iCE40 UltraPlus Key Phrase Detection Quick Start Guide

Application Note

FPGA-AN-02008-1.0

October 2019

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 and with all faults, and all risk associated with such information is entirely with 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.

© 2019 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.

Contents

Figures

© 2019 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.

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
CKPT	Checkpoint
FPGA	Field-Programmable Gate Array

© 2019 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.

1. Introduction

This document provides a quick guide on how to train a machine and create a frozen file for the Lattice Machine Learning development using the iCE40 UltraPlus™ HIMAX HM01B0 Upduino Shield board. It assumes that the reader is familiar with the basic Lattice FPGA design flow and mainly focuses on the Machine Learning part of the overall development process. This document refers to the Key Phrase Detection Using Compact CNN Accelerator IP (FPGA-RD-02066) for the detailed steps of the design flow.

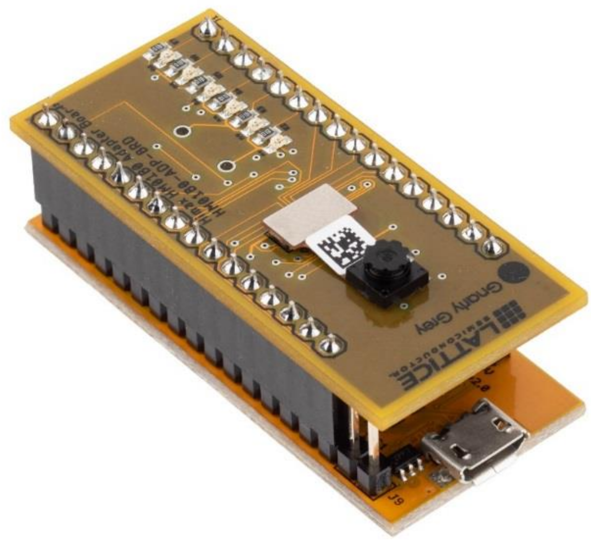

Figure 1.1. Himax HM01B0 Upduino Shield Board

1.1. Design Process Overview

The design process involves the following steps:

- Setting up the basic environment
- Preparing the dataset
- Training the machine
- Creating the frozen file (*.pb)
- Creating the binary file with sensAI™ 2.1 program
- Creating the FPGA bitstream file
- Programming the binary and bitstream files to iCE40 UltraPlus Upduino hardware

© 2019 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.

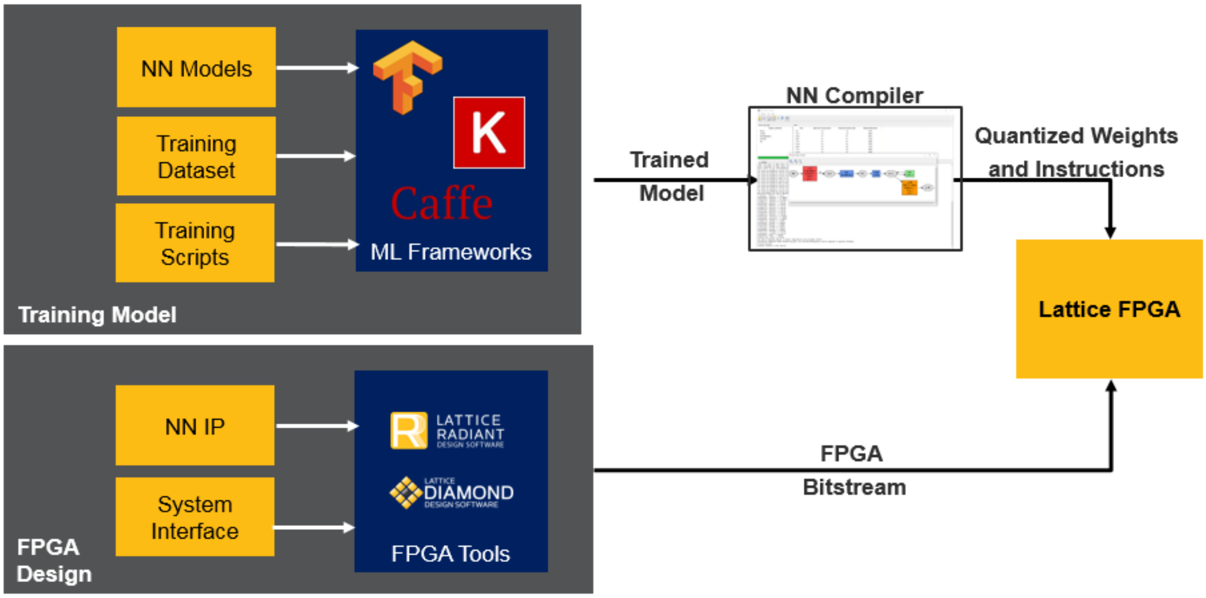

Figure 1.2. Lattice Machine Learning Design Flow

© 2019 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.

2. Machine Training and Creating Frozen File

2.1. Verifying TensorFlow and Tool Environment

Check if TensorFlow and your tool environment is installed correctly. For the detailed procedure in creating the basic environment on PC, refer to the Setting up the Basic Environment section in Key Phrase Detection Using Compact CNN Accelerator IP (FPGA-RD-02066).

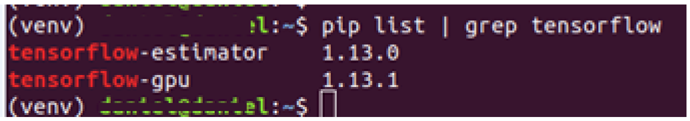

Figure 2.1. TensorFlow Installation Check

2.2. Preparing the Dataset

Below are the two dataset options for the key phrase detection models, which can be used for machine training.

- Google speech command dataset v0.01
- Google speech command dataset v0.02

To download a dataset:

1. Click the Google speech command dataset link.
   - Google speech command dataset v0.01: https://storage.cloud.google.com/download.tensorflow.org/data/speech_commands_v0.01.tar.gz
   - Google speech command dataset v0.02: https://storage.cloud.google.com/download.tensorflow.org/data/speech_commands_v0.02.tar.gz
2. The .wav files in the dataset have a sample rate of 16 kHz. Convert the sample rate to 8 kHz to align with the model input.
   a. Install necessary dependency:

   ```
   $ sudo apt-get install sox
   ```

   b. Convert the dataset:

   ```
   $python convert-samplerate.py
   ```

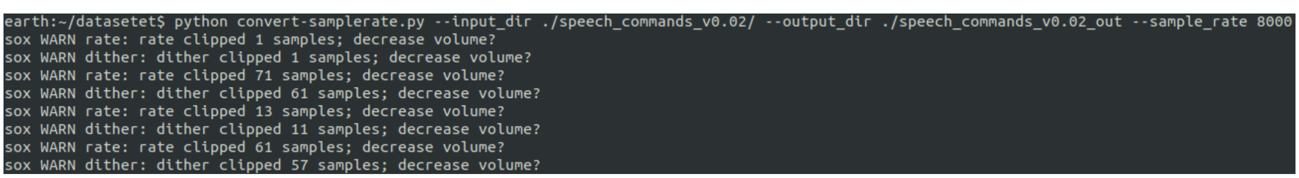

Figure 2.2. Convert Sample Rate Script Execution

© 2019 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.

2.3. Training the Machine

Machine training for the key phrase detection model consists of two phases.

- Phase 1 – Training with all keywords available in the dataset (Filter training).
- Phase 2 – Training for selected keywords using the checkpoint of the Phase 1 model (Keyword training).

To train the machine:

1. Modify training keywords and other configurations.
   To configure keywords or other parameters, modify *config.sh* under the training directory. The default parameter values for the Key Phrase Detection training are shown in Figure 2.3.

```
config.sh
#
# This file sets common params for key phrase training.
# Update log directory and dataset path if required.
#

# Update dataset path if needed
export DATA_DIR=./data/speech_commands_sAI2/

# Configure training log directory if needed
export FILTER_TRAIN_DIR=./logs/set8_seven.filter
export TRAIN_DIR=./logs/set_prefilter

export NETWORK=tinyvgg_conv
export TRAIN_OPT=--optimizer=Adam

# Network ID
export FILTER_NET_ID=cmd_seven.filter
export NET_ID=cmd_seven

# Keywords to train
export FILTER_TRAIN_KEYWORD="marvin,sheila,on,off,up,down,go,stop,left,right,yes,learn,visual,follow,\
no,cat,dog,bird,tree,house,bed,wow,happy,zero,one,two,three,four,five,six,seven,eight,nine,forward,backward"
export TRAIN_KEYWORD="seven,marvin,on,happy"

export COMMON_OPT="--model_architecture=$NETWORK \
--sample_rate=8000 \
--downsample=1 \
--no_prefilter_bias \
--clip_duration_ms=1040 \
--time_shift_ms=140.0 \
--window_size_ms=32.0 \
--window_stride_ms=16.0 \
--dct_coefficient_count=64 \
--background_volume=0.5"
```

Figure 2.3. Config.sh

- Update *DATA_DIR* with the path to root directory of speech commands dataset.
- Update FILTER_TRAIN_KEYWORD with all available keywords in the dataset.
- Update TRAIN_KEYWORD with all keywords that need to be trained by model.

You can also configure additional parameters in *train_filter.sh* and *train.sh* if needed.

© 2019 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.

2. Perform filter training.
   After configuring the parameters (only if required) in Figure 2.3, run the script to start filter training from scratch.

```
./train_filter.sh
```

```
earth:$ ./train_filter.sh
name: GeForce RTX 2080 Ti major: 7 minor: 5 memoryClockRate(GHz): 1.755
pciBusID: 0000:01:00.0
totalMemory: 10.73GiB freeMemory: 10.34GiB
2019-09-13 16:23:10.864151: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1512] Adding visible gpu devices: 0
[256, 1, 64]
[None, 8320, 1]
[None, 64, 64]
[None, 64, 64, 1]
INFO:tensorflow:Training from step: 1
2019-09-13 16:23:14.807664: I tensorflow/stream_executor/dso_loader.cc:152] successfully opened CUDA library libcublas.so.10.0 locally
INFO:tensorflow:Step #1: rate 0.010000, accuracy 3.0%, cross entropy 4.603645
INFO:tensorflow:Step #2: rate 0.010000, accuracy 7.0%, cross entropy 4.603001
INFO:tensorflow:Step #3: rate 0.010000, accuracy 5.0%, cross entropy 4.349138
INFO:tensorflow:Step #4: rate 0.010000, accuracy 7.0%, cross entropy 4.221001
INFO:tensorflow:Step #5: rate 0.010000, accuracy 5.0%, cross entropy 3.846268
INFO:tensorflow:Step #6: rate 0.010000, accuracy 7.0%, cross entropy 3.923666
INFO:tensorflow:Step #7: rate 0.010000, accuracy 5.0%, cross entropy 3.963864
```

Figure 2.4. Trigger Training with Default Options (Phase 1)

3. Perform keyword training.
   Update the path of the Phase 1 checkpoint in *train.sh* as shown below.

```
TRAIN_OPT = "$TRAIN_OPT –set_prefilter=<path_to_traindir/tinyvgg_conv.ckpt-50000> --lock_prefilter"
```

   After configuring (only if required) the parameters mentioned in Figure 2.3, run the script to start keyword training from scratch..

```
./train.sh
```

```
earth:$ ./train.sh
name: GeForce RTX 2080 Ti major: 7 minor: 5 memoryClockRate(GHz): 1.755
pciBusID: 0000:01:00.0
totalMemory: 10.73GiB freeMemory: 10.34GiB
2019-09-13 16:25:19.800218: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1512] Adding visible gpu devices: 0
[256, 1, 64]
[None, 8320, 1]
[None, 64, 64]
[None, 64, 64, 1]
INFO:tensorflow:Restoring parameters from ./logs/set8_seven.filter/tinyvgg_conv.ckpt-50000
INFO:tensorflow:Training from step: 1
2019-09-13 16:25:23.611075: I tensorflow/stream_executor/dso_loader.cc:152] successfully opened CUDA library libcublas.so.10.0 locally
INFO:tensorflow:Step #1: rate 0.010000, accuracy 8.0%, cross entropy 3.645020
INFO:tensorflow:Step #2: rate 0.010000, accuracy 35.0%, cross entropy 2.330117
INFO:tensorflow:Step #3: rate 0.010000, accuracy 48.0%, cross entropy 2.245769
INFO:tensorflow:Step #4: rate 0.010000, accuracy 46.0%, cross entropy 2.318952
INFO:tensorflow:Step #5: rate 0.010000, accuracy 45.0%, cross entropy 2.491566
INFO:tensorflow:Step #6: rate 0.010000, accuracy 48.0%, cross entropy 2.219422
INFO:tensorflow:Step #7: rate 0.010000, accuracy 43.0%, cross entropy 2.098553
INFO:tensorflow:Step #8: rate 0.010000, accuracy 33.0%, cross entropy 2.271026
INFO:tensorflow:Step #9: rate 0.010000, accuracy 39.0%, cross entropy 1.720827
```

Figure 2.5. Keyword Training from Scratch (Phase 2)

4. Perform transfer learning.
   For transfer learning, $FILTER_TRAIN_DIR and/or $TRAIN_DIR should point to the last iteration's log directory in *config.sh* and the latest checkpoint name should be updated in the training script. New checkpoints are stored at the path provided in $TRAIN_DIR and/or $FILTER_TRAIN_DIR.
   Modify the line below in *train_filter.sh* and *train.sh* to specify checkpoints from where the training should resume.

```
TRAIN_OPT="$TRAIN_OPT --start_checkpoint=$TRAIN_DIR/$NETWORK.ckpt-50000"
```

© 2019 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.

```
earth:$ ./train.sh
name: GeForce RTX 2080 Ti major: 7 minor: 5 memoryClockRate(GHz): 1.755
pciBusID: 0000:01:00.0
totalMemory: 10.73GiB freeMemory: 10.34GiB
2019-09-13 16:49:03.041945: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1512] Adding visible gpu devices: 0
[256, 1, 64]
[None, 8320, 1]
[None, 64, 64]
[None, 64, 64, 1]
INFO:tensorflow:Restoring parameters from ./logs/set_prefilter/tinyvgg_conv.ckpt-200
INFO:tensorflow:Training from step: 200
2019-09-13 16:49:07.087444: I tensorflow/stream_executor/dso_loader.cc:152] successfully opened CUDA library libcublas.so.10.0 locally
INFO:tensorflow:Step #200: rate 0.010000, accuracy 53.0%, cross entropy 1.226098
INFO:tensorflow:Saving to "./logs/set_prefilter/tinyvgg_conv.ckpt-200"
INFO:tensorflow:Step #201: rate 0.001000, accuracy 67.0%, cross entropy 1.009409
INFO:tensorflow:Step #202: rate 0.001000, accuracy 69.0%, cross entropy 0.885104
INFO:tensorflow:Step #203: rate 0.001000, accuracy 76.0%, cross entropy 0.848068
INFO:tensorflow:Step #204: rate 0.001000, accuracy 67.0%, cross entropy 0.932998
INFO:tensorflow:Step #205: rate 0.001000, accuracy 70.0%, cross entropy 0.847289
INFO:tensorflow:Step #206: rate 0.001000, accuracy 58.0%, cross entropy 1.115180
```

Figure 2.6. Trigger Training with Transfer Learning

5. Start TensorBoard by running the command below.

```
$ tensorboard –logdir=<log directory of training>
```

```
earth:~$ tensorboard --logdir logs/set8_seven/
TensorBoard 1.12.0 at http://earth:6006 (Press CTRL+C to quit)
```

Figure 2.7. TensorBoard – Generated Link

6. Check the training status on TensorBoard.

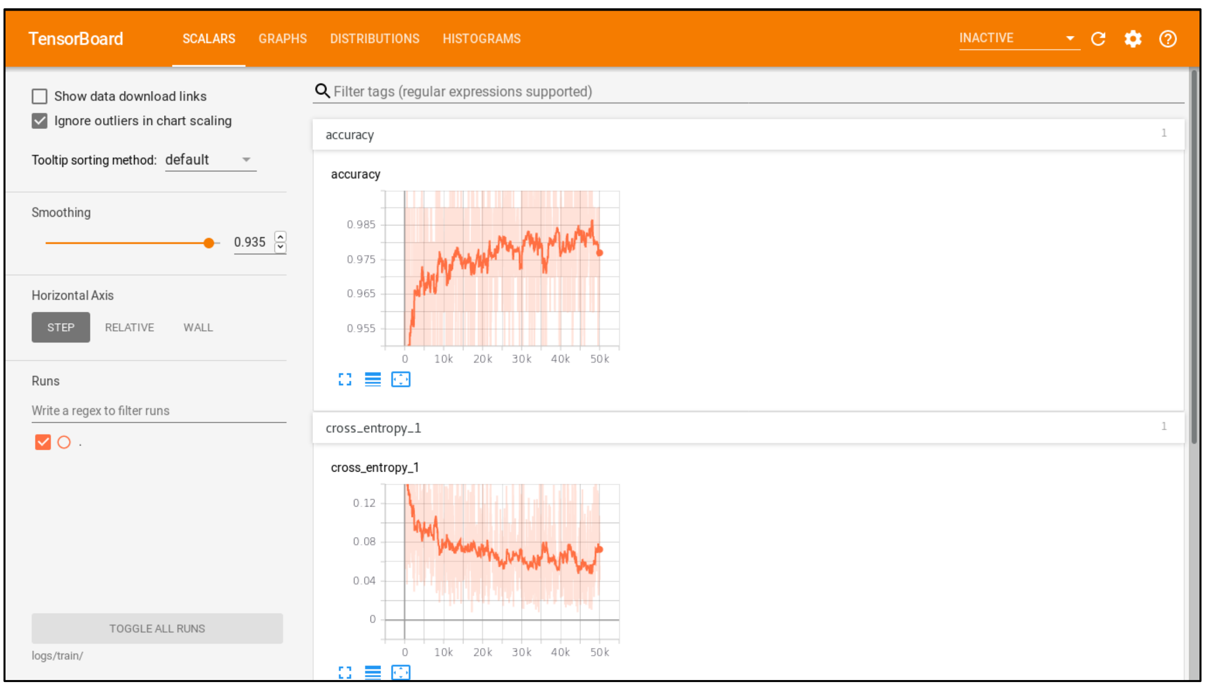

Figure 2.8. TensorBoard

© 2019 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.

7. Check if the checkpoint, data, meta, index, and events (if using TensorBoard) files are created at the log directory. These files are used for creating the frozen file (*.pb).

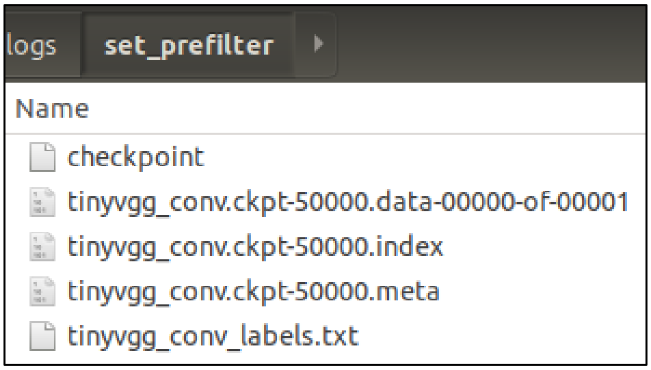

Figure 2.9. Checkpoint Data Files at Log Folder

© 2019 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.

2.4. Generating *.pbtxt File

To generate the *.pbtxt file:

1. Run the command below to generate the .pbtxt file.
   Note: Do not modify config.sh after training.

```
$ ./genpbtxt.sh
```

```
earth:$ ./genpbtxt.sh
[256, 1, 64]
[None, 8320, 1]
[None, 64, 64]
[None, 64, 64, 1]
saved pbtxt for inference at log direcory:./logs/set_prefilter
```

Figure 2.10. Create *.pbtxt File

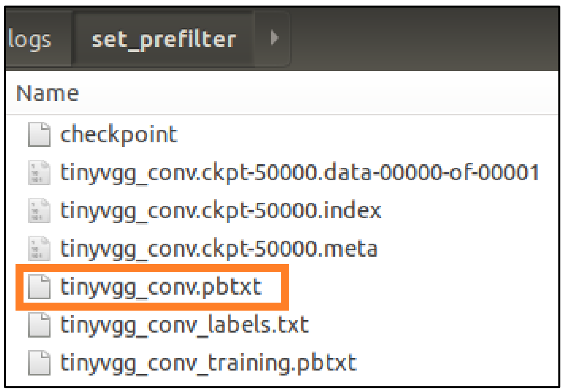

Figure 2.11. Generated .pbtxt for Inference

The .pbtxt for inference is generated under train log directory.

2.5. Generating the Frozen (.pb) File

To generate Frozen file (*.pb):

1. Run genpb.py.

```
$ python genpb.py --ckpt_dir <COMPLETE_PATH_TO_LOG_DIRECTORY>
```

```
earth:$ python genpb.py --ckpt_dir logs/set_prefilter/
using checkpoint :tinyvgg_conv.ckpt-500.meta
inputShape shape [None, None, None, None]
inputShape shapes [None, None, None, None]
output_shapes of input Node [None, None, None, None]
 **TensorFlow**: can not locate input shape information at: audio_input
node to modify name: "audio_input"
```

Figure 2.12. Create *.pb File

genpb.py uses .pbtxt and the latest checkpoint in train directory to generate the frozen .pb file.
Once the genpb.py is executed successfully, the log directory shows *ckpt-prefix>_frozenforinference.pb* file as shown in Figure 2.13.

© 2019 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.

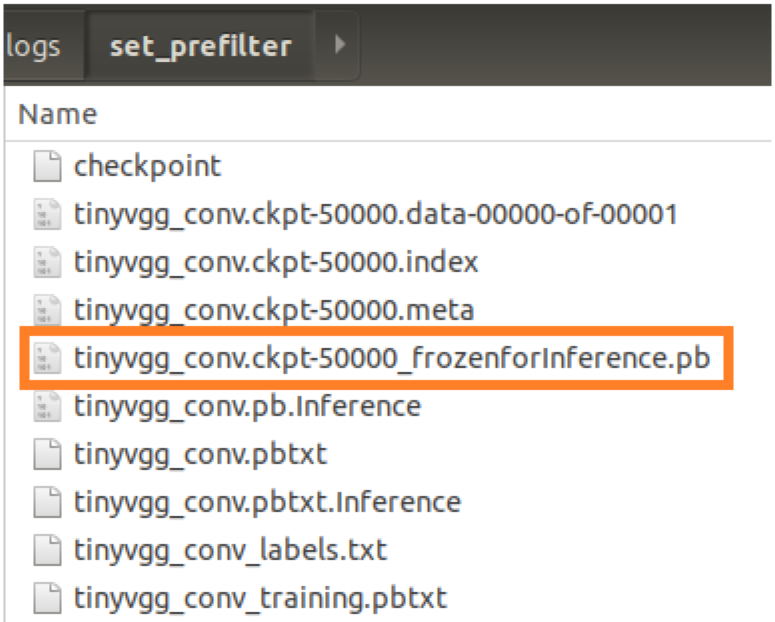

Figure 2.13. Check Frozen File

© 2019 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.

3. Generating the Binary File

For the detailed procedure in creating the binary file, refer to the Creating Binary File with sensAI section in Key Phrase Detection Using Compact CNN Accelerator IP (FPGA-RD-02066).

4. Programming the Bitstream and Binary Files to HIMAX HM01B0 Upduino Shield Board

For the detailed procedure in flashing the bitstream and binary files to the iCE40 HIMAX HM01B0 Upduino Shield board, refer to the Programming the Key Phrase Detection Demo section in Key Phrase Detection Using Compact CNN Accelerator IP (FPGA-RD-02066).

© 2019 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.

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

© 2019 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.

Revision History

Revision 1.0, October 2019

Section	Change Summary
All	Initial release.

© 2019 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.

www.latticesemi.com