Setting Up TensorFlow With OpenCL Using SYCL

Posted on March 30, 2017 by Luke Iwanski.

> This blog post is out of date, a guide to using TensorFlow with ComputeCpp is available **on our website here** that explains how to get set up and start using SYCL.

Intro

This short post aims to guide through set-up process for TensorFlow with OpenCL support. It's a "copy-paste" type of post. There is nothing ground breaking in it, all the instructions can be found across other websites / forums.

The aim was to put all relevant information in one place which should make it more convenient/easier to go through the set-up process.

Requirements

- ComputeCpp CE 0.1.2
- Python 2.7
- Ubuntu 14.04.05 LTS Headless Server
- AMD R9 Nano GPU

Dependencies

The assumption is that you have a vanilla server with 14.04.05 LTS Ubuntu64 installed.

Install Ubuntu kernel 3.19.0-79-generic using:

Install Git, LLVM and Build Essential using:

```
$ sudo apt-get install build-essential git llvm
```

Install JDK8 & Bazel using:

```
$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
$ sudo apt-get install oracle-java8-installer
$ echo "deb [arch=amd64] http://storage.googleapis.com/bazel-apt
stable jdk1.8" | sudo tee /etc/apt/sources.list.d/bazel.list
$ curl https://bazel.build/bazel-release.pub.gpg | sudo apt-key
add -
$ sudo apt-get update && sudo apt-get install bazel
$ sudo apt-get upgrade bazel
```

Install the AMD OpenCL 1.2 headers using:

```
$ sudo apt-get install ocl-icd-opencl-dev opencl-headers
$ wget --referer=http://support.amd.com
https://www2.ati.com/drivers/linux/fglrx_15.302-
0ubuntu1_amd64_ub_14.01.deb
$ wget --referer=http://support.amd.com
https://www2.ati.com/drivers/linux/fglrx-core_15.302-
0ubuntu1_amd64_ub_14.01.deb
$ sudo dpkg -i fglrx-core_15.302-0ubuntu1_amd64_ub_14.01.deb
fglrx_15.302-0ubuntu1_amd64_ub_14.01.deb
$ sudo adduser $(whoami) video
$ sudo reboot
```

Install ComputeCpp CE 0.1.2:

Visit **https://computecpp.codeplay.com**, register and download Ubuntu 14.04 release, unzip and copy the content – bin, include and lib are essential – to /usr/local/computecpp (you will need sudo to do it)

Install Numpty using:

```
$ sudo apt-get install python-numpy python-dev python-wheel
python-mock python-psutil
$ sudo pip install --upgrade pip
$ sudo pip install py-cpuinfo portpicker numpy
$ sudo pip install --upgrade scipy
```

```
$ export COMPUTECPP_TOOLKIT_PATH=/usr/local/computecpp
$ export COMPUTE=:0
```

You might consider adding the above to the ~/.bashrc and apply them to the current terminal session by:

```
$ . ~/.bashrc
```

Verification

At this point you should be able to verify your set up.

```
$ clinfo
```

Should have returned something along these lines:

```
Number of platforms: 1
Platform Profile: FULL_PROFILE
Platform Version: OpenCL 2.0 AMD-APP (1912.5)
Platform Name: AMD Accelerated Parallel Processing
Platform Vendor: Advanced Micro Devices, Inc.
Platform Extensions: cl_khr_icd cl_amd_event_callback
cl_amd_offline_devices
Platform Name: AMD Accelerated Parallel Processing
Number of devices: 2
Device Type: CL_DEVICE_TYPE_GPU
Vendor ID: 1002h
Board name:
Device Topology: PCI[ B#1, D#0, F#0 ]
Max compute units: 64
Max work items dimensions: 3
Max work items[0]: 256
Max work items[1]: 256
Max work items[2]: 256
Max work group size: 256
Preferred vector width char: 4
Preferred vector width short: 2
Preferred vector width int: 1
Preferred vector width long: 1
Preferred vector width float: 1
Preferred vector width double: 1
Native vector width char: 4
Native vector width short: 2
Native vector width int: 1
Native vector width long: 1
Native vector width float: 1
```

```
Max number of images write arguments: 64
Max image 2D width: 16384
Max image 2D height: 16384
Max image 3D width: 2048
Max image 3D height: 2048
Max image 3D depth: 2048
Max samplers within kernel: 16
Max size of kernel argument: 1024
Alignment (bits) of base address: 2048
Minimum alignment (bytes) for any datatype: 128
Single precision floating point capability
Denorms: No
Quiet NaNs: Yes
Round to nearest even: Yes
Round to zero: Yes
Round to +ve and infinity: Yes
IEEE754-2008 fused multiply-add: Yes
Cache type: Read/Write
Cache line size: 64
Cache size: 16384
Global memory size: 4242251136
Constant buffer size: 65536
Max number of constant args: 8
Local memory type: Scratchpad
Local memory size: 32768
Max pipe arguments: 16
Max pipe active reservations: 16
Max pipe packet size: 3002336256
Max global variable size: 2702102528
Max global variable preferred total size: 4242251136
Max read/write image args: 64
Max on device events: 1024
Queue on device max size: 8388608
Max on device queues: 1
Queue on device preferred size: 262144
SVM capabilities:
Coarse grain buffer: Yes
Fine grain buffer: Yes
Fine grain system: No
Atomics: No
Preferred platform atomic alignment: 0
Preferred global atomic alignment: 0
Preferred local atomic alignment: 0
Kernel Preferred work group size multiple: 64
Error correction support: 0
Unified memory for Host and Device: 0
Profiling timer resolution: 1
Device endianess: Little
Available: Yes
```

```
Profiling : Yes
Queue on Device properties:
Out-of-Order: Yes
Profiling : Yes
Platform ID: 0x7f8a310c4a18
Name: Fiji
Vendor: Advanced Micro Devices, Inc.
Device OpenCL C version: OpenCL C 2.0
Driver version: 1912.5 (VM)
Profile: FULL_PROFILE
Version: OpenCL 2.0 AMD-APP (1912.5)
Extensions: cl_khr_fp64 cl_amd_fp64
cl_khr_global_int32_base_atomics
cl_khr_global_int32_extended_atomics
cl_khr_local_int32_base_atomics
cl_khr_local_int32_extended_atomics cl_khr_int64_base_atomics
cl_khr_int64_extended_atomics cl_khr_3d_image_writes
cl_khr_byte_addressable_store cl_khr_gl_sharing
cl_khr_gl_depth_images cl_ext_atomic_counters_32
cl_amd_device_attribute_query cl_amd_vec3 cl_amd_printf
cl_amd_media_ops cl_amd_media_ops2 cl_amd_popcnt
cl_khr_image2d_from_buffer cl_khr_spir cl_khr_subgroups
cl_khr_gl_event cl_khr_depth_images cl_khr_mipmap_image
cl_khr_mipmap_image_writes
[...]
```

The above output means that there is an OpenCL platform with devices – GPU (The R9 Nano) and the CPU (most likely) available

Call computecpp_info which is located in the bin folder of the ComputeCpp installation:

```
$ /usr/local/computecpp/bin/computecpp_info
```

The output should look something like thisL

```
********************************************************************************
ComputeCpp Info (CE 0.1.2)
********************************************************************************
Toolchain information:
GLIBCXX: 20150426
This version of libstdc++ is supported.
********************************************************************************
Device Info:
Discovered 2 devices matching:
platform :
device type :
```

```
CL_DEVICE_VENDOR : Advanced Micro Devices, Inc.
CL_DRIVER_VERSION : 1912.5 (VM)
CL_DEVICE_TYPE : CL_DEVICE_TYPE_GPU
--------------------------------------------------------------------------------
Device 1:
Device is supported : YES - Tested internally by Codeplay Software Ltd.
CL_DEVICE_NAME : Intel(R) Core(TM) i7-2600K CPU @ 3.40GHz
CL_DEVICE_VENDOR : Intel(R) Corporation
CL_DRIVER_VERSION : 1.2.0.25
CL_DEVICE_TYPE : CL_DEVICE_TYPE_CPU
********************************************************************************
```

At this point, we have verified that we have SYCL 1.2/OpenCL 1.2 supported device(s) available in our system.

TensorFlow

Finally, lets get TensorFlow, there are three places worth consideration:

Official upstream (stable branch):

```
$ git clone https://github.com/tensorflow/tensorflow.git
```

Development branch – Benoit Steiner fork OpenCL hub:

```
$ git clone https://github.com/benoitsteiner/tensorflow-opencl.git
```

Experimental branch – Mine for high pace experimental:

```
$ git clone https://github.com/lukeiwanski/tensorflow-opencl.git
```

```
$ cd tensorflow-opencl
$ ./configure
```

Configure is an interactive bash script that sets-up environment for TensorFlow.

If you exported the variables mentioned in “Set Up” section you should be able to press enter for all the options. Otherwise you will have to answer few questions:

```
Please specify the location of python. [Default is
/usr/bin/python]:
Do you wish to build TensorFlow with Google Cloud Platform
support? [y/N]
```

```
/usr/lib/python2.7/dist-packages
Please input the desired Python library path to use. Default is
[/usr/local/lib/python2.7/dist-packages]
Using python library path: /usr/local/lib/python2.7/dist-
packages
Do you wish to build TensorFlow with OpenCL support? [y/N] y
OpenCL support will be enabled for TensorFlow
Do you wish to build TensorFlow with CUDA support? [y/N]
No CUDA support will be enabled for TensorFlow
Please specify which C++ compiler should be used as the host C++
compiler. [Default is /usr/bin/clang++-3.6 ]:
/usr/bin/clang++-3.6
Please specify which C compiler should be used as the host C
compiler. [Default is /usr/bin/clang-3.6 ]: /usr/bin/clang-3.6
Please specify the location where ComputeCpp for SYCL 1.2 is
installed. [Default is /usr/local/computecpp]:
```

After that is finished you can either run TensorFlow testsuite (recommended):

```
$ bazel test --config=sycl -k --test_timeout 1600 --
//tensorflow/... -//tensorflow/contrib/...
-//tensorflow/java/... -//tensorflow/compiler/...
```

or, build pip package (pip package):

```
$ bazel build --local_resources 2048,.5,1.0 -c opt --config=sycl
//tensorflow/tools/pip_package:build_pip_package
```

```
$ mkdir _python_build
$ cd _python_build
$ ln -s ../bazel-
bin/tensorflow/tools/pip_package/build_pip_package.runfiles/org_t
 .
$ ln -s ../tensorflow/tools/pip_package/* .
```

Keep in mind OpenCL support at this point is experimental and at an early stage in development.

Next Steps

TensorFlow introduction:
https://www.tensorflow.org/get_started/basic_usage
OpenCL support status:
https://github.com/tensorflow/tensorflow/issues/22

Similar (4)

Cookie Privacy Policy

Please note that by closing this message dialog you are agreeing to our privacy and cookie usage policy.

Codeplay are attending SC18, Dallas 2018

Codeplay are attending DemoFest: Bringing Research to Life, Edinburgh 2018

Codeplay are attending LLVM, San Jose 2018

By Author (4)

Setting Up TensorFlow With OpenCL Using SYCL

Bringing the Acceleration of OpenCL to TensorFlow with SYCL

TensorFlow™ for OpenCL™ using SYCL™

Codeplay is attending Edinburgh Centre for Robotics Annual Conference