iCE40 UltraPlus Human Presence Detection Quick Start Guide

Application Note

FPGA-AN-02005-2.1

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.

Contents

Figures

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
CKPT	Checkpoint
FPGA	Field-Programmable Gate Array

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 Human Presence Detection Using Compact CNN Reference Design (FPGA-RD-02059) for the detailed steps of the design flow.

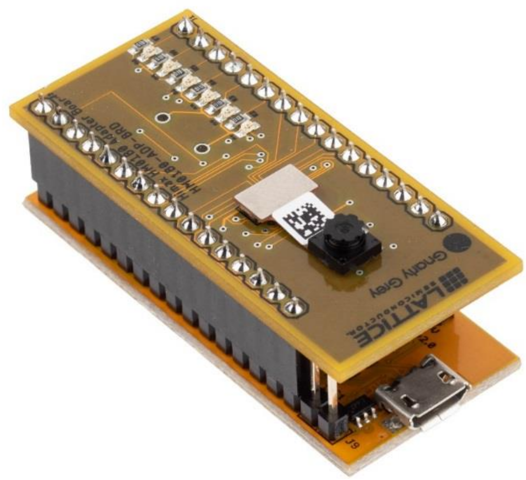

Figure 1.1. Himax HM01B0 Upduino Shield Board

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 Upduino hardware

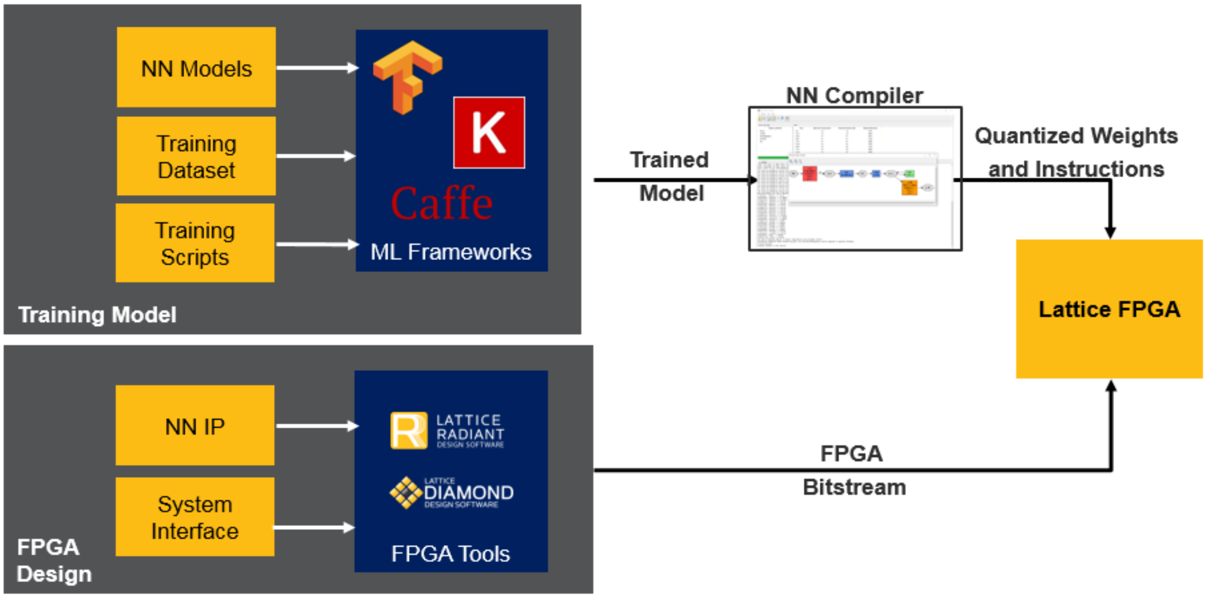

Figure 1.2. Lattice Machine Learning Design Flow

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 Human Presence Detection Using Compact CNN Reference Design (FPGA-RD-02059).

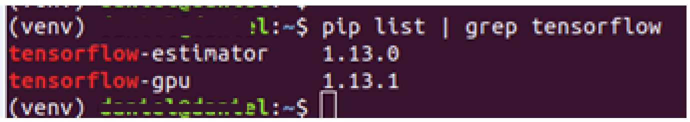

Figure 2.1. Tensorflow Installation Check

2.2. Preparing the Dataset

Prepare the image and label data (KITTI format). The image size for this design is 64 x 64 pixels.

For the detailed procedure in preparing the dataset, refer to the Preparing the Dataset section in Human Presence Detection Using Compact CNN Reference Design (FPGA-RD-02059).

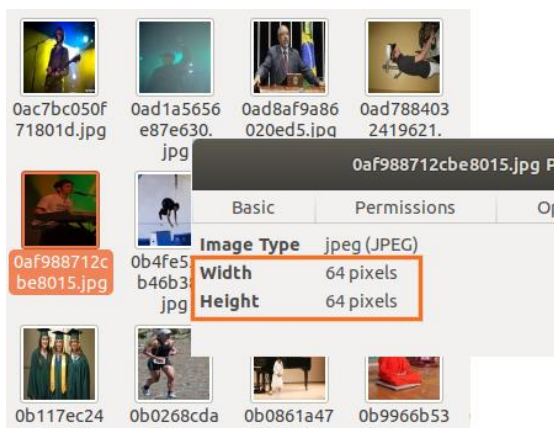

Figure 2.2. Dataset Image Size Check

2.3. Training the Machine

For the detailed procedure in machine training, refer to the Training the Machine section in Human Presence Detection Using Compact CNN Reference Design (FPGA-RD-02059).

To train the machine:

1. Check the training dataset path in the training script file *train.sh*.

```
export GPUID=0
export NET="squeezeDet"
export TRAIN_DIR="./logs/humandet/"

export TRAIN_DATA_DIR="./data/humandet"
```

Figure 2.3. Dataset Folder Path Check

The subdataset path can be set in the training code *@src/dataset/kitti.py* and can be used in combination with the --*data_path* option while triggering training using *train.py* to get the desired path. For example, you can have *<data_path>/training/images* and *<data_path>/training/labels*.

```
def __init__(self, image_set, data_path, mc):
  imdb.__init__(self, 'kitti_'+image_set, mc)
  self._image_set = image_set
  self._data_root_path = data_path
  self._image_path = os.path.join(self._data_root_path, 'training', 'images')
  self._label_path = os.path.join(self._data_root_path, 'training', 'labels')
  self._classes = self.mc.CLASS_NAMES
```

Figure 2.4. Dataset List, Image, and Label Data Path

2. Create a train.txt file.

```
$ cd data/humandet/
$ python dataset_create.py
```

```
k$ python dataset_create.py
k$ _
```

Figure 2.5. Create a Label File

3. Run machine training. In the command prompt, execute the *./run* command.

```
k$ ./run
Using TensorFlow backend.
self.preds: Tensor("conv12/convolution:0", shape=(20, 4, 4, 42), dtype=float32, device=/device:GPU:0)
ANCHOR_PER_GRID: 7
CLASSES: 1
preds2: Tensor("interpret_output/strided_slice:0", shape=(20, 4, 4, 7), dtype=float32, device=/device:GPU:0)
ANCHORS: 112
```

Figure 2.6. Execute the Script

4. Start TensorBoard.

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

For example: tensorboard –logdir='./logs/'

5. Open the local host port on your web browser.

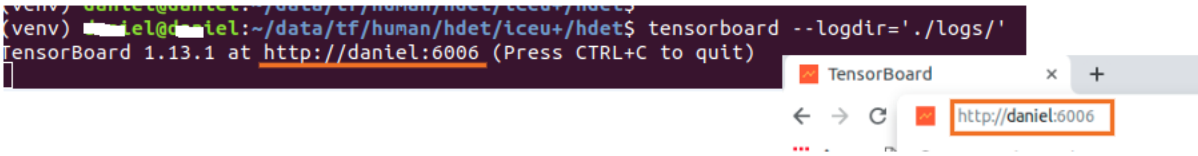

Figure 2.7. TensorBoard – Generated Link

6. Check the training status on TensorBoard

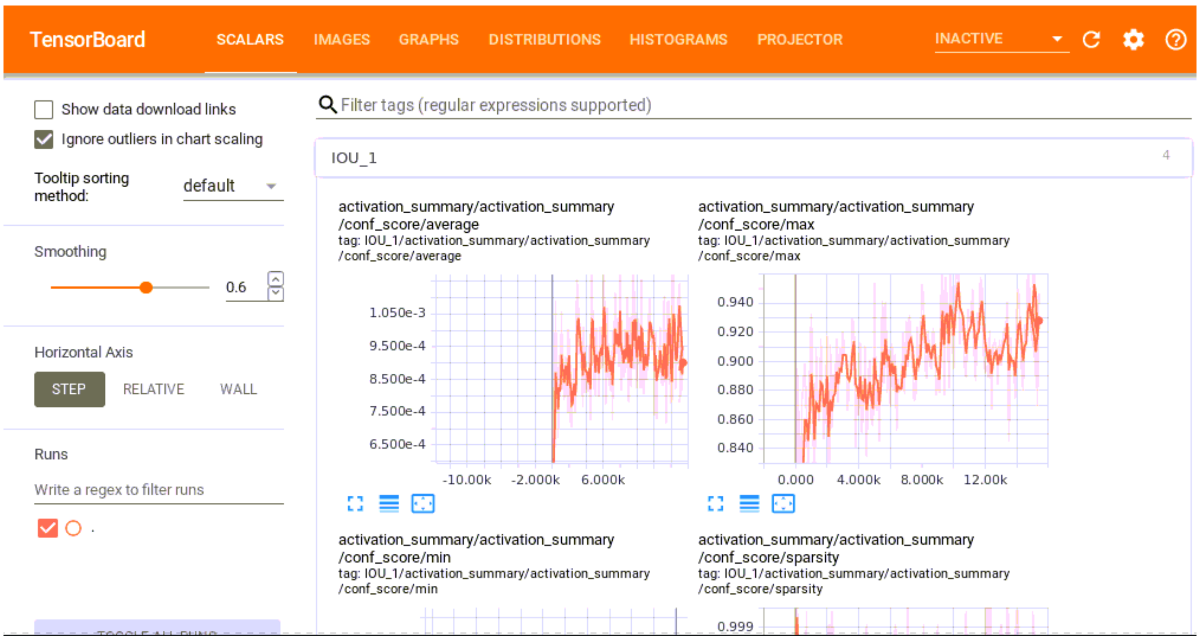

Figure 2.8. TensorBoard Training Status

Figure 2.9 shows the image menu of TensorBoard.

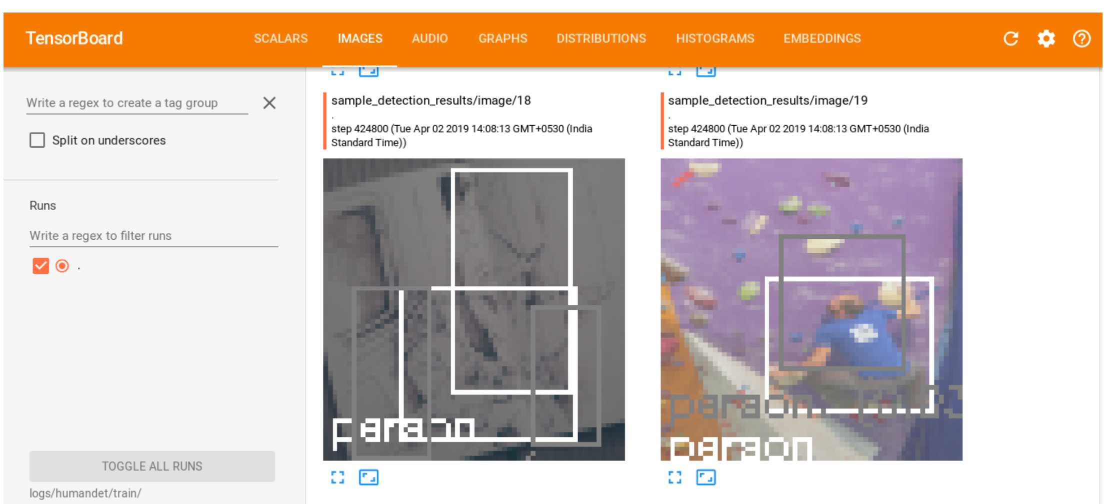

Figure 2.9. TensorBoard Image Menu

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

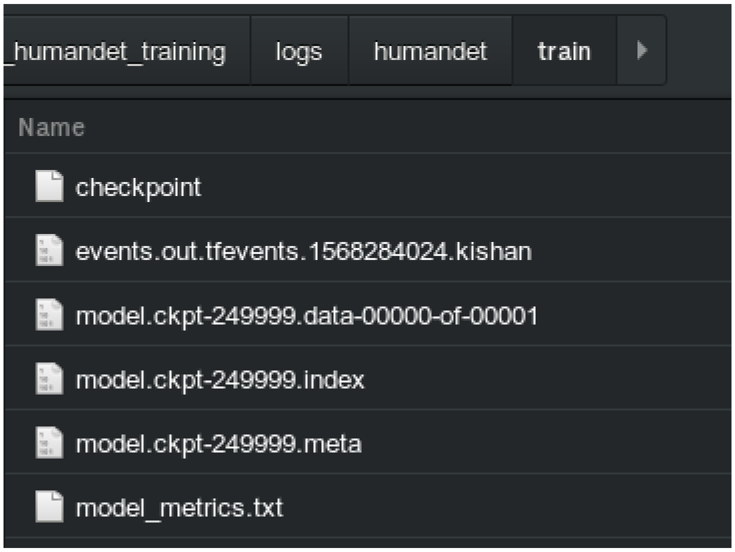

Figure 2.11 Checkpoint Data Files at Log Folder

2.4. Generating Frozen (*.pb) File

To generate Frozen file(*.pb):

1. Run genpb.py.

```
$ python src/genpb.py –ckpt_dir <log directory> --freeze
```

For example, python src/genpb.py –ckpt_dir './logs/humandet/train/' –freeze.

```
earth:~/human_presence-2.1$ python src/genpb.py --ckpt_dir=logs/humandet/train/ --freeze
genrating pbtxt
self.preds: Tensor("conv12/convolution:0", shape=(20, 4, 4, 42), dtype=float32, device=/device:GPU:0)
ANCHOR_PER_GRID: 7
CLASSES: 1
preds2: Tensor("interpret_output/strided_slice:0", shape=(20, 4, 4, 7), dtype=float32, device=/device:GPU:0)
ANCHORS: 112
Using checkpoint: ./model.ckpt-249999
saved pbtxt at checkpoint direcory Path
('inputShape shape', [1, 64L, 64L, 3L])
```

Figure 2.10. Create *.pb File

2. Verify and find the generated frozen file.

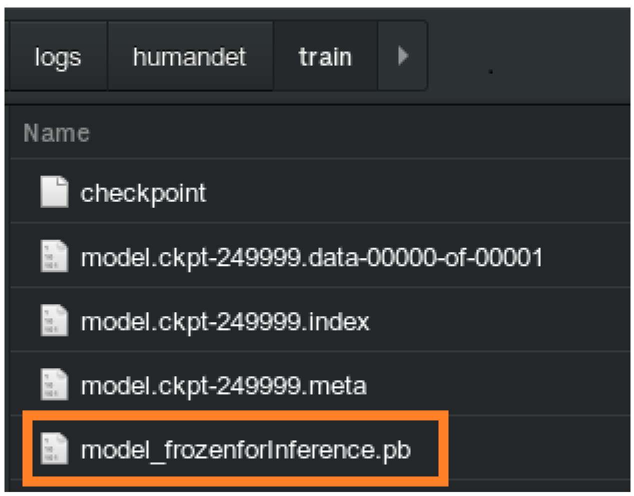

Figure 2.11. Check Frozen File

3. Generating the Binary File

For the detailed procedure in creating the binary file, refer to the Creating Binary File with SensAI section in Human Presence Detection Using Compact CNN Reference Design (FPGA-RD-xxxx).

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 Running the iCE40 Human Presence Detection Demo section in Human Presence Detection Using Compact CNN Reference Design (FPGA-RD-02059).

Technical Support Assistance

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

Revision History

Revision 2.1, October 2019

Section	Change Summary
—	Changed document title to iCE40 UltraPlus Human Presence Detection Quick Start Guide.
Introduction	Updated senseAI version to 2.1.
Machine Training and Creating Frozen File	Updated procedure steps.

Revision 1.1, April 2019

Section	Change Summary
All	Corrected link to Human Presence Detection Using Compact CNN Reference Design (FPGA-RD-02059).

Revision 1.0, May 2019

Section	Change Summary
All	Initial release.

www.latticesemi.com