Getting Started Guide for the Movidius™ Neural Compute Stick

<July 2017>
The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting: http://www.intel.com/design/literature.htm

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at http://www.intel.com/ or from the OEM or retailer.

Intel, Movidius and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2017, Intel Corporation. All rights reserved.
Contents

1.0 Quick Start for the Movidius™ Neural Compute Stick..................................................5
  1.1 Hardware and Software........................................................................................................6
    1.1.1 Connecting the Movidius NCS to a Development Computer........................................6
    1.1.2 Downloading the Movidius™ NC SDK.........................................................................7
  1.2 Install and Verify the Movidius Neural Compute (NC) Toolkit........................................10
  1.3 Install and Verify the Movidius NC API............................................................................12
  1.4 Using the Movidius™ NC Toolkit........................................................................................13
  1.5 Using the Movidius NC API................................................................................................17

2.0 Appendix A: Toolkit Utilities and Examples ......................................................................20
  2.1 Utilities..................................................................................................................................20
    2.1.1 Movidius™ Neural Compute compiler..........................................................................20
    2.1.2 Movidius™ Neural Compute checker............................................................................20
    2.1.3 Movidius™ Neural Compute profiler............................................................................21
  2.2 Examples................................................................................................................................21
    2.2.1 top_5_over_a_dataset.....................................................................................................21

3.0 Appendix B: API Examples.................................................................................................22
  3.1 C Examples..............................................................................................................................22
    3.1.1 ncs-check.......................................................................................................................22
    3.1.2 ncs-threadcheck.............................................................................................................23
    3.1.3 ncs-fullcheck................................................................................................................23
  3.2 Python3 Examples ..................................................................................................................24
    3.2.1 ncs_camera....................................................................................................................24
    3.2.2 stream_infer..................................................................................................................25
    3.2.3 age_gender_classification.............................................................................................25
    3.2.4 age_gender_example......................................................................................................26
## Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7/13/2017</td>
<td>0.5</td>
<td>Initial version covering SDK RC2b2 release</td>
</tr>
<tr>
<td>7/18/2017</td>
<td>0.9</td>
<td>Updates for SDK RC3b2</td>
</tr>
<tr>
<td>7/19/2017</td>
<td>1.0</td>
<td>Updates for SDK Gold release 1.07.06</td>
</tr>
</tbody>
</table>

§
1.0 **Quick Start for the Movidius™ Neural Compute Stick**

This guide is provided as a starting point for using the Movidius™ Neural Compute Stick (NCS). It will help you get up and running with the hardware and software and guide you through some example scenarios to make sure your installation and configuration were successful.

Users are expected to use the NCS in two scenarios:

- **Profiling, tuning, and compiling a CNN on a development computer (host system) with the Movidius™ Neural Compute (NC) Toolkit** software package, while tuning and validating a Convolutional Neural Network (CNN). In this scenario the host system is typically a desktop or laptop machine.

- **Prototyping a user application on a development computer (host system) which accesses the hardware of the NCS to accelerate CNN inferences. The Movidius™ Neural Compute (NC) API** software package provides the programming interface to enable these user applications. In this scenario the host system may be an embedded system or any developer machine.

The following diagram shows the typical workflow for development with the NCS. In this diagram you will see a training phase for which the NCS and associated software will not be used. The subsequent phases of “profiling, tuning and compiling” and “prototyping” do make use of the NCS and its accompanying software, the Movidius™ Neural Compute SDK.
Using the NCS in conjunction with the following scenarios ("profiling, tuning, and compiling" and "prototyping") will be covered in this guide.

1.1 Hardware and Software

The hardware and software setup required to use the Movidius™ NCS on a development computer is covered in this section.

This guide assumes that the following versions of hardware and software are available and used for the two configurations mentioned above:

- Movidius™ Neural Compute Stick
- Development computer with an available USB 2.0 port running Ubuntu 16.04 LTS
- USB Camera for some samples
- The latest Movidius™ Neural Compute (NC) SDK software package which contains:
  - The latest Movidius™ NC Toolkit
  - The latest Movidius™ NC API

1.1.1 Connecting the Movidius NCS to a Development Computer

The NCS connects to the development computer over a USB 2.0 High Speed interface. Plug the NCS directly to a USB port on your development computer or into a powered USB hub that is plugged into your development computer.
Figure: Connecting the Movidius™ NCS

The Movidius™ NCS plugged into a development computer.

1.1.2 Downloading the Movidius™ NC SDK

The Movidius™ NC SDK provides the software tools, development libraries, and samples that enable the NCS. To download and install the SDK you will need an internet connection and a development computer with Ubuntu 16.04 installed. The SDK includes the Movidius™ NC API and Movidius™ NC Toolkit. The SDK is distributed via the Movidius™ NCS User Forum ([https://ncsforum.movidius.com](https://ncsforum.movidius.com)).

You can get the latest release here:
You may want to create an account and get involved with the Movidius™ developer community to share ideas but creating an account is not required to download and install the SDK. If you want to create an account click the sign in button and then the "Don't have an account? Create one" link.
When you have navigated to the forum post linked above, click on the link to go to the specific post for the latest version. The SDK will be available for download as an attachment to the post. The SDK will come as a single compressed archive file. When downloading from the forum the file name may be changed by the browser unless you right click and save as. In this screenshot below it is named \textit{iuy3rj448jzo.tgz}

After the archive is downloaded, copy it to a directory of your choice such as \textasciitilde/ncsdk. Next expand the archive with the tar command below. This will create two new archive files, one for the toolkit and one for the API.

- $ cp \textasciitilde/Downloads/iuy3rj448jzo.tgz \textasciitilde/ncsdk
- $ cd \textasciitilde/ncsdk
- tar -xvf <downloaded sdk file>
1.2 Install and Verify the Movidius Neural Compute (NC) Toolkit

The Movidius™ NC Toolkit provides tools to enable rapid tuning, validation and profiling of Convolutional Neural Networks. The Toolkit also includes a tool to compile CNNs to binary graph files that the Neural Compute Stick can load and execute from within a user’s software application.

Before you install the Toolkit you should update your system with these commands

- $ sudo apt-get update
- $ sudo apt-get upgrade

After the SDK has been downloaded and extracted, the Toolkit archive will be created. Follow the steps below to install the Toolkit. Note that your development host computer will need to be connected to the internet for the installation to work properly.

- Expand the Toolkit archive with:
  $ tar –xvf <MvNC_Toolkit file name>
After the toolkit has been extracted run the following commands to install it.

- $ cd bin
- $ ./setup.sh (This may take 15 minutes or more)
- The setup.sh script will prompt for a password and also prompt you for a location to install Caffe.
- After Toolkit setup.sh has finished be sure to open a new terminal session if you haven’t already. This is also indicated at the end of the setup.sh output as shown below.
In the new terminal execute the following command to verify that your $PYTHONPATH includes the python directory inside your caffe installation directory. Output should be similar to the following:

```bash
$ echo $PYTHONPATH
:/opt/Movidius/caffe/python:
```

If your $PYTHONPATH is not set, run the following command:

```bash
$ source ~/.bashrc
```

```bash
$ cd <toolkit bin dir>/data
```

The command below downloads sample caffe models for the example code:

```bash
$ ./dlnets.sh
```

### 1.3 Install and Verify the Movidius NC API

The Movidius™ NC API provides user software applications with an Application Programming Interface to take advantage of the Movidius™ NCS features.

After the SDK has been downloaded and extracted, the API archive will be created. Follow the steps below to install the API.

**Note:** Your development host computer will need to be connected to the internet for the installation to work properly.
Note: The Toolkit must be setup prior to the API.

- Expand the API archive with this tar command to expand the ncap directory.

```
$ tar --xvf <API archive file name>
```

- After the API archive has been expanded follow the steps below to install it.
  - `$ cd ncap`
  - `$ ./setup.sh` (this will take a few minutes to finish.)

### 1.4 Using the Movidius™ NC Toolkit

Now that the Toolkit is installed you have access to the three main tools it provides:

- `mvNCProfile.pyc`, and `mvNCCheck.pyc`, `mvNCC Compile.pyc`,

Follow the steps below to try out these tools.

- Plug in the Movidius™ NCS to an available USB port on host system.
- `$ cd <toolkit directory>/bin`
- `$ make example00`
  
  This will compile a prototxt file into a graph file that applications built with the API can load onto the NCS for hardware accelerated inferences.
You should see the following output:

```
$ make example01
```

This will create `bin/output_report.html` with profile details for the GoogleNet as well as the following output on the terminal.

- $ make example01
  This will create `bin/output_report.html` with profile details for the GoogleNet as well as the following output on the terminal.
Similar to example01, this will create a profile report for another network (lenet8.) You should see the output below and get a new

<table>
<thead>
<tr>
<th>Layer</th>
<th>MFLOPs</th>
<th>Bandwidth MB/s</th>
<th>time(ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv1/7x7_s2</td>
<td>236.028</td>
<td>2588.67</td>
<td>5.62</td>
</tr>
<tr>
<td>pool1/3x3_s2</td>
<td>1.886</td>
<td>1440.52</td>
<td>1.96</td>
</tr>
<tr>
<td>pool1/1</td>
<td>0.680</td>
<td>714.97</td>
<td>0.54</td>
</tr>
<tr>
<td>conv2/3x3</td>
<td>25.690</td>
<td>465.32</td>
<td>0.66</td>
</tr>
<tr>
<td>conv2/3x3</td>
<td>693.633</td>
<td>315.43</td>
<td>11.59</td>
</tr>
<tr>
<td>conv2/norm2</td>
<td>0.000</td>
<td>795.01</td>
<td>1.44</td>
</tr>
<tr>
<td>pool2/3x3_s2</td>
<td>1.355</td>
<td>1493.75</td>
<td>0.77</td>
</tr>
<tr>
<td>inception3b/1x1</td>
<td>19.268</td>
<td>462.52</td>
<td>0.67</td>
</tr>
<tr>
<td>inception3b/3x3</td>
<td>28.981</td>
<td>399.90</td>
<td>0.81</td>
</tr>
<tr>
<td>inception3b/3x3</td>
<td>173.408</td>
<td>324.83</td>
<td>4.63</td>
</tr>
<tr>
<td>inception3b/3x5</td>
<td>4.017</td>
<td>791.58</td>
<td>0.17</td>
</tr>
<tr>
<td>inception3b/5x5</td>
<td>20.670</td>
<td>852.28</td>
<td>0.73</td>
</tr>
<tr>
<td>inception3b/5x5</td>
<td>1.355</td>
<td>655.40</td>
<td>0.42</td>
</tr>
<tr>
<td>inception3b/5x5</td>
<td>9.034</td>
<td>536.64</td>
<td>0.54</td>
</tr>
<tr>
<td>inception3b/5x5</td>
<td>51.388</td>
<td>486.30</td>
<td>0.95</td>
</tr>
<tr>
<td>inception3b/5x5</td>
<td>136.817</td>
<td>271.67</td>
<td>7.92</td>
</tr>
<tr>
<td>inception3b/5x5</td>
<td>12.845</td>
<td>1096.07</td>
<td>0.36</td>
</tr>
<tr>
<td>inception3b/5x5</td>
<td>120.422</td>
<td>586.66</td>
<td>2.12</td>
</tr>
<tr>
<td>inception3b/pool</td>
<td>1.086</td>
<td>899.19</td>
<td>6.65</td>
</tr>
<tr>
<td>inception3b/pool</td>
<td>25.690</td>
<td>682.73</td>
<td>0.61</td>
</tr>
<tr>
<td>pool3/3x3_s2</td>
<td>0.847</td>
<td>1307.68</td>
<td>0.55</td>
</tr>
<tr>
<td>inception4a/1x1</td>
<td>36.127</td>
<td>386.54</td>
<td>0.91</td>
</tr>
<tr>
<td>inception4a/3x3</td>
<td>18.063</td>
<td>573.13</td>
<td>8.47</td>
</tr>
<tr>
<td>inception4a/3x3</td>
<td>70.467</td>
<td>322.64</td>
<td>2.07</td>
</tr>
<tr>
<td>inception4a/5x5</td>
<td>3.011</td>
<td>1032.97</td>
<td>0.19</td>
</tr>
<tr>
<td>inception4a/5x5</td>
<td>7.526</td>
<td>610.02</td>
<td>0.31</td>
</tr>
<tr>
<td>inception4a/5x5</td>
<td>0.847</td>
<td>632.88</td>
<td>0.31</td>
</tr>
<tr>
<td>inception4a/5x5</td>
<td>12.042</td>
<td>662.47</td>
<td>0.36</td>
</tr>
<tr>
<td>inception4a/5x5</td>
<td>32.113</td>
<td>287.77</td>
<td>1.21</td>
</tr>
<tr>
<td>inception4b/1x1</td>
<td>22.479</td>
<td>377.30</td>
<td>0.80</td>
</tr>
<tr>
<td>inception4b/3x3</td>
<td>80.510</td>
<td>339.01</td>
<td>2.63</td>
</tr>
<tr>
<td>inception4b/3x3</td>
<td>4.087</td>
<td>614.93</td>
<td>0.26</td>
</tr>
<tr>
<td>inception4b/5x5</td>
<td>15.053</td>
<td>374.02</td>
<td>0.80</td>
</tr>
<tr>
<td>inception4b/5x5</td>
<td>0.983</td>
<td>614.64</td>
<td>0.31</td>
</tr>
<tr>
<td>inception4b/5x5</td>
<td>12.845</td>
<td>552.18</td>
<td>0.40</td>
</tr>
</tbody>
</table>
$ make example03
This is checking for the correctness of network results for the lenet8 network
You should see the following output:
• For more information on the tools in the Toolkit, take a look at the NCS Toolkit Documentation which you can download from the Movidius™ NCS user forum here [https://ncsforum.movidius.com](https://ncsforum.movidius.com).

• Try the other examples and options shown in the Toolkit Documentation.

### 1.5 Using the Movidius NC API

Now that you have installed the API, follow these steps to explore the API.

• Plug in the Movidius™ NCS to an available USB port on a host system.

• `$ cd <ncapi directory>/c_examples`

• `$ make`

• `./ncs-check -1`

• You should see output similar to this screenshot

![Screenshot of Movidius NC API output](image)

• Make sure you have a camera plugged into your developer computer (a built-in laptop camera is probably fine.)

• `$ cd <ncapi directory>

• Look in the `<sdk directory>/ncapi/networks/SqueezeNet` directory and verify that the following files exist there:
  
  - `categories.txt`
  - `graph`
If these files don’t exist then run the following commands:

$ cd <ncapi directory>/tools

$ ./get_models.sh

$ ./convert_models.sh

$ cd <ncapi directory>/py_examples/stream_infer

$ python3 ./stream_infer.py

You will see the example running as in the following screen shot.

Place different items in front of the camera to see if they are recognized.

Take a look at the readme.html file in the stream_infer directory to get an understanding of how the sample works.

Read through the API User Guide which you can download from the Movidius User Forum at https://ncsforum.movidius.com and take a look at the other
examples provided with the API. See appendix B for more information on API examples.
Appendix A: Toolkit Utilities and Examples

All Toolkit utilities and samples must be executed from the Toolkit's bin directory.

To run the included examples and utilities you will need to have specific networks available in the Toolkit bin/data directory. The dlnets.sh script in that directory will download these networks.

Run the following command from the Toolkit bin/data directory prior to running Toolkit utilities and examples:

- $ dlnets.sh

Utilities

2.1.1 Movidius™ Neural Compute compiler

The compiler is used to create an optimized binary graph file for the NCS.

Usage:
- $ python3 ./mvNCCompile.pyc <network.prototxt> [--help] [-w <weights file>] [-s <number of shaves] [-in <input node name>] [-on <output node name>] [-is <image width> <image height>] [-o <path>]

See the Toolkit documentation for more usage details.

Example command:
- $ python3 ./mvNCCompile.pyc ./data/lenet8.prototxt -w ./data/lenet8.caffemodel -s 12 -o ./lenet8_graph

2.1.2 Movidius™ Neural Compute checker

The checker runs a single inference on the NCS, allowing for the calculation of classification correctness.

A sample data set can be found in the bin/examples/data/test directory.

Usage:
- $ python3 ./mvNCCheck.pyc <network.prototxt> [--help] [-w <weights file>] [-s <number of shaves] [-in <input node name>] [-on <output node name>] [-is <image width> <image height>] [-i <image>] [-id <expected id>] [-S <scale factor>] [-M <number or numpy mean file>]
See the Toolkit documentation for more usage details.

Example command:
- $ python3 ./mvNCCheck.pyc ./data/lenet8.prototxt -w ./data/lenet8.caffemodel -s 12 -i ./examples/data/test/5/00015.png -id 5 -S 255 -M ./data/imagenet_mean.npy

2.1.3 Movidius™ Neural Compute profiler
The profiler provides a detailed stage-by-stage breakdown of network performance.

Usage:
- $ python3 ./mvNCProfile.pyc <network.prototxt> [--help] [-w <weights file>] [-s <number of shaves] [-in <input node name>] [-on <output node name>] [-is <image width> <image height>]

See the Movidius NC Toolkit documentation for more usage details.

Example command:
- $ python3 ./mvNCProfile.pyc ./data/lenet8.prototxt -w ./data/lenet8.caffemodel -s 12

2.2 Examples

2.2.1 top_5_over_a_dataset
This is an example user script for calculating the top-5 accuracy of a network over a given dataset.

Usage:
- $ python3 ./examples/top_5_over_a_dataset.py [<data directory>] [<network.prototxt>] [<weights file>]

Example commands:
- $ python3 ./examples/top5_over_a_dataset.py

- $ python3 ./examples/top_5_over_a_dataset.py ./examples/data/test ./data/lenet8.prototxt ./data/lenet8.caffemodel
3.0 Appendix B: API Examples

3.1 C Examples

C examples must be built before they can be run. To build these examples, enter the following command in the ncapi/c_examples directory:

- $ make

3.1.1 ncs-check

This example opens the device, allocates a graph, sends some random data representing an input, and gets the result. When it has done this <count> number of times, it deallocates the graph and closes the device.

Usage:

- $ ncs-check [-l<loglevel>] -1 (try one device)
- $ ncs-check [-l<loglevel>] -2 (try two devices)
- $ ncs-check [-l<loglevel>] [-c<count>] <network directory>

The <loglevel> can be 0 for no log output (default), 1 for errors only, or 2 for verbose log output.

The <count> is the number of times to run inferences (default 2).

The <network directory> should be a directory containing graph, stat.txt, categories.txt, and inputsize.txt. Sample networks can be found in the ncapi/networks directory.

Example commands:

These example commands must be called from the ncapi/c_examples directory.

- $ ./ncs-check -l
- $ ./ncs-check -l2 -c3 ../networks/SqueezeNet
3.1.2  ncs-threadcheck

This example executes the same tasks as ncs-check but uses a threaded approach.

Usage:

- $ ncs-threadcheck [-l<loglevel>] [-c<count>] <network directory>

The <loglevel> can be 0 for no log output (default), 1 for errors only, or 2 for verbose log output.

The <count> is the number of times to run inferences (default 2).

The <network directory> should be a directory containing graph, stat.txt, categories.txt, and inputsize.txt. Sample networks can be found in the API ncapi/networks directory.

Example command:

This example command must be called from the ncapi/c_examples directory.

- $ ./ncs-threadcheck ../networks/SqueezeNet

3.1.3  ncs-fullcheck

This example executes the same tasks as ncs-check but requires an input image.

Usage:

- $ ncs-fullcheck [-l<loglevel>] [-c<count>] <network directory> <image>

The <loglevel> can be 0 for no log output (default), 1 for errors only, or 2 for verbose log output.

The <count> is the number of times to run inferences (default 2).

The <network directory> should be a directory containing graph, stat.txt, categories.txt, and inputsize.txt. Sample networks can be found in the API ncapi/networks directory.
The `<image>` is a path to an input image. Sample images can be found in the bin/images directory.

**Example command:**

This example command must be called from the ncapi/c_examples directory.

- $ ./ncs-fullcheck ../networks/SqueezeNet ../images/512_Ball.jpg

### 3.2 Python3 Examples

#### 3.2.1 ncs_camera

This script performs inferences on streaming video using GStreamer.

**Usage:**


- `<V4L2 source>` is the v4l2 source device name (e.g. /dev/video0) (default: /dev/video0)

- `<video source>` is the video source (default: None).

- `<picture source>` is the filename of an input picture (default: None).

- `<network directory>` is a directory containing the graph, categories.txt, inputsize.txt and stat.txt files (default: ../networks/GoogLeNet).

- `<device>` is the name of the NCS device to use for inference (default: 1).

- `<loglevel>` is the API logging level (0 = none, 1 = errors, 2 = verbose) (default: 0).

The verbose option (--v, --verbose) prints out additional information (default: False).
The opengl option (\texttt{--opengl}) will cause this script to use OpenGL instead of Xv extension for preview (default: False)

Example commands:

These example commands must be called from the \texttt{ncapi/python_examples} directory.

- $ python3 ./ncs_camera.py
- $ python3 ./ncs_camera.py -g ../networks/SqueezeNet -v

\section*{3.2.2 \texttt{stream_infer}}

This script performs inference on streaming video using GStreamer. See the readme in the \texttt{ncapi/py_examples/stream_infer} directory for more detailed usage information and a code walkthrough.

Usage:

- $ python3 stream_infer.py

Example command:

This example command must be called from the \texttt{ncapi/py_examples/stream_infer} directory.

- $ python3 ./stream_infer.py

\section*{3.2.3 \texttt{age_gender_classification}}

This script performs inference on a sample image using either the Age or the Gender network. The included sample image is located at \texttt{ncapi/py_examples/image.jpg}.

This script requires OpenCV for Python. To easily install OpenCV Python bindings, you can use the \texttt{install_opencv.sh} script found in the \texttt{ncapi/py_examples} directory.

Usage:

- $ python3 age_gender_classification.py <1 | 2>

Use 1 for Age or 2 for Gender.
Example commands:

These commands must be called from the ncapi/py_examples directory.

- $ python3 ./age_gender_classification.py 1
- $ python3 ./age_gender_classification.py 2

3.2.4 age_gender_example

This script performs inference on a sample image using either the GoogLeNet, AlexNet, or SqueezeNet network. The included sample image is located at ncapi/images/cat.jpg.

This script requires OpenCV for Python. To easily install OpenCV Python bindings, you can use the install_opencv.sh script found in the ncapi/py_examples directory.

Usage:

- $ python3 classification_example.py <1 | 2 | 3>

Use 1 for GoogLeNet, 2 for AlexNet, or 3 for SqueezeNet

Example commands:

These commands must be called from the ncapi/py_examples directory.

- $ python3 ./classification_example.py 1
- $ python3 ./classification_example.py 2
- $ python3 ./classification_example.py 3