OS Platform and Distribution

- Ubuntu 16.04 LTS on norco EMB-7502 board
- kernel 3.14
- arm-linux-gnueabihf

TensorFlow installed from

- git https://github.com/tensorflow

TensorFlow version

- v1.11.0

Bazel version

- bazel release 0.18.0

GCC version 5.4.0

Python version 3.6.5

## Step 1.1: Preprocessing for Bazel (Build tool for Tensorflow)

### Before we build Bazel, run following commands

```
sudo apt-get install cmake pkg-config zip g++ zlib1g-dev unzip default-jdk autoconf
automake libtool

# For Python 2.7
sudo apt-get install python-pip python-numpy swig python-dev
sudo pip install wheel

# For Python 3.3+
sudo apt-get install python3-pip python3-numpy swig python3-dev
sudo pip3 install wheel
```

### Install JDK 8

```
sudo apt install openjdk-8-jdk
```

Check java and javac version.

```
root@norco-desktop:/gc100# javac -version
javac 1.8.0_191
root@norco-desktop:/gc100# java -version
java version "1.8.0_191"
Java(TM) SE Runtime Environment (build 1.8.0_191-b12)
Java HotSpot(TM) Client VM (build 25.191-b12, mixed mode)
```

## Step 1.2: Adding swap memory

Creat swap iso file

```
dd if=/dev/zero of=/gc100/2048Mb.swap bs=1M count=2048
```

Format swap iso file

```
mkswap /gc100/2048Mb.swap
```

Mount swap  file

```
swapon /gc100/2048Mb.swap
```

## Step 1.3: Get Bazel

```
wget https://github.com/bazelbuild/bazel/releases/download/0.18.0/bazel-0.18.0-dist.zip
unzip -d bazel bazel-0.18.0-dist.zip
```

Once it's done downloading and extracting, we can move into the directory to make a few changes:

```
cd bazel
sudo chmod u+w ./* -R
```

We need to update Java heap size before building bazel. Edit scripts/bootstrap/compile.sh or By setting environment variables

scripts/bootstrap/compile.sh

```
nano scripts/bootstrap/compile.sh
```

Update

```
run "${JAVAC}" -classpath "${classpath}" -sourcepath "${sourcepath}" \
      -d "${output}/classes" -source "$JAVA_VERSION" -target "$JAVA_VERSION" \
```

```
      -encoding UTF-8 "@${paramfile}"
```

To

```
  run "${JAVAC}" -classpath "${classpath}" -sourcepath "${sourcepath}" \
      -d "${output}/classes" -source "$JAVA_VERSION" -target "$JAVA_VERSION" \
      -encoding UTF-8 "@${paramfile}" -J-Xmx500M
```

Save file and exit.

Or

By setting environment variables

```
export BAZEL_JAVAC_OPTS="-J-Xmx1g -J-Xms200m"
```

## Step 1.4: Build Bazel (~2 Hours)

```
./compile.sh
```

After build finishes, run following command

```
ln -sf out/bazel /bin/bazel
```

Run

```
bazel
```

To make sure it's working properly, run `bazel` on the command line and verify it prints help text. Note: this may take 5-10 seconds to run, so be patient!

```
root@norco-desktop:/gc100# bazel
WARNING: --batch mode is deprecated. Please instead explicitly shut down your Bazel
server using the command "bazel shutdown".
                                                [bazel release 0.18.0- (@non-git)]
Usage: bazel <command> <options> ...

Available commands:
  analyze-profile     Analyzes build profile data.
  aquery              Analyzes the given targets and queries the action graph.
  build               Builds the specified targets.
  canonicalize-flags  Canonicalizes a list of bazel options.
```

```
  clean               Removes output files and optionally stops the server.
  coverage            Generates code coverage report for specified test targets.
  cquery              Loads, analyzes, and queries the specified targets w/
configurations.
  dump                Dumps the internal state of the bazel server process.
  fetch               Fetches external repositories that are prerequisites to the
targets.
  help                Prints help for commands, or the index.
  info                Displays runtime info about the bazel server.
  license             Prints the license of this software.
  mobile-install      Installs targets to mobile devices.
  print_action        Prints the command line args for compiling a file.
  query               Executes a dependency graph query.
  run                 Runs the specified target.
  shutdown            Stops the bazel server.
  sync                Syncs all repositories specifed in the workspace file
  test                Builds and runs the specified test targets.
  version             Prints version information for bazel.

Getting more help:
  bazel help <command>
                  Prints help and options for <command>.
  bazel help startup_options
                  Options for the JVM hosting bazel.
  bazel help target-syntax
                  Explains the syntax for specifying targets.
  bazel help info-keys
                  Displays a list of keys used by the info command.
```

Now come out of bazel direcotry

```
cd ..
```

# Step 2: Tensorflow r1.11 (Python 3.6)

## 2.1 Let's install some pre-requisites

```
sudo apt-get install python3-numpy python3-dev python3-pip python3-mock
```

```
sudo apt-get install libhdf5-serial-dev
sudo apt-get install libhdf5-dev
sudo pip3 install keras_applications==1.0.6 --no-deps
sudo pip3 install keras_preprocessing==1.0.5 --no-deps
sudo pip3 install h5py==2.8.0
sudo apt-get install -y openmpi-bin libopenmpi-dev
sudo apt install git
```

## 2.2 Clone Tensorflow and checkout r1.11

```
git clone https://github.com/tensorflow/tensorflow/releases/tag/v1.11.0
cd tensorflow
git checkout r1.11
```

## 2.3 Config bazel

```
./configure
```

```
odroid@odroid:~/Desktop/tensorflow$ ./configure
Extracting Bazel installation...
WARNING: --batch mode is deprecated. Please instead explicitly shut down your Bazel server using the command "bazel sh
You have bazel 0.15.0- (@non-git) installed.
Please specify the location of python. [Default is /usr/bin/python]: /usr/bin/python3


Found possible Python library paths:
  /usr/lib/python3/dist-packages
  /usr/local/lib/python3.6/dist-packages
Please input the desired Python library path to use.  Default is [/usr/lib/python3/dist-packages]

Do you wish to build TensorFlow with jemalloc as malloc support? [Y/n]: n
No jemalloc as malloc support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Google Cloud Platform support? [Y/n]: n
No Google Cloud Platform support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Hadoop File System support? [Y/n]: n
No Hadoop File System support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Amazon AWS Platform support? [Y/n]: n
No Amazon AWS Platform support will be enabled for TensorFlow.

Do you wish to build TensorFlow with Apache Kafka Platform support? [Y/n]: n
No Apache Kafka Platform support will be enabled for TensorFlow.

Do you wish to build TensorFlow with XLA JIT support? [y/N]: n
No XLA JIT support will be enabled for TensorFlow.

Do you wish to build TensorFlow with GDR support? [y/N]: n
No GDR support will be enabled for TensorFlow.

Do you wish to build TensorFlow with VERBS support? [y/N]: n
No VERBS support will be enabled for TensorFlow.

Do you wish to build TensorFlow with nGraph support? [y/N]: n
No nGraph support will be enabled for TensorFlow.

Do you wish to build TensorFlow with OpenCL SYCL support? [y/N]: n
No OpenCL SYCL support will be enabled for TensorFlow.

Do you wish to build TensorFlow with CUDA support? [y/N]: n
No CUDA support will be enabled for TensorFlow.

Do you wish to download a fresh release of clang? (Experimental) [y/N]: n
Clang will not be downloaded.

Do you wish to build TensorFlow with MPI support? [y/N]: n
No MPI support will be enabled for TensorFlow.

Please specify optimization flags to use during compilation when bazel option "--config=opt" is specified [Default is


Would you like to interactively configure ./WORKSPACE for Android builds? [y/N]: n
Not configuring the WORKSPACE for Android builds.

Preconfigured Bazel build configs. You can use any of the below by adding "--config=<>" to your build command. See too
        --config=mkl            # Build with MKL support.
        --config=monolithic     # Config for mostly static monolithic build.
Configuration finished
```

## 2.4 Let's start building. It will take really long time. (~ 5-8 hours)

```
bazel build -c opt --copt=-march=armv7-a --copt=-mfpu=vfpv3-d16 --copt=-mfloat-abi=hard --copt=-DS_IREAD=S_IRUSR --copt=-DS_IWRITE=S_IWUSR --copt=-U__GCC_HAVE_SYNC_COMPARE_AND_SWAP_1 --copt=-U__GCC_HAVE_SYNC_COMPARE_AND_SWAP_2 --copt=-U__GCC_HAVE_SYNC_COMPARE_AND_SWAP_8 --config=monolithic --copt=-funsafe-math-
```

```
optimizations --copt=-ftree-vectorize --copt=-fomit-frame-pointer --verbose_failures
//tensorflow/tools/pip_package:build_pip_package
```

### After it finishes

```
Target //tensorflow/tools/pip_package:build_pip_package up-to-date:
  bazel-bin/tensorflow/tools/pip_package/build_pip_package
INFO: Elapsed time: 38434.091s, Critical Path: 12645.60s
INFO: 4471 processes: 4471 local.
INFO: Build completed successfully, 4825 total actions
```

## 2.5 Let's create wheel file

```
bazel-bin/tensorflow/tools/pip_package/build_pip_package /gc100/tensorflow_pkg
```

## 2.6 Install it

```
sudo pip3 install /gc100/tensorflow_pkg/tensorflow-1.11.0-cp36-cp36m-linux_armv7l.whl
```

# Step3 Cleaning Up

```
swapoff /gc100/2048Mb.swap
rm -rf  /gc100/2048Mb.swap
```