Lattice SensAI Neural Network Compiler Software

User Guide

FPGA-UG-02052 Version 1.0

May 18
LICENSE AGREEMENT

This is a legal agreement between you, the end user, and Lattice Semiconductor Corporation if you are an end user located in the United States or Lattice SG Pte. Ltd. if you are an end user located in a country other than the United States. By proceeding with the installation or use of the Software you acknowledge you have read this Agreement, you understand it, and you agree to be bound by the terms and conditions of this Agreement. If you do not agree to the terms and conditions of this Agreement, do not use, download, install the Software, and if you have already obtained the Software from an authorized source, promptly return the media package and all accompanying items (including written materials and binders or other containers) to the place you obtained them for a full refund of any applicable license fees.

Lattice Semiconductor Corporation or Lattice SG Pte. Ltd. ("Lattice") and the individual or entity acquiring the Software ("Licensee") agree as follows:

1. DEFINITIONS

"Software" means the computer program(s) in machine-readable form furnished to Licensee by Lattice, in whatever media and by whatever method, which are enabled for use pursuant to Lattice’s software protection mechanism, and for which Licensee has paid any applicable license fees. Software includes any related update or upgrade programs that may be added from time to time.

2. SOFTWARE LICENSE

a. Lattice hereby grants to Licensee a non-exclusive, nontransferable license to use the Software for Licensee’s internal purposes only on any computer possessed by Licensee on which the Software is designed to operate, such use to be in accordance with and subject to the terms and conditions of this Agreement.

b. Pursuant to this Agreement, Licensee may (i) physically transfer any Software from one computer to another; and (ii) use the Software and any output files generated by the Software for the sole purpose of designing and programming semiconductor components manufactured by or for Lattice and sold by Lattice or its authorized distributors ("Lattice Devices"); and (iii) make copies of the Software solely for Licensee’s own use.

c. Third Party Components. Certain open source or free software ("Open Source Software") distributed with the Software are licensed from third parties and subject to separate license terms (the “Open Source Software License Terms”). The applicable Open Source Software License Terms are set forth in a directory named "Open Source Licenses" that is generated as part of the Software install process. If you do not want to be bound by the Open Source License Terms, you must immediately cease use of the Software and uninstall the Software.

d. Licensee shall include Lattice’s (and Lattice’s suppliers’, as applicable) copyrights, trademarks, and other proprietary notices on any copies of the Software.

e. Licensee shall not distribute, copy, transfer, lend, incorporate, modify, or use the Software for any purpose except as expressly provided herein.

f. If Licensee fails to comply with the provisions of this Agreement, this license is automatically terminated.

g. Except for the rights expressly granted herein to Licensee, the title and all intellectual property rights in and to the Software and any copy of the Software which may be made by Licensee hereunder remain the sole and exclusive property of Lattice and/or Lattice’s licensors.

h. Licensee acknowledges that obtaining a license for the Software does not entitle Licensee to technical support from Lattice regarding the Software. However, Lattice will use commercially reasonable efforts to respond to technical support inquiries from Licensee regarding the Software. Response times to inquiries for technical support are not guaranteed and will be established by Lattice in its sole discretion.

3. LIMITED WARRANTY AND REMEDIES

a. If the Software has been provided on tangible media, then Lattice warrants to Licensee that the media containing the Software will be free from defects in materials and workmanship under normal use and service for a period of ninety (90) days from the date of delivery.

b. Subject to applicable laws, during the 90-day warranty period, the entire liability of Lattice and its licensors to Licensee, and Licensee’s exclusive remedy under this warranty after Licensee’s return of the defective Software media, will be for Lattice, at its option, either to replace any such Software media or refund any applicable license fee paid by Licensee to Lattice and terminate this License Agreement. Any replacement Software or media will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer.

c. Any products which are not returned to Lattice within the warranty period or which have been subject to accident, abuse, misuse, alteration, neglect, or unauthorized repair or installation are not covered by warranty.

4. WARRANTY DISCLAIMER

EXCEPT FOR THE ABOVE EXPRESSED LIMITED WARRANTIES, LATTICE MAKES NO WARRANTIES ON THE SOFTWARE, WHETHER EXPRESSED, IMPLIED, STATUTORY, OR IN ANY OTHER PROVISION OF THIS AGREEMENT OR COMMUNICATION WITH LICENSEE, AND LATTICE SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. LATTICE DOES NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE SOFTWARE WILL MEET LICENSEE’S REQUIREMENTS, OR THAT LICENSEE’S OPERATION OF THE SOFTWARE WILL BE UNINTERRUPTED OR ERROR FREE, OR THAT DEFECTS IN THE SOFTWARE WILL BE CORRECTED. LICENSEE ASSUMES RESPONSIBILITY FOR SELECTION OF THE SOFTWARE TO ACHIEVE ITS INTENDED RESULTS, AND FOR THE PROPER INSTALLATION, USE, AND RESULTS OBTAINED FROM THE SOFTWARE.

EXCEPT FOR THE ABOVE EXPRESSED LIMITED WARRANTIES, LICENSEE ASSUMES THE ENTIRE RISK OF THE SOFTWARE PROVING DEFECTIVE OR FAILING TO PERFORM PROPERLY AND IN SUCH EVENT, LICENSEE SHALL ASSUME THE ENTIRE COST AND RISK OF ANY REPAIR, SERVICE, CORRECTION, OR ANY OTHER LIABILITIES OR DAMAGES CAUSED BY OR ASSOCIATED WITH THE SOFTWARE. LATTICE’S SOLE LIABILITY, AND LICENSEE’S SOLE REMEDY, IS SET FORTH ABOVE. LATTICE DOES NOT WARRANT THAT USE OF THE SOFTWARE DOES NOT INFRINGE ON THIRD PARTIES’ INTELLECTUAL PROPERTY RIGHTS.

5. SOURCE CODE

Licensee shall not attempt to reverse translate, decompile or otherwise attempt to derive the source code of the Software. In the event any source code is explicitly licensed to Licensee as part of the Software, such limitation will not apply to such source code. Licensee shall not alter or remove from the Software any copyright, trademark or other proprietary notices of Lattice and/or Lattice’s licensors. Any use or attempted use of the Software in violation of the foregoing restrictions is a breach of the Agreement which will cause irreparable harm to Lattice, entitling Lattice to injunctive relief in addition to all legal remedies.

6. LIMITATION OF LIABILITY

a. Licensee agrees that Lattice’s entire liability to Licensee and Licensee’s sole remedy hereunder for any cause whatsoever, regardless of the form of the action, shall be limited to the price paid to Lattice for the Software.
b. IN NO EVENT WILL LATTICE OR ANY OF ITS SUPPLIERS BE LIABLE TO LICENSEE OR ANY OTHER PERSON FOR ANY DAMAGES, INCLUDING ANY DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL, OR SPECIAL DAMAGES, INCLUDING EXPENSES, LOST PROFITS, LOST SAVINGS, OR OTHER DAMAGES OF ANY SORT ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE, EVEN IF LATTICE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME JURISDICTIONS DO NOT ALLOW THE LIMITATION OR EXCLUSION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

7. DEFAULT AND TERMINATION

Lattice licenses the Software for an indefinite term. Accordingly, this Agreement will continue indefinitely, until and unless terminated. This Agreement will terminate automatically in the event Licensee fails to perform any of its obligations hereunder. Licensee may terminate this Agreement at any time by returning to Lattice the original and all copies of the Software or by destroying the Software together with all copies thereof. Upon termination of this Agreement for any reason, Licensee shall either return to Lattice the original and all copies of the Software, or, upon Lattice’s request, destroy such original and copies and provide Lattice with written certification of their destruction. The obligations of Lattice and Licensee under the provisions of Sections 2.g, 4, 5, 6, 7, 8, 9, 10 and 11 will survive any termination of this Agreement.

8. EXPORT CONTROL

Licensee shall not export the Software or the direct product thereof without first obtaining any necessary U.S. or other governmental licenses and approvals. In connection with such export control compliance, Licensee certifies as follows:
* that Licensee is not on the Denied Persons List maintained by the U.S. Bureau of Industry and Security;
* that Licensee is not on the list of Specially Designated Nationals and Blocked Persons maintained by the U.S. Department of the Treasury;
* that Licensee is not a citizen or resident of, or an agent of, Cuba, Iran, Iraq, North Korea, Sudan, or Syria, or any other country to which export of the referenced Software is prohibited; and
* that Licensee is legally permitted, under all applicable export and commerce control laws and regulations, to receive the referenced Software.

9. U.S. GOVERNMENT RESTRICTED RIGHTS

The Software and any accompanying documentation provided to agencies of the U.S. Government are "commercial computer software" and "commercial computer software documentation" pursuant to DFARS 227.7202 and FAR 12.212, and their successors. All use, reproduction, release, performance, display or disclosure of the Software and related documentation by or for the U.S. Government shall be in strict accordance with the terms and conditions of this Agreement. Contractor/manufacturer is Lattice Semiconductor Limited c/o Lattice Semiconductor Corporation, 111 SW 5th Ave., Suite 700, Portland, Oregon 97204 and its licensors.

10. INFORMATION REGARDING PERSONAL DATA.

If you downloaded this Software from our website, we have collected information about you, including your name and contact information, from the information you provided when you registered to use the website. If you acquired the Software from a source other than our website, we will ask you for certain information, including your name and contact information, as part of the installation procedure.

11. GENERAL.

IF LICENSEE IS LOCATED IN THE UNITED STATES, THIS AGREEMENT WILL BE GOVERNED BY THE LAWS OF THE STATE OF OREGON, U.S.A. WITHOUT REFERENCE TO ANY CONFLICT OF LAW PRINCIPLES. IF LICENSEE IS LOCATED IN A COUNTRY OTHER THAN THE UNITED STATES, THIS AGREEMENT WILL BE GOVERNED BY THE LAWS OF THE REPUBLIC OF SINGAPORE WITHOUT REFERENCE TO ANY CONFLICT OF LAW PRINCIPLES. THE APPLICATION OF THE UNITED NATIONS CONVENTION ON CONTRACTS FOR THE INTERNATIONAL SALE OF GOODS IS EXPRESSLY EXCLUDED. NOTHING IN THIS AGREEMENT WILL BE INTERPRETED OR CONSTRUED SO AS TO LIMIT OR EXCLUDE THE RIGHTS OR OBLIGATIONS OF LICENSEE OR LATTTICE WHICH IT IS UNLAWFUL TO LIMIT OR EXCLUDE UNDER APPLICABLE LAWS, INCLUDING THE LAWS OF ANY MEMBER STATE OF THE EUROPEAN UNION WHICH IMPLEMENTS RELEVANT EUROPEAN COMMUNITIES COUNCIL DIRECTIVES. Except as specifically provided in this Agreement, Licensee may not sublicense, assign, or transfer this license or the Software. Any attempted assignment, transfer or sublicense by Licensee in violation of this provision shall be void. Subject to the foregoing, this Agreement shall be binding upon and inure to the benefit of the successors and permitted assigns of the parties. The prevailing party in any legal action or arbitration arising out of this Agreement shall be entitled to reimbursement for reasonable attorneys fees and expenses, in addition to any other rights and remedies such party may have. This Agreement is the entire agreement between the parties with respect to the use of the Software and supersedes any other communications or prior agreements, oral or written, regarding the Software. If any provision of this Agreement is held invalid, the remainder of the Agreement shall continue in full force and effect.

Please direct all inquiries, in writing, to Lattice Semiconductor Corporation, 111 SW 5th Ave, Suite 700, Portland, Oregon 97204.

Lattice Patent Rights

This Software is made available for use under a license from Lattice to all applicable U.S. and foreign patents, subject to the conditions and restrictions provided by this License Agreement. Applicable patents are contained in the Legal Notice located at:
http://www.latticesemi.com/About/LegalNotices.aspx

Lattice does not represent that products described herein are free from patent infringement or from any third-party right.

Trademark Rights

This Software may contain registered trademarks of Lattice or third parties. Such trademarks are the property of their respective owners. The Software governed by this License Agreement is Copyright (c) 2018 Lattice Semiconductor Corporation. All rights reserved.

2018.04.
Contents

Chapters
LICENSE AGREEMENT ......................................................................................................................... 2
Contents .................................................................................................................................................. 4
Chapters ................................................................................................................................................ 4
Figures .................................................................................................................................................. 6
1. Introduction ...................................................................................................................................... 7
  1.1. Prerequisites ............................................................................................................................... 7
    1.1.1. Hardware Requirements ........................................................................................................ 7
    1.1.2. Software Requirements .......................................................................................................... 7
    1.1.3. Connection Requirements .................................................................................................... 7
    1.1.4. General Requirements ........................................................................................................... 7
  1.2. Purpose ......................................................................................................................................... 7
  1.3. Limitations ................................................................................................................................... 7
2. Installing the Software ....................................................................................................................... 8
3. Getting Started ................................................................................................................................ 12
  3.1. Creating a New Project ................................................................................................................. 12
  3.2. Opening an Existing Project ........................................................................................................ 15
  3.3. Saving a Project ............................................................................................................................ 16
  3.4. Help ............................................................................................................................................ 16
    3.4.1. About .................................................................................................................................... 16
    3.4.2. User Guide ............................................................................................................................. 16
  3.5. Command Line Interface ............................................................................................................. 17
    3.5.1. Usage ..................................................................................................................................... 17
    3.5.2. Arguments .............................................................................................................................. 17
  3.6. Next Steps .................................................................................................................................. 19
4. Working with Projects ....................................................................................................................... 20
  4.1. Implementations ............................................................................................................................ 20
    4.1.1. Creating a New Implementation .............................................................................................. 20
    4.1.2. Editing an Implementation .................................................................................................... 20
  4.2. Project Flow .................................................................................................................................. 21
    4.2.1. Analyze .................................................................................................................................. 21
    4.2.2. Compile ................................................................................................................................... 22
    4.2.3. Simulate .................................................................................................................................. 23
    4.2.4. Download .............................................................................................................................. 23
    4.2.5. Run ........................................................................................................................................ 24
  4.3. Views ........................................................................................................................................... 25
    4.3.1. Input Network ......................................................................................................................... 25
    4.3.2. Analyzed Network .................................................................................................................. 25
    4.3.3. Simulation Output ................................................................................................................... 25
    4.3.4. Log File .................................................................................................................................. 25
5. Advanced Topics .............................................................................................................................. 27
  5.1. Project Implementation Settings .................................................................................................. 27
    5.1.1. Number of Convolution Engines .......................................................................................... 27
    5.1.2. On-Chip Memory Block Size ............................................................................................... 27
    5.1.3. Number of On-Chip Memory Blocks .................................................................................... 27
    5.1.4. Off-Chip Data Memory Start Address ................................................................................... 28
    5.1.5. Mean Value for Data Pre-Processing ..................................................................................... 28
    5.1.6. Scale Value for Data Pre-Processing ....................................................................................... 28
6. Supported Frameworks .................................................................................................................... 29
  6.1. Caffe .............................................................................................................................................. 29
6.1.1. Binary Neural Networks ........................................................................................................ 29
6.1.2. Supported Layers List ........................................................................................................ 29
6.2. Tensorflow ............................................................................................................................ 31
   6.2.1. Conversion ..................................................................................................................... 31
   6.2.2. Supported Operations List ............................................................................................ 33
7. References .................................................................................................................................. 34
8. Technical Support Assistance ..................................................................................................... 35
Appendix A: Supported and Added Caffe Layers ......................................................................... 36
   Accuracy .................................................................................................................................. 36
   BatchNorm .............................................................................................................................. 36
   Binarize ................................................................................................................................... 36
   BinaryInnerProduct ................................................................................................................ 36
   BinaryConvolution ................................................................................................................... 36
   Convolution ............................................................................................................................ 36
   Eltwise ...................................................................................................................................... 36
   InnerProduct .......................................................................................................................... 36
   Input ......................................................................................................................................... 37
   Pooling ...................................................................................................................................... 37
   ReLU ......................................................................................................................................... 37
   Scale ......................................................................................................................................... 37
Appendix B: Supported Tensorflow Operations ............................................................................. 38
   Batch Normalization ............................................................................................................... 38
   Conv2D ..................................................................................................................................... 39
   Elementwise Vector Add ......................................................................................................... 39
   Matmul ...................................................................................................................................... 39
   Placeholder ............................................................................................................................... 39
   Pooling ...................................................................................................................................... 39
   ReLU ......................................................................................................................................... 39
Revision History ............................................................................................................................ 40
Figures

Figure 2.1: Installation Location Specification ................................................................. 8
Figure 2.2: Installation Component Specification .............................................................. 9
Figure 2.3: Installation Ready to Install Dialog Box ........................................................... 10
Figure 2.4: Lattice SensAI Software for Windows Splash Screen ..................................... 11
Figure 3.1: Project Settings Window ................................................................................. 12
Figure 3.2: Proto File Selection Window ........................................................................... 13
Figure 3.3: Project Implementation Options Window ......................................................... 14
Figure 3.4: Project Window ............................................................................................... 15
Figure 3.5: Load Project Window ....................................................................................... 16
Figure 4.1: Project Implementation Options Window ......................................................... 20
Figure 4.2: Analyze Results ............................................................................................. 21
Figure 4.3: Compile Results ............................................................................................. 22
Figure 4.4: Simulate Results ............................................................................................ 23
Figure 4.5: Download Results .......................................................................................... 24
Figure 4.6: Run Results .................................................................................................... 25
Figure 5.1: Project Implementation Window ....................................................................... 27
6.1: Original Tensorflow Training Model ........................................................................... 32
6.2: Simplified Tensorflow Inference Model ...................................................................... 33
Figure B.1: Batch Normalization ...................................................................................... 38
1. Introduction

This document describes the usage and troubleshooting of Lattice SensAI Software.

1.1. Prerequisites

The hardware, software, connection, and general requirements for this demonstration are provided in the following sections.

1.1.1. Hardware Requirements

The software requires the following hardware components:

- PC with either Windows 7 x64 or newer, or PC with compatible Ubuntu x64 distribution for running software flow only.
- Lattice Inference Machine-compatible FPGA.

1.1.2. Software Requirements

This software product requires the following software components:

- Lattice SensAI Software for Windows or Linux.
- Diamond Programmer System software for downloading the FPGA bitstream.
- Lattice Diamond® Design Software for modifying the platform and regenerating the bitstream.

1.1.3. Connection Requirements

Programming the device and running Lattice SensAI Software directly from the GUI requires a Windows installation, and a Windows-compatible connection such as the PCIE driver for Lattice FPGA development boards.

1.1.4. General Requirements

This document requires some knowledge of the following:

- Familiarity with either Caffe or Tensorflow Machine Learning Frameworks.
- Familiarity with Lattice FPGA development including basic concepts and troubleshooting skills and experience establishing basic connectivity between the device and computer, or else utilizing some other hardware (such as an SD card) for transferring data onto the intended hardware.

1.2. Purpose

This application shows the ability and features of Lattice SensAI Software to:

- Analyze and Compile a neural network for use with select Lattice Semiconductor FPGA products.
- Simulate hardware to obtained expected fixed- and floating-point output.
- Download and Run neural networks directly onto hardware.
- Manage multiple implementations per project to view effects of different strategies.

1.3. Limitations

The following cautions apply to the software:

- Operations are conducted in fixed point notation on the hardware as a result of floating point values being converted to and from fixed point representation.
- Specific neural network features, such as layers or functions, require certain configurations to function or may not be supported.
- Features that interface with Lattice hardware over the PCIE connection are limited to Windows systems only.
2. Installing the Software

The demonstration package of the Lattice SensAI Software is available as an executable installer for Windows and Linux systems. The software is installed on Windows by using the Machine Learning Software Setup executable installer (.exe) or on Ubuntu Linux by using the run file (.run). Launch the installation process and customize the options, as detailed in this section.

To install Lattice SensAI Software:

1. Close all applications before starting the Lattice SensAI Software installation.
2. Double-click on the Lattice SensAI Software installer you downloaded.
3. The Welcome to the Lattice SensAI Software 1.0 Software Setup dialog box opens.
4. Click Next to select Installation Folder.
5. On Windows, the default destination folder is C:\lsc\ml\1.0. On Linux, the default installation directory is ~/lscc/rel/1.0. Click Browse to change the destination.

![Figure 2.1: Installation Location Specification](image)

6. Click Next to open the Product Options dialog box.
7. Select Machine Learning Software components that you want to install by selecting or clearing each of the listed options.
8. Click **Next** to open the License Agreement dialog box.

9. Read the license agreement. If you agree, click **I accept the license** to open the Start Menu shortcuts dialog box.

10. Click **Next** to open the Select Program Folder dialog box. The default name is Lattice SensAI Software 1.0. If you want to change the name, change it in the Program Folder text box.

11. Click **Next** to display the Ready to Install dialog box. Review the current settings: the destination folder and components selected. If everything is correct, select Install to start the installation.
Figure 2.3: Installation Ready to Install Dialog Box

12. In the Installation Wizard Complete dialog box, read the confirmation note and click Finish.

13. Run the executable, either by using the desktop or start menu shortcuts if it created, or by navigating to your installation directory and running `lsc_ml_compl.exe` on Windows, or `lsc_ml_compl` on Ubuntu Linux. You will be greeted by the main window, as shown below in Figure 2.4
The installed software is now ready for use.
3. Getting Started

In this chapter, you will learn how to use Lattice SensAI Software to create new projects and edit existing projects.

3.1. Creating a New Project

A project is a collection of all files necessary to create and download your design to the selected device. The New Project window guides you through the steps of specifying a project name and adding existing sources to the new project.

To create a new project:

1. From the main window, click File> New.

   The Project Settings window opens, as shown in Figure 3.1

   Figure 3.1: Project Settings Window

2. Enter a project name into the Project field at the top-left.

3. Click on Network File. The Proto File Selection window opens, as shown in Figure 3.2
4. Navigate to your proto file and select it in the window.
5. Click Open to load the proto file into your project.
6. Click Model File and follow a similar process to step 3-5, this time selecting your model file.
7. Click Image/Video Data and follow a similar process to step 3-5, this time selecting your image or video file. Check the Capture button if you plan on using footage from an attached camera as the input.
8. Click Next to open the Project Implementation Options Window, as shown in Figure 3.3.
9. The Project Implementation Window is automatically filled with default settings for the Implementation Name, as well as the parameters. You can change the name and parameters if desired. For more information on how each parameter works and their limitations, read the chapter “Advanced Topics > Project Implementation Settings”.

10. Click **Ok** to create your project. The Project Window opens, as shown in Figure 3.4.
3.2. Opening an Existing Project

1. Use one of the following methods to open an existing Lattice SensAI Software project:
   - On the Main Window, click the **Open Project** button.
   - From the **File** menu, choose **Open**.

   The Open Project Window will open, as shown in Figure 3.4.
2. Navigate to an existing LDNN type file and select it.
3. Click Open to open the project.

### 3.3. Saving a Project

When working on a project you wish to save, click on the floppy disk icon, or navigate to **File>Save**, in order to save your project. This will save the files with the project name into the project directory, as specified in your project settings.

### 3.4. Help

For more software help, the **Help** menu contains links to relevant help topics.

#### 3.4.1. About

To find out more version and license information, navigate to **Help>About** to bring up the About window, which has tabs for different software information sections. The **About** tab contains information about the software. Your current version and build number are displayed here. The **License** tab provides a convenient way to view the license agreement.

#### 3.4.2. User Guide

This User Guide is routinely updated, and may not be the latest version. To quickly go to the Lattice Semiconductor web page which contains the latest version of the User Guide, as well as supplemental material, navigate to **Help>User Guide** and you will be taken to the correct page.
3.5. Command Line Interface

The executable can be used from a command line interface if you would prefer not to use the GUI. To execute a command, launch the executable from the command line and pass it the arguments you wish to use.

For example, to bring up the help Windows CLI in Cygwin, the command is:

   lsc_ml_compl.exe --help

While on Linux, execute it as:

   ./lsc_ml_compl --help

This will bring up the help menu for the CLI. You can see the usage and arguments in the following sections in this chapter.

3.5.1. Usage

- `-h`
- `--gui [GUI]`
- `--cmd CMD`
- `--framework FRAMEWORK`
- `--network_file NETWORK_FILE`
- `--model_file MODEL_FILE`
- `--image_files IMAGE_FILES`
- `--class_file CLASS_FILE`
- `--dfrac DFRAC_LIST DFRAC_LIST`
- `--num_conv_eng NUM_CONV_ENG`
- `--num_ebr NUM_EBR`
- `--ebr_blk_size EBR_BLK_SIZE`
- `--extmem_start_addr EXTMEM_START_ADDR`
- `--mean MEAN`
- `--scale SCALE`
- `--extmem_off EXTMEM_OFF`
- `--load_from_extmem LOAD_FROM_EXTMEM`
- `--store_to_extmem STORE_TO_EXTMEM`
- `--hwcfg_file HWCFG_FILE`
- `--out_file OUT_FILE`
- `--sim`
- `--project_name PROJECT_NAME`
- `--project_dir PROJECT_DIR`
- `--skip SKIP_LAYERS`
- `--dumpmem DUMP_MEMS`
- `--dump DUMP_BLOBS`
- `--check CHECK_LAYERS`

3.5.2. Arguments

- `-h, --help`
  - Show this help message and exit
- `--gui [GUI]`
Invoke GUI tool

- --cmd CMD
  - Valid commands are analyze, compile, simulate, download, run, all

- --framework FRAMEWORK
  - Framework used to train network. Currently, Caffe and TensorFlow are supported

- --network_file NETWORK_FILE
  - Caffe .prototxt or .proto file

- --model_file MODEL_FILE
  - .Caffemodel file

- --image_files IMAGE_FILES
  - .jpg Image file

- --class_file CLASS_FILE
  - Class file

- --dfrac DFRAC_LIST DFRAC_LIST
  - Set blob fraction by user

- --num_conv_eng NUM_CONV_ENG
  - Number of convolution engines used

- --num_ebr NUM_EBR
  - Number of embedded block ram

- --ebr_blk_size EBR_BLK_SIZE
  - Size of each embedded block ram

- --extmem_start_addr EXTMEM_START_ADDR
  - Starting address of external DRAM to store data

- --mean MEAN
  - Mean value used to preprocess data during training

- --scale SCALE
  - Scale value used to preprocess data during training

- --extmem_off EXTMEM_OFF
  - Turn-off using external memory to store data. By default external memory is used to store input/output and scratch data.

- --load_from_extmem LOAD_FROM_EXTMEM
  - By default we load data from external memory to internal memory. If this option is '0' it will make sure data is directly loaded to EBR from sensor or host

- --store_to_extmem STORE_TO_EXTMEM
  - By default we store output data to external memory. If this option is '0' make sure to read data from internal memory

- --hwcfg_file HWCFG_FILE
  - HW configuration file

- --out_file OUT_FILE
  - Output file to write firmware

- --sim
  - Enable Simulation

- --project_name PROJECT_NAME
3.6. Next Steps

Now that you have created or opened a project, you are ready to edit your project and run through the design flow, as detailed in the next chapter.
4. Working with Projects

4.1. Implementations
Implementations organize the structure of your design and allow you to try alternate structures and tool settings to determine which ones will give you the best results. To help determine which scenario best meets your project goals, use a different implementation of a design using the different settings. Each implementation has associated active settings; when you create a new implementation you must select its active settings.

4.1.1. Creating a New Implementation
To try a new implementation with different strategies within an existing project, you must create a new implementation.
1. Choose File > Add Impl to bring up the Implementation Options window.
2. The Implementation Options window has the same parameters as the one you encountered creating your project initially. You can change the implementation name to a unique string if desired. Within the project, each implementation must have a unique name.
3. Change the implementation settings from the default settings, if desired.

4.1.2. Editing an Implementation
You can edit an existing implementation to change the specific input and output files.
1. Choose File > Edit Impl to bring up the Project Settings window.
2. The Project Implementation Settings Window will open, as shown in Figure 4.1.

![Figure 4.1: Project Implementation Options Window](image)

3. Edit your existing settings and click OK to apply them to your Project Implementation. For more information on parameters and their limitations, read the chapter “Advanced Topics > Project Implementation Settings”.

© 2018 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal. All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.
4.2. Project Flow

4.2.1. Analyze

You must first run the Analyze function on your project before you can progress to the Compile or Simulate stages. It will analyze your code to verify compatibility with the Lattice DNN Compiler. You can run the Analyzer by selecting Process>Analyze.

![Analyze Results](image)

**Figure 4.2: Analyze Results**

After successfully analyzing a neural network file, the implementation window will update with a set of columns listing the properties of your neural network under the current settings.

- **Blobs:** Each blob that is detected and implemented by the software is listed in this column. Some blobs that are in the network file will not be implemented in the hardware, such as those used for external data processing, and not listed here.

- **Signed Data Format:** This column lists the breakdown of the fixed-point representation of the blob. The number preceding the period is the number of bits used to represent the integer component of the number, while the number following it is the number of bits used in the fractional component. The total number of bits is one less than the total number of bits used, as one bit is always used for signage. For clarification, the following examples are provided for a 16-bit number, using 15 bits to represent the integer and fractional component:
  - 3.12 represents a number with 3 integer bits and 12 fractional bits.
  - 10.5 represents a number with 10 integer bits and 5 fractional bits.

- **Stored Data Format:** This column is a user-editable list of the fixed-point representations of each blob. It is populated with the default values that are automatically calculated by the software. Values are written in the same format as the Signed Data Format entry above. In order to edit the Stored Data Format for a blob, double-click the entry in that column for the blob in question.

You can allocate how many bits you want dedicated to the integer and fractional component for EBR storage for the specified blob. You will have to specify whether the EBR accepts 16-bit mode or 8-bit mode. To use 16-bit mode, your two values will need to add up to 15. To use 8-bit mode, your two values will need to add up to 7.
Lattice SensAI Neural Network Compiler Software
User Guide

- 12.3 represents EBR storage in 16-bit mode with 12 integer bits and 3 fraction bits
- 6.1 represents EBR storage in 8-bit mode with 6 integer bits and 1 fraction bit.

- Required Memory Bytes: The memory required to implement each blob is listed in this column. See Chapter 5.1 “Project Implementation Settings” for more details on the effects your settings may have on this.

4.2.2. Compile
You can create a firmware file for your analyzed network by running the compilation flow. This will generate a lscml-type file which can be used to download the network to your hardware by the software, or by another tool. You can run the Compiler by selecting Process＞Compile.

Figure 4.3: Compile Results

After your network has successfully compiled, you will be presented with performance information. The cycles used by your neural network given the specified settings are reported, with a breakdown of cycles spent on DRAM access, convolution, pooling, fully connected, and scale.

- DRAM: These are the cycles that are spent accessing or storing data in the DRAM. Designs that use more of the EBR for storage will have fewer cycles used in the DRAM stage, and this number will increase as your settings offload more storage from the EBR to the DRAM.
- Conv: The cycles used performing convolution are reported here. In a conventional neural network this represents the standard convolution cycles. In a binary neural network it displays the cycles used during binary convolution. In designs utilizing EBR, it typically represents the largest share of cycles in your design.
- Pool: These cycles are used to implement pooling in your neural network.
- FC: This entry corresponds to cycles used to implement fully connected (or inner product) vector operation
- Scale: Scaling cycles are spent performing the scale operation.
4.2.3. **Simulate**

It is recommended that you run the Simulation to verify results. This is not a required step to Compile, Download, or Run your project. You can simulate your analyzed network using the Simulate feature. By selecting the green or red check boxes in the process window of the left pane, the simulation type can be changed between the network or hardware model. By default, the Fixed Point Hardware Model simulation is selected. You can run the Simulator by selecting **Process>Simulate**.

![Figure 4.4: Simulate Results](image)

The inputs and outputs of the simulation are determined by your neural network and your source file. The total cycles reported are identical to those found in the Compilation stage.

4.2.4. **Download**

Lattice SensAI Software for Windows is capable of directly downloading a project to a compatible board that is connected to the computer. The test board must be connected via PCIe. You can run the download tool by selecting **Process>Download**.
The software will begin the download process and display a message when finished, indicating its status. The download may fail for several reasons, such as if your connection to the board is interrupted, or if you are attempting to load an incompatible network. After a successful download, your network is ready to be run on the board in the next stage of the flow.

4.2.5. Run

Lattice SensAI Software can run a downloaded design on a board over PCIE. This requires a compatible hardware connection, as detailed in the Download stage. To run your design, click the Run button.
Figure 4.6: Run Results

The run time is dependent on your input data and your cycle time, the latter of which can be derived from the total cycle count displayed during the compile stage. Once the process is complete, the software will display the output of your test.

4.3. Views

The View menu in the software allow you to view the input network, analyzed network, simulation output, and logfile in different windows.

4.3.1. Input Network

The Input Network view displays a visualization of your input network, consisting of the layers, blobs, and connections in your network file.

4.3.2. Analyzed Network

The Analyzed Network View displays a visualization of your analyzed network. This is only available after the Analyze stage of project flow. In addition to its entry in the view menu, you can also click the “View Analyzed Network” button to the right of the “Run” button to bring up the display.

4.3.3. Simulation Output

The Simulation Output view will display the predicted values from your simulation run. This view is accessible after completing a software simulation.

4.3.4. Log File

The Log File view allows you to view the output log of your project. This is a history of operations you have initiated and the output that was generated as a result. If you would prefer to use a text viewer of your choice, the contents of your log file are stored in a .log file in your project directory.
5. **Advanced Topics**

5.1. **Project Implementation Settings**

Each project has several main settings for customizing your neural network’s implementation. These settings are accessed either during new project creation (see Chapter 3.1: Creating a New Project) or by editing an existing implementation (see chapter 4.1.2: Editing an Implementation). These settings are visible in the Project Implementation Window, as shown in Figure 6.1 below.

![Figure 5.1: Project Implementation Window](image)

#### 5.1.1. **Number of Convolution Engines**

You can change the number of convolution engines used by your design – whether they are standard convolution engines or binary convolution engines – to be less than the maximum amount supported on your device. The ability to use less than the maximum depends on the specific device. For example, certain LatticeECP5 products can support up to 8 CNN engines, allowing you to reduce the usage.

#### 5.1.2. **On-Chip Memory Block Size**

This is the size of the storage space used by each block implemented in the Embedded Block RAM. EBR is quicker to access than DRAM, and does not require as many swapping operations. The more of your data that is stored in the EBR instead of off-chip, the faster your design will run. Using less EBR storage can allow a network to fit in a smaller device by utilizing DRAM, at the cost of decreasing the frequency at which it runs.

#### 5.1.3. **Number of On-Chip Memory Blocks**

This setting specifies the number of discrete blocks in the EBR that are utilized in the DNN Inference Machine. You are required to have a minimum of one, plus an additional one for each Convolution Engine used by your design.
5.1.4. Off-Chip Data Memory Start Address
This setting determines the memory address in DRAM where the convolution design will start storing and loading data to. The amount of DRAM required depends on your neural network and your EBR settings, with larger networks or implementations with lower EBR usage requiring more DRAM.

5.1.4.1. Do Not Use
The Do Not Use option disables all DRAM usage. In addition to not storing the input or output in DRAM, it will also disable the ability to store data from intermediate stages in the DRAM. This mode may not be compatible with all networks.

5.1.4.2. Store Input
Enabling Store Input indicates that DRAM will be used for input rather than another source. Disabling this setting will prevent DRAM from being used to store input and you must have a way of providing input in your design.

5.1.4.3. Store Output
Functioning similarly to Store Input, enabling Store Output indicates that DRAM will be used for output rather than another source.

5.1.5. Mean Value for Data Pre-Processing
The Mean Value is used for normalizing input data. You must specify a value, or use the default. If you wish to use something other than the default, it must be specified in this setting, it is not inferred from your neural network files.

5.1.6. Scale Value for Data Pre-Processing
The Scale Value is used for scaling data values. You must specify a value, or use the default. If you wish to use a value other than the default Scale Value, it must be specified in this setting, it is not inferred from your neural network files.
6. Supported Frameworks

Currently Lattice SensAI Software supports both the Caffe and Tensorflow machine learning frameworks. Caffe protofiles are natively supported while Tensorflow requires creating a frozen deployment model file. Each supported framework is clearly defined in the appendix sections. These following sections explain how to customize or alter the neural network.

6.1. Caffe

Lattice SensAI Software supports Caffe natively. You can quickly import a Caffe neural network if you have the required files. You are required to provide a protofile (.proto), a caffemodel file (.caffemodel), and a reference data file (such as a .jpg image or .mp4 video file). For detailed information regarding the Caffe Framework, and in-depth explanations of features and limitations, please see Appendix A: Supported and Added Caffe Layers.

You must follow these requirements when creating your protofile:

- Do not include blobs intended for training purposes only, such as Accuracy or Loss.
- An Input layer with a clearly-defined input size must be present in the network.
- ReLU must be an in-place layer. Its top and bottom blobs must be the same.
- Every BatchNorm layer must have a Scale layer immediately following it.
- When using a kernel size of 3x3, pad and stride must both be 1.

In addition to the above requirements, you may find the following guidelines useful for protofile creation:

- The first blob should include the Input layer to indicate to the tool that it is the desired first blob and potentially improve runtime by reducing cycles required for operations.
- Mean and Scale are not read from the protofile. They must be specified in the tool itself, otherwise the default values will be used. The default Mean value is 128, and the default Scale value is 255.
- Use Scaling and BatchNorm layers every few layers to optimize performance due to the fixed point notation constraints of hardware.
- Use an input size that is a power of 2 for better computational speed and to minimize memory alignment issues.

6.1.1. Binary Neural Networks

The software utilizes a custom implementation of Caffe for incorporating Binary Neural Networks. The Binarize, BinaryInnerProduct, and BinaryConvolution layers are not supported in official Caffe releases, and cannot be trained using those distributions. You are required to use a version of Caffe that has been supplemented by these layers in order to train binary neural networks.

6.1.2. Supported Layers List

Finally, before creating your protofile, consult the list of supported layers. The following layers are supported in some capacity, please see Appendix A: Supported and Added Caffe Layers for specific features and limitations of each layer:

- BatchNorm
- Binarize
- BinaryInnerProduct
- BinaryConvolution
- Convolution
- Eltwise
- InnerProduct
- Input
- Pooling
- ReLU
- Scale
6.2. Tensorflow

Lattice SensAI Software is able to run designs made using the Tensorflow framework. This is done by using the provided tool for analyzing and converting Tensorflow neural networks into a compatible Caffe model internally. You are required to provide a Tensorflow inference frozen model file (.pb), which contains both graph and parameter values. The frozen .pb file requires both network topology and constant weights that are made for the purpose of inference. Please follow manual instructions on how to convert a training .pb model to an inference frozen .pb model. For detailed information regarding the Tensorflow Framework, please see Appendix B: Supported Tensorflow Operations.

You must follow these requirements when creating your Tensorflow inference frozen model file:

- Data preprocessing-related subgraphs and operations will be ignored. A separate script is required to preprocess input data, which can then be used directly as the input when testing your Tensorflow model in Lattice SensAI Software.
- Using a frozen model from a training session is not supported and cannot be used to create a compatible project.
- Data post-processing (such as softmax, sigmoid, etc.) is not supported. Supported output layers are: Conv2D, Inner Product, Full Connect, and Global Average Pool.

The following guidelines are not required, but strongly suggested:

- Call tf.reset_default_graph() immediately before initializing a new inference session. Within the inference session, only do inference-related Tensorflow operations, and then save the session graph definition as a .pb file.
- Mean and scale are not read from the pb. It must be specified in the tool itself, otherwise, the default values will be used. The default Mean value is 128, and the default Scale value is 255.
- Use Scaling and BatchNorm layers every few layers to optimize performance due to the fixed point notation constraints of hardware.
- Use an input size that is a power of 2 for better computational speed and to minimize memory alignment issues.

6.2.1. Conversion

There are three main steps in the process for converting a Tensorflow training model in the checkpoint directory into the supported Tensorflow inference frozen model, detailed below:

1. Identify the input and output nodes needed for inference.
2. Use tensorflow.python.tools.optimize_for_inference_lib.optimize_for_inference to remove nodes which are not related to inference and use tf.train.write_graph to save the output in binary .pb format.
3. Copy the output of step 2 (the simplified inference .pb) into the checkpoint folder and use tensorflow.python.tools.freeze_graph to freeze the checkpoint weight as Tensorflow inference frozen model file (.pb).

An example graph before and after inference optimization is shown in the following two figures:
6.1: Original Tensorflow Training Model

Figure 6.1 displays an example training model. This one is not yet frozen and has extraneous nodes.
6.2: Simplified Tensorflow Inference Model

After being converted, the example model resembles the one in Figure 6.2. This is the frozen model, with only inference nodes remaining.

6.2.2. Supported Operations List
The following Tensorflow operations are supported, full details for these operations are listed in Appendix B: Supported Tensorflow Operations.

- batch_normalization
- Conv2D
- Element wise vector Add
- Matmul
- Placeholder
- Pooling
- ReLU
7. References

The following documents provide more information on topics discussed in this guide:

- Lattice Diamond Software User Guide
- Lattice Radiant Software User Guide
8. Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.
Appendix A: Supported and Added Caffe Layers

This appendix is intended to provide information for all supported and added Caffe layers.

Accuracy

The Accuracy layer is not internally supported by the software, but can remain in your network file without causing an issue.

BatchNorm

The BatchNorm Caffe layer is supported for implementing batch normalization operations. You are required to put a Scale layer in your network after each BatchNorm layer, see Scale below for more information.

Binarize

Binarize fulfils the same purpose in binary neural networks as the ReLU layer in standard neural networks. The Binarize layer should be used in your binary neural networks instead of ReLU, because there is no need for that method of rectification to be used.

BinaryInnerProduct

BinaryInnerProduct calculates the inner product for a binary network, and should be used instead of the InnerProduct layer when dealing with binary neural networks.

BinaryConvolution

The BinaryConvolution layer is an added layer that functions similarly to the Convolution layer in Caffe, using binary weights and activations, and employing the same parameters. Your design must implement the BNN Accelerator IP to utilize this functionality, as the CNN Accelerator IP cannot perform binary convolution.

Convolution

Convolution is the layer type utilized by the CNN Accelerator IP for implementing convolution into your neural network, and users already familiar with Caffe can use it as they normally would without any major adjustments. Your design must implement the CNN Accelerator IP to utilize this functionality, as the BNN Accelerator IP cannot perform non-binary convolution. The group attribute is not fully supported, while the following parameters are supported by the CNN Accelerator IP for the Convolution layer:

- kernel_size
- num_output
- bias_term
- pad
- stride

Eltwise

The Eltwise layer currently supports only the SUM operation. Other operations such as MULT are not implemented.

InnerProduct

The num_output parameter is supported for specifying the number of filters. The bias_term parameter is supported for training purposes only. Inference uses the bias from training during compilation.
Input
The Input layer is supported, along with the shape parameter.

Pooling
Pooling layers are supported, while the Ave and Stochastic pooling are unable to be implemented. The Pooling layer supports the following Caffe parameters:

- MAX
- global_pooling
- kernel_size
- pad
- stride

Only square-shaped kernels are supported in the Pooling layer. The parameters kernel_h, kernel_w, stride_h, stride_w, pad_h, and pad_w are ignored.

ReLU
The ReLU layer is supported for rectifying values. It supports the negative_slope parameter, which is suggested to be between 0 and 0.25.

Scale
The Scale layer in Caffe is supported. You are required to put a Scale layer in your network after each BatchNorm layer.
Appendix B: Supported Tensorflow Operations
This appendix is intended to provide information for all supported Tensorflow operations.

Batch Normalization
Currently Rsqrt is the operation flag used to locate the batch normalization subgraph (a group of multiple operations), based on tf.nn.batch_normalization. Therefore, the software does not support the case where Rsqrt is used in the graph but not for batch normalization. If you do not use tf.nn.batch_normalization to create a batch normalization subgraph, then the batch normal subgraph should in same computation order and structured as shown below:

Figure B.1: Batch Normalization
Conv2D
The software only supports regular Conv2D. The Conv2D node is required to be the bias node's direct input in order to apply the bias to the Conv2D layer. Other convolution operations (such as dilated, depth wise, quantized convolution, etc.) are not supported.

Elementwise Vector Add
Elementwise Vector Add is only supported for residual net, with two tensor objects as the only input where the coefficients for each is 1. Other level elementwise operations (add mul div sub, max, etc.) are not supported.

Matmul
Matmul is used in regular fully connected/dense layers. Sparse, advanced transpose and adjoint mode are not supported. Unofficial operation (TF contribute, customized open source) implementation are not supported, such as tf.contrib.layers.fully_connected.

Placeholder
Support is limited to inputs with 3, 4 dimension shape where only one placeholder exists in graph. Preprocess operations on input are not supported. Expected input is a single image (gray or color without the need to be preprocessed). Group image and video are not supported.

Pooling
The software currently support two types of Pooling:
- Maxpool: tf.nn.max_pool
- Global Average Pooling: tf.reduce_mean

ReLU
Normal ReLU which is implemented by tf.nn.relu (slope =1 in positive region and slope =0 in negative region) is supported
For leaky ReLU (non-zero alpha slope in negative region), the software only supports implementations based on tf.nn.relu. For example: tf.nn.relu(x) - alpha * tf.nn.relu(-x).
# Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 2018</td>
<td>1.0</td>
<td>Initial release.</td>
</tr>
</tbody>
</table>