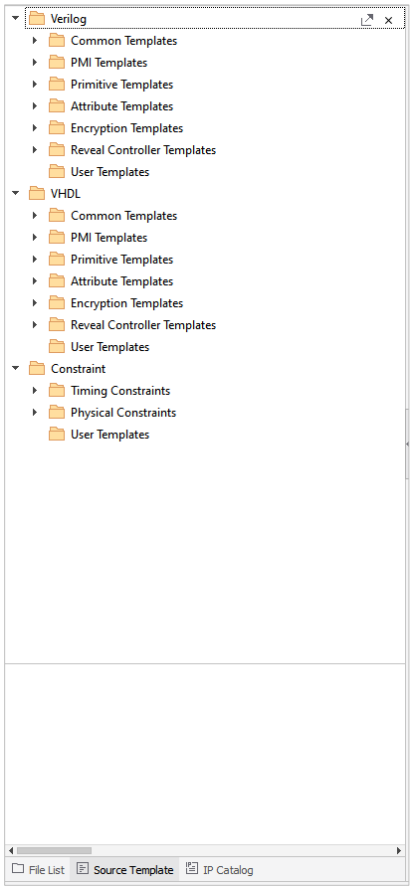


Figure 4.9. Source Template Tab

4.3.2. IP Catalog

In the Quartus software, you can open the IP catalog by clicking on the IP Catalog icon on the toolbar. The IP catalog pane can be detached into a window where you can select the different IPs. In the Lattice Radiant software, the IP catalog is accessed by clicking either the IP Catalog icon on the toolbar or the IP Catalog tab.

On the IP Catalog tab, you can select installed IPs on local folder or download and install additional IPs from the IP Server.

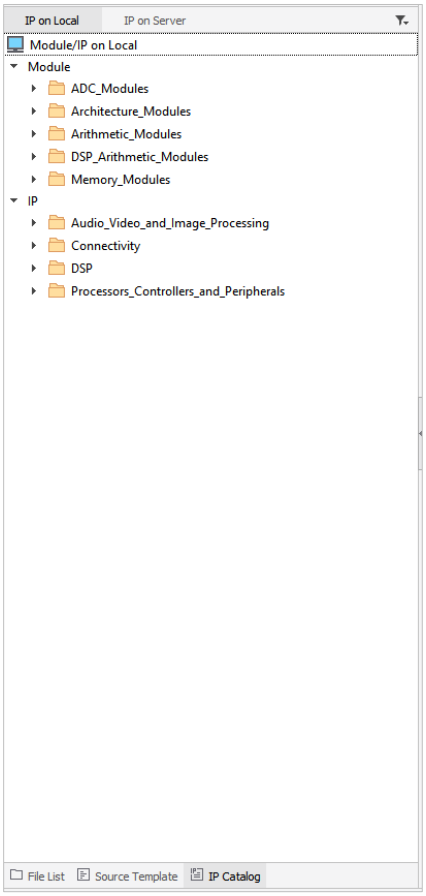


Figure 4.10. IP Catalog Tab

For more information, you may refer to the Lattice Radiant Help or the following documents:

- Lattice Radiant Software IP User Guide
- Lattice Radiant Software 3.1 User Guide sections:
  - IP Catalog
  - Packaging IP Using IP Packager
  - Running Radiant IP Packager and Viewing Documentation
  - Installing IP Created with IP Packager into IP Catalog

4.4. Design Constraints

Design constraints are used to specify the performance requirements desired for the FPGA design. Various tools helps the designers to meet those conditions. Constraints are instructions applied to the design elements that guide the design toward desired results and performance goals. They are critical to achieving timing closure or managing reusable intellectual property (IP). The most common constraints are those for timing and pin assignments, but constraints are also available for placement, routing, and many other functions.

Table 4.2 Design Constraint Tools Comparison

Intel Quartus	Lattice Radiant Software
Prime Assignment Editor, Timing Analyzer Text Editor	Device/Physical and Timing Constraint Editor

The Lattice Radiant software provides tools with user interface for assigning constraints: Device/Physical/Timing Constraint Editor. Constraints can include timing or physical constraints defined in Constraint Files (.ldc/.pdc/.sdc/.fdc) or HDL attributes synthesis tool and specify design goals. Synthesis, map, and place-and-route work to meet these goals. Post-synthesis constraints (.pdc) can also be specified. The flow combines these based on different entry points with in the Lattice Radiant software design flow. The timing analysis tool reports whether or not the goals are met.

The pre-synthesis constraint flow depends on the synthesis tool selection.

- .ldc (through LSE)
- .sdc (through Synplify Pro and LSE)
- .fdc (through Synplify Pro)

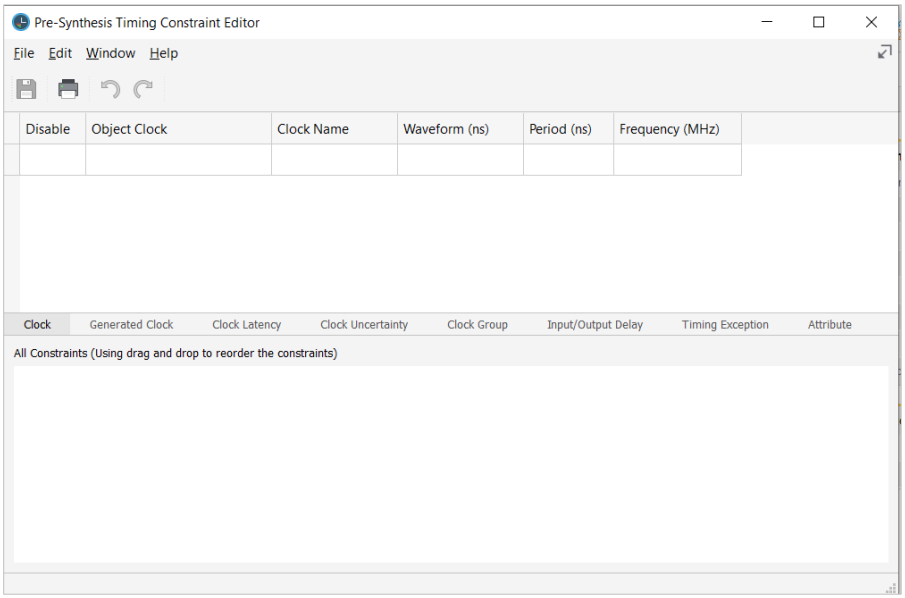


Figure 4.11. Pre-Synthesis Timing Constraint Editor

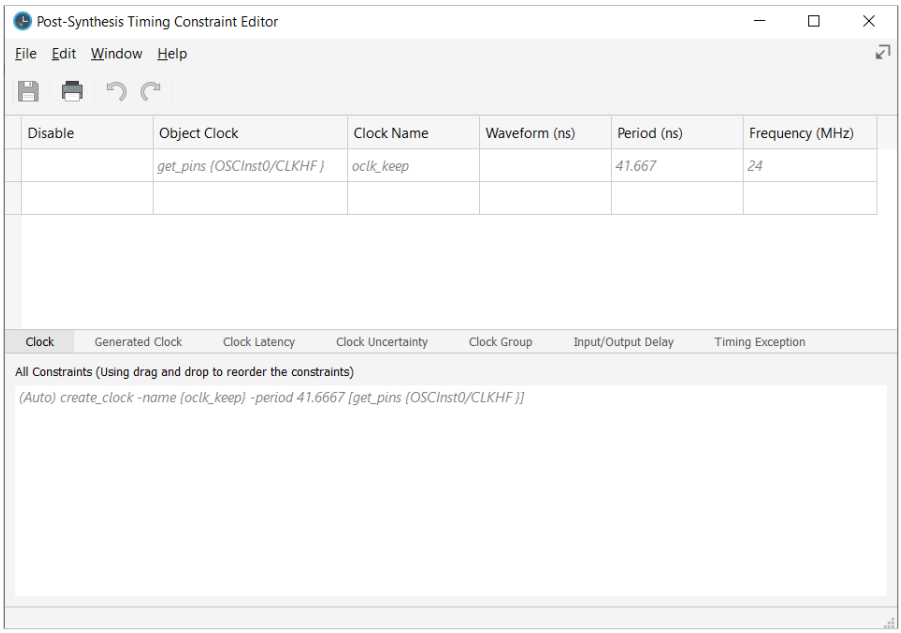


Figure 4.12. Post-Synthesis Timing Constraint Editor

The post-synthesis constraint flow is the same for all projects and should be completed before the MAP and PAR General Timing Constraint Flow, as shown in Figure 4.13.

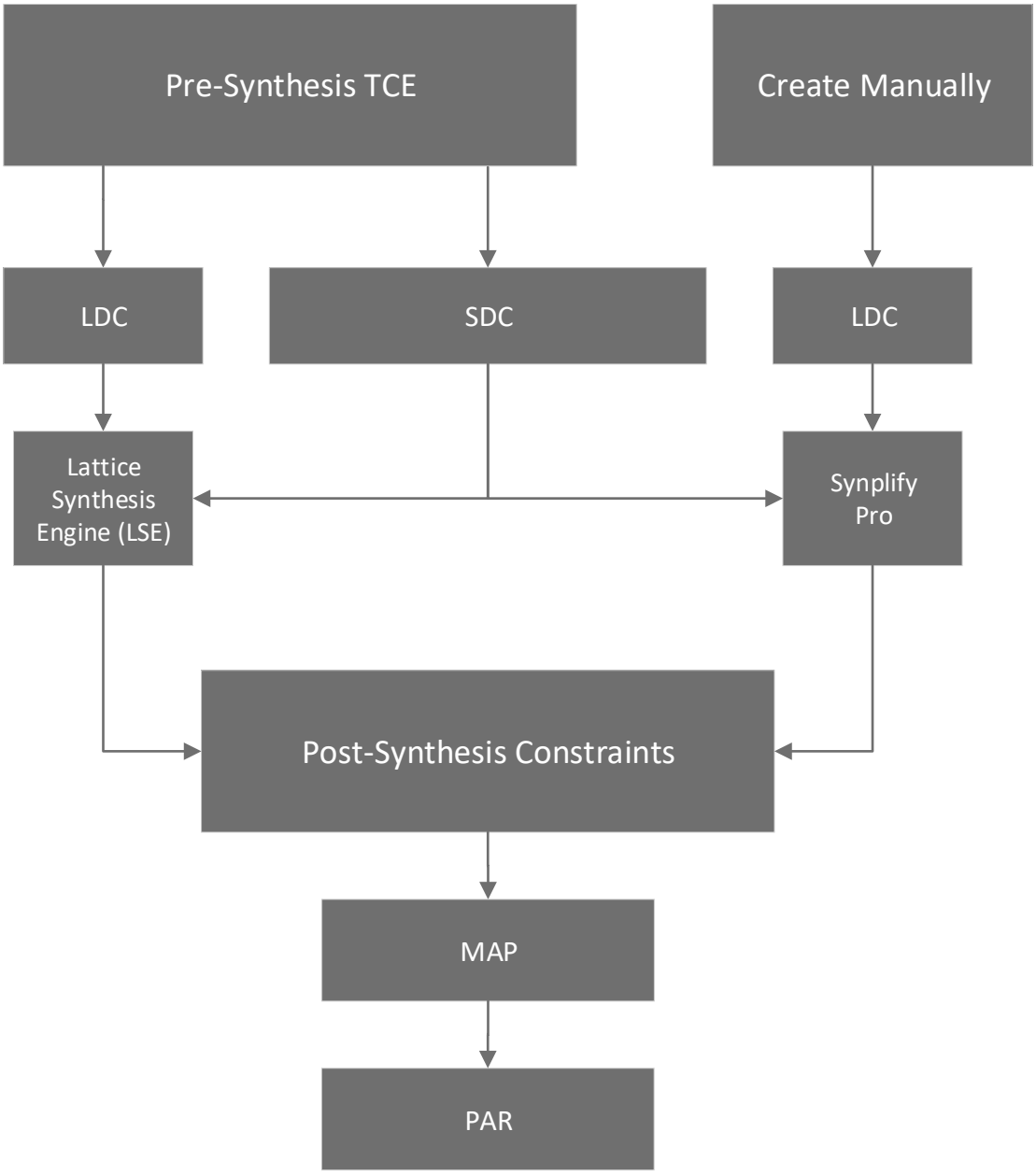


Figure 4.13. General Constraint Flow

The Quartus Assignment Editor provides a spreadsheet-like interface for assigning all instance-specific settings and constraints. The Lattice Radiant software Device Editor, on the other hand, shows the pin layout of the device and displays the assignments of signals to device pins. This view allows you to edit these assignments and reserve sites on the layout to exclude from placement and routing. It is also the entry mechanism for physical constraints. The Device Constraint Editor views enable you to develop constraints that shorten turn-around time and achieve a design that conforms to critical circuit performance requirements.

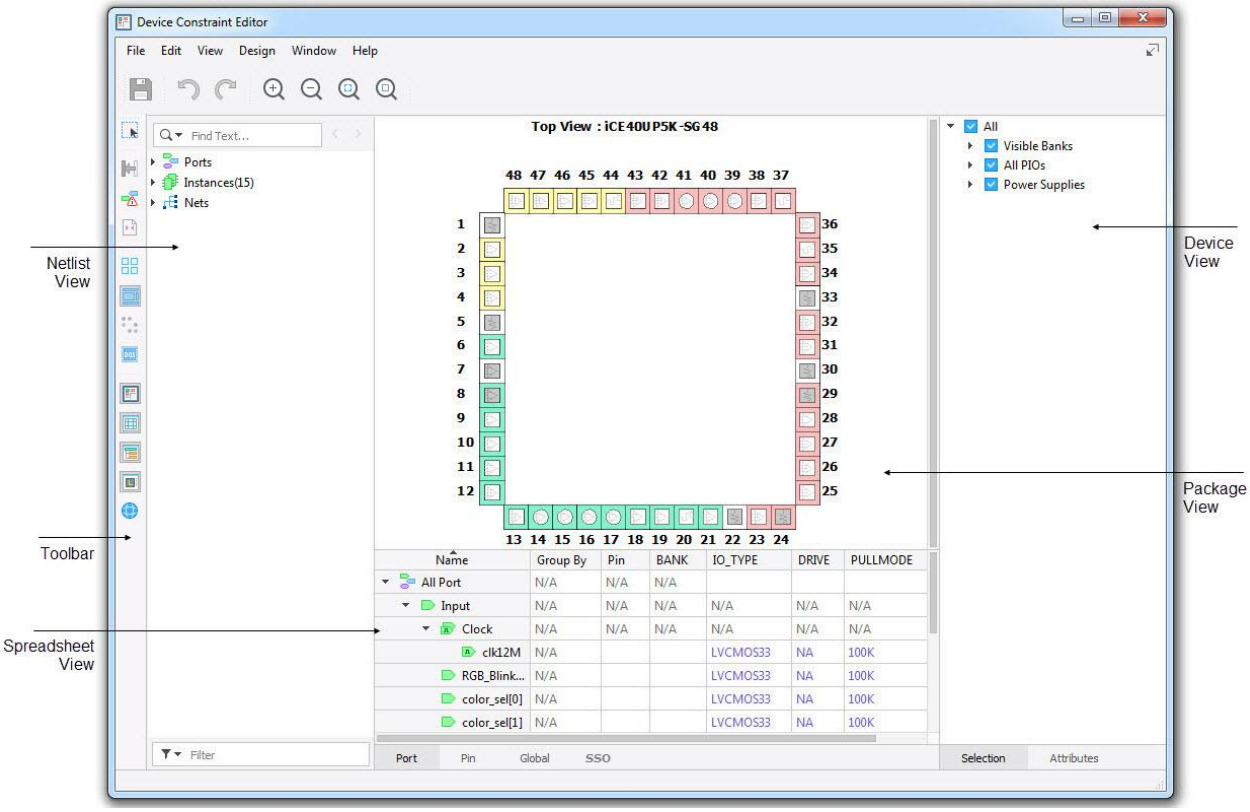


Figure 4.14. Device Constraints Editor

4.5. Synthesis

Similar to Intel Quartus, the Lattice Radiant software has a proprietary synthesis tool in Lattice Synthesis Engine (LSE) and supports a third party synthesis tool in Synplify Pro.

Table 4.3. Synthesis Tools Comparison

Intel Quartus	Lattice Radiant Software
Precision Synthesis, Synplify, and Synplify Pro	LSE and Synplify Pro

You can select synthesis tools in Quartus by clicking Analysis & Synthesis and selecting Edit Settings > Electronic Design Automation (EDA) Tool Settings. In the Lattice Radiant software, you can do this by right-clicking the current implementation and then choosing Select Synthesis Tool from the drop-down menu.

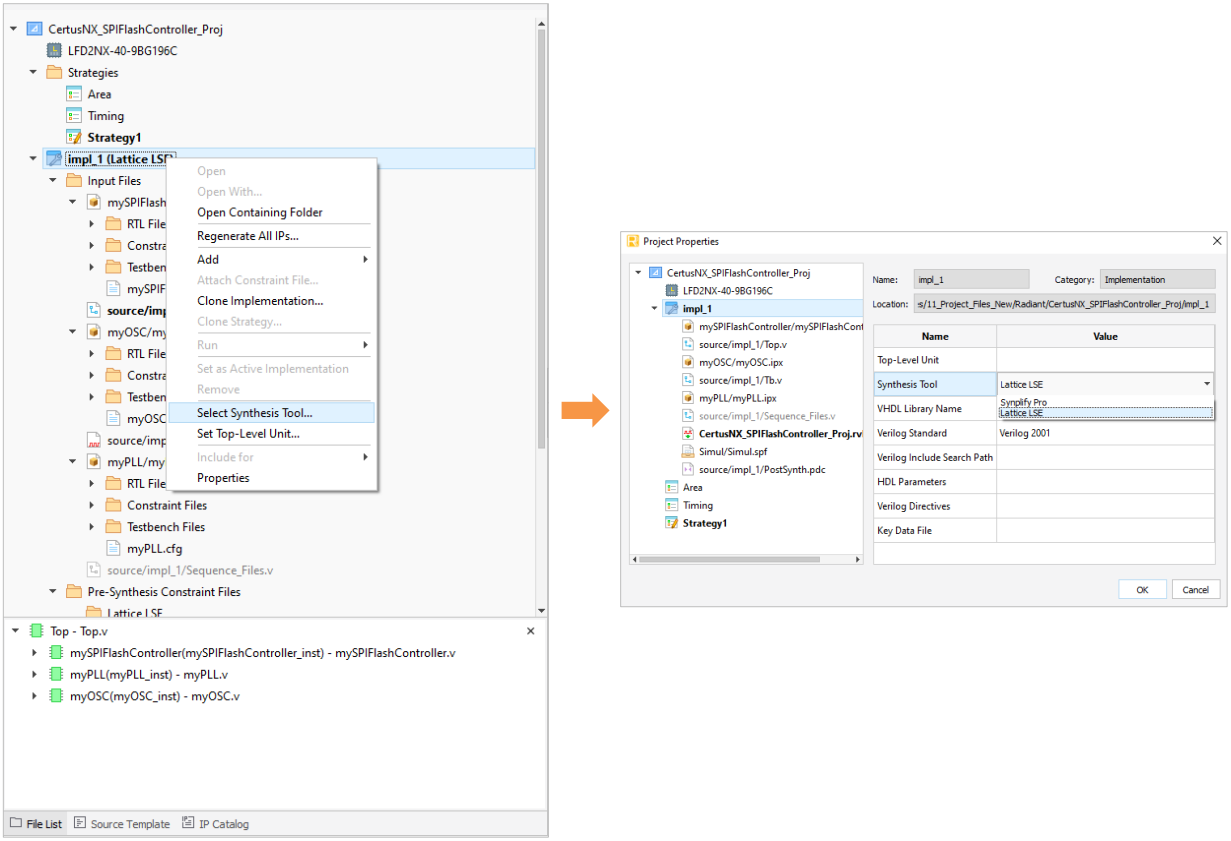


Figure 4.15. Selecting Synthesis Tools

In Quartus, synthesis is performed by double-clicking Analysis & Synthesis on the Tasks tab. In the Lattice Radiant software, synthesis is performed by clicking on the Synthesis Design play icon in the Process toolbar.

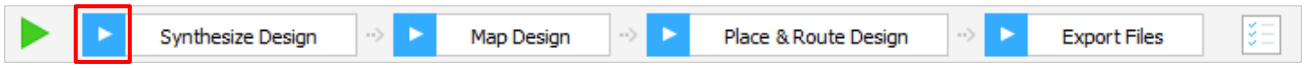


Figure 4.16. Running Synthesis on the Process Toolbar.

To modify the optimization and other settings related to synthesis, you can refer to the Design Strategy section.

4.6. Netlist Viewer

A generated netlist can be viewed through one or more schematic views and a browser that shows the lists of modules, instances, ports, and nets. In Intel Quartus, this can be performed using RTL Viewer and Technology Map Viewer. While in the Lattice Radiant software, this can be performed using Netlist Analyzer of LSE or Technology Viewer of Synplify Pro.

Table 4.4. Tools Comparison

Intel Quartus	Lattice Radiant Software
RTL Viewer and Technology Map Viewer	Netlist Analyzer and Technology Viewer

To open Netlist Analyzer, click the Netlist Analyzer icon on the toolbar.

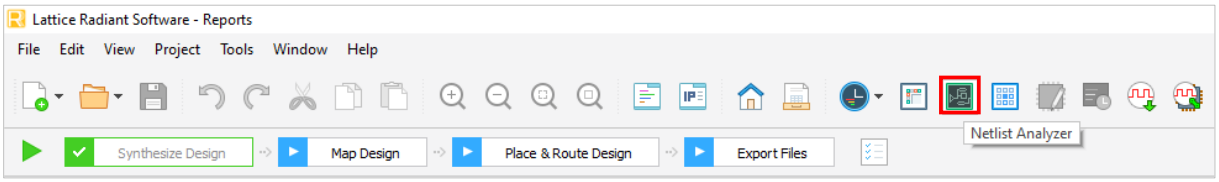


Figure 4.17. Opening Netlist Analyzer

To open Technology Viewer, open Synplify Pro from the Lattice Radiant Software and then click on the Technology View icon on the toolbar.

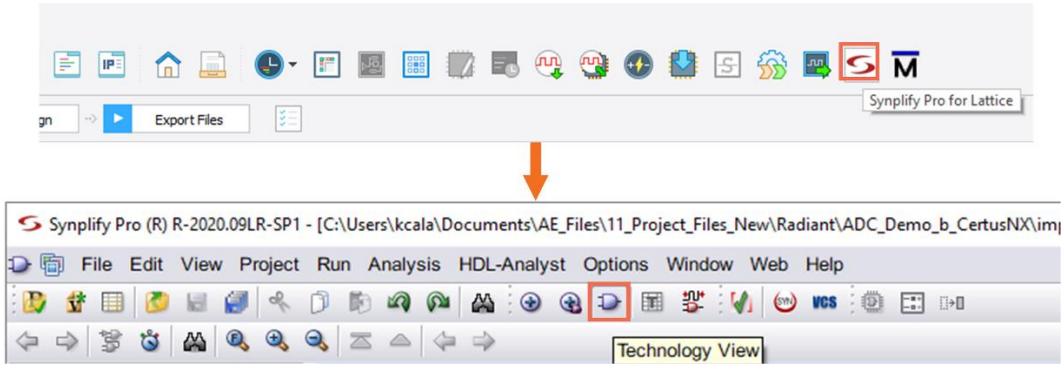


Figure 4.18. Opening Technology View.

For more information, refer to the Lattice Radiant Software Help or to the Lattice Radiant Software 3.1 User Guide Netlist Analyzer section.

4.7. Design Mapping

Design mapping converts the logical design into a network of physical components or configurable logic blocks. In Quartus, this process is combined with the Analysis & Synthesis process. In the Lattice Radiant software, this is a separate process in the design flow that can be optimized through the design strategy settings.

To map a design in Lattice Radiant, click the Map Design play icon.

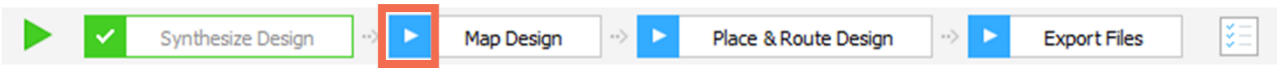


Figure 4.19. Running Map on the Process Toolbar

4.8. Place and Route

After a design is translated to physical design format during mapping, it is ready for placement and routing. Placement is the process of assigning the device-specific components produced by the mapping process to specific locations on the device floorplan. After placement is complete, the route phase establishes physical connections to join components in an electrical network. The place and route process takes a mapped physical design and places and routes the design. Placement and routing of a design can be cost-based or timing driven.

In Quartus, the Quartus II Fitter, which is also known as the PowerFit Fitter, performs place and route, also referred to as *fitting* in the software. In the Lattice Radiant software environment, on the other hand, the place and route process automatically assigns device-specific components to locations and connects them.

Table 4.5. Place and Route Tools Comparison

Intel Quartus Prime Pro Edition	Lattice Radiant Software
Quartus Fitter (Plan, Early Place, Place, Route, Retime and Finalize)	Place and Route

Placement and routing options can influence the performance and utilization of the design implementation and ease incremental design changes. Some options affect the way the results are reported. Experimenting with place and route settings in the Strategies dialog box can help improve your placement and routing results.

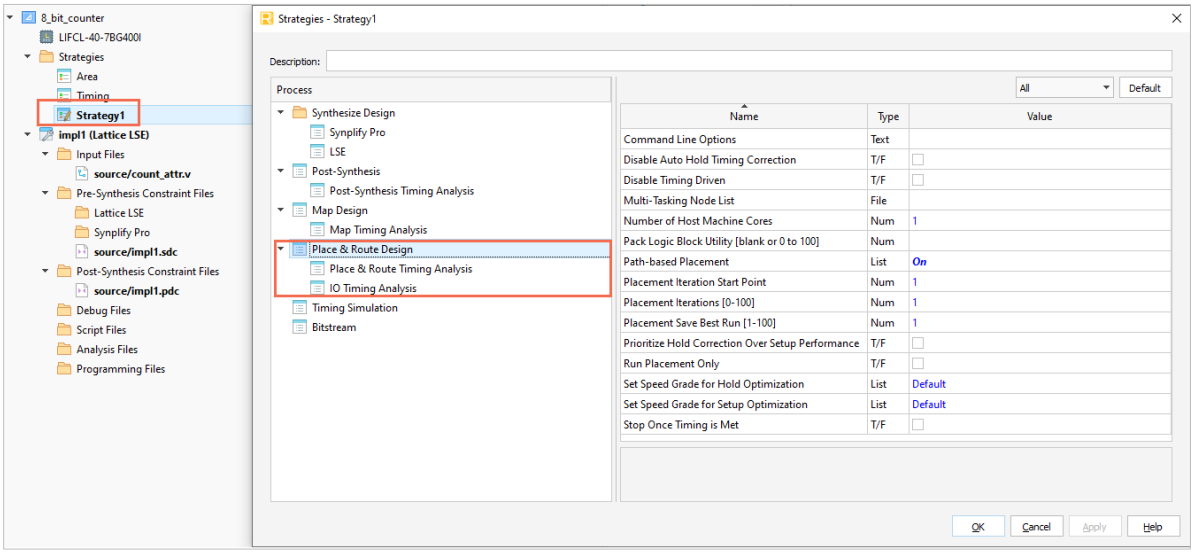


Figure 4.20. PAR Design Strategy Options

In most cases, your design requires timing-driven placement and routing, where the timing criteria you specify influences the implementation of the design. In the Lattice Radiant software, static timing analysis results show how constrained nets meet or do not meet your timing. In the Task Detail View, the Place & Route Timing Analysis process runs static timing analysis. This process reports any timing errors and generates a report.

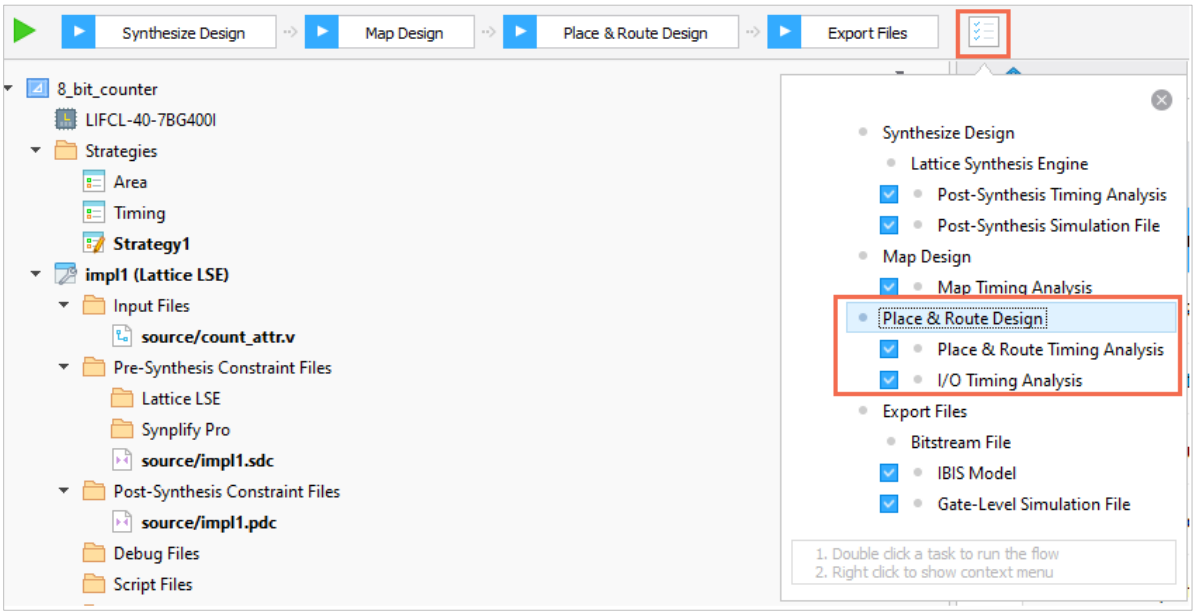


Figure 4.21. PAR Timing Analysis Option.

After place and route, the Quartus software offers the Chip Planner tool, where you can view post-compilation placement, connections, and routing paths. You can also make assignment changes, such as creating and deleting resource assignments. In the Lattice Radiant software environment, the Physical Designer tool is the combined user interface of both the Placement Mode and Routing Mode. This provides for one central location where you can perform all the floor-planning and view the physical layout of the design. The tool has three mode options:

- Placement Mode
- Input/Output (I/O) Mode
- Routing Mode.

Placement Mode provides a large-component layout of your design. All connections are displayed as fly-lines. Placement Mode allows you to create REGIONs and bounding boxes for GROUPS and specify the types of components and connections to be displayed. As you move your mouse pointer slowly over the floorplan layout, details are displayed in tool tips: the number of resources for each GROUP and REGION; the number of utilized slices for each programmable logic controller (PLC) component; and the name and location of each component, port, net, and site

I/O Mode is used for I/O planning and I/O assignment such as Double Data Rate (DDR) interface, DQS and clock assignments. As you move your mouse pointer slowly over the I/O Abstract layout, details are displayed in tool tips: the number of resources for each items such as ECLCK, DDR, Phase Locked Loop (PLL), and I/O Banks utilization are displayed.

Routing Mode provides a read-only detailed layout of your design that includes switch boxes and physical wire connections. Routed connections are displayed as Manhattan-style lines, and unrouted connections are displayed as fly-lines. As you move your mouse slowly over the layout, the name and location of each REGION, group, component, port, net, and site are displayed as tool tips.

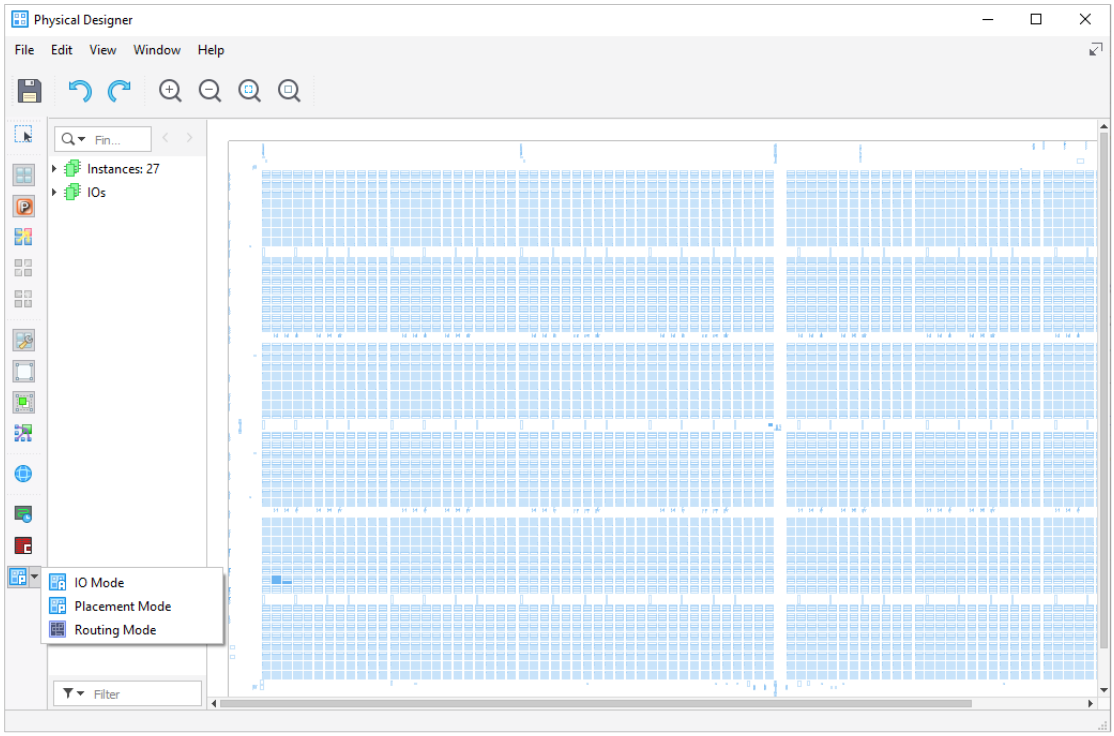


Figure 4.22. Physical Designer

4.9. Cross-Probing

Cross-probing is a feature that allows seamless bi-directional communication between the different tools within the software itself. You can select a design element from one tool and locate them in another tool. Both Intel Quartus and Lattice Radiant Software have cross-probing ability.

In the Lattice Radiant software, aside from the tools that can use cross-probing, you can also click a hyperlink icon in the reports to cross probe into that tool.

- Post-Synthesis & Map timing report links to Netlist Analyzer.
- In the PAR timing report, you can cross probe to Netlist Analyzer, Physical Designer Placement Mode, and Routing Mode.

4.10. Programming Files

After you have created and verified your design, you can use the final output data file to download or upload a bitstream to or from an FPGA device using the Lattice Radiant Programmer. Use the Process Toolbar to generate files for exporting. Programmer supports serial and microprocessor programming of Lattice devices in PC and Linux environments.

Table 4.6. File Formats

File Format	Description
Data File	A data file can be a hex, or bitstream file. Each of these files is based upon an IEEE programming standard:
Bitstream	Data files used for configuring volatile memory (SRAM) of our FPGAs.
Hex	Hexadecimal PROM data files used for Programming into external non-volatile memory, such as parallel or Serial Peripheral Interface (SPI) Flash devices.

For more information, you may refer to the Lattice Radiant Software Help or to the following sections in Lattice Radiant Software 3.1 User Guide:

- Programming Files
- Programmer

4.11. Reports

In Quartus, you can view the reports of each step in the Task View by clicking View Report. In the Lattice Radiant software, you can click the Reports tab, which is open by default on the workspace. You can also click View > Reports.

Each step on the design flow has its own set of reports from resource usage to timing analysis.

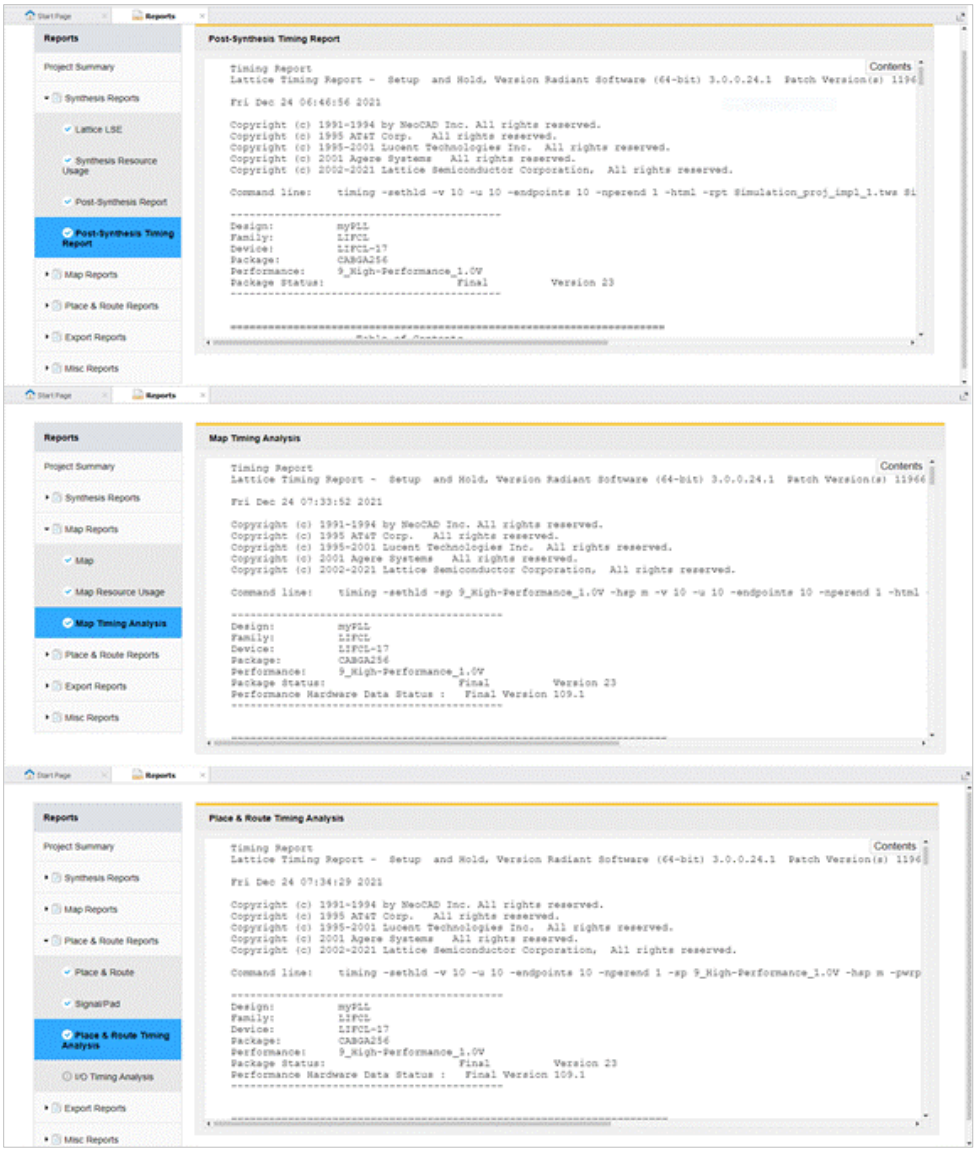


Figure 4.23. Reports Tab

For more information, you may refer to the Lattice Radiant Software Help or to the following sections in Lattice Radiant Software 3.1 User Guide:

- Reports and Messages Views
- Reports

5. Design Verification Tools

5.1. Simulation

Both Intel Quartus and Lattice Radiant Software support a number of third-party simulators, as shown in Table 5.1.

Table 5.1. Supported Simulators.

Intel Quartus	Lattice Radiant Software
Aldec Active-HDL, Aldec Rivera-PRO, Cadence Incisive, Synopsys VCS, Questasim, and Mentor Graphics ModelSim	Cadence Xcelium , Synopsys VCS, Questasim, and Mentor Graphics ModelSim

In Quartus, you can run a simulation through the Tasks tab by Selecting the RTL or Gate-Level Simulation. In the Lattice Radiant software, you can click on the Simulation Wizard icon to run simulation using ModelSim, which is a directly linked third-party simulator.

Figure 5.1. Opening Simulation Wizard from the Toolbar.

The Lattice Radiant software also comes with a standalone version of ModelSim that can be used for project simulation.

5.1.1. Simulation Levels

Depending on the output files chosen on the process toolbar, you can perform different simulation levels. In Quartus, there are only two levels of simulation: RTL and gate-level simulations. In the Lattice Radiant software, there are four simulation levels: RTL, post-synthesis, post-route gate-level, and post-route gate-level plus timing simulations. These provide you more options when simulating your designs prior to hardware verification.

You can choose between the simulation levels on the Process Stage of the Simulation Wizard window.

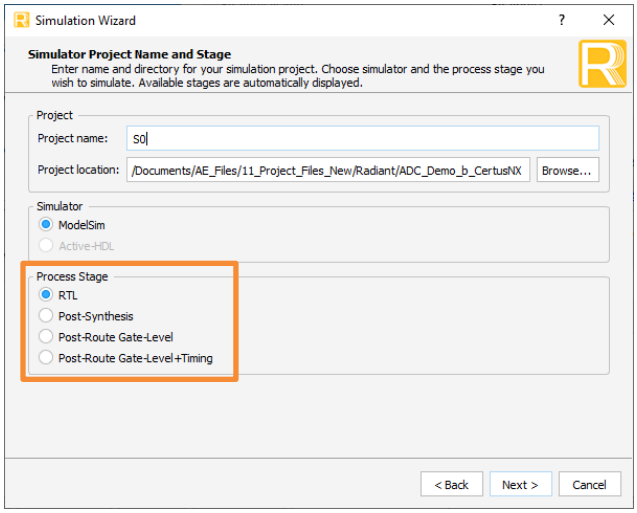


Figure 5.2. Simulation Levels in Simulation Wizard

Each simulation requires specific output files that can be generated on the design flow before simulation can be performed:

- RTL Simulation can be run with just an HDL file and a testbench file.
- Post-Synthesis Simulation requires a post-synthesis netlist file in addition to an HDL file and a testbench file. The post-synthesis netlist file is generated during the Synthesize Design process when you select Post-Synthesis Simulation File.

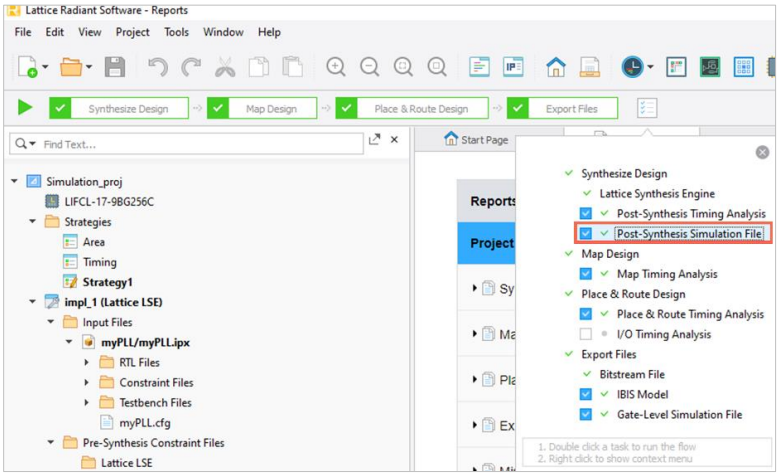


Figure 5.3. Generating a Post-Synthesis Simulation File

- Post-Route Gate-Level requires a gate-level netlist file in addition to an HDL file and a testbench file. The gate-level netlist file is generated during the Export Files process when Gate-Level Simulation File is selected.
- Post-Route Gate-Level+Timing requires a gate-level netlist file and a standard delay format file (.sdf) in addition to an HDL file and a testbench file. The gate-level netlist file is generated during the Export Files process when Gate-Level Simulation File is selected.

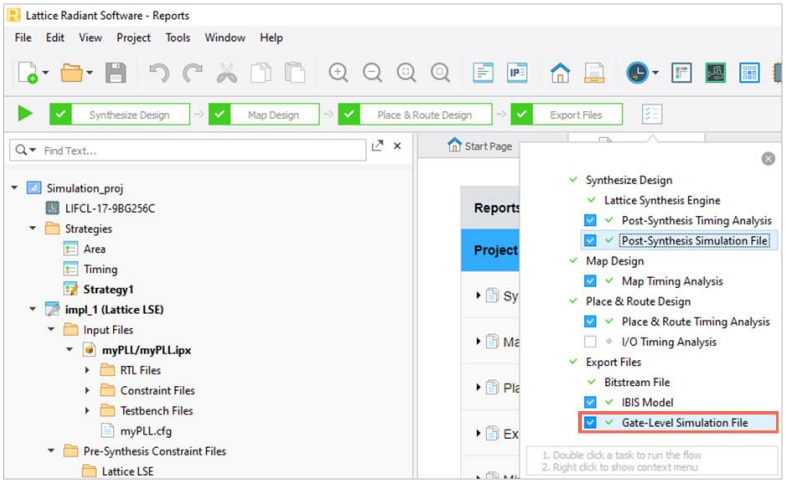


Figure 5.4. Generating Gate-Level Simulation Files

For more information, you may refer to the Lattice Radiant Software Help or to the following sections in Lattice Radiant Software 3.1 User Guide:

- Mentor ModelSim
- Simulation Wizard
- Simulation Flow
- Simulation Wizard Flow

5.2. Static Timing Analysis (STA)

Similar to the timing analyzer in Intel Quartus, Lattice Radiant timing analyzer analyses and reports timing performance of all logic in your design while checking all possible paths with timing violations.

Timing Analyzer analyzes timing constraints that are present in the .ldc and .pdc files. These timing constraints are defined in the Timing Constraint Editor or in a text editor before the design is mapped. A Timing Analysis report file, which shows the results of timing constraints, is generated each time you run the synthesis engine, Map Timing Analysis process, or the PAR Timing Analysis process. PAR Timing Analysis results can then be viewed in the Timing Analyzer window, which can be opened by clicking on the Timing Analyzer icon on the toolbar.

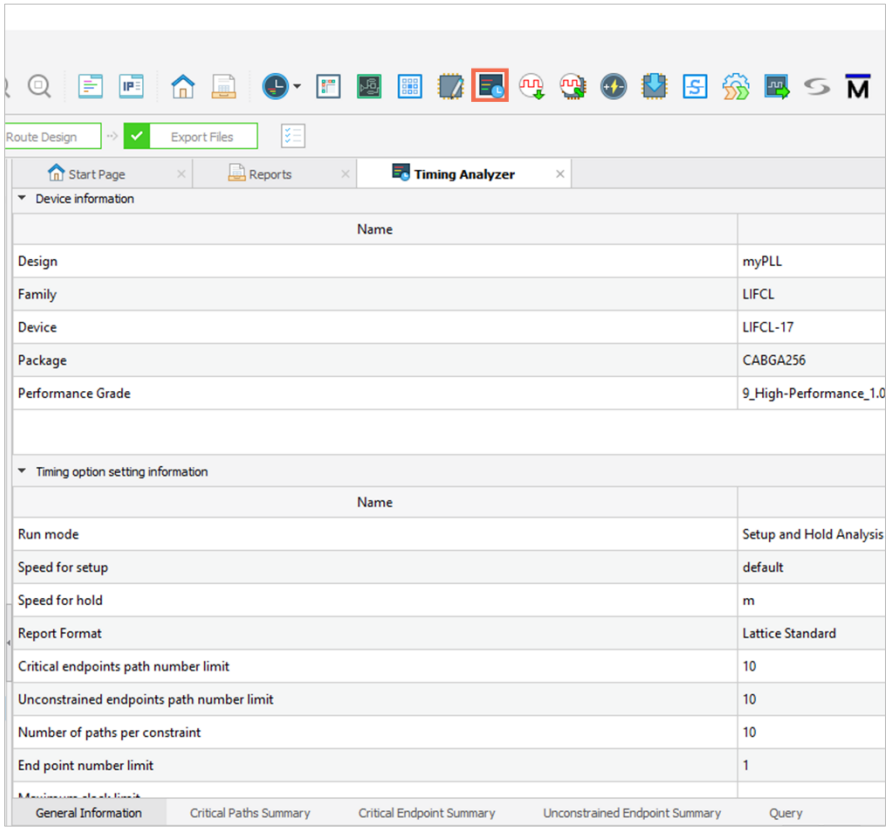


Figure 5.5. Opening the Timing Analyzer Tab

For more information regarding Timing Analyzer, you may refer to the *Analyzing Static Timing* topic in Lattice Radiant Software Help or to the following sections in Lattice Radiant Software 3.1 User Guide:

- Timing Analyzer
- Using Stand Alone Timing Analyzer

5.3. On-Chip Hardware Debugging using Reveal

The final stage of developing the design is the actual verification process either on a test board or in your system. The on-chip debugging tools allow live hardware aspect checking in your design, which helps to quickly do a verification without the use of any external equipment.

While Intel Quartus software offers multiple portfolios of on-chip debugging tools, in the Lattice Radiant software environment, the Reveal Inserter and Reveal Analyzer/Controller tool continuously monitor signals within the FPGA for specific conditions, which can range from simple to complex. These tools observe what is happening inside the FPGA and even change register values while your system is running.

- Reveal Inserter, which you use to create a debug module and add it to your design.
- Reveal Analyzer/Controller, which you use to control the debug module and to view test results. Reveal Analyzer/Controller is used after programming the FPGA with your combined design and debug module.

For more information on using Reveal, you may refer to the following documents:

- Lattice Radiant Software Reveal User Guide
- Reveal Troubleshooting Guide for Lattice Radiant Software

You may also refer to the following sections in the Lattice Radiant Software 3.1 User Guide:

- Reveal Inserter
- Reveal Analyzer
- Reveal Controller

5.4. Power Analysis

Similar to Intel Quartus' Power Estimator (EPE) and Power Analyzer tool, Lattice Radiant Software offers the Power Calculator, which estimates the power dissipation for a given design.

Table 5.2. Power Analysis Tools Comparison.

Intel Quartus	Lattice Radiant Software
Power Estimator (EPE) and Power Analyzer tool	Power Calculator

Power Calculator uses parameters such as voltage, temperature, process variations, air flow, heat sink, resource utilization, activity, and frequency to calculate the device power consumption. It reports both static and dynamic estimated power consumption. The tool also allows you to import frequency and activity factors from the post-PAR simulation *value change dump* file (.vcd file). After the design information is added, Power Calculator provides accurate power consumption analysis for the design.

Power Calculator provides two modes for reporting power consumption:

- Estimation mode is used before completing the design.
- Calculation mode is based on the physical netlist file (.udb) after placement and routing.

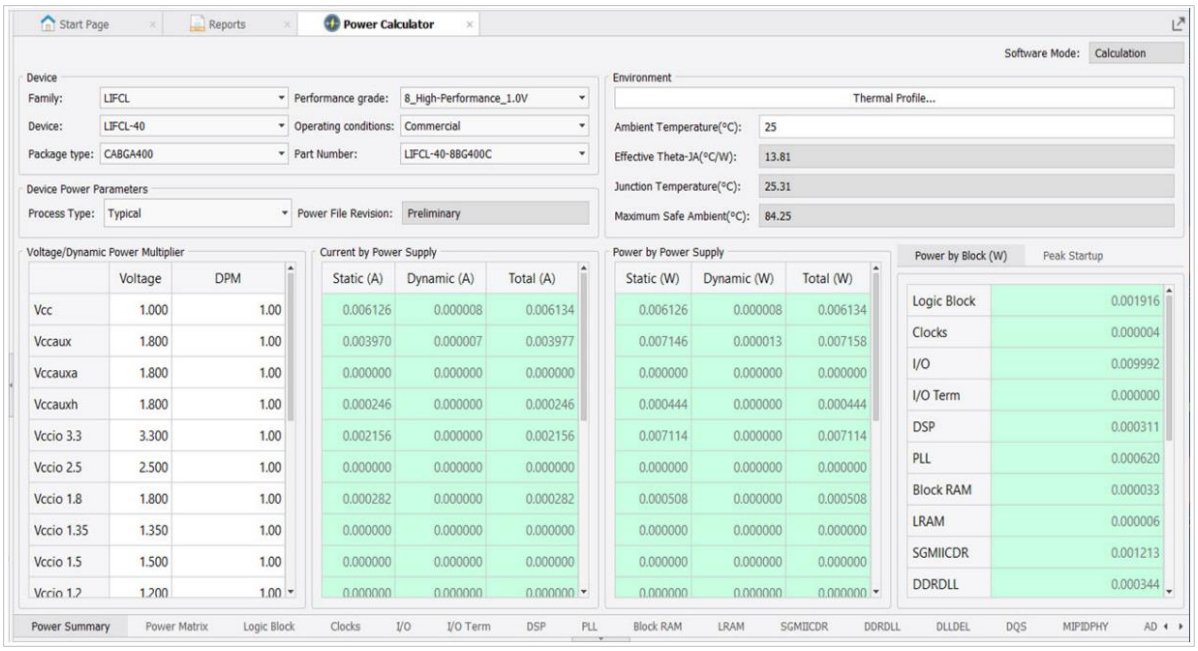


Figure 5.6. Power Calculator

For more information, you may refer to the Lattice Radiant Software Help or to the Lattice Radiant Software 3.1 User Guide Power Calculator section.

6. TCL Scripting

Similar to Quartus, Lattice Radiant Software also supports TCL (Tool Command Language) scripting feature that enable a batch capability for running tools in the Lattice Radiant software user interface. TCL commands can be used through command line/terminal or the Lattice Radiant software Stand-Alone TCL console that is included in the software package. The command set and the Tcl Console used to run it affords you the speed, flexibility and power to extend the range of useful tasks that the Lattice Radiant software tools are already designed to perform.

Using the command line tools allows you to do the following:

- Develop a repeatable design environment and design flow that eliminates setup errors that are common in GUI design flows
- Create test and verification scripts that allow designs to be checked for correct implementation
- Run jobs on demand automatically without user interaction

Note: The environments for both the Lattice Radiant software TCL Console window and Lattice Radiant software Standalone TCL Console window are preset. You can start entering TCL tool commands or core tool commands in the console and the software executes them. '

When running the Lattice Radiant software from the Windows command line (through cmd.exe) or Linux terminal (bash), set up the environment variables as described in the *Setting Up the Environment to Run Command Line* section of the Lattice Radiant Software Help.

For more information, you may refer to the Lattice Radiant Software 3.1 User Guide.

7. Lattice Propel

Lattice Propel is a design environment for Lattice FPGA-based processor system designs. Its development suite includes:

- Integrated development environment (IDE)
- Lattice Propel Builder graphical user interface for System-on-Chip (SoC) design
- Lattice Propel Software Development Kit (SDK) for system software development based on Eclipse Embedded C/C++ Development tools (CDT)

The Lattice Propel flow starts with the creation on an SoC system design in Lattice Propel Builder using the RISC-V CPU together with APB and AHB-L buses, and peripherals.

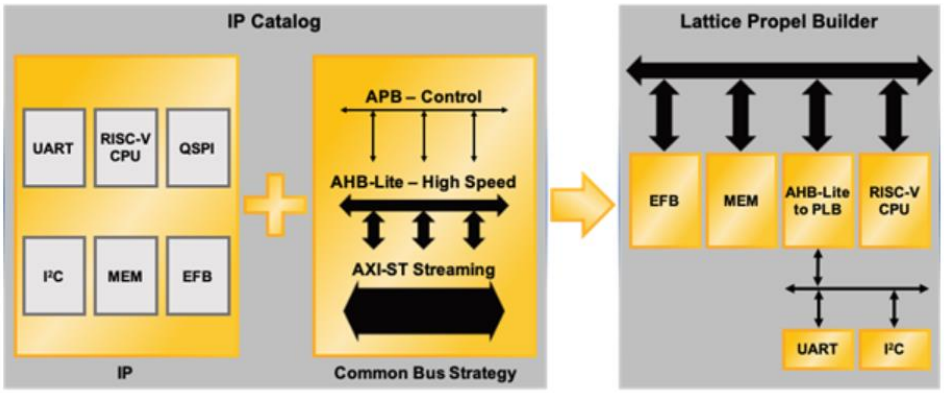


Figure 7.1 Lattice Propel Builder Design Flow

A C/C++ project is then created using the environment file of the SoC design. You can then input your executable code on the SoC design as part of the RTL. Typically, this process can be done by loading the memory file on the system memory of the SoC design.

The next step is to generate the RTL files and then go through the design flow using Lattice Diamond® or Lattice Radiant Software. Once the configuration is loaded onto the SRAM. Design can be verified through Reveal or through GNU debugger of the SDK.

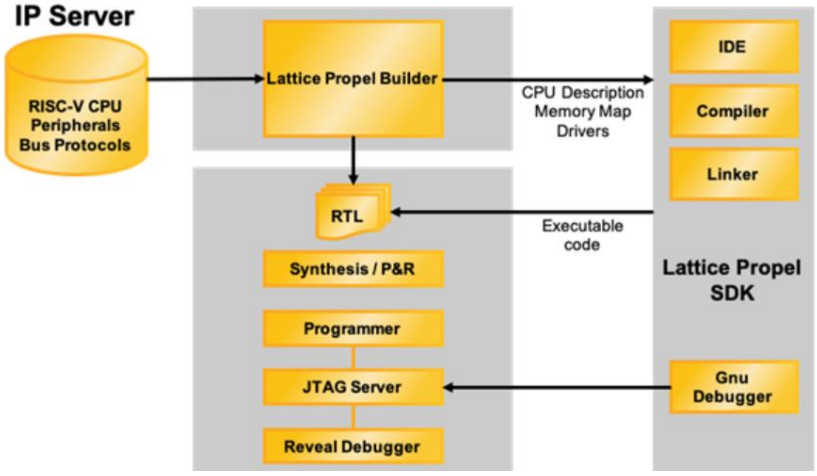


Figure 7.2. Lattice Propel Design Environment

For more information on Lattice Propel, you can find all materials on the Lattice Propel page of the Lattice website.

References

- Lattice Radiant Software 3.1 Release Notes
- Lattice Radiant Software 3.1 User Guide
- Lattice Radiant Software Guide for Lattice Diamond Users
- Migrating iCEcube2 iCE40 UltraPlus Designs to Lattice Radiant Software
- Lattice Radiant Software IP User Guide
- Programming Tools User Guide for Radiant Software
- Reveal User Guide for Radiant Software
- Reveal Troubleshooting Guide for Lattice Radiant Software
- Lattice Radiant Software 3.1 Help (PDF)
- Lattice Radiant Software 3.1 Installation Guide for Windows
- Lattice Radiant Software 3.1 Installation Guide for Linux/Ubuntu
- Quartus II Help v13.0

Revision History

Revision 1.1, April 2022

Section	Change Summary
All	• Changed document title to Lattice Radiant Software Design Flow Overview for Intel Quartus Users • Added copyright and disclaimers information. • Minor editorial and style changes.

Revision 1.0, February 2022

Section	Change Summary
All	Initial release

www.latticesemi.com