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 all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. 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. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. 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.

Inclusive Language
This document was created consistent with Lattice Semiconductor’s inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice’s inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.
Contents

Acronyms in This Document .......................................................................................................................... 3

1. Introduction .............................................................................................................................................. 4

2. Recommended Revision Control Systems .............................................................................................. 5

3. Lattice Propel Revision Control Strategies ............................................................................................. 6
   3.1. Lattice Propel Project Directory Structure ......................................................................................... 6
   3.2. Lattice Propel File Types .................................................................................................................... 7
   3.3. Files Committed for Lattice Propel Revision Control Strategies ....................................................... 7

4. Lattice Propel Revision Control Workflows ............................................................................................. 8
   4.1. Lattice Propel Revision Control Normal Workflow .............................................................................. 8
   4.2. Multiple Developers Working on the Same File Workflow ................................................................. 8
   4.3. Multiple Developers Adding Files Workflow ....................................................................................... 9

5. Reproduce the System using TCL Script ................................................................................................ 11
   5.1. Generate TCL Script for Existing System ......................................................................................... 11
   5.2. Reproduce the System ....................................................................................................................... 11

6. References ................................................................................................................................................. 15

Technical Support Assistance .................................................................................................................... 16

Revision History .......................................................................................................................................... 17

Figures

Figure 3.1. Source Files ................................................................................................................................. 6
Figure 3.2. Verification-related Files .............................................................................................................. 6
Figure 4.1. Lattice Propel Revision Control Normal Workflow ................................................................. 8
Figure 4.2. Multiple Developers Working on the Same File Workflow ..................................................... 9
Figure 4.3. Multiple Developers Adding Files Workflow ......................................................................... 10
Figure 5.1. Generate TCL Script for Existing System ............................................................................... 11
Figure 5.2. IP Catalog View ....................................................................................................................... 12
Figure 5.3. IP Catalog List Command ........................................................................................................ 13
Figure 5.4. Launch TCL Script to Create New Project ............................................................................. 13
Figure 5.5. Relayout Option ....................................................................................................................... 14
Figure 5.6. dut_inst of SoC project ............................................................................................................. 14
Acronyms in This Document
A list of acronyms used in this document.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPGA</td>
<td>Field Programmable Gate Array</td>
</tr>
<tr>
<td>IP</td>
<td>Intellectual Property</td>
</tr>
<tr>
<td>TCL</td>
<td>Tool Command Language</td>
</tr>
</tbody>
</table>
1. Introduction

This user guide introduces how to use revision control for Lattice Propel™ projects. Lattice Propel supports revision control for your projects. You can access the revision control system to get all the change logs and switch to previous milestones if necessary. Using revision control for Lattice Propel projects, the compile time can be reduced. Compilation only starts when inputs are changed.

2. Recommended Revision Control Systems

The following revision control systems are recommended to be used with Lattice Propel:

- GIT
- SVN
- Perforce
3. Lattice Propel Revision Control Strategies

Lattice Propel is friendly to revision control systems. There are many revision control systems available. You do not need to consider version compatibility. Lattice Propel does not directly integrate specific revision control into it. You can select different revision control systems to work with Lattice Propel.

Only the necessary files that constitute the project are managed by the revision control system. Other files, including some intermediate files and your configuration files, are only used in your own project and are not submitted to the version control system.

3.1. Lattice Propel Project Directory Structure

In Lattice Propel, it is recommended that all source codes and IP designs be placed under the same disk partition instead of being used across partitions.

Lattice Propel projects use one directory to put all source and project files. After creating a project, the directory structure of the project is shown in Figure 3.1 and Figure 3.2. All source files are stored in the folder named project name, such as the "hello_world" folder shown in Figure 3.1.

![Figure 3.1. Source Files](image)

If the Propel project has a verification design, all relative files are placed in a verification directory, as shown in Figure 3.2.

![Figure 3.2. Verification-related Files](image)
3.2. Lattice Propel File Types

The project and constraint file types under a Lattice Propel project directory include:

- .socproject file: Propel project file, internal file
- .pdc file: Radiant constraint file
- .lpf file: Diamond constraint file

The basic source or design file types include:

- .v file: Verilog file
- .vhdl file: VHDL file
- .sv file: System Verilog file
- .sbx file: Builder Project file
- lib folder: IP instance files under the folder
- application folder: files of C project under the folder

3.3. Files Committed for Lattice Propel Revision Control Strategies

The following files are committed for revision control strategies of Lattice Propel:

- the .socproject project file and .pdc/.lpf constraint file
- all the source files under directory named project name
- verification directory files
4. Lattice Propel Revision Control Workflows

The Lattice Propel revision control workflows are presented in the following sections.

4.1. Lattice Propel Revision Control Normal Workflow

Each developer has a working directory. You can get the latest code from the Lattice Propel version server and develop in your own development environment. Later, you can submit the modified code or new design to the Propel version server if needed.

![Diagram of Lattice Propel Revision Control Normal Workflow]

Figure 4.1. Lattice Propel Revision Control Normal Workflow

4.2. Multiple Developers Working on the Same File Workflow

Most of the time, developers have their own code changes, and may modify the same source file or design. If developer A submits a file and developer B has updated the file, then developer A gets an error. At this time, developer A needs to update the latest code locally from the server, integrate it with his own modifications, and submit it again. This is the collaborative operation of multiple developers for revision control.
4.3. Multiple Developers Adding Files Workflow

Developer A may add a new file `a.v` to the design, submitting this file and the Propel project file to the revision control server. After that, Developer B can get the latest code from the server, merge the local code, resolve the conflict, and continue the subsequent development.
Revision Control Server for Propel Projects (Project1, Project2, ...)

- Get latest code
- Developer A
  - Add a.v of Project1
  - Commit/Push (a.v and project1.rdf)
  - Update latest code
- Merge code, resolve conflicts
- Continue developing
- Commit/Push

- Get latest code
- Developer B
  - Get latest a.v and project1.rdf on server
- Push succeeded

✓ Push succeeded

Figure 4.3. Multiple Developers Adding Files Workflow
5. Reproduce the System using TCL Script

You can generate a TCL script from the current SoC project, and then set up a new project by launching this TCL script in the new environment.

5.1. Generate TCL Script for Existing System

To generate the TCL file:

1. In Lattice Propel Builder, save the current SoC project by clicking the icon.
2. Type ‘sbp_design gen_tcl’ in the TCL console. The output of this command shows the location of the TCL script, as shown in the following example:

   Tcl file C:/lscc/workspace/Example/Example/gen_newsbx.tcl was generated successfully (Figure 5.1).

![Figure 5.1. Generate TCL Script for Existing System](image)

5.2. Reproduce the System

You can use the TCL script generated for the existing system to reproduce a new SoC project.

1. Before launching this TCL script in a new environment, make sure all the IPs mentioned in the script are installed in the new environment. Both name and version of the IPs should match.

   You can check all IPs from the IP Catalog view (Figure 5.2).

![Figure 5.2. Reproduce the System](image)
2. Or, you can check all IPs by typing the following command in TCL console:

```
ip_catalog_list (Figure 5.3).
```
3. Fetch the TCL script and change corresponding settings in the script to use in the new environment:
   - targetDir: The new project directory. Make sure this directory already exists before launching TCL.
   - projectName: Project name.
   - propelRegenLscBase: The Propel installation path. Make sure this directory exists.

4. To launch this TCL script to create a new project, make sure no design is currently open in Propel Builder. Type in `source <tcl_path>` in the TCL console:
   ```
   source D:/iscc/workspace/Example/Example/gen_newsbx.tcl
   ```
   (Figure 5.4).
5. (Optional) Right-click on the Schematic view and select Relayout if the new design is out of proportion (Figure 5.5).

6. Switch to Verification Project from Design Project.

In a new Lattice Propel SoC design project, you can switch to the verification project of this design by clicking on the icon from the toolbar, and then a dut_inst of the SoC project can be seen in the Schematic view (Figure 5.6).

Note: If the original SoC project does not contain a corresponding verification project, the Switch Verification and SoC Design icon is grayed out.
6. References

- Lattice Propel 2023.2 SDK User Guide (FPGA-UG-02195)
- IP Packager 2023.2 User Guide (FPGA-UG-02197)
- Lattice Propel 2023.2 Installation for Windows User Guide (FPGA-AN-02069)
- Lattice Propel 2023.2 Installation for Linux User Guide (FPGA-AN-02070)

For more information, refer to:
- Lattice Propel Web Page
- Lattice Insights for Training Series and Learning Plans
Technical Support Assistance
Submit a technical support case through www.latticesemi.com/techsupport.
For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.
## Revision History

**Revision 1.0, November 2023**

<table>
<thead>
<tr>
<th>Section</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Production release.</td>
</tr>
</tbody>
</table>