Building An Android Rom from AOSP Source Code:
Android Rom Version: Nougat 7.0

System Configuration

Processor: Intel® Core™ i7-3770 CPU @ 3.40GHz × 8
Graphics: Intel® Ivybridge Desktop
Hard-Drive : 1 TB
Physical Memory : 8gb Ram.
Os Version: Ubuntu 16.04 LTS

Procedure:

The Android AOSP Source code was already downloaded to our Local Server , To prevent Network Bandwidth Issues, Apart From forking the git repository from the Local Server instead of the Android AOSP site, All steps are same for building the rom. Make sure your system is updated using:
sudo apt get update && sudo apt-get upgrade

1. Make sure You have /bin directory in your Home path and include it using:

```
$mkdir ~/bin

$PATH=~/bin:$PATH
```

2.Download The Repo tool and make it Executable:

```
$curl https://storage.googleapis.com/git-repo-downloads/repo > ~/bin/repo

$chmod a+x ~/bin/repo
```

3. make a folder to copy the repository to that folder, I named it Android and install git.

```
`       $mkdir Android
        $cd Android/
        $sudo apt-get install git
        $sudo apt-get update && sudo apt-get upgrade
```

4. Configure Git If this is the first time using your username and email registered with git.

```
$git config --global user.name  “Username9”

$git config --global user.email “email-id”
```

5. Generate Your ssh key using ssh-keygen and then copy the key to the android server repository to prevent frequents checks asking for username and password.

```
$cd /Android

$ssh-keygen

$ssh-copy-id android@10.30.53.57

$repo init -u android@10.30.53.57:/home/seshagiri/Android/platform/manifest.git
```

To check out branches other than master:

```
$repo init -u android@10.30.53.57:/home/seshagiri/Android/platform/manifest.git -b android-X.X.X_rXX

$repo init -u android@10.30.53.57:/home/seshagiri/Android/platform/manifest.git -b android-7.0.0_r1
```

6. Finally Pulldown the repository:

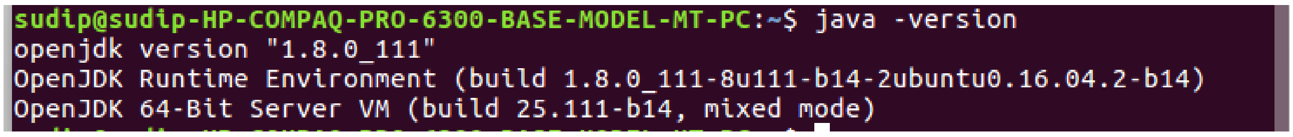
```
$repo sync
```

7. After the repository is downloaded Install all the necessary packages for buliding the android rom.

```
$sudo apt-get install git ccache automake lzop bison gperf build-essential zip curl zlib1g-dev zlib1g-dev:i386 g++-multilib python-networkx libxml2-utils bzip2 libbz2-dev libbz2-1.0  libghc-bzlib-dev squashfs-tools pngcrush schedtool dpkg-dev liblz4- tool make optipng  maven
```

8. Updating your package list and install the latest version of java ,Sometimes jadk versions less than 1.8 creates compatibility issues.

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



```
sudip@sudip-HP-COMPAQ-PRO-6300-BASE-MODEL-MT-PC:~$ java -version
openjdk version "1.8.0_111"
OpenJDK Runtime Environment (build 1.8.0_111-8u111-b14-2ubuntu0.16.04.2-b14)
OpenJDK 64-Bit Server VM (build 25.111-b14, mixed mode)
```

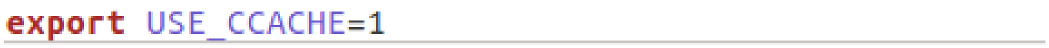
```
source ~/.bashrc
```

9. Setup ccache, ccache is a c/c++ development tool that caches the output of c/c++ compilation, so that the next time the if the same compilation is performed , it can be avoided and results taken from ccache instead.

```
$cd prebuilts/

$export CCACHE_DIR=/home/sudip/Android/

$export CCACHE_DIR=/home/sudip/Android//cache

$export CCACHE_DIR=/home/sudip/Android/.ccache

$prebuilts/misc/linux-x86/ccache/ccache -M 50G
```

add the following lines to .bashrc

```
$sudo gedit ~/.bashrc
```

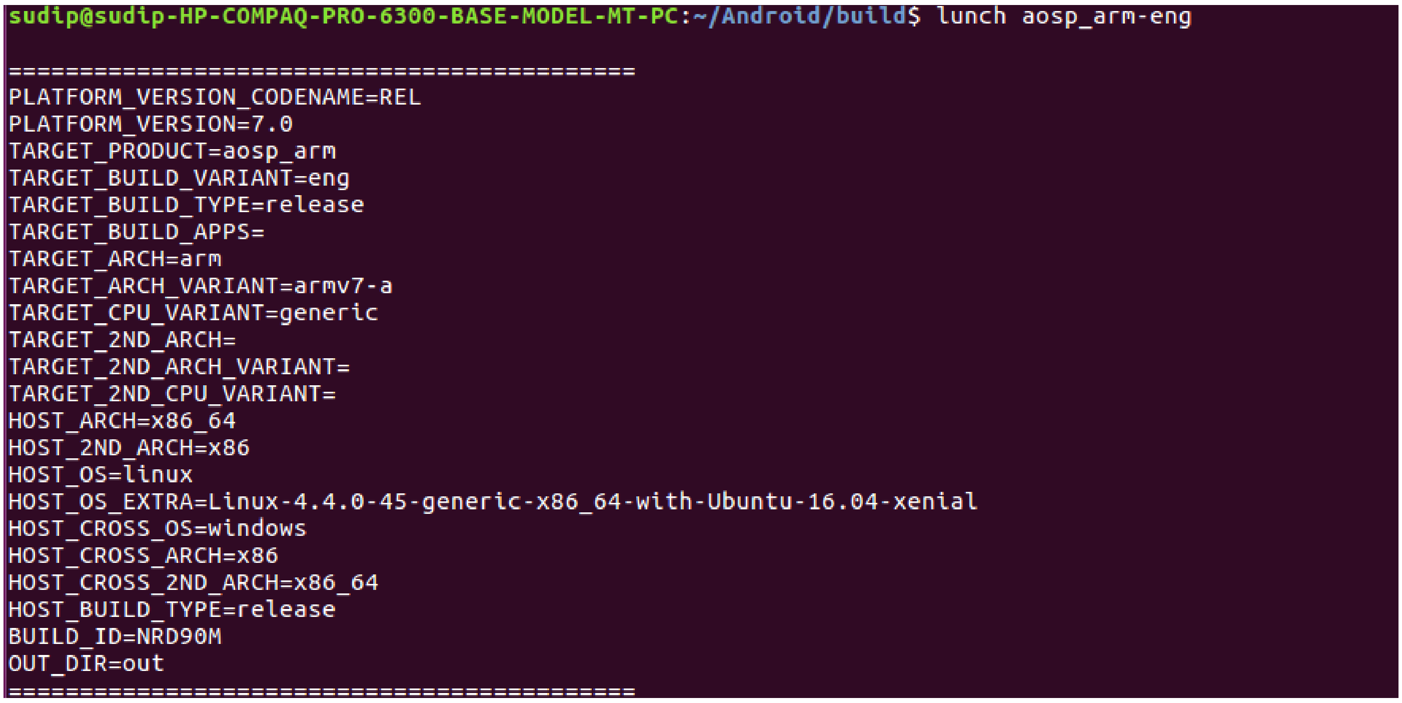


```
export USE_CCACHE=1
```

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

10. Now we are ready to srart the process of building the Android Rom:

```
$chmod -R 755 Android/

$cd Android/

$cd build/

$source build/envsetup.sh

$source envsetup.sh

$lunch aosp_arm-eng
```



```
sudip@sudip-HP-COMPAQ-PRO-6300-BASE-MODEL-MT-PC:~/Android/build$ lunch aosp_arm-eng

============================================
PLATFORM_VERSION_CODENAME=REL
PLATFORM_VERSION=7.0
TARGET_PRODUCT=aosp_arm
TARGET_BUILD_VARIANT=eng
TARGET_BUILD_TYPE=release
TARGET_BUILD_APPS=
TARGET_ARCH=arm
TARGET_ARCH_VARIANT=armv7-a
TARGET_CPU_VARIANT=generic
TARGET_2ND_ARCH=
TARGET_2ND_ARCH_VARIANT=
TARGET_2ND_CPU_VARIANT=
HOST_ARCH=x86_64
HOST_2ND_ARCH=x86
HOST_OS=linux
HOST_OS_EXTRA=Linux-4.4.0-45-generic-x86_64-with-Ubuntu-16.04-xenial
HOST_CROSS_OS=windows
HOST_CROSS_ARCH=x86
HOST_CROSS_2ND_ARCH=x86_64
HOST_BUILD_TYPE=release
BUILD_ID=NRD90M
OUT_DIR=out
============================================
```

$make -j16 // No of cpu threads the build is allowed to use.

Jack is a new Android toolchain that compiles Java source into Android dex bytecode. It replaces the previous Android toolchain, which consists of multiple tools, such as javac, ProGuard, jarjar, and dx.

To prevent Error regarding low heap size:

$export JACK_SERVER_VM_ARGUMENTS="-Dfile.encoding=UTF-8 -XX:+TieredCompilation -Xmx4096m"

$out/host/linux-x86/bin/jack-admin kill-server

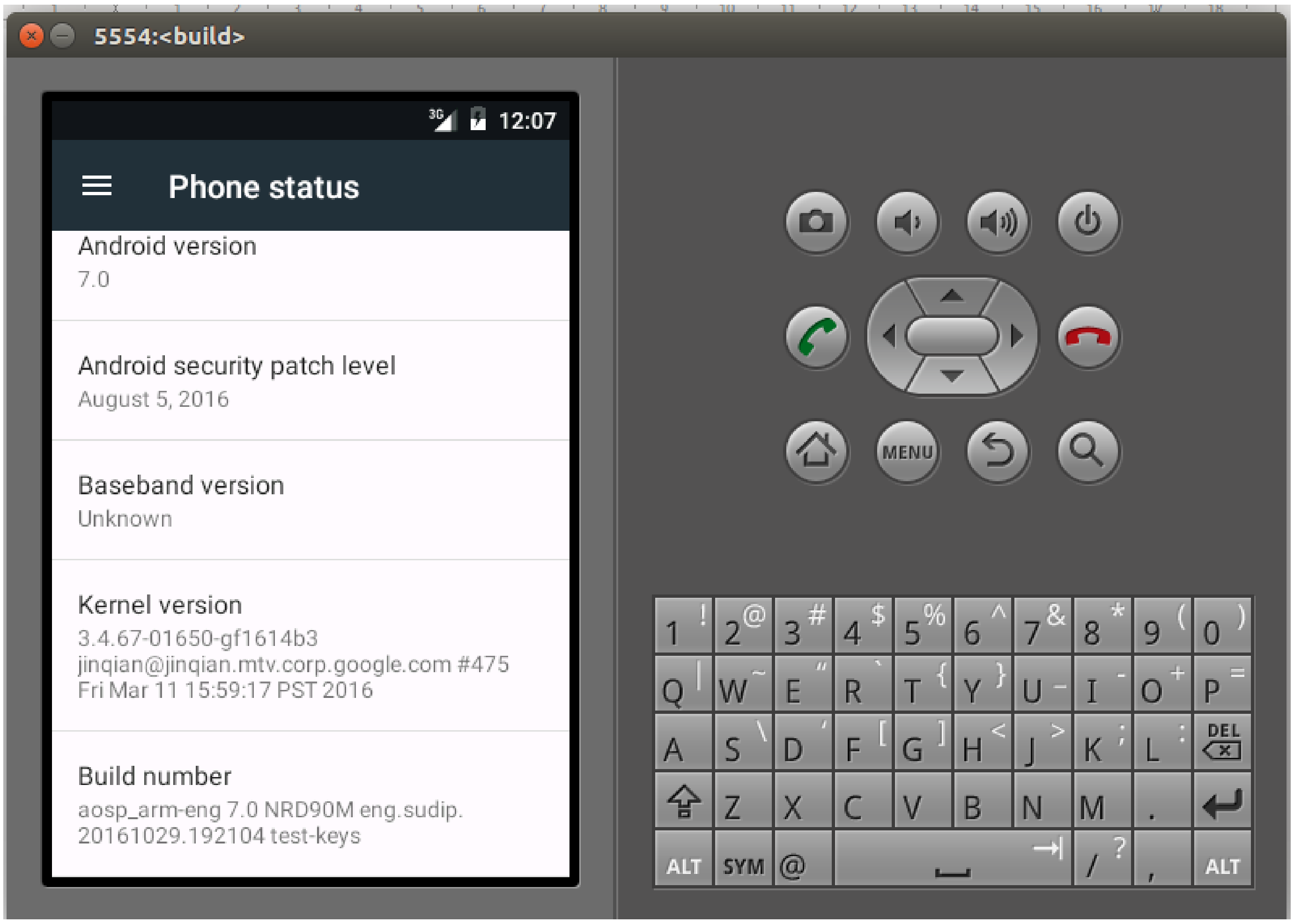
$out/host/linux-x86/bin/jack-admin start-server

$make -j16

It will take approximately 4 hours to build the source code and get the android rom.

After the ROM is built succesfully,in the same directory type:

$emulator



5554:<build>

12:07

Phone status

Android version
7.0

Android security patch level
August 5, 2016

Baseband version
Unknown

Kernel version
3.4.67-01650-gf1614b3
jinqian@jinqian.mtv.corp.google.com #475
Fri Mar 11 15:59:17 PST 2016

Build number
aosp_arm-eng 7.0 NRD90M eng.sudip.
20161029.192104 test-keys