

QCS40X Smart Audio – SSRD System Bring up Guide

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

80-YB420-4 Rev. B

Confidential and Proprietary – Qualcomm Technologies, Inc.

NO PUBLIC DISCLOSURE PERMITTED: Please report postings of this document on public servers or websites to: DocCtrlAgent@qualcomm.com.

Restricted Distribution: Not to be distributed to anyone who is not an employee of either Qualcomm Technologies, Inc. or its affiliated companies without the express approval of Qualcomm Configuration Management.

Confidential
2019-12-30 02:00:15 PST
ldi.setadi@prasimax.com

Confidential and Proprietary – Qualcomm Technologies, Inc.

NO PUBLIC DISCLOSURE PERMITTED: Please report postings of this document on public servers or websites to: DocCtrlAgent@qualcomm.com.

Restricted Distribution: Not to be distributed to anyone who is not an employee of either Qualcomm Technologies, Inc. or its affiliated companies without the express approval of Qualcomm Configuration Management.

Not to be used, copied, reproduced, or modified in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm Technologies, Inc.

All Qualcomm products mentioned herein are products of Qualcomm Technologies, Inc. and/or its subsidiaries.

Qualcomm, Hexagon, Snapdragon, and Snapdragorn are trademarks of Qualcomm Incorporated, registered in the United States and other countries. Qualcomm ChipCode is a trademark of Qualcomm Incorporated. AllPlay is a trademark of Qualcomm Connected Experiences, Inc, registered in the United States and other countries. aptX is a trademark of Qualcomm Technologies International, Ltd, registered in the United States and other countries. IPS is a trademark of CSR Technology, Inc. Other product and brand names may be trademarks or registered trademarks of their respective owners.

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited.

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

© 2019 Qualcomm Technologies, Inc. and/or its subsidiaries. All rights reserved.

Revision History

Revision	Date	Description
A	June 2019	Initial release
B	July 2019	Update for r1.0_00007.0 software

Qualcomm Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

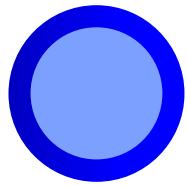
Agenda

- System Overview
- Setup Build System
- Install Tools for Windows
- Obtain Software
- Build Software
- Device Boot Configuration
- Writing Images
- Basic Operation Test
- References

Qualcomm Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Conventions

No	Mark	Description
1	>	Prefix for commands on Windows Command Prompt
2	\$	Prefix for commands on Linux Terminal
3	/ #	Prefix for commands on adb shell
4	—	RED Line: Line to highlight ERROR MESSAGE on terminal window
5	—	YELLOW Line: Line to highlight USER INPUT COMMAND on terminal window
6	—	ORANGE Line: Line to highlight CHECK POINT on terminal window



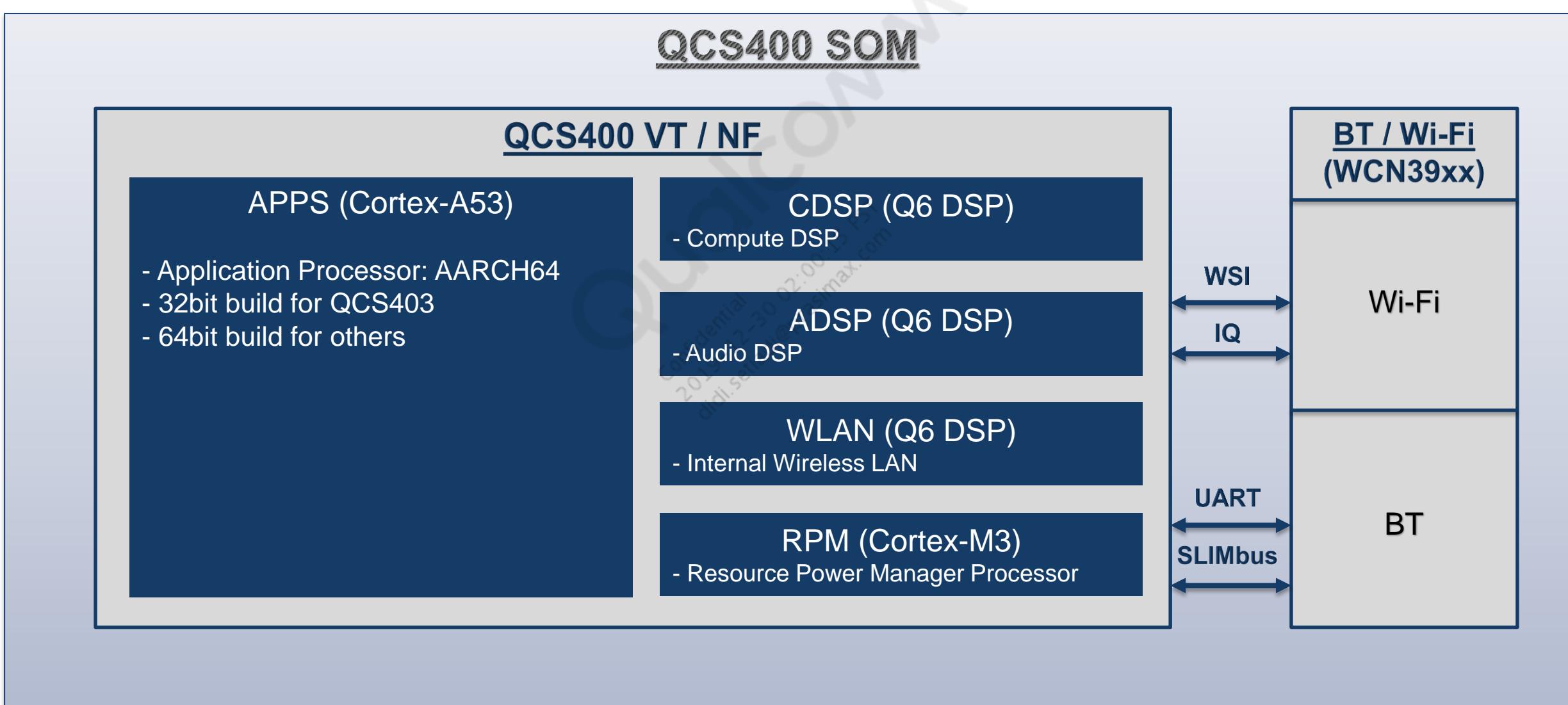
Section 1

System Overview

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

QCS400 Sub-system Overview



QCS400 Sub-system Overview – Image Files



Software Overview

**ADK Reference
Applications**

**Mobile Applications
(iOS, Android)**

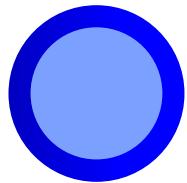
Tools / Utilities

MiddleWare
(Voice UI, Gstreamer, PulseAudio, and so on...)

HAL & QCOM Native Libs
(BT Stack, Wi-Fi Driver, OpenGL, Audio HAL, ST HAL, and so on...)

Kernel / BSP
(HDMI, I2C, TinyALSA, TZ/QSEE, DDR, Ethernet, and so on...)

ADSP / CDSP Firmware

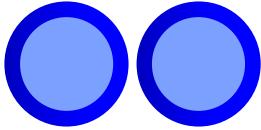


Section 2

Setup Build System

Qualcomm
Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

2.1 Basic Setup	11
2.2 Installing Toolchains	26
2.2.1 Qualcomm® Snapdragon™ Low Light Vision (LLV)M ARM Toolchain	28
2.2.2 Qualcomm® Hexagon™ LLVM Toolchain	41
2.2.3 GCC Linaro Toolchain	50



Section 2.1

Basic Setup

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

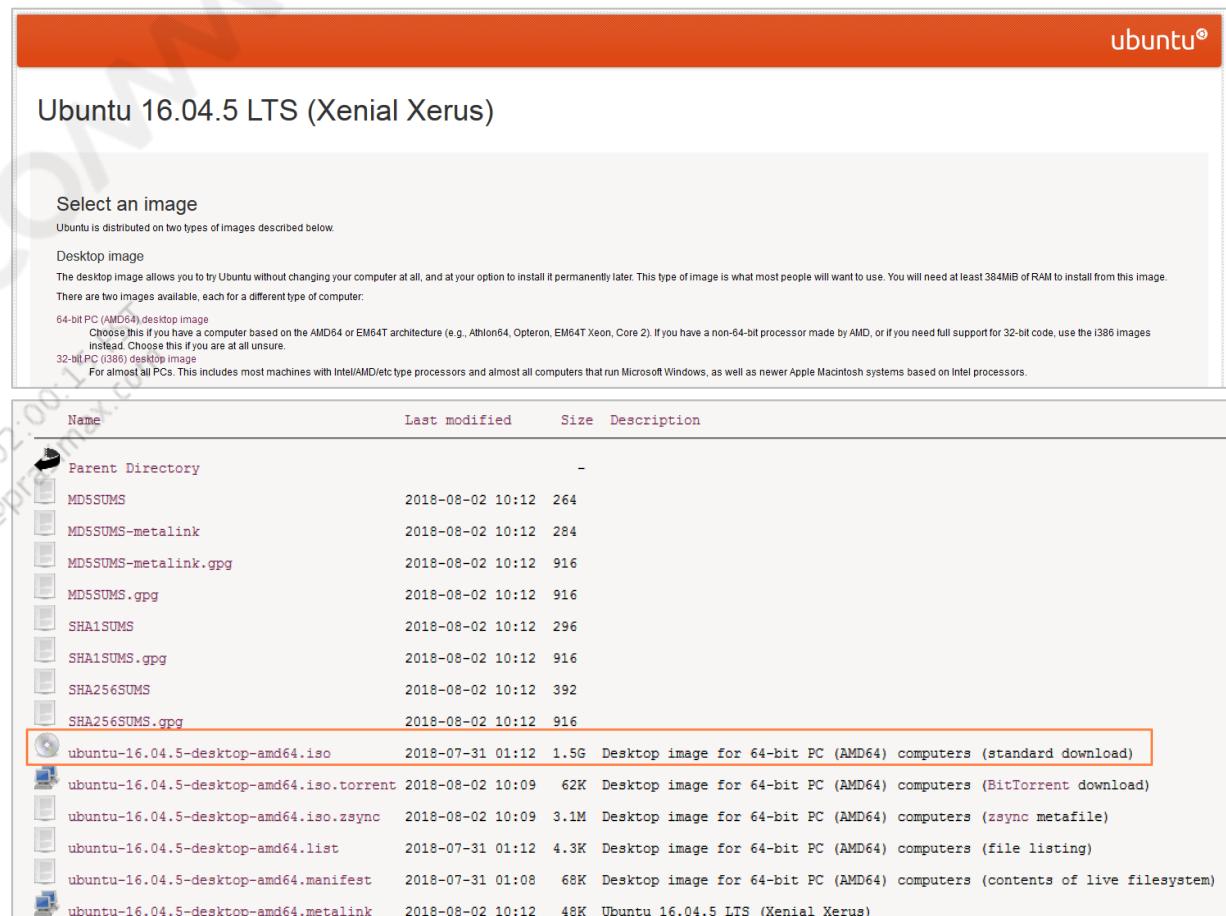
Basic Setup - Download Ubuntu

- Ubuntu 16.04 LTS for 64bit desktop PC is recommended
 - Download ‘ubuntu-16.04.5-desktop-amd64.iso’ file from:
<http://releases.ubuntu.com/xenial/>
- Virtual Machine
 - Tested VMWare setting example:

VMWare@Workstation 14 Pro

- VMWare Version : Workstation 14.x
- Memory 8GB
- Processors = 1
- Cores per processor = 2
- HDD 200GB pre-allocated
- OS : Ubuntu 16.04 LTS (64bit build)

- NOTE** on using VMWare:
 - Unexpected build errors can occur during HLOS software build
 - You can avoid this problem by adjusting the number of build tasks
 - This will be mentioned in the [Build HLOS Software](#) chapter



The screenshot shows the Ubuntu 16.04.5 LTS (Xenial Xerus) download page. At the top, there's an orange header with the Ubuntu logo. Below it, the title "Ubuntu 16.04.5 LTS (Xenial Xerus)" is displayed. A section titled "Select an image" explains that Ubuntu is distributed on two types of images: Desktop image and Server image. Under "Desktop image", it describes the 64-bit PC (AMD64) desktop image, noting it's for AMD64 or EM64T architecture. It also mentions the 32-bit PC (i386) desktop image, which is suitable for most PCs. A file list table follows, showing various files including the ISO image and its metadata.

Name	Last modified	Size	Description
Parent Directory		-	
MD5SUMS	2018-08-02 10:12	264	
MD5SUMS-metalink	2018-08-02 10:12	284	
MD5SUMS-metalink.gpg	2018-08-02 10:12	916	
MD5SUMS.gpg	2018-08-02 10:12	916	
SHA1SUMS	2018-08-02 10:12	296	
SHA1SUMS.gpg	2018-08-02 10:12	916	
SHA256SUMS	2018-08-02 10:12	392	
SHA256SUMS.gpg	2018-08-02 10:12	916	
ubuntu-16.04.5-desktop-amd64.iso	2018-07-31 01:12	1.5G	Desktop image for 64-bit PC (AMD64) computers (standard download)
ubuntu-16.04.5-desktop-amd64.iso.torrent	2018-08-02 10:09	62K	Desktop image for 64-bit PC (AMD64) computers (BitTorrent download)
ubuntu-16.04.5-desktop-amd64.iso.zsync	2018-08-02 10:09	3.1M	Desktop image for 64-bit PC (AMD64) computers (zsync metafile)
ubuntu-16.04.5-desktop-amd64.list	2018-07-31 01:12	4.3K	Desktop image for 64-bit PC (AMD64) computers (file listing)
ubuntu-16.04.5-desktop-amd64.manifest	2018-07-31 01:08	68K	Desktop image for 64-bit PC (AMD64) computers (contents of live filesystem)
ubuntu-16.04.5-desktop-amd64.metalink	2018-08-02 10:12	48K	Ubuntu 16.04.5 LTS (Xenial Xerus)

Basic Setup - Installing development packages

- Update Linux System Packages
 - \$ sudo apt-get update
 - \$ sudo apt-get upgrade
- Setup bash
 - \$ sudo ln -sf /bin/bash /bin/sh
- Install Android Tools
 - \$ sudo apt-get install android-tools-adb android-tools-fastboot
- Install Build Tools
 - \$ sudo apt-get install curl git git-core gawk chrpath build-essential texinfo libz-dev
 - \$ sudo apt-get install gcc-arm-linux-gnueabi
 - \$ sudo apt-get install gcc-multilib zlib1g:i386 libssl-dev clang llvm llvm-3.8

apt-get will choose 'zlib1g-dev' for 'libz-dev' on ubuntu 16.04

Basic Setup - Installing Java v1.7.0 JDK

- Since Java v1.7.0 JDK is not in the Linux default repository, you need to add repository separately
 - \$ sudo `add-apt-repository ppa:openjdk-r/ppa`
 - \$ sudo `apt-get update`
- Install Java v1.7.0 JDK
 - \$ sudo `apt-get install openjdk-7-jdk`
- Make sure the default Java version is 1.7.0
 - \$ `java -version`
- If multiple Java packages installed on your Linux build system and the default Java version is not v1.7.0,
- Then you can change the default Java version to v1.7.0 as follows:
 - \$ `update-java-alternatives --list`
 - \$ sudo `update-java-alternatives -s java-1.7.0-openjdk-amd64`

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Basic Setup - Checking Linux LLVM Toolchain

- Check the installed LLVM version: v3.8 or later is required

- \$ cd /usr/lib
 - /usr/lib\$ ls | grep llvm

```
donald@ubuntu:~$ cd /usr/lib
donald@ubuntu:/usr/lib$ ls -al | grep llvm
drwxr-xr-x  8 root root  4096 Feb 28 02:28 llvm-3.8
donald@ubuntu:/usr/lib$
```

- Check the symbolic link for 'llvm-propdata'

- \$ cd /usr/bin
 - /usr/bin\$ ls -al | grep llvm-propdata

```
donald@ubuntu:~$ cd /usr/bin
donald@ubuntu:/usr/bin$ ls -al | grep llvm-propdata
lrwxrwxrwx  1 root root      33 Jul 12  2016 llvm-propdata-3.8 -> ../../lib/llvm-3.8/bin/llvm-propdata
donald@ubuntu:/usr/bin$
```

- If there is no the symbolic link for 'llvm-propdata', then create the link

- /usr/bin\$ sudo ln -s ../../lib/llvm-3.8/bin/llvm-propdata

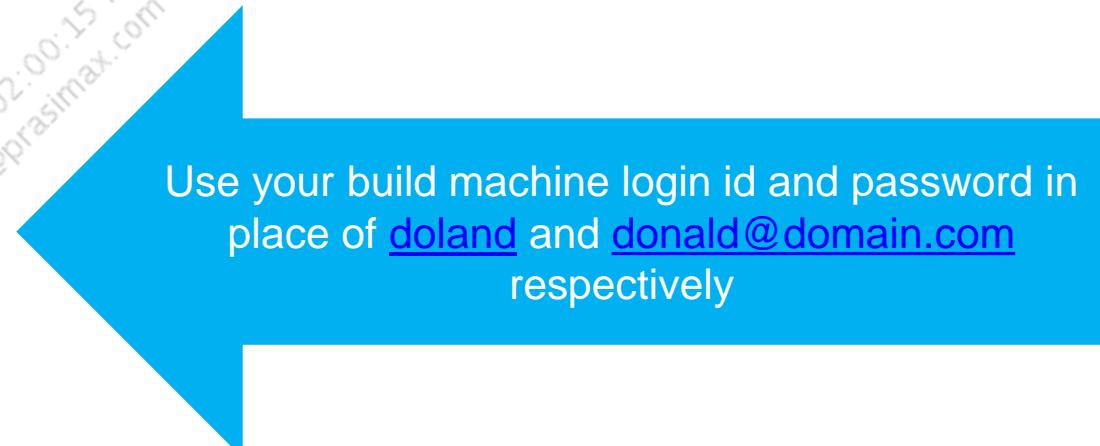
- Make sure the symbolic link for 'llvm-propdata'

- /usr/bin\$ ls -al | grep llvm-propdata

```
donald@ubuntu:/usr/bin$ sudo ln -s ../../lib/llvm-3.8/bin/llvm-propdata
[sudo] password for donald:
donald@ubuntu:/usr/bin$ ls -al | grep llvm-propdata
lrwxrwxrwx  1 root root      33 Mar 26 01:14 llvm-propdata -> ../../lib/llvm-3.8/bin/llvm-propdata
lrwxrwxrwx  1 root root      33 Jul 12  2016 llvm-propdata-3.8 -> ../../lib/llvm-3.8/bin/llvm-propdata
donald@ubuntu:/usr/bin$
```

Basic Setup - Installing repo

- Install repo
 - \$ `mkdir ~/bin`
 - \$ `PATH=~/bin:$PATH`
 - \$ `curl https://storage.googleapis.com/git-repo-downloads/repo > ~/bin/repo`
 - \$ `chmod a+x ~/bin/repo`
- Initialize repo client
 - \$ `git config --global user.name doland`
 - \$ `git config --global user.email doland@domain.com`
- Reference
 - <https://source.android.com/setup/downloading>



Use your build machine login id and password in place of doland and doland@domain.com respectively

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Basic Setup - Installing samba

- Samba is required for file sharing with Windows test machine
- The following processes require Samba:
 - [Writing Images using QFIL](#)
 - [Writing Images using fastboot on Windows](#)
 - [Packaging Images using QFIL](#)

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Basic Setup - Installing samba

- Install samba
 - \$ sudo apt-get install samba
- Add account
 - \$ sudo gedit /etc/samba/smb.conf

```
[donald]
comment = Donald Duck
path = /home/donald
valid users = doland
public = yes
writable = yes
```

- Password setup
 - \$ sudo smbpasswd -a donald
- Restart samba service
 - \$ sudo service smbd restart



Use your build machine login id and password in place of doland and doland@domain.com respectively

Basic Setup - Connecting to samba

- Check IP address of Linux machine

```
donald@ubuntu:~$ ifconfig
ens33      Link encap:Ethernet HWaddr 00:0c:29:ed:2b:61
           inet addr 192.168.130.128 Bcast:192.168.130.255 Mask:255.255.255.0
                     inet6 addr fe80::f6f2:c54f:f34c:49c2/64 Scope:Link
                           UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
                           RX packets:285653 errors:0 dropped:0 overruns:0 frame:0
                           TX packets:87996 errors:0 dropped:0 overruns:0 carrier:0
                           collisions:0 txqueuelen:1000
                           RX bytes:353154476 (353.1 MB) TX bytes:5349289 (5.3 MB)

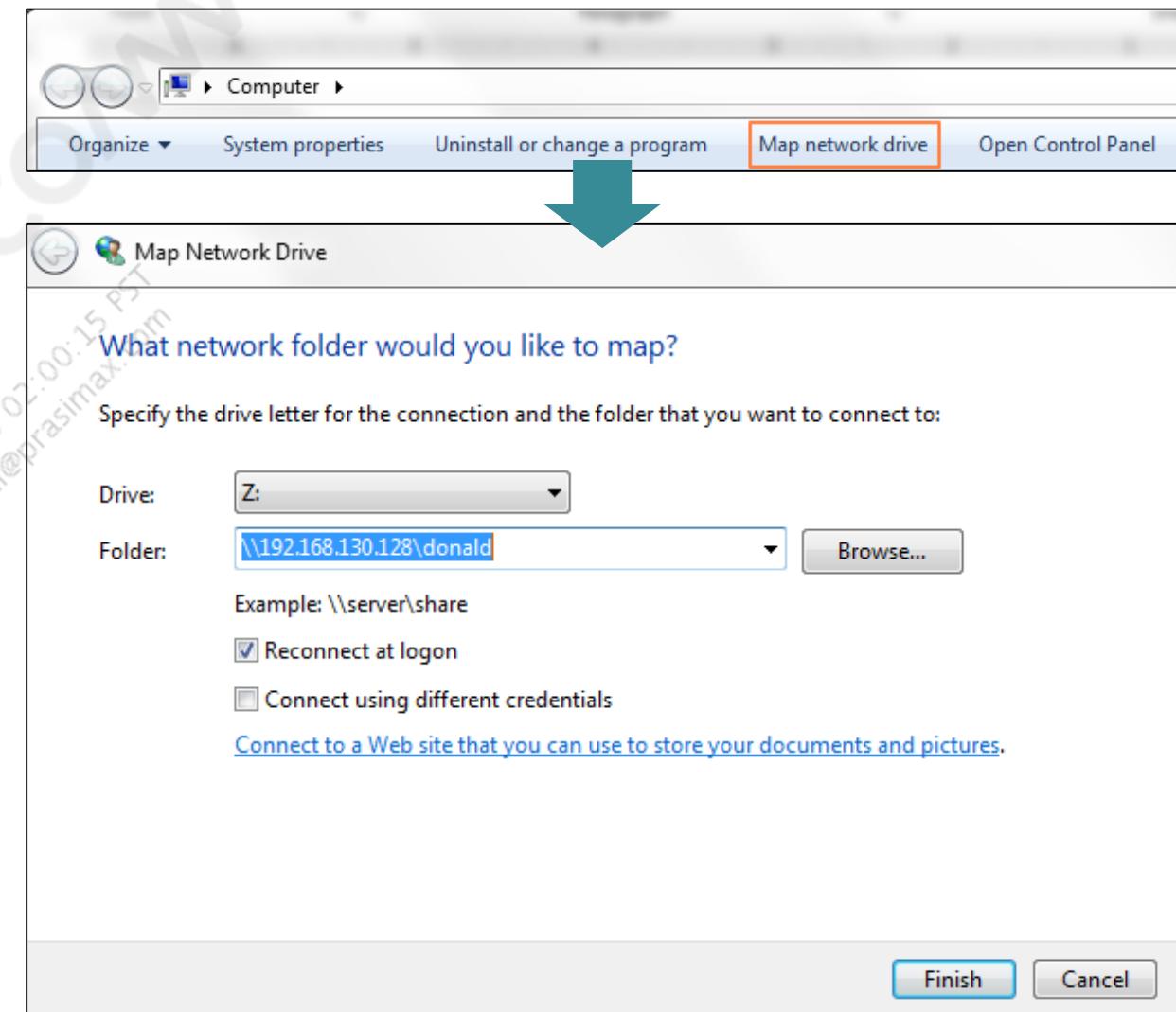
lo         Link encap:Local Loopback
           inet addr:127.0.0.1 Mask:255.0.0.0
                     inet6 addr ::1/128 Scope:Host
                           UP LOOPBACK RUNNING MTU:65536 Metric:1
                           RX packets:327 errors:0 dropped:0 overruns:0 frame:0
                           TX packets:327 errors:0 dropped:0 overruns:0 carrier:0
                           collisions:0 txqueuelen:1000
                           RX bytes:27365 (27.3 KB) TX bytes:27365 (27.3 KB)

donald@ubuntu:~$
```

Confidential
2019-12-30 09:15:15
didi.setiadi@raspberrypi.org

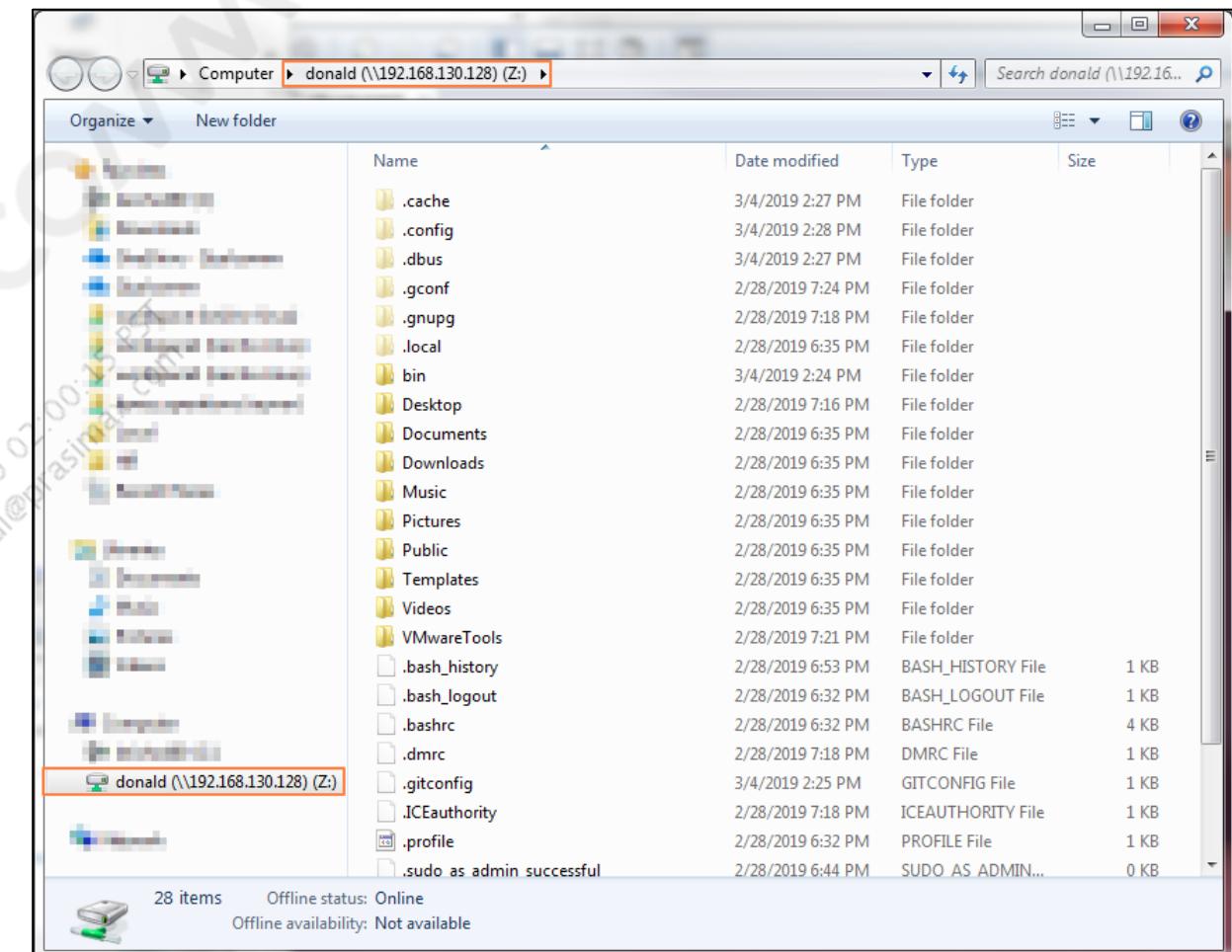
Basic Setup - Connecting to samba

- Check IP address of Linux machine
- Open “Map Network Drive” windows from Windows File Explorer
- Specify a network folder to map
 - e.g. <\\192.168.130.128\donald>
- Click on ‘Finish’ button



Basic Setup - Connecting to samba

- Check IP address of Linux machine
- Open “Map Network Drive” windows from Windows File Explorer
- Specify a network folder to map
 - e.g. <\\192.168.130.128\donald>
- Click on ‘Finish’ button
- Make sure connection state on Windows File Explorer
- DONE !



Basic Setup - Installing ssh

- Ubuntu Terminal requires large resource
- Build machine may crash during build process unless RAM size of the build system is big enough
- The terminal window launched from Ubuntu desktop is known to consume good amount of system resource
- Therefore, it's recommended for light weight build machine users to rely on remote terminal which requires SSH

Confidential

2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Basic Setup - Installing ssh

- Install ssh
 - \$ sudo apt-get install ssh
- Check IP address of Linux machine

```
donald@ubuntu:~$ ifconfig
ens33      Link encap:Ethernet HWaddr 00:0c:29:ed:2b:61
           inet addr:192.168.130.128 Bcast:192.168.130.255 Mask:255.255.255.0
                     inet6 addr: fe80::f6f2:c54f:f34c:49c2/64 Scope:Link
                           UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
                           RX packets:285653 errors:0 dropped:0 overruns:0 frame:0
                           TX packets:87996 errors:0 dropped:0 overruns:0 carrier:0
                           collisions:0 txqueuelen:1000
                           RX bytes:353154476 (353.1 MB) TX bytes:5349289 (5.3 MB)

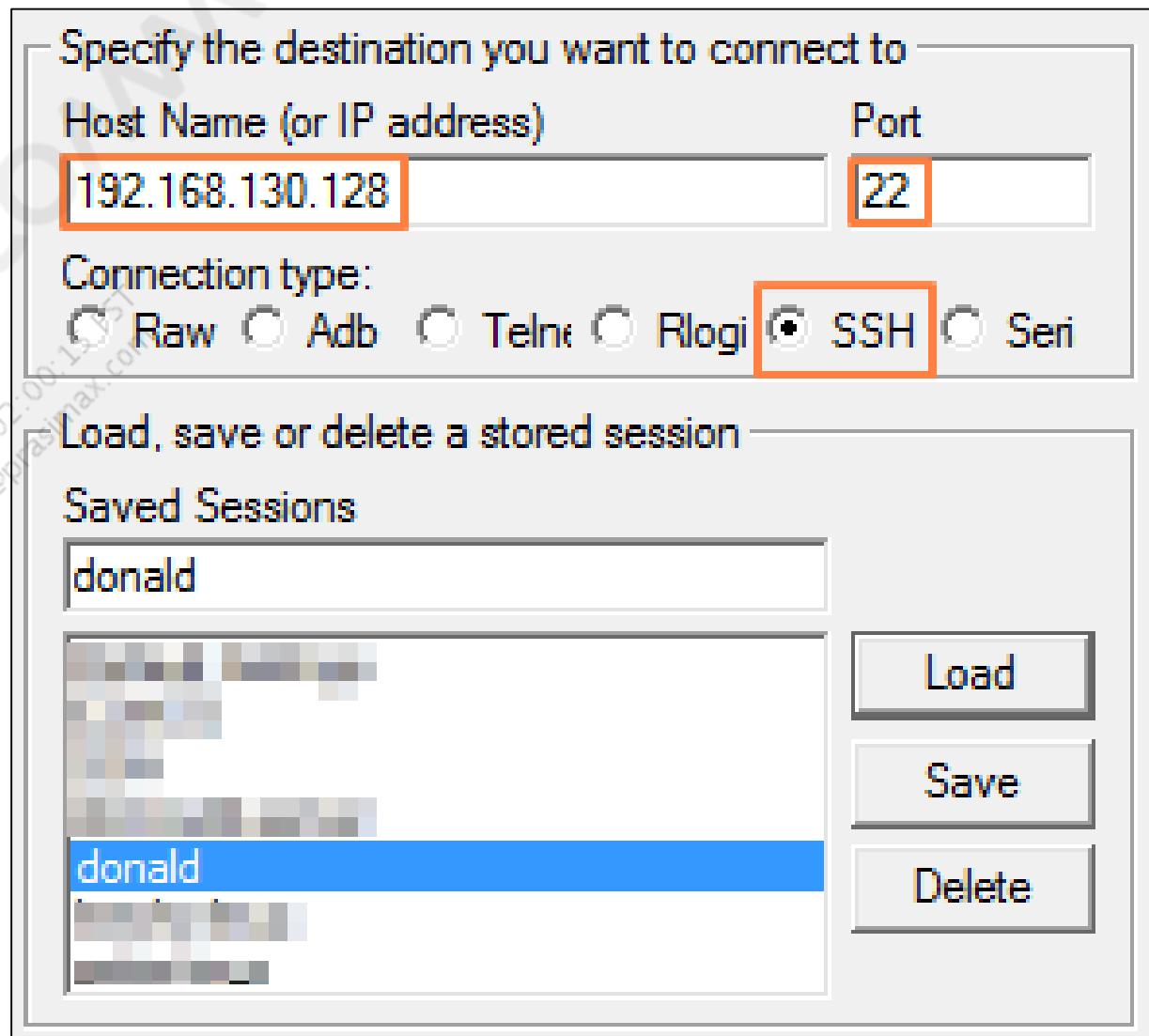
lo         Link encap:Local Loopback
           inet addr:127.0.0.1 Mask:255.0.0.0
                     inet6 addr: ::1/128 Scope:Host
                           UP LOOPBACK RUNNING MTU:65536 Metric:1
                           RX packets:327 errors:0 dropped:0 overruns:0 frame:0
                           TX packets:327 errors:0 dropped:0 overruns:0 carrier:0
                           collisions:0 txqueuelen:1000
                           RX bytes:27365 (27.3 KB) TX bytes:27365 (27.3 KB)

donald@ubuntu:~$
```

Confidential
2019-12-30 09:15:15
didi.setiadi@raspberrypi.org

Basic Setup - Installing ssh

- Ubuntu Terminal requires large resource so user may encounter system crash during build process unless there is big enough RAM in the build system. Therefore, we are suggesting SSH which is known to use less resource
- Install ssh
 - \$ sudo apt-get install ssh
- Check IP address of Linux machine
- Open your preferred terminal application and connect with following setting
 - e.g. : [Putty]
 - Connection type : SSH
 - Host IP address : 192.168.130.128
 - Port : 22 (is default ssh port number)



Basic Setup - Installing ssh

- Ubuntu Terminal requires large resource so user may encounter system crash during build process unless there is big enough RAM in the build system. Therefore, we are suggesting SSH which is known to use less resource
- Install ssh
 - \$ sudo apt-get install ssh
- Check IP address of Linux machine
- Open your preferred terminal application and connect with following setting
 - e.g. : [Putty]
 - Connection type : SSH
 - Host IP address : 192.168.130.128
 - Port : 22 (is default ssh port number)
- Check login as your account
- DONE !

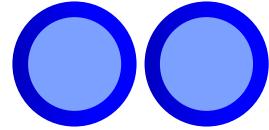
The screenshot shows a terminal window titled "donald@ubuntu: ~". It displays the following text:

```
login as: donald
donald@192.168.130.128's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-29-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

3 packages can be updated.
1 update is a security update.

*** System restart required ***
Last login: Thu Feb 28 01:53:02 2019 from 192.168.130.1
donald@ubuntu:~$
```



Section 2.2

Installing Toolchains

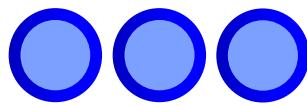
Qualcomm
Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

2.2.1 Qualcomm® Snapdragon™ Low Light Vision (LLV)M ARM Toolchain	28
2.2.2 Qualcomm Hexagon LLVM Toolchain	41
2.2.3 GCC Linaro Toolchain	50

Installing Toolchains

- Required Toolchains for building non-HLOS images:
 - Snapdragon LLVM ARM Toolchain
 - Qualcomm Hexagon LLVM Toolchain
 - GCC Linaro Toolchain
- Qualcomm Toolchains are available in Qualcomm CreatePoint
 - <https://createpoint.qti.qualcomm.com/dashboard/>

The screenshot shows the Qualcomm CreatePoint Product Kits dashboard. The top navigation bar includes links for Home, Documents, Hardware Components, Software Code, Tools, Support, and Hardware Purchase. A search bar and a 'My Account' dropdown are also present. The main content area is titled 'Product Kits' and displays a list of 'Product Kits: All Product Types'. The list is filtered by 'Kit Level' (All) and shows various Qualcomm products such as APQ8009 Android Audio for Distributors (Proxy), APQ8009 Android for General IoT (Platform), APQ8009 Android Home Hub (Platform), APQ8009 Android Smart Assistant (Platform), APQ8009 Android Smart Speaker (Developer), APQ8009 Android Smart Speaker (Platform), APQ8009 Android Things Smart Speaker (Developer), APQ8009 Android Things Smart Speaker (Platform), APQ8009 Linux Audio + FluencePro for Distributors (Proxy), APQ8009 Linux Audio for Distributors (Proxy), APQ8009 Linux Audio Player (Platform), APQ8009 Linux Camera (Platform), APQ8009 Linux Drone (Platform), and APQ8009 Linux for General IoT (Developer). Each item in the list includes a 'Kit Info' column, a 'Product Kit' column, a 'Kit Level' column (e.g., DEV, MRKT, STRT), and a 'Favorites' column with a star icon.



Section 2.2.1

Snapdragon LLVM ARM Toolchain

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Snapdragon LLVM ARM Toolchain

- First, check required SD(Snapdragon) LLVM version in [80-YB405-129] QCS40x Software User Guide document
- Below are required SD LLVM version by each SW module based on [80-YB405-129] QCS40x Software User Guide

SW module	SD LLVM version
RPM	3.9.2
Boot Loader	4.0.2
Trust Zone	4.0.11

Confidential
2019-12-30 02:00:15 PFT
didi.setiadi@prasimax.com

QCS40x Software

User Guide

80-YB405-129 Rev. F

December 31, 2018

2.5.5 Build and compile RPM

The compiler version is Qualcomm® Snapdragon™ Mobile Platform LLVM 3.9.2.

2.5.6 Build and compile boot loader

The compiler version is Qualcomm Snapdragon Mobile Platform LLVM v4.0.2.

2.5.7 Build and compile TrustZone

The following are the compiler versions:

- QTI – Snapdragon LLVM v4.0.11

Snapdragon LLVM ARM Toolchain – Download

- To download SD LLVM,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “LLVM” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Find required SD LLVM version
 - e.g. ‘3.9’ for SD LLVM 3.9.2
 - Make sure ‘Snapdragon LLVM ARM Toolchain for Linux’
 - Click on the item in ‘Name’ column

The screenshot shows the Qualcomm createpoint dashboard. A large watermark in the center reads "Confidential 2019-12-30 didi.setadi@praktikus.com". At the top, there's a navigation bar with links: Home, Documents, Hardware Components, Software Code, Tools (which is highlighted with a red circle labeled 1), Support, and Hardware Purchase. On the right, there's a search bar with the text "keicho". Below the navigation, a sub-menu titled "Tools: Choose a Tools Suite" is displayed. It includes a "All Suites" button, a "Filter Results" section with a search input containing "LLVM" (redboxed) and an "Apply" button (redboxed), and a "Filters" section with a "Clear All" link. The main area is titled "All Tools" and contains a table with columns: Name, Full Name, Version, and Release Date. The table lists four items:

Name	Full Name	Version	Release Date
<input type="checkbox"/> Snapdragon_SD_LLVM_ARM.LNX.3.9 Installer	Qualcomm® Snapdragon™ LLVM ARM Toolchain for Linux	39130.1	01-26-2019
<input type="checkbox"/> Snapdragon_SD_LLVM_ARM.WIN.3.9 Installer	Qualcomm® Snapdragon™ LLVM ARM Toolchain for Windows	39132.1	01-26-2019
<input type="checkbox"/> Hexagon.LNX.8.3 Installer	Qualcomm® Hexagon™ LLVM Toolchain Release	08302.1	01-22-2019

Red circles numbered 2, 3, and 4 point to the "Filters" section, the "Apply" button, and the first listed item in the table respectively.

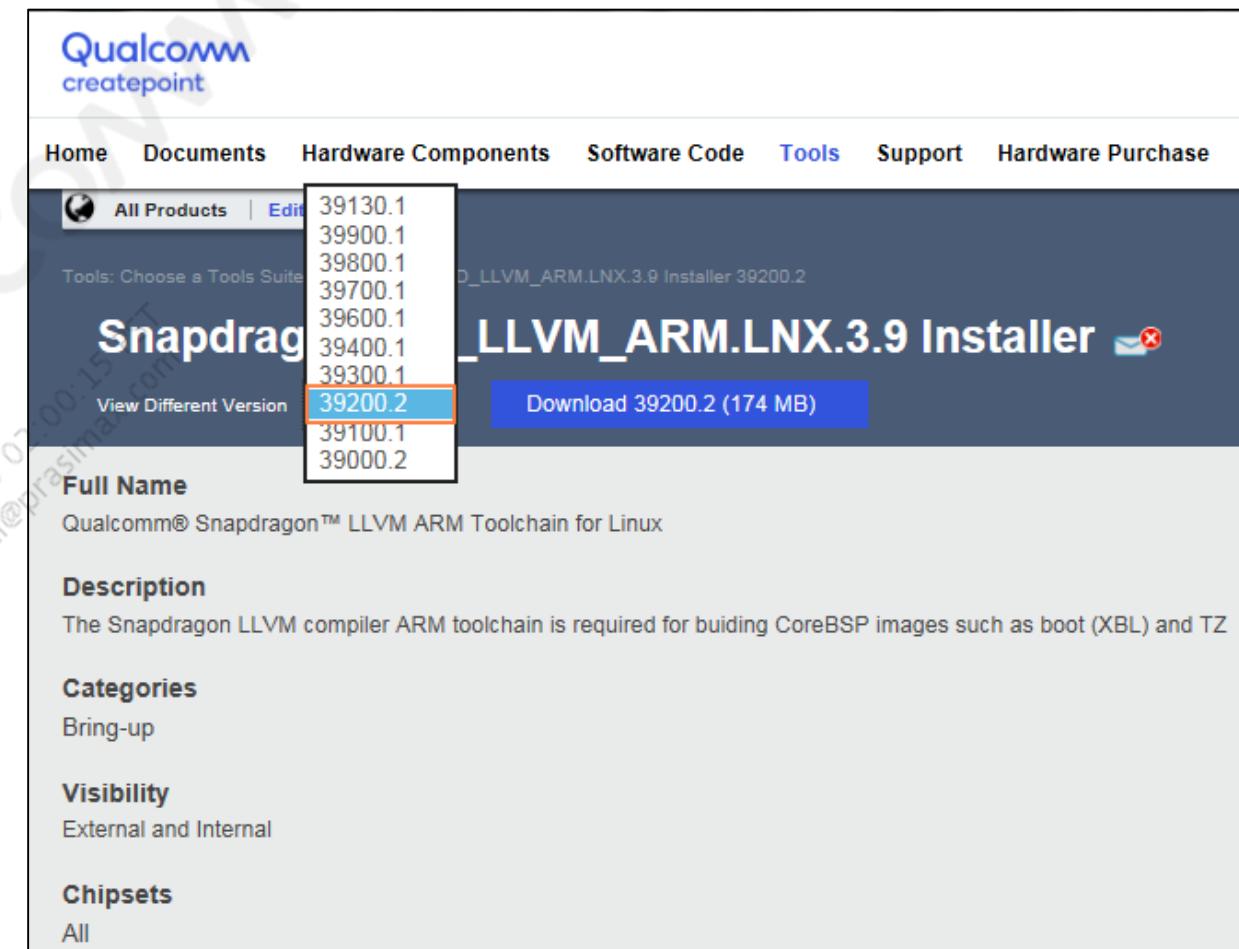
Snapdragon LLVM ARM Toolchain – Download

- To download SD LLVM,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “LLVM” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Find required SD LLVM version
 - e.g. ‘3.9’ for SD LLVM 3.9.2
 - Make sure ‘Snapdragon LLVM ARM Toolchain for Linux’
 - Click on the item in ‘Name’ column
- Click on ‘View Different Version’ combo box for specific version setting

The screenshot shows a web browser displaying the Qualcomm createpoint website. The URL in the address bar is https://createpoint.qti.qualcomm.com/. The page title is "Qualcomm createpoint". The navigation menu includes Home, Documents, Hardware Components, Software Code, Tools, Support, and Hardware Purchase. The "Tools" tab is selected. A sub-menu for "Tools" shows "Choose a Tools Suite" and "Snapdragon_SD_LLVM_ARM.LNX.3.9 Installer 39130.1". The main content area displays the "Snapdragon_SD_LLVM_ARM.LNX.3.9 Installer" page. It features a large blue header with the product name. Below the header, there is a "View Different Version" dropdown set to "39130.1" (which is highlighted with a red box), and a "Download 39130.1 (175 MB)" button. To the right of the download button is an envelope icon with a red 'X' over it. The page also contains sections for "Full Name" (Qualcomm® Snapdragon™ LLVM ARM Toolchain for Linux), "Description" (The Snapdragon LLVM compiler ARM toolchain is required for building CoreBSP images such as boot (XBL) and TZ), "Categories" (Bring-up), and "Visibility" (External and Internal). A watermark with the text "Confidential 2019-12-30 01:00:15 didi.setiadi@prasmania.com" is visible across the page.

Snapdragon LLVM ARM Toolchain – Download

- To download SD LLVM,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “LLVM” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Find required SD LLVM version
 - e.g. ‘3.9’ for SD LLVM 3.9.2
 - Make sure ‘Snapdragon LLVM ARM Toolchain for Linux’
 - Click on the item in ‘Name’ column
- Click on ‘View Different Version’ combo box for specific version setting
- Select specific version
 - e.g. ‘**39200.2**’ for SD LLVM 3.9.2



Snapdragon LLVM ARM Toolchain – Download

- To download SD LLVM,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “LLVM” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Find required SD LLVM version
 - e.g. ‘3.9’ for SD LLVM 3.9.2
 - Make sure ‘Snapdragon LLVM ARM Toolchain for Linux’
 - Click on the item in ‘Name’ column
- Click on ‘View Different Version’ combo box for specific version setting
- Select specific version
 - e.g. ‘39200.2’ for SD LLVM 3.9.2
- Click on ‘Download’ button to start downloading
 - A compressed file will be downloaded

The screenshot shows a web interface for Qualcomm's createpoint software catalog. At the top, there is a navigation bar with links for Home, Documents, Hardware Components, Software Code, Tools, Support, and Hardware Purchase. Below the navigation bar, a search bar contains the text "Tools: Choose a Tools Suite > Snapdragon_SD_LLVM_ARM.LNX.3.9 Installer 39200.2". The main content area features a large title "Snapdragon_SD_LLVM_ARM.LNX.3.9 Installer" with a red 'x' icon. Below the title, there is a "View Different Version" dropdown set to "39200.2" and a blue "Download 39200.2 (174 MB)" button. To the right of the download button is an envelope icon with a red 'x'. Further down, there are sections for "Full Name" (Qualcomm® Snapdragon™ LLVM ARM Toolchain for Linux), "Description" (The Snapdragon LLVM compiler ARM toolchain is required for building CoreBSP images such as boot (XBL) and TZ), "Categories" (Bring-up), "Visibility" (External and Internal), and "Chipsets" (All). A watermark with the text "Confidential 2019-12-30 01:00:15 didi.setiadi@prasina.com" is visible across the page.

Snapdragon LLVM ARM Toolchain – Download

- Repeat Download step for all SD LLVM Toolchains

- [3.9.2]
 - [4.0.2]
 - [4.0.11]

Version	Capture from CreatePoint
3.9.2	Snapdragon_SD_LLVM_ARM.LNX.3.9 Installer  View Different Version 39200.2  Download 39200.2 (174 MB)
4.0.2	Snapdragon_SD_LLVM_ARM.LNX.4.0 Installer  View Different Version 40200.1  Download 40200.1 (221 MB)
4.0.11	Snapdragon_SD_LLVM_ARM.LNX.4.0 Installer  View Different Version 41100.1  Download 41100.1 (221 MB)

- Make sure that the downloaded files are in <Download_Folder>

```
snapdragon_sd_llvm_arm.lnx.3.9_installer_39200.2.tar  
snapdragon_sd_llvm_arm.lnx.4.0_installer_40200.103-04-19_08_25_16.zip  
snapdragon_sd_llvm_arm.lnx.4.0_installer_41100.103-04-19_08_24_36.zip
```

Snapdragon LLVM ARM Toolchain – Install

- Create <SD LLVM version> folders in <Download_Folder>
 - <Download_Folder>\$ mkdir 3.9.2
 - <Download_Folder>\$ mkdir 4.0.2
 - <Download_Folder>\$ mkdir 4.0.11

```
~/Downloads$ mkdir 3.9.2
~/Downloads$ mkdir 4.0.2
~/Downloads$ mkdir 4.0.11
~/Downloads$
~/Downloads$ ls -al

donald donald 4096 Apr 21 21:31 .
donald donald 4096 Apr 21 19:32 ..
donald donald 4096 Apr 21 21:31 3.9.2
donald donald 4096 Apr 21 21:31 4.0.11
donald donald 4096 Apr 21 21:31 4.0.2
donald donald 551893237 Dec 6 2017 Hexagon_LLVM_linux_installer_8.2.02.bin
donald donald 548279768 Mar 4 19:22 hexagon.lnx.8.2_installer_08202.303-05-19_02_56_23.zip
donald donald 551895040 Mar 4 18:56 Hexagon_LNX_8_2_Installer_08202_3.tar
donald donald 182108160 Apr 21 21:28 snapdragon_sd_llvm_arm.lnx.3.9_installer_39200.2.tar
donald donald 230600801 Apr 21 21:29 snapdragon_sd_llvm_arm.lnx.4.0_installer_40200.103-04-19_08_25_16.zip
donald donald 230590321 Apr 21 21:29 snapdragon_sd_llvm_arm.lnx.4.0_installer_41100.103-04-19_08_24_36.zip
~/Downloads$
```

Confidential
2019-12-30 02:11:11
didi.setiadi@prasina

Snapdragon LLVM ARM Toolchain – Install

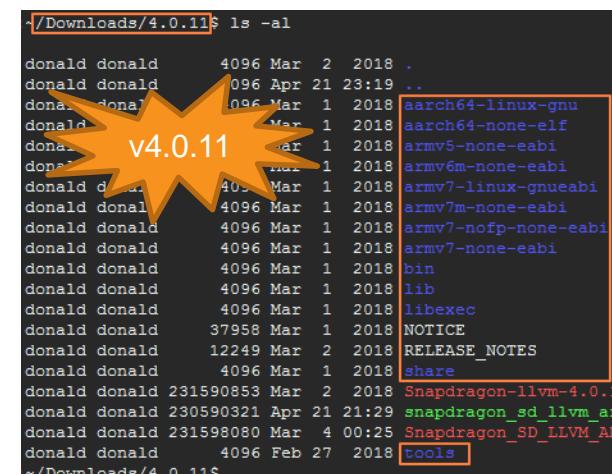
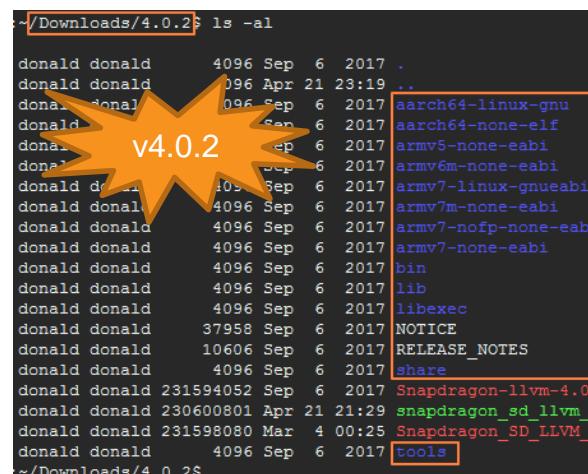
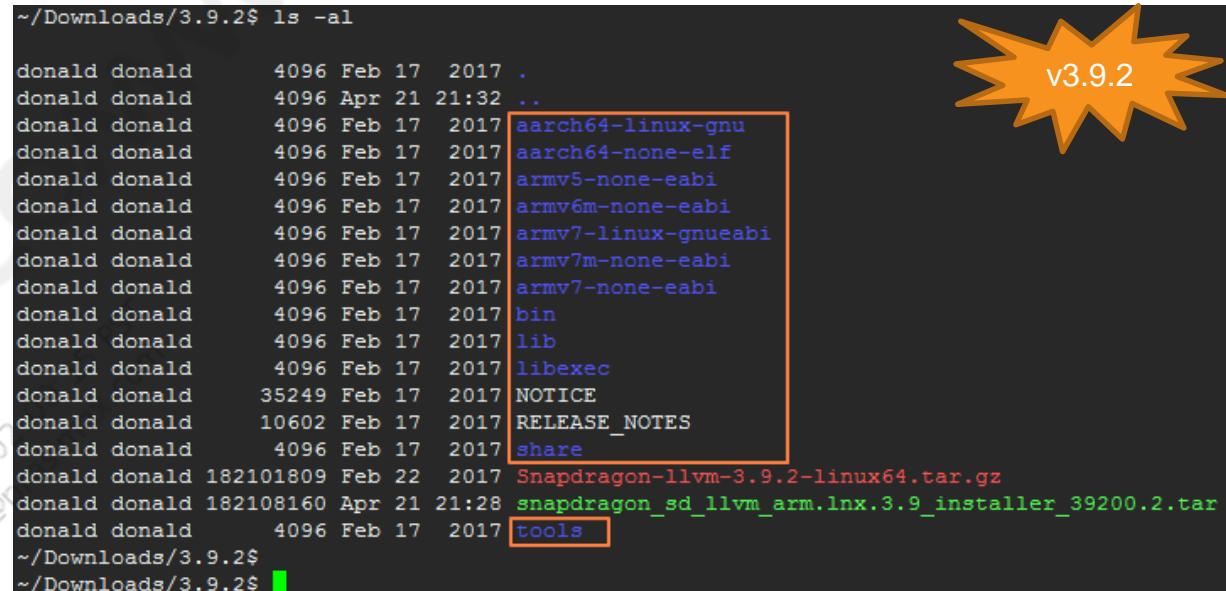
- Create <SD LLVM version> folders in <Download_Folder>
 - <Download_Folder>\$ mkdir 3.9.2
 - <Download_Folder>\$ mkdir 4.0.2
 - <Download_Folder>\$ mkdir 4.0.11
- Move each **downloaded file** to each <SD LLVM version> folder
 - \$ mv **snapdragon_sd_llvm_arm.lnx.3.9_installer_39200.2.tar** **3.9.2/**
 - \$ mv **snapdragon_sd_llvm_arm.lnx.4.0_installer_40200.103-04-19_08_25_16.zip** **4.0.2/**
 - \$ mv **snapdragon_sd_llvm_arm.lnx.4.0_installer_41100.103-04-19_08_24_36.zip** **4.0.11/**

```
donald@ubuntu:~/Downloads$ mv snapdragon_sd_llvm_arm.lnx.3.9_installer_39200.2.tar 3.9.2/
donald@ubuntu:~/Downloads$ mv snapdragon_sd_llvm_arm.lnx.4.0_installer_40200.103-04-19_08_25_16.zip 4.0.2/
donald@ubuntu:~/Downloads$ mv snapdragon_sd_llvm_arm.lnx.4.0_installer_41100.103-04-19_08_24_36.zip 4.0.11/
donald@ubuntu:~/Downloads$ 
donald@ubuntu:~/Downloads$ cd 3.9.2
donald@ubuntu:~/Downloads/3.9.2$ ls
snapdragon_sd_llvm_arm.lnx.3.9_installer_39200.2.tar
donald@ubuntu:~/Downloads/3.9.2$ 
donald@ubuntu:~/Downloads/3.9.2$ cd ../4.0.2
donald@ubuntu:~/Downloads/4.0.2$ ls
snapdragon_sd_llvm_arm.lnx.4.0_installer_40200.103-04-19_08_25_16.zip
donald@ubuntu:~/Downloads/4.0.2$ 
donald@ubuntu:~/Downloads/4.0.2$ cd ../4.0.11
donald@ubuntu:~/Downloads/4.0.11$ ls
snapdragon_sd_llvm_arm.lnx.4.0_installer_41100.103-04-19_08_24_36.zip
donald@ubuntu:~/Downloads/4.0.11$ 
```

Confidential
2019-12-30
didi.setiani@qti.com

Snapdragon LLVM ARM Toolchain – Install

- Create <SD LLVM version> folders in <Download_Folder>
 - <Download_Folder>\$ mkdir 3.9.2
 - <Download_Folder>\$ mkdir 4.0.2
 - <Download_Folder>\$ mkdir 4.0.11
- Move each downloaded file to each <SD LLVM version> folder
 - \$ mv snapdragon_sd_llvm_arm.lnx.3.9_installer_39200.2.tar 3.9.2/
 - \$ mv snapdragon_sd_llvm_arm.lnx.4.0_installer_40200.103-04-19_08_25_16.zip 4.0.2/
 - \$ mv snapdragon_sd_llvm_arm.lnx.4.0_installer_41100.103-04-19_08_24_36.zip 4.0.11/
- Uncompress each **downloaded file** with following commands and make sure the result
 - <Download_Folder>\$ cd <SD LLVM version>
 - <SD LLVM version>\$ unzip <file_name>.zip
 - <SD LLVM version>\$ tar -xvf <file_name>.tar
 - <SD LLVM version>\$ tar -xzvf <file_name>.tar.gz



v3.9.2

```
~/Downloads/3.9.2$ ls -al
donald donald 4096 Feb 17 2017 .
donald donald 4096 Apr 21 21:32 ..
donald donald 4096 Feb 17 2017 aarch64-linux-gnu
donald donald 4096 Feb 17 2017 aarch64-none-elf
donald donald 4096 Feb 17 2017 armv5-none-eabi
donald donald 4096 Feb 17 2017 armv6m-none-eabi
donald donald 4096 Feb 17 2017 armv7-linu
donald donald 4096 Feb 17 2017 armv7m-none-eabi
donald donald 4096 Feb 17 2017 armv7-none-eabi
donald donald 4096 Feb 17 2017 bin
donald donald 4096 Feb 17 2017 lib
donald donald 4096 Feb 17 2017 libexec
donald donald 35249 Feb 17 2017 NOTICE
donald donald 10602 Feb 17 2017 RELEASE_NOTES
donald donald 4096 Feb 17 2017 share
donald donald 182101809 Feb 22 2017 Snapdragon-llvm-3.9.2-linux64.tar.gz
donald donald 182108160 Apr 21 21:28 snapdragon_sd_llvm_arm.lnx.3.9_installer_39200.2.tar
donald donald 4096 Feb 17 2017 tools
~/Downloads/3.9.2$
~/Downloads/3.9.2$
```

```
~/Downloads/4.0.2$ ls -al
donald donald 4096 Sep 6 2017 .
donald donald 4096 Apr 21 23:19 ..
donald donald 4096 Sep 6 2017 aarch64-linux-gnu
donald donald 4096 Sep 6 2017 aarch64-none-elf
donald donald 4096 Sep 6 2017 armv5-none-eabi
donald donald 4096 Sep 6 2017 armv6m-none-eabi
donald donald 4096 Sep 6 2017 armv7-linu
donald donald 4096 Sep 6 2017 armv7m-none-eabi
donald donald 4096 Sep 6 2017 armv7-nofp-none-eabi
donald donald 4096 Sep 6 2017 armv7-none-eabi
donald donald 4096 Sep 6 2017 bin
donald donald 4096 Sep 6 2017 lib
donald donald 4096 Sep 6 2017 libexec
donald donald 37958 Sep 6 2017 NOTICE
donald donald 10606 Sep 6 2017 RELEASE_NOTES
donald donald 4096 Sep 6 2017 share
donald donald 231594052 Sep 6 2017 Snapdragon-llvm-4.0.2-
donald donald 230600801 Apr 21 21:29 snapdragon_sd_llvm_ar
donald donald 231598080 Mar 4 00:25 Snapdragon_SD_LLVM_AR
donald donald 4096 Sep 6 2017 tools
~/Downloads/4.0.2$
```

```
~/Downloads/4.0.11$ ls -al
donald donald 4096 Mar 2 2018 .
donald donald 4096 Apr 21 23:19 ..
donald donald 4096 Mar 1 2018 aarch64-linux-gnu
donald donald 4096 Mar 1 2018 aarch64-none-elf
donald donald 4096 Mar 1 2018 armv5-none-eabi
donald donald 4096 Mar 1 2018 armv6m-none-eabi
donald donald 4096 Mar 1 2018 armv7-linu
donald donald 4096 Mar 1 2018 armv7m-none-eabi
donald donald 4096 Mar 1 2018 armv7-nofp-none-eabi
donald donald 4096 Mar 1 2018 armv7-none-eabi
donald donald 4096 Mar 1 2018 bin
donald donald 4096 Mar 1 2018 lib
donald donald 4096 Mar 1 2018 libexec
donald donald 37958 Mar 1 2018 NOTICE
donald donald 12249 Mar 2 2018 RELEASE_NOTES
donald donald 4096 Mar 1 2018 share
donald donald 231590853 Mar 2 2018 Snapdragon-llvm-4.0.1-
donald donald 230590321 Apr 21 21:29 snapdragon_sd_llvm_ar
donald donald 231598080 Mar 4 00:25 Snapdragon_SD_LLVM_AR
donald donald 4096 Feb 27 2018 tools
~/Downloads/4.0.11$
```

Snapdragon LLVM ARM Toolchain – Install

- Create <SD LLVM version> folders in <Download_Folder>
 - \$ mkdir 3.9.2
 - \$ mkdir 4.0.2
 - \$ mkdir 4.0.11
- Move each downloaded file to each <SD LLVM version> folder
 - \$ mv snapdragon_sd_llvm_arm.lnx.3.9_installer_39200.2.tar 3.9.2/
 - \$ mv snapdragon_sd_llvm_arm.lnx.4.0_installer_40200.103-04-19_08_25_16.zip 4.0.2/
 - \$ mv snapdragon_sd_llvm_arm.lnx.4.0_installer_41100.103-04-19_08_24_36.zip 4.0.11/
- Uncompress each downloaded file with following commands and make sure the result
 - <Download_Folder>\$ cd <SD LLVM version>
 - <SD LLVM version>\$ unzip <file_name>.zip
 - <SD LLVM version>\$ tar -xvf <file_name>.tar
 - <SD LLVM version>\$ tar -xzvf <file_name>.tar.gz
- Create '/pkg/qct/software/llvm/release/arm/' folder
 - \$ sudo mkdir -p /pkg/qct/software/llvm/release/arm
 - \$ sudo chmod 755 /pkg/qct/software/llvm/release/arm

Qualcomm Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

```
donald@ubuntu:~/Downloads$ sudo mkdir -p /pkg/qct/software/llvm/release/arm
donald@ubuntu:~/Downloads$ sudo chmod 755 /pkg/qct/software/llvm/release/arm
donald@ubuntu:~/Downloads$ ls -al /pkg/qct/software/llvm/release
total 12
drwxr-xr-x 3 root root 4096 Mar  3 23:57 .
drwxr-xr-x 3 root root 4096 Mar  3 23:57 ..
drwxr-xr-x 2 root root 4096 Apr 21 23:19 arm
donald@ubuntu:~/Downloads$
```

Snapdragon LLVM ARM Toolchain – Install

- Create <SD LLVM version> folders in <Download_Folder>
 - \$ mkdir 3.9.2
 - \$ mkdir 4.0.2
 - \$ mkdir 4.0.11
- Move each downloaded file to each <SD LLVM version> folder
 - \$ mv snapdragon_sd_llvm_arm.lnx.3.9_installer_39200.2.tar 3.9.2/
 - \$ mv snapdragon_sd_llvm_arm.lnx.4.0_installer_40200.103-04-19_08_25_16.zip 4.0.2/
 - \$ mv snapdragon_sd_llvm_arm.lnx.4.0_installer_41100.103-04-19_08_24_36.zip 4.0.11/
- Uncompress each downloaded file with following commands and make sure the result
 - <Download_Folder>\$ cd <SD LLVM version>
 - <SD LLVM version>\$ unzip <file_name>.zip
 - <SD LLVM version>\$ tar -xvf <file_name>.tar
 - <SD LLVM version>\$ tar -xzvf <file_name>.tar.gz
- Create '/pkg/qct/software/llvm/release/arm/' folder
 - \$ sudo mkdir -p /pkg/qct/software/llvm/release/arm
 - \$ sudo chmod 755 /pkg/qct/software/llvm/release/arm
- Move all <SD LLVM version> folders to '.../arm/' folder
 - <Download_Folder>\$ sudo mv 3.9.2 /pkg/qct/software/llvm/release/arm/
 - <Download_Folder>\$ sudo mv 4.0.2 /pkg/qct/software/llvm/release/arm/
 - <Download_Folder>\$ sudo mv 4.0.11 /pkg/qct/software/llvm/release/arm/

Qualcomm Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

```
~/Downloads$ sudo mv 3.9.2 /pkg/qct/software/llvm/release/arm/
~/Downloads$ sudo mv 4.0.2 /pkg/qct/software/llvm/release/arm/
~/Downloads$ sudo mv 4.0.11 /pkg/qct/software/llvm/release/arm/
```

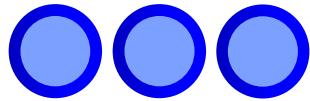
Snapdragon LLVM ARM Toolchain – Install

- Make sure the result
 - \$ cd /pkg/qct/software/llvm/release/arm
 - \$ ls -al
- SD LLVM installation DONE !

```
:~/Downloads$ cd /pkg/qct/software/llvm/release/arm
:/pkg/qct/software/llvm/release/arm$
:/pkg/qct/software/llvm/release/arm$ ls -al

root      root    4096 Apr 21 23:32 .
root      root    4096 Mar  3 23:57 ..
donald   donald  4096 Feb 17 2017 3.9.2
donald   donald  4096 Mar  2 2018 4.0.11
donald   donald  4096 Sep  6 2017 4.0.2
:/pkg/qct/software/llvm/release/arm$
```

Confidential
2019-12-30
didi.setiadi@ptt.qcom



Section 2.2.2

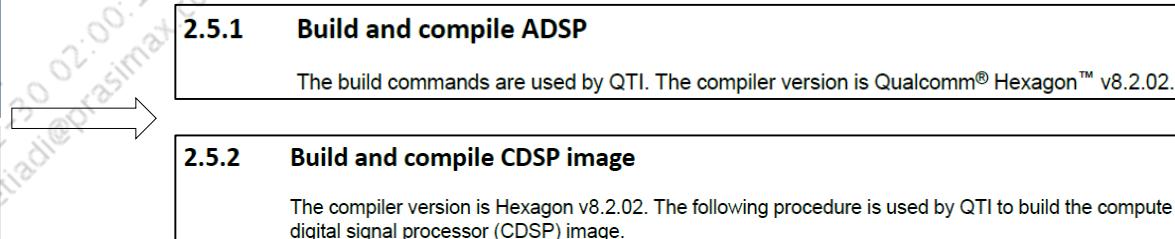
Qualcomm Hexagon LLVM Toolchain

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Qualcomm Hexagon LLVM Toolchain

- First, make sure required Hexagon Tool version in [80-YB405-129] QCS40x Software User Guide document
- Below are required Hexagon Tool version based on [80-YB405-129] QCS40x Software User Guide

SW module	Hexagon Tool version
ADSP, CDSP	8.2.02



Qualcomm Hexagon LLVM Toolchain – Download

- To download Hexagon Tool,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “Hexagon 8.2” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Click on ‘Hexagon.LNX.8.2 Installer’ in ‘Name’ column

The screenshot shows the Qualcomm createpoint dashboard. A large watermark in the center reads "Confidential 2019-12-30 didi.setiadi@praktikum.com".

Top Navigation: Qualcomm createpoint, Search (keicho), Home, Documents, Hardware Components, Software Code, **Tools** (highlighted with orange circle 1), Support, Hardware Purchase.

Left Sidebar: All Products, Edit, Tools: Choose a Tools Suite.

Filter Results: All Suites, Filter Results (input: Hexagon 8.2, highlighted with orange circle 2), Apply (button, highlighted with orange circle 3).

System OS: Linux (1), Windows (1).

Full Name: Qualcomm® Hexagon™ LLVM Toolchain Release (2).

Build Type: Installer (2).

Chipsets: (dropdown menu).

Categories: (dropdown menu).

All Tools Table: Headers: Name, Full Name, Version, Release Date. Data rows:

- Hexagon.LNX.8.2 Installer, Qualcomm® Hexagon™ LLVM Toolchain Release, 08208.1, 10-15-2018 (highlighted with orange circle 4)
- Hexagon.WIN.8.2 Installer, Qualcomm® Hexagon™, 08208.1, 10-15-2018

Qualcomm Hexagon LLVM Toolchain – Download

- To download Hexagon Tool,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “Hexagon 8.2” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Click on ‘Hexagon.LNX.8.2 Installer’ in ‘Name’ column
- Click on ‘View Different Version’ combo box for specific version setting

The screenshot shows a web browser displaying the Qualcomm createpoint portal. The URL in the address bar is https://createpoint.qti.qualcomm.com/. The page has a dark header with the Qualcomm createpoint logo and navigation links for Home, Documents, Hardware Components, Software Code, Tools, Support, and Hardware Purchasing. The main content area is titled "Hexagon.LNX.8.2 Installer". It features a large blue "Download 08208.1 (526 MB)" button. Below the button, there is a "View Different Version" dropdown menu which is currently set to "08208.1". The page also includes sections for "Full Name" (Qualcomm® Hexagon™ LLVM Toolchain Release), "Description" (a brief overview of the toolchain), and "Categories" (Design, Logging & Debug, Bring-up, Tuning). A watermark with the text "Confidential 2019-12-30 didi.setiadi@presintech.com" is visible across the page.

Qualcomm Hexagon LLVM Toolchain – Download

- To download Hexagon Tool,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “Hexagon 8.2” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Click on ‘Hexagon.LNX.8.2 Installer’ in ‘Name’ column
- Click on ‘View Different Version’ combo box for specific version setting
- Select specific version
 - e.g. ‘**08202.3**’ for Hexagon Tool 8.2.02 version

The screenshot shows the Qualcomm createpoint software interface. At the top, there is a navigation bar with links for Home, Documents, Hardware Components, Software Code, Tools, Support, and Hardware Purchasing. The 'Tools' link is highlighted in blue. Below the navigation bar, there is a search bar labeled 'All Products' with a dropdown menu showing various tool versions: 08208.1, 18207.1, 18206.1, 18205.1, 08204.1, 08203.1, 18200.1, 08202.3 (which is selected and highlighted in blue), and 08201.1. To the right of the dropdown, there is a large button labeled '8.2 Installer' with a download icon. Below the dropdown, there is a link 'View Different Version'. On the right side of the screen, there is a detailed description of the selected tool version, including its full name ('Qualcomm® Hexagon™ LLVM Toolchain Release'), description ('The Hexagon LLVM Tools Release version 8.2 is the first release with support for Hexagon v66 architecture, based on LLVM.org enhanced with optimizations from the Qualcomm LLVM team. As a cross compiler, it runs'), and categories ('Design', 'Logging & Debug', 'Bring-up', 'Tuning'). A watermark with the text 'Confidential 2019-12-30 didi.setadi@presidio.com' is visible across the page.

Qualcomm Hexagon LLVM Toolchain – Download

- To download Hexagon Tool,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “Hexagon 8.2” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Click on ‘Hexagon.LNX.8.2 Installer’ in ‘Name’ column
- Click on ‘View Different Version’ combo box for specific version setting
- Select specific version
 - e.g. ‘08202.3’ for Hexagon Tool 8.2.02 version
- Click on ‘Download’ button to start downloading
 - A compressed file will be downloaded

The screenshot shows a web browser displaying the Qualcomm createpoint portal. The URL in the address bar is https://createpoint.qti.qualcomm.com/. The page has a dark header with the Qualcomm createpoint logo and navigation links for Home, Documents, Hardware Components, Software Code, Tools (which is highlighted in blue), Support, and Hardware Purchasing.

The main content area displays a product listing for the "Hexagon.LNX.8.2 Installer". The product name is "Hexagon.LNX.8.2 Installer" with a small mail icon. Below the name, there is a "View Different Version" dropdown menu set to "08202.3" and a large blue "Download 08202.3 (526 MB)" button.

Below the download button, there are sections for "Full Name" (Qualcomm® Hexagon™ LLVM Toolchain Release), "Description" (The Hexagon LLVM Tools Release version 8.2 is the first release with support for Hexagon v66 architecture, based on LLVM.org enhanced with optimizations from the Qualcomm LLVM team. As a cross compiler, it run), and "Categories" (Design, Logging & Debug, Bring-up, Tuning).

A watermark with the text "Confidential 2019-12-30 didi.setiadi@prasinam.com" is visible across the page.

Qualcomm Hexagon LLVM Toolchain – Install

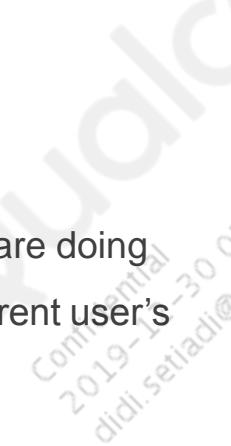
- Uncompress with following commands until .bin file appears
 - \$ cd <download_folder>
 - \$ unzip hexagon.lnx.8.2_installer_08202.303-05-19_02_56_23.zip
 - \$ tar -xvf Hexagon_LNX_8_2_Installer_08202_3.tar

```
donald@ubuntu:~$ cd Downloads/
donald@ubuntu:~/Downloads$ ls
Hexagon.LLVM_linux_installer_8.2.02.bin
hexagon.lnx.8.2_installer_08202.303-05-19_02_56_23.zip
Hexagon_LNX_8_2_Installer_08202_3.tar
donald@ubuntu:~/Downloads$
```

Confidential
2019-12-30 02:00:12
didi.setiadi@prasimax.com

Qualcomm Hexagon LLVM Toolchain – Install

- Uncompress with following commands until .bin file appears
 - \$ cd <download_folder>
 - \$ unzip hexagon.lnx.8.2_installer_08202.303-05-19_02_56_23.zip
 - \$ tar -xvf Hexagon_LNX_8_2_Installer_08202_3.tar
- Start to install with following commands
 - \$ chmod +x Hexagon.LLVM_linux_installer_8.2.02.bin
 - \$./Hexagon.LLVM_linux_installer_8.2.02.bin
- Use recommended settings unless you know what you are doing
- Check the ‘Install Folder’, the default install folder is current user’s home folder
 - /home/donald/Qualcomm/HEXAGON_Tools/8.2.02



```
=====
Introduction
-----
InstallAnywhere will guide you through the installation of
Qualcomm_Hexagon_LLVM_Tools.

It is strongly recommended that you quit all programs before continuing with
this installation.

Respond to each prompt to proceed to the next step in the installation. If
you want to change something on a previous step, type 'back'.

You may cancel this installation at any time by typing 'quit'.

PRESS <ENTER> TO CONTINUE:
=====
```

```
=====
Choose Install Folder
-----
Please choose a destination folder for this installation. This installation
will install the tools to this location.

Where would you like to install?

Default Install Folder: /home/donald/Qualcomm/HEXAGON_Tools/8.2.02
=====
ENTER AN ABSOLUTE PATH, OR PRESS <ENTER> TO ACCEPT THE DEFAULT
: [REDACTED]
=====
```

```
=====
Pre-Installation Summary
-----
Please Review the Following Before Continuing:

Product Name:
  Qualcomm_Hexagon_LLVM_Tools

Install Folder:
  /home/donald/Qualcomm/HEXAGON_Tools/8.2.02

Link Folder:
  /home/donald

Disk Space Information (for Installation Target):
  Required: 1,233,141,438 Bytes
  Available: 94,901,194,752 Bytes

PRESS <ENTER> TO CONTINUE:
=====
```

```
=====
Installation Complete
-----
Congratulations. Qualcomm_Hexagon_LLVM_Tools has been successfully installed
to:

  /home/donald/Qualcomm/HEXAGON_Tools/8.2.02

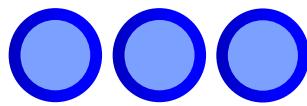
PRESS <ENTER> TO EXIT THE INSTALLER:
=====
```

Qualcomm Hexagon LLVM Toolchain – Install

- Uncompress with following commands until .bin file appears
 - \$ cd <download_folder>
 - \$ unzip hexagon.lnx.8.2_installer_08202.303-05-19_02_56_23.zip
 - \$ tar -xvf Hexagon_LNX_8_2_Installer_08202_3.tar
- Start to install with following commands
 - \$ chmod +x Hexagon.LLVM_linux_installer_8.2.02.bin
 - \$./Hexagon.LLVM_linux_installer_8.2.02.bin
- Use recommended settings unless you know what you are doing
- Check the ‘Install Folder’, the default install folder is current user’s home folder
 - /home/donald/Qualcomm/HEXAGON_Tools/8.2.02
- Add following environment variables to build ADSP and CDSP
 - \$ export HEXAGON_ROOT=~/Qualcomm/HEXAGON_Tools
 - \$ export HEXAGON_RTOS_RELEASE=8.2.02
 - \$ export HEXAGON_Q6VERSION=v66
 - \$ export HEXAGON_IMAGE_ENTRY=0x8AC00000
- **HEXAGON_ROOT** depends on where **the tools are installed**
- Hexagon Tool installation DONE !

Confidential and Proprietary – Qualcomm Technologies, Inc. | MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION | 80-YB420-4 Rev. B
didi.setiadi@prasimax.com
2019-12-30 02:00:15 PST

```
donald@ubuntu:~/Downloads$ export HEXAGON_ROOT=~/Qualcomm/HEXAGON_Tools
donald@ubuntu:~/Downloads$ export HEXAGON_RTOS_RELEASE=8.2.02
donald@ubuntu:~/Downloads$ export HEXAGON_Q6VERSION=v66
donald@ubuntu:~/Downloads$ export HEXAGON_IMAGE_ENTRY=0x8AC00000
donald@ubuntu:~/Downloads$ env | grep HEXA
HEXA GON_Q6VERSION=v66
HEXA GON_ROOT=/home/donald/Qualcomm/HEXA GON_Tools
HEXA GON_IMAGE_ENTRY=0x8AC00000
HEXA GON_RTOS_RELEASE=8.2.02
donald@ubuntu:~/Downloads$
```



Section 2.2.3

GCC Linaro Toolchain

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

GCC Linaro Toolchain

- All the required GCC Linaro Toolchains

SW module	Linaro Toolchain version	Download Site	File Name
Boot Loader	4.8-2014.02	https://releases.linaro.org/archive/14.02/components/toolchain/binaries/	gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2
Trust Zone	4.9-2014.07	https://releases.linaro.org/archive/14.07/components/toolchain/binaries/	gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2

GCC Linaro Toolchain – Download

- Download Linaro Toolchain files for v4.8 and v4.9 with following ‘`wget`’ commands on `<Download_Folder>`

- `<Download_Folder>$ wget https://releases.linaro.org/archive/14.02/components/toolchain/binaries/gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2`
- `<Download_Folder>$ wget https://releases.linaro.org/archive/14.07/components/toolchain/binaries/gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2`

Version	Download Command
4.8	<code>\$ wget https://releases.linaro.org/archive/14.02/components/toolchain/binaries/gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2</code>
4.9	<code>\$ wget https://releases.linaro.org/archive/14.07/components/toolchain/binaries/gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2</code>

```
~/Downloads$ ls -al
donald donald      4096 Apr 22 00:44 .
donald donald      4096 Apr 21 23:49 ..
donald donald  54692263 Feb 26 2018 gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2
donald donald 100166972 Feb 26 2018 gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2
donald donald 551893237 Dec  6 2017 Hexagon.LLVM_linux_installer_8.2.02.bin
donald donald 548279768 Mar  4 19:22 hexagon.lnx.8.2_installer_08202.303-05-19_02_56_23.zip
donald donald 551895040 Mar  4 18:56 Hexagon_LNX_8_2_Installer_08202_3.tar
~/Downloads$
```

GCC Linaro Toolchain – Install

- Uncompress the downloaded file for each downloaded file
 - <Download_Folder>\$ tar -xvf gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2
 - <Download_Folder>\$ tar -xvf gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2

Version	Install Commands
4.8	\$ tar -xvf gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2
4.9	\$ tar -xvf gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2


```
Confidential  
2019-12-30 02:00:15 PST  
didi.setiadi@prasimay.com  
~/Downloads$ ls -al  
donald donald 4096 Apr 22 00:56 .  
donald donald 4096 Apr 21 23:49 ..  
donald donald 4096 Jul 30 2014 gcc-linaro-aarch64-none-elf-4.9-2014.07_linux  
donald donald 54692263 Feb 26 2018 gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2  
donald donald 4096 Feb 20 2014 gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux  
donald donald 100166972 Feb 26 2018 gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2  
donald donald 551893237 Dec 6 2017 Hexagon_LLVM_linux_installer_8.2.02.bin  
donald donald 548279768 Mar 4 19:22 hexagon.lnx.8.2_installer_08202.303-05-19_02_56_23.zip  
donald donald 551895040 Mar 4 18:56 Hexagon_LNX_8_2_Installer_08202_3.tar  
~/Downloads$
```

GCC Linaro Toolchain – Install

- Uncompress the downloaded file for each downloaded file
 - <Download_Folder>\$ tar -xvf gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2
 - <Download_Folder>\$ tar -xvf gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2
- Create following folders for each Toolchain
 - \$ sudo mkdir -p /prj/llvm-arm/home/common/build_tools
 - \$ sudo chmod 755 /prj/llvm-arm/home/common/build_tools
 - \$ sudo mkdir -p /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf
 - \$ sudo chmod 755 /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf

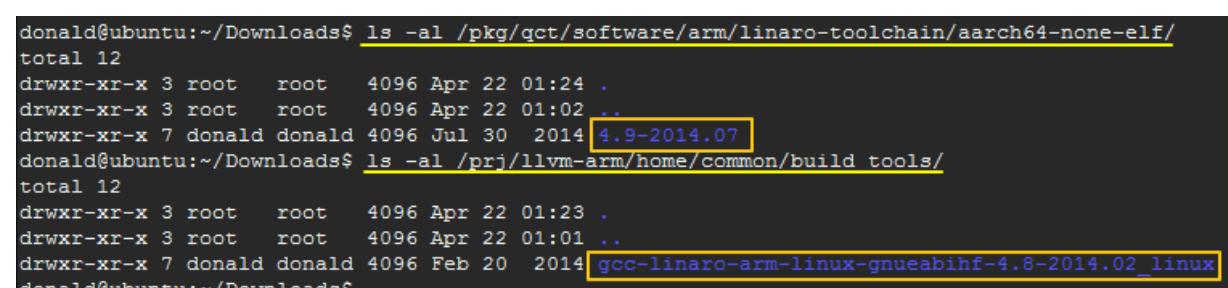
Version	Install Commands
4.8	\$ tar -xvf gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2 \$ sudo mkdir -p /prj/llvm-arm/home/common/build_tools \$ sudo chmod 755 /prj/llvm-arm/home/common/build_tools
4.9	\$ tar -xvf gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2 \$ sudo mkdir -p /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf \$ sudo chmod 755 /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf

```
donald@ubuntu:~/Downloads$ ls -al /prj/llvm-arm/home/common/
total 12
drwxr-xr-x 3 root root 4096 Apr 22 01:01 .
drwxr-xr-x 3 root root 4096 Mar  4 20:22 ..
drwxr-xr-x 2 root root 4096 Apr 22 01:01 build_tools
donald@ubuntu:~/Downloads$ 
donald@ubuntu:~/Downloads$ ls -al /pkg/qct/software/arm/linaro-toolchain/
total 12
drwxr-xr-x 3 root root 4096 Apr 22 01:02 .
drwxr-xr-x 3 root root 4096 Mar  4 20:28 ..
drwxr-xr-x 2 root root 4096 Apr 22 01:02 aarch64-none-elf
donald@ubuntu:~/Downloads$ 
```

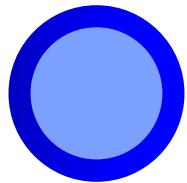
GCC Linaro Toolchain – Install

- Uncompress the downloaded file for each downloaded file
 - <Download_Folder>\$ tar -xvf gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2
 - <Download_Folder>\$ tar -xvf gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2
- Create following folders for each Toolchain
 - \$ sudo mkdir -p /prj/llvm-arm/home/common/build_tools
 - \$ sudo chmod 755 /prj/llvm-arm/home/common/build_tools
 - \$ sudo mkdir -p /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf
 - \$ sudo chmod 755 /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf
- Move the uncompressed folders to each Toolchain install target folder
- NOTE: For the v4.9, the target folder name should be changed to '**4.9-2014.07**'
 - <Download_Folder>\$ sudo mv gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux /prj/llvm-arm/home/common/build_tools/
 - <Download_Folder>\$ sudo mv gcc-linaro-aarch64-none-elf-4.9-2014.07_linux /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf/4.9-2014.07
- Linaro Toolchain installation DONE !

Version	Install Commands
4.8	\$ tar -xvf gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux.tar.bz2 \$ sudo mkdir -p /prj/llvm-arm/home/common/build_tools \$ sudo chmod 755 /prj/llvm-arm/home/common/build_tools \$ sudo mv gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux /prj/llvm-arm/home/common/build_tools/
4.9	\$ tar -xvf gcc-linaro-aarch64-none-elf-4.9-2014.07_linux.tar.bz2 \$ sudo mkdir -p /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf \$ sudo chmod 755 /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf \$ sudo mv gcc-linaro-aarch64-none-elf-4.9-2014.07_linux /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf/ 4.9-2014.07



```
donald@ubuntu:~/Downloads$ ls -al /pkg/qct/software/arm/linaro-toolchain/aarch64-none-elf/
total 12
drwxr-xr-x 3 root root 4096 Apr 22 01:24 .
drwxr-xr-x 3 root root 4096 Apr 22 01:02 ..
drwxr-xr-x 7 donald donald 4096 Jul 30 2014 4.9-2014.07
donald@ubuntu:~/Downloads$ ls -al /prj/llvm-arm/home/common/build_tools/
total 12
drwxr-xr-x 3 root root 4096 Apr 22 01:23 .
drwxr-xr-x 3 root root 4096 Apr 22 01:01 ..
drwxr-xr-x 7 donald donald 4096 Feb 20 2014 gcc-linaro-arm-linux-gnueabihf-4.8-2014.02_linux
donald@ubuntu:~/Downloads$
```



Section 3

Installing Tools for Windows

Qualcomm
Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

- | | |
|---|--------------------|
| 3.1 Installing QPST | 59 |
| 3.2 Installing Qualcomm Qualcomm® Smart Speaker Platform Configuration Tool | 62 |
| 3.3 Installing Android SDK Tools for Windows | 65 |

Installing Tools for Windows

- In this chapter, only basic required tools will be explained for Windows
 - Qualcomm Product Support Tool (QPST)
 - Qualcomm Smart Audio Platform Configuration Tool
 - Android SDK Tools (fastboot, adb, and so on)
- For more details, refer to [80-YB405-129] QCS40x Software User Guide document

Confidential

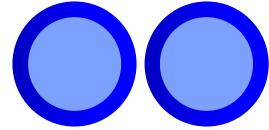
2019-12-30 02:00:55 PST
didi.setiadi@prasimax.com

Installing Tools for Windows

- Here are required tools information

Tool / Software	Version	Source / Vendor	Main function
Qualcomm Product Support Tool (QPST)	✓ 2.7.477 or later	QTI	✓ OS: Windows ✓ Download firmware images ✓ QFIL
Qualcomm SAP Configuration Tool	✓ 1.0.4 or later	QTI	✓ OS: Windows, Linux ✓ Configure numerous parameters for Linux Smart Speaker
Android SDK Tools	✓ r10 or higher ✓ ADB 1.0.39 or later	Android open-source project	✓ OS: Windows, Linux ✓ Host USB driver ✓ adb ✓ Fastboot

- For more details for QPST and QFIL, refer to:
 - [80-V1400-3] QUALCOMM PRODUCT SUPPORT TOOL (QPST) 2.7 USER GUIDE
 - [80-NN120-1] QUALCOMM FLASH IMAGE LOADER (QFIL) USER GUIDE



Section 3.1

Installing QPST

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Installing QPST

- To download QPST,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “QPST” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Click on ‘QPST.WIN.2.7 Installer’ item

Confidential
2019-12-30 or earlier
didi.setadi@praktikum.com

1

Search
keicho

Home Documents Hardware Components Software Code Tools Support Hardware Purchase

All Products | Edit

Tools: Choose a Tools Suite

Tools: Choose a Tools Suite

All Suites

2 results

QPST

3

Apply

Your Filters

Clear All

Visibility

External and Internal

System OS

Windows (3)

Full Name

Qualcomm® Development Acceleration Resource Toolkit (1)

Qualcomm® Development Acceleration Resource Toolkit for Manufacturers (1)

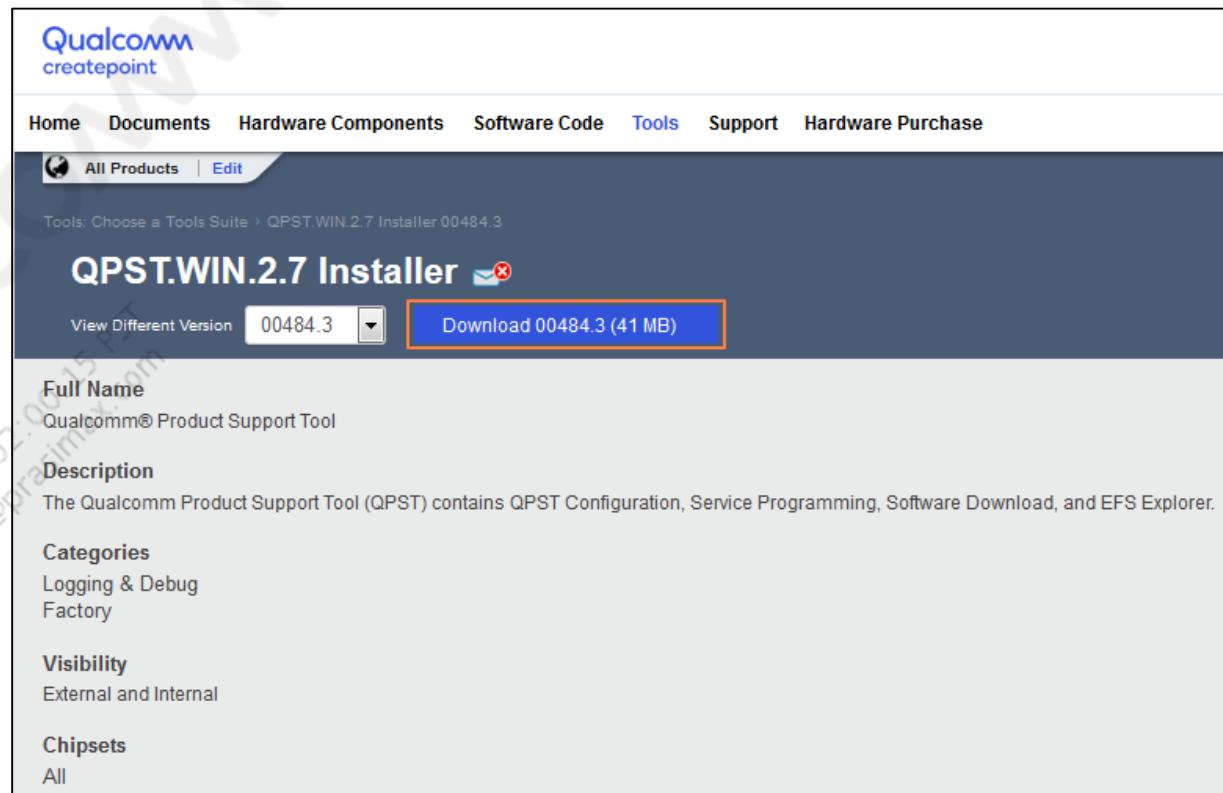
All Tools

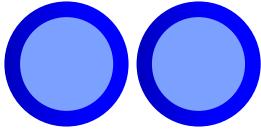
Subscribe Unsubscribe

Name	Full Name	Version	Release Date	
QDART.WIN.4.8 Installer	Qualcomm® Development Acceleration Resource Toolkit	00066.1	04-02-2019	
4	QPST.WIN.2.7 Installer	Qualcomm® Product Support Tool	00483.3	03-04-2019
QDART_MFG.WIN.4.9 Installer	Qualcomm® Development Acceleration Resource Toolkit for Manufacturers	00007.15	12-11-2014	

Installing QPST

- To download QPST,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “QPST” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Click on ‘QPST.WIN.2.7 Installer’ item
- Click on ‘Download’ button to start downloading
 - A compressed file will be downloaded
- Unzip and run setup file (QPST.<version>.exe) to start installation and follow the guidelines for a successful installation
- DONE!





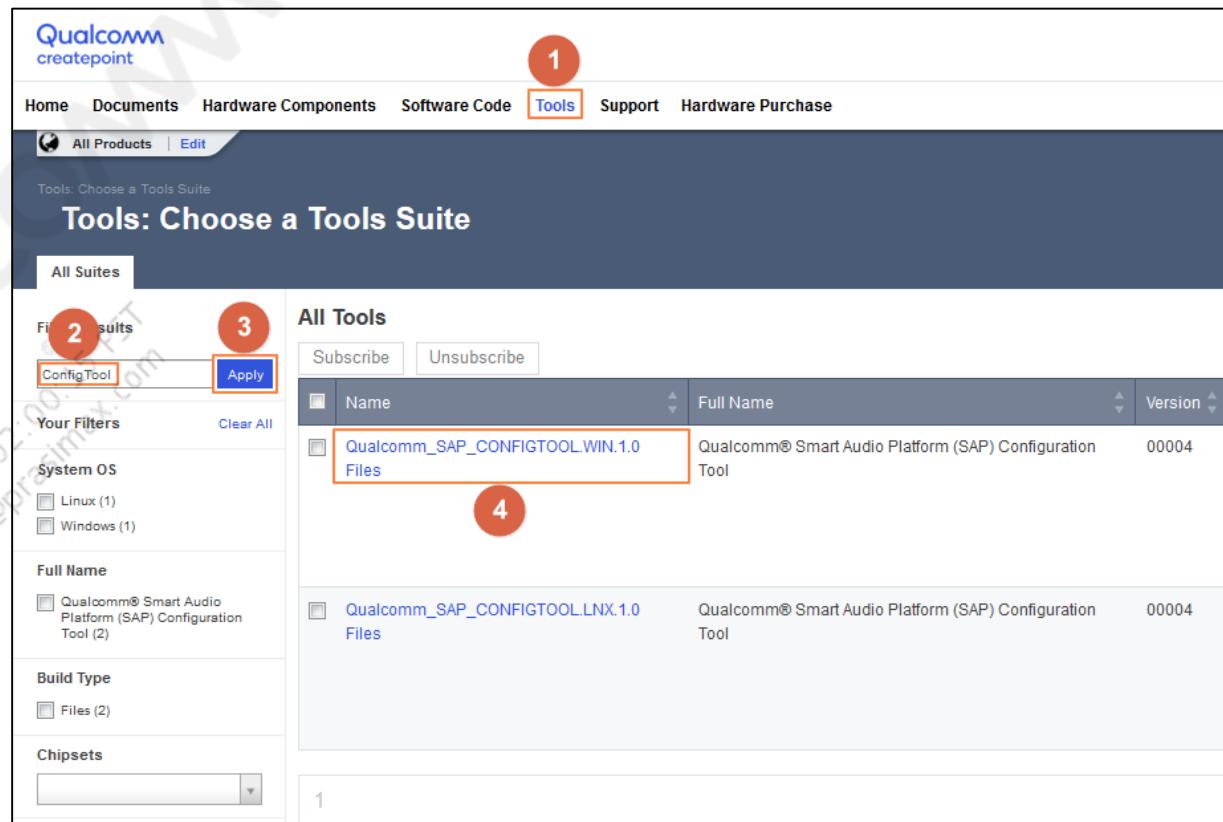
Section 3.2

Installing Qualcomm Smart Audio Platform Configuration Tool

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

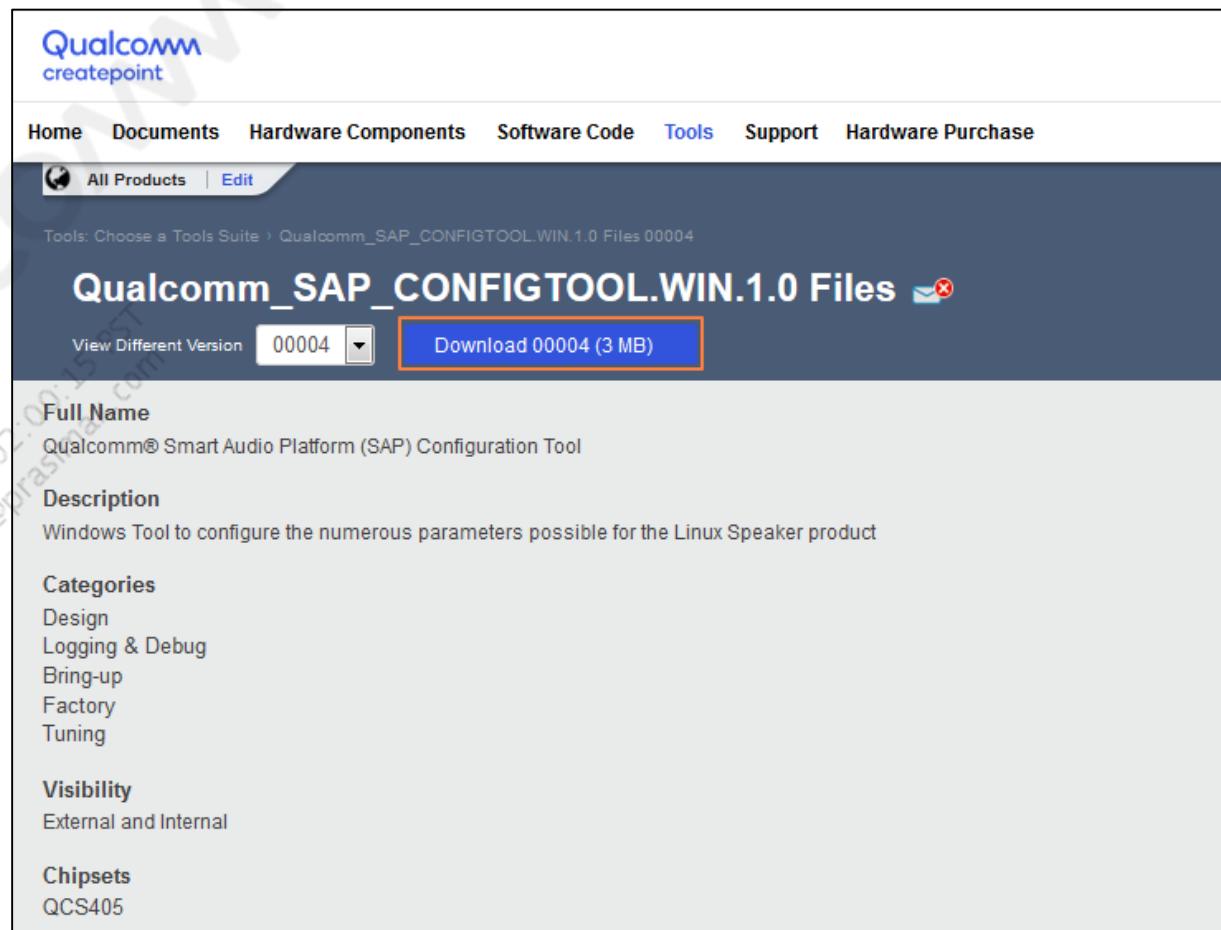
Installing Qualcomm Smart Audio Platform Configuration Tool

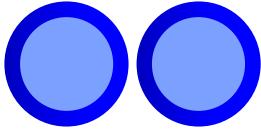
- To download SAP Configuration Tool,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “ConfigTool” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Click on ‘Qualcomm_SAP_CONFIGTOOL.WIN.1.0 Files’ item



Installing Qualcomm Smart Audio Platform Configuration Tool

- To download SAP Configuration Tool,
 - Go : <https://createpoint.qti.qualcomm.com/dashboard/>
- (1) Click on ‘Tools’ tab
- (2) Input “ConfigTool” into ‘Filter Results’
- (3) Click on ‘Apply’ button
- (4) Click on ‘Qualcomm_SAP_CONFIGTOOL.WIN.1.0 Files’ item
- Click on ‘Download’ button to start downloading
 - A compressed file will be downloaded
- Unzip and run setup file (QualcommDeviceConfigTool-<version>.exe) to start installation and follow the guidelines for a successful installation
- DONE!
- NOTE: There is also SAP Configuration Tool for Linux





Section 3.3

Installing Android SDK Tools for Windows

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

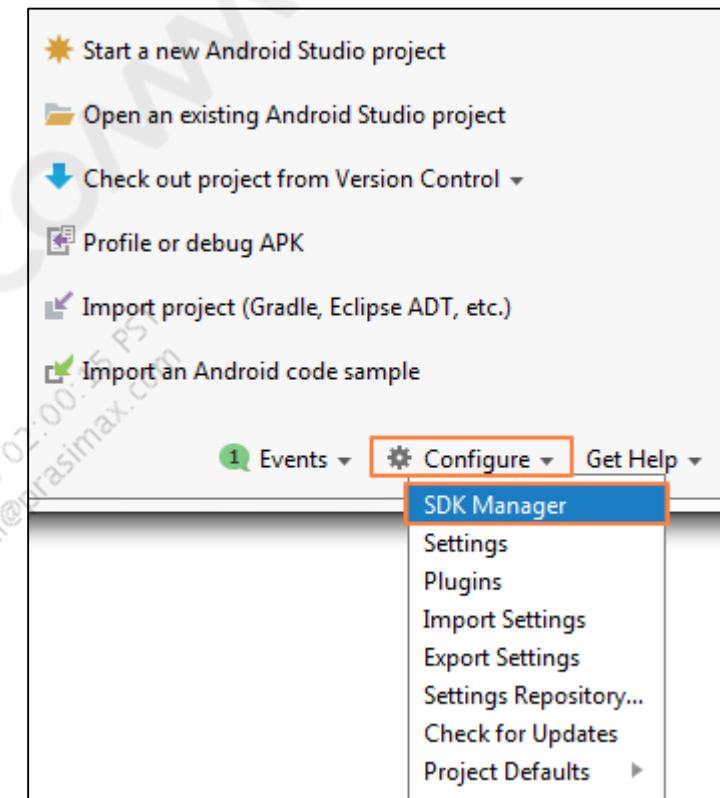
Installing Android SDK Tools for Windows

- Android Studio installation is recommended to use Android SDK Tools
- To download Android Studio, refer to:
 - <https://developer.android.com/studio>
 - <https://developer.android.com/studio/install>

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

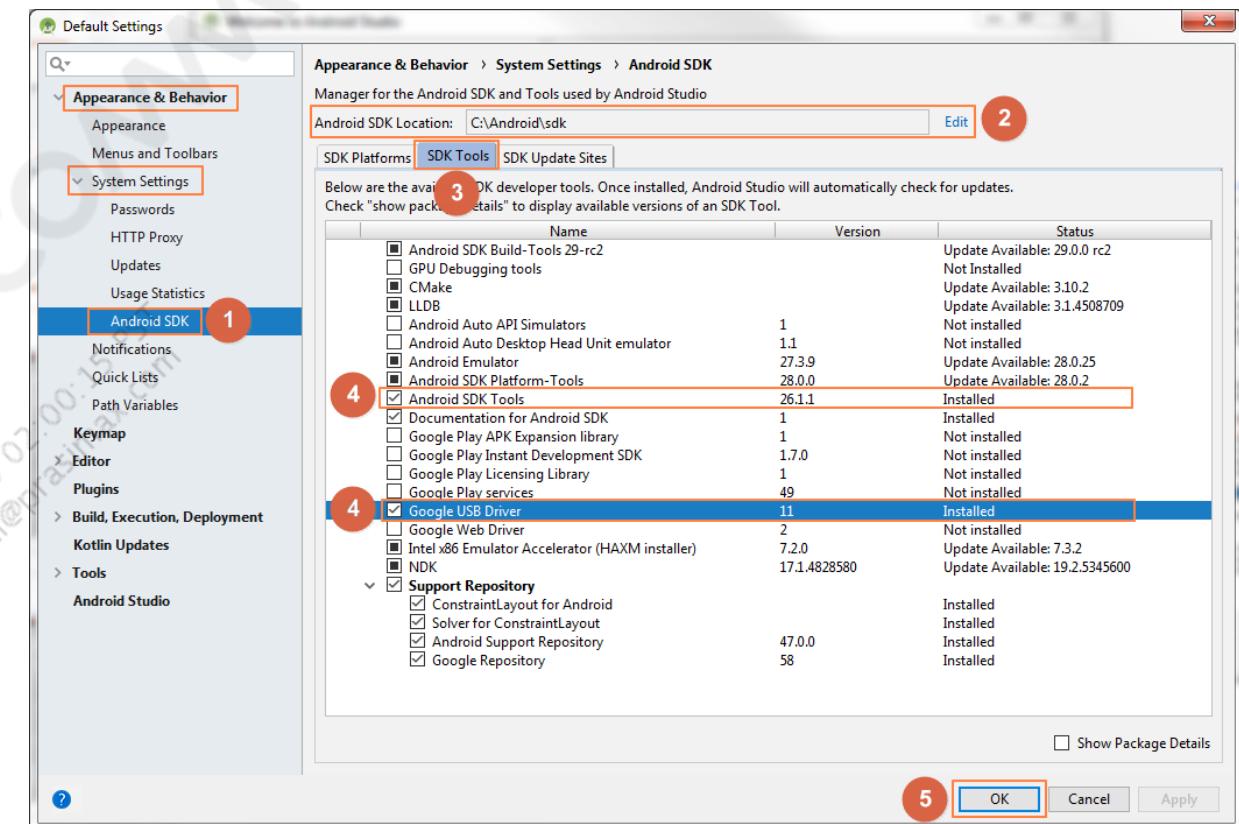
Installing Android SDK Tools for Windows

- When Android Studio is launched, 'Welcome to Android Studio' window appears
- Click on 'Configure' > 'SDK Manager' menu



Installing Android SDK Tools for Windows

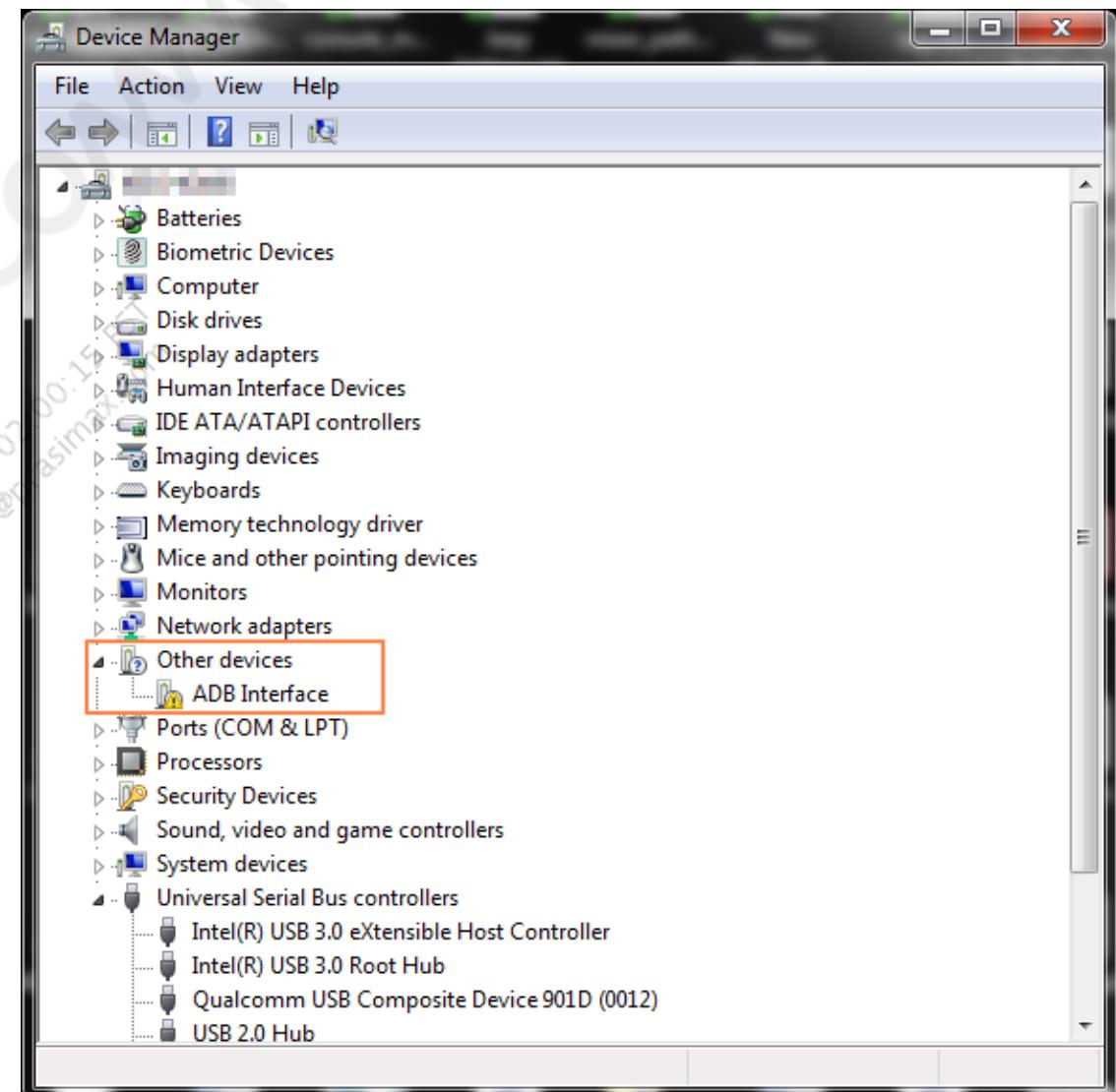
- When Android Studio is launched, ‘Welcome to Android Studio’ window appears
- Click on ‘Configure’ > ‘SDK Manager’ menu
- (1) On the left tree control, Select ‘Appearance & Behavior > System Settings > Android SDK’
- (2) Select ‘Android SDK Location’ with Edit button
- (3) Click on ‘SDK Tools’ tab
- (4) Check on ‘Android SDK Tools’ and ‘Google USB Driver’ to install adb, fastboot and usb driver
- (5) Click on ‘OK’ button to install the required tools
- DONE !



Installing Android SDK Tools for Windows – Install USB Driver (adb)

- After the device is plugged in, open Windows ‘Device Manager’ to install USB Driver for adb interface

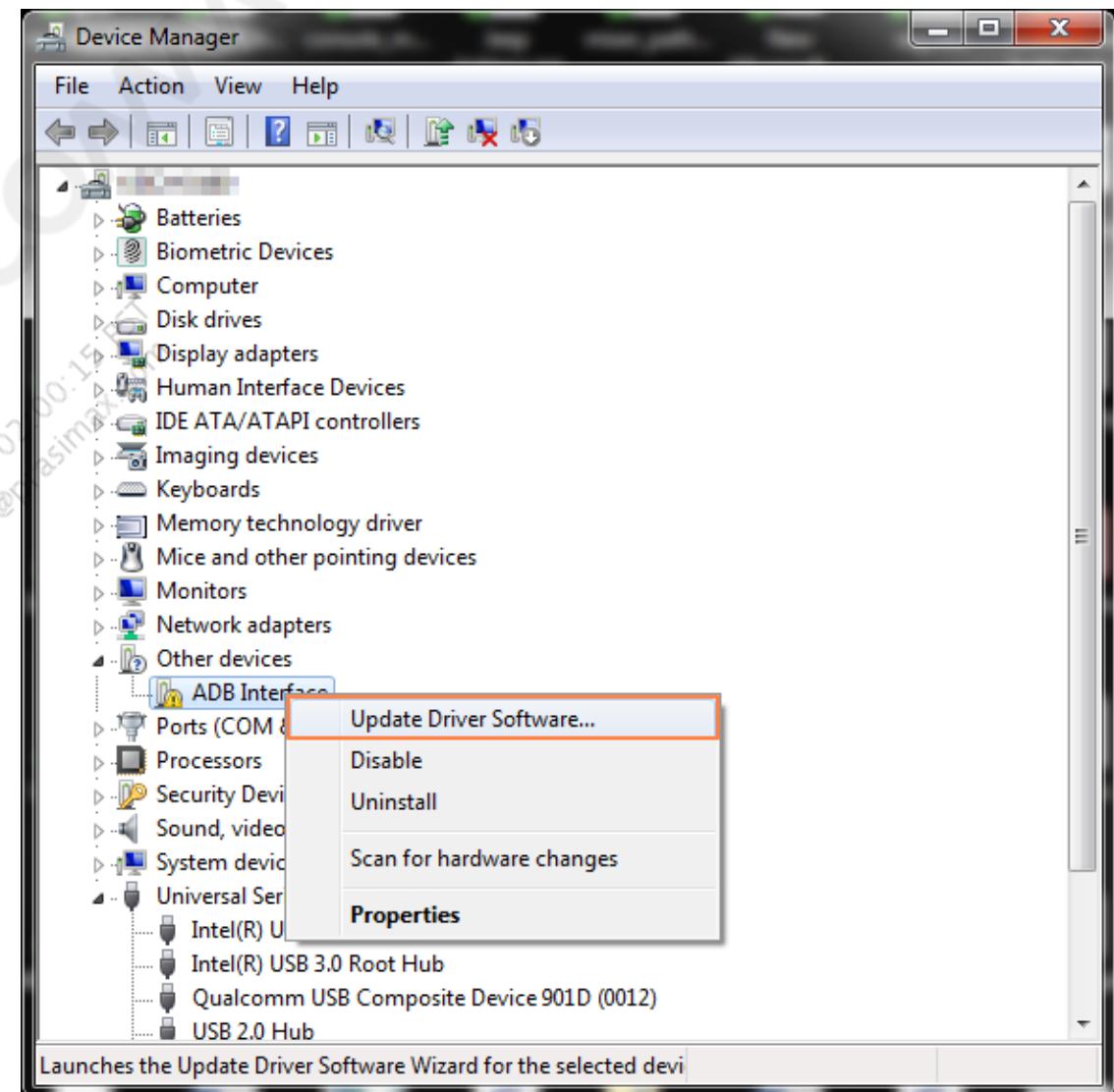
Confidential
2019-12-30 01:00:15
didi.setiadi@rasimq.com



Installing Android SDK Tools for Windows – Install USB Driver (adb)

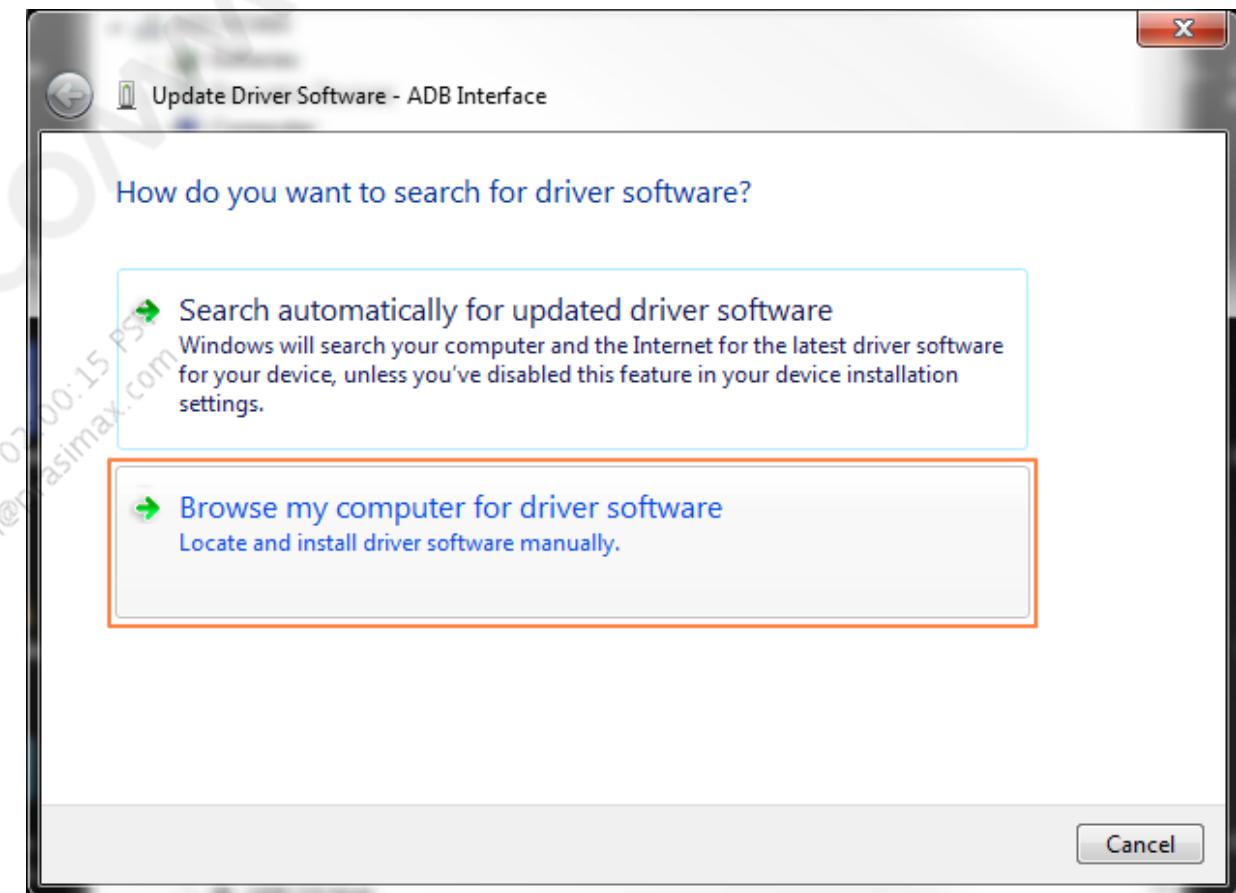
- After the device is plugged in, open Windows ‘Device Manager’ to install USB Driver for adb interface
- Right-click on ‘ADB Interface’ and select ‘Update Driver Software...’

Confidential
2019-12-30 01:00:15
didi.setiadi@rasimq.com



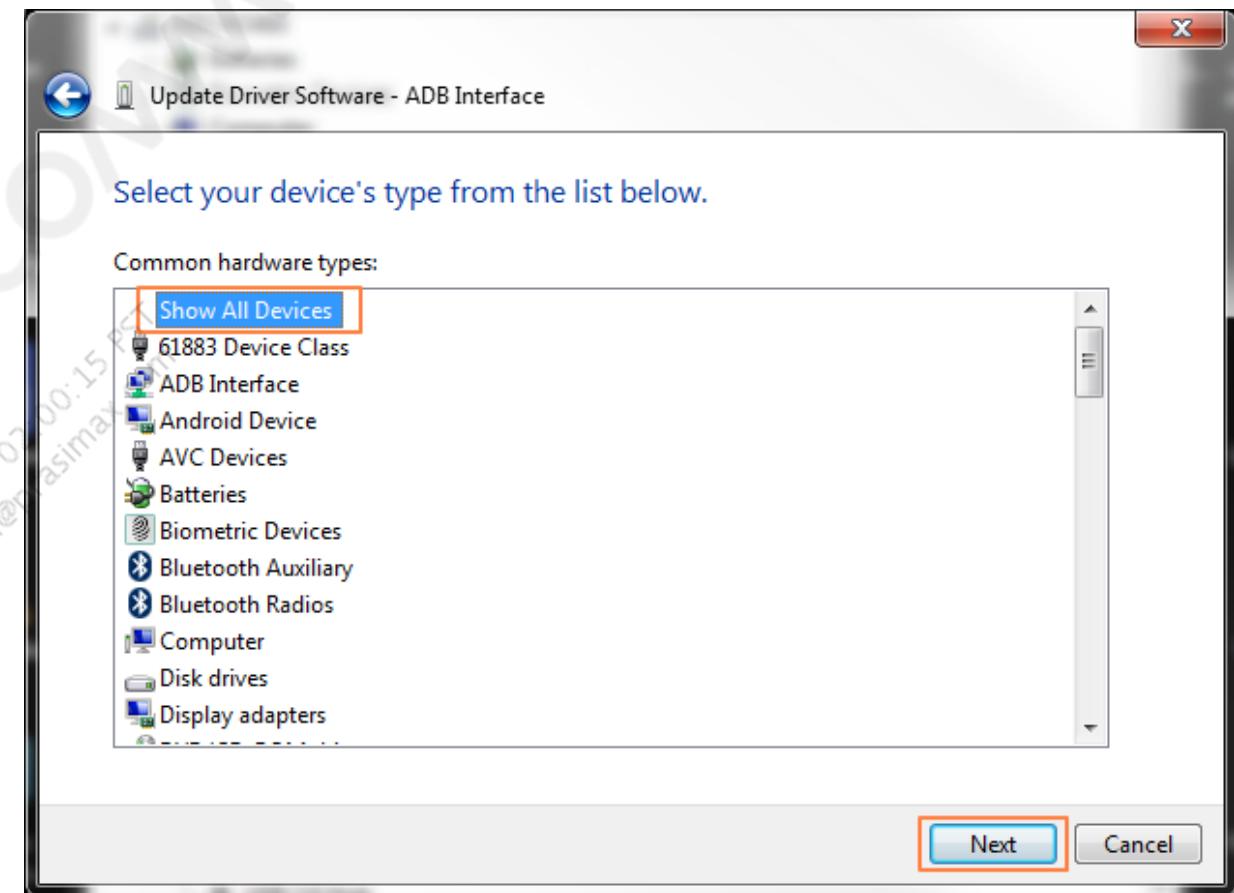
Installing Android SDK Tools for Windows – Install USB Driver (adb)

- After the device is plugged in, open Windows ‘Device Manager’ to install USB Driver for adb interface
- Right-click on ‘ADB Interface’ and select ‘Update Driver Software...’
- Click on ‘Browse my computer for driver software’



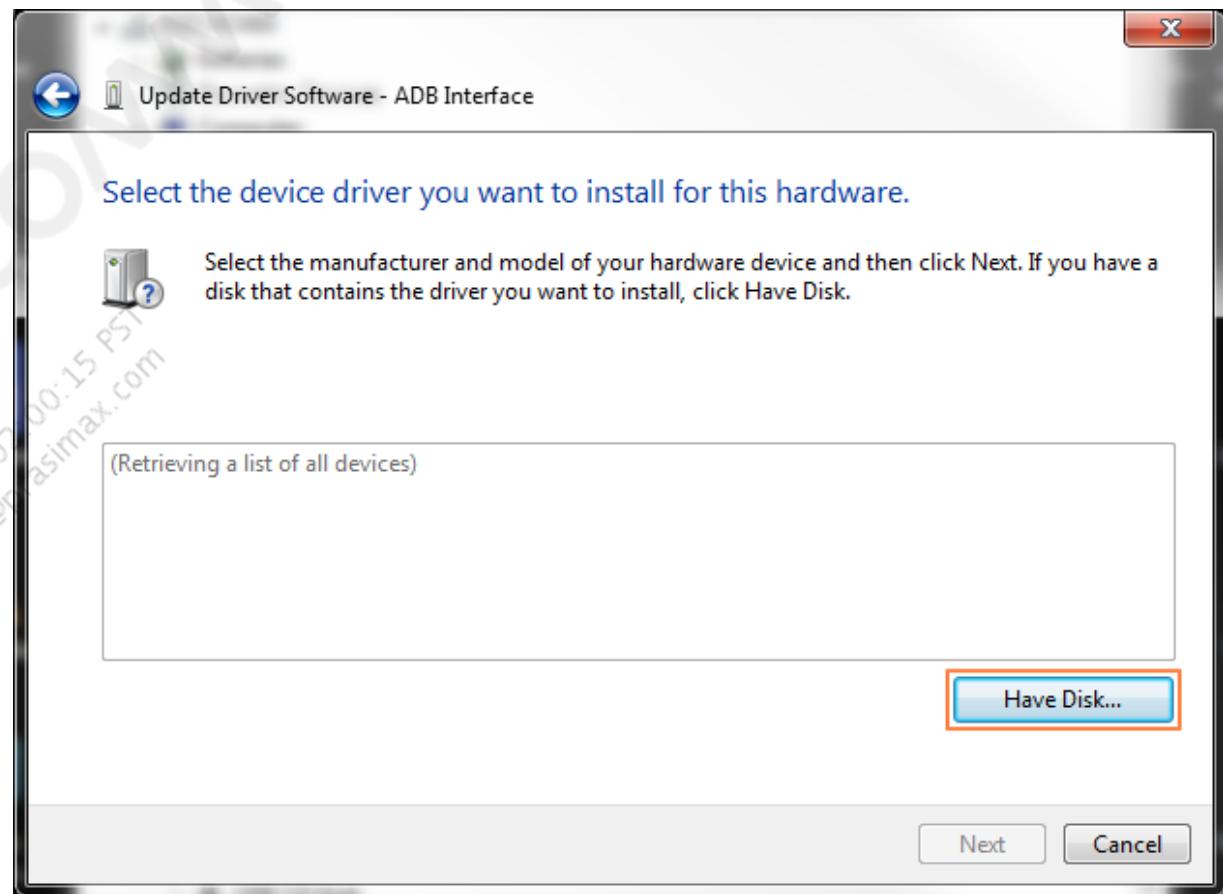
Installing Android SDK Tools for Windows – Install USB Driver (adb)

- After the device is plugged in, open Windows ‘Device Manager’ to install USB Driver for adb interface
- Right-click on ‘ADB Interface’ and select ‘Update Driver Software…’
- Click on ‘Browse my computer for driver software’
- Select ‘Show All Devices’ and Click on ‘Next’ button



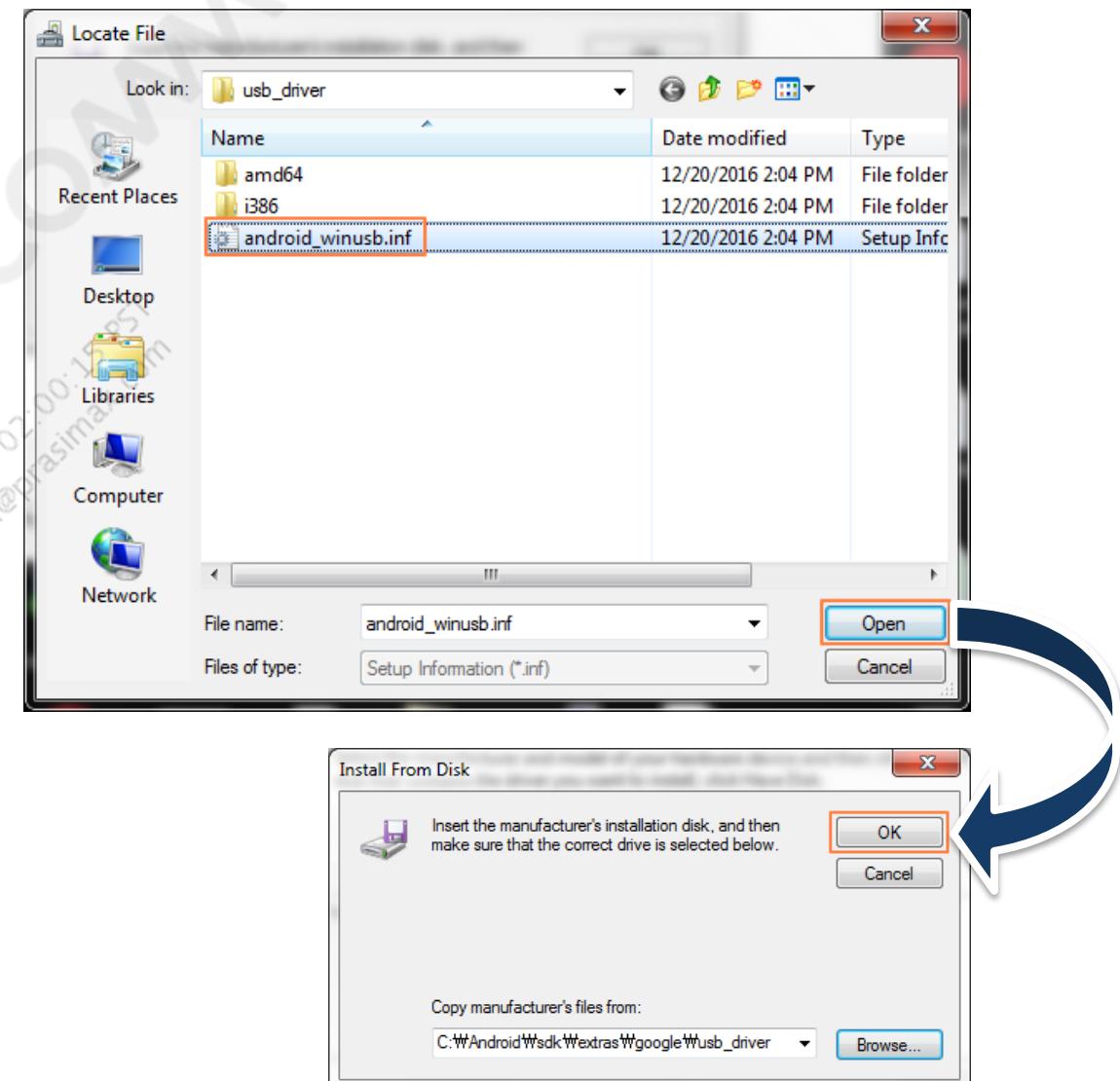
Installing Android SDK Tools for Windows – Install USB Driver (adb)

- After the device is plugged in, open Windows ‘Device Manager’ to install USB Driver for adb interface
- Right-click on ‘ADB Interface’ and select ‘Update Driver Software…’
- Click on ‘Browse my computer for driver software’
- Select ‘Show All Devices’ and Click on ‘Next’ button
- Click on ‘Have Disk…’ button



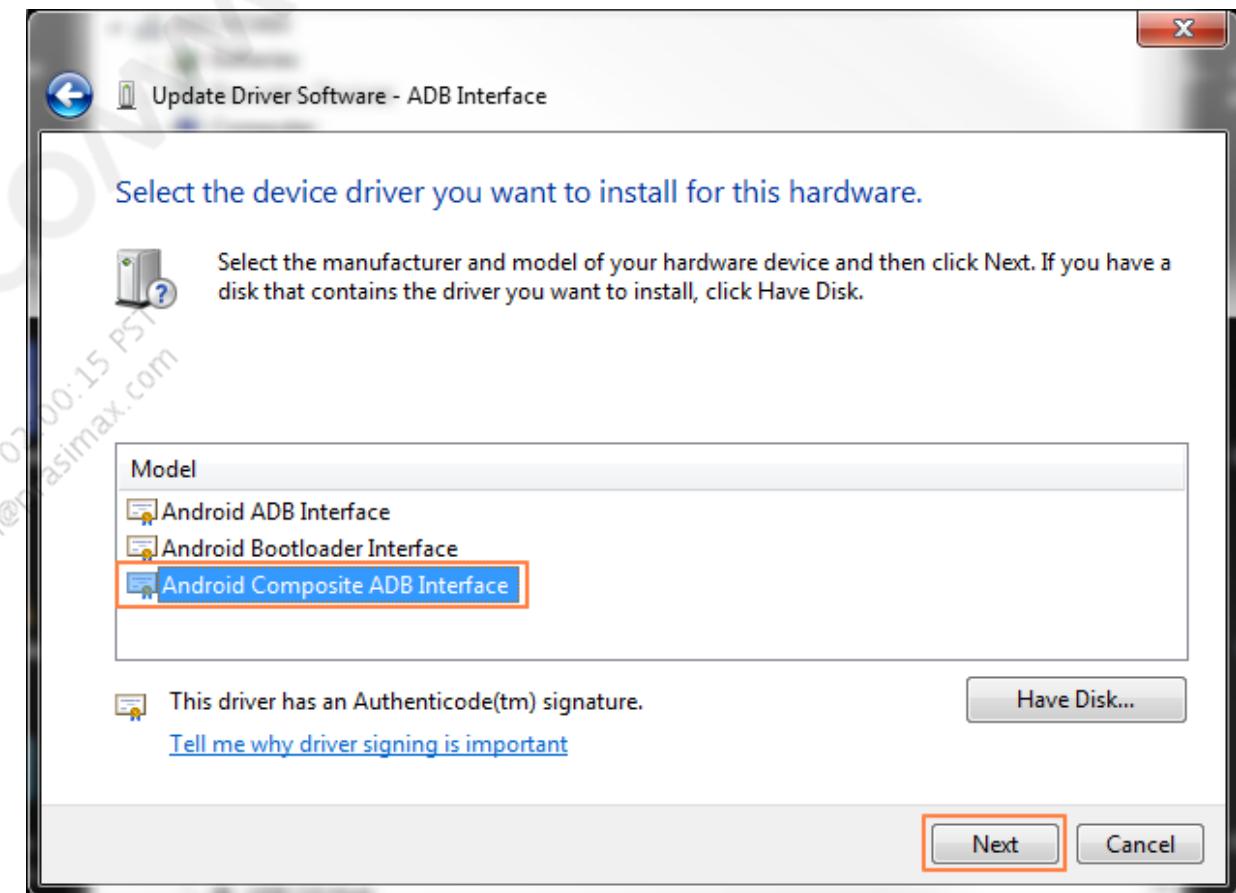
Installing Android SDK Tools for Windows – Install USB Driver (adb)

- After the device is plugged in, open Windows ‘Device Manager’ to install USB Driver for adb interface
- Right-click on ‘ADB Interface’ and select ‘Update Driver Software…’
- Click on ‘Browse my computer for driver software’
- Select ‘Show All Devices’ and Click on ‘Next’ button
- Click on ‘Have Disk…’ button
- Select ‘**android_winusb.inf**’ file on ‘<Android SDK Location>\extras\google\usb_driver\’ directory and then click on ‘Open’ button
- Click on ‘OK’ button



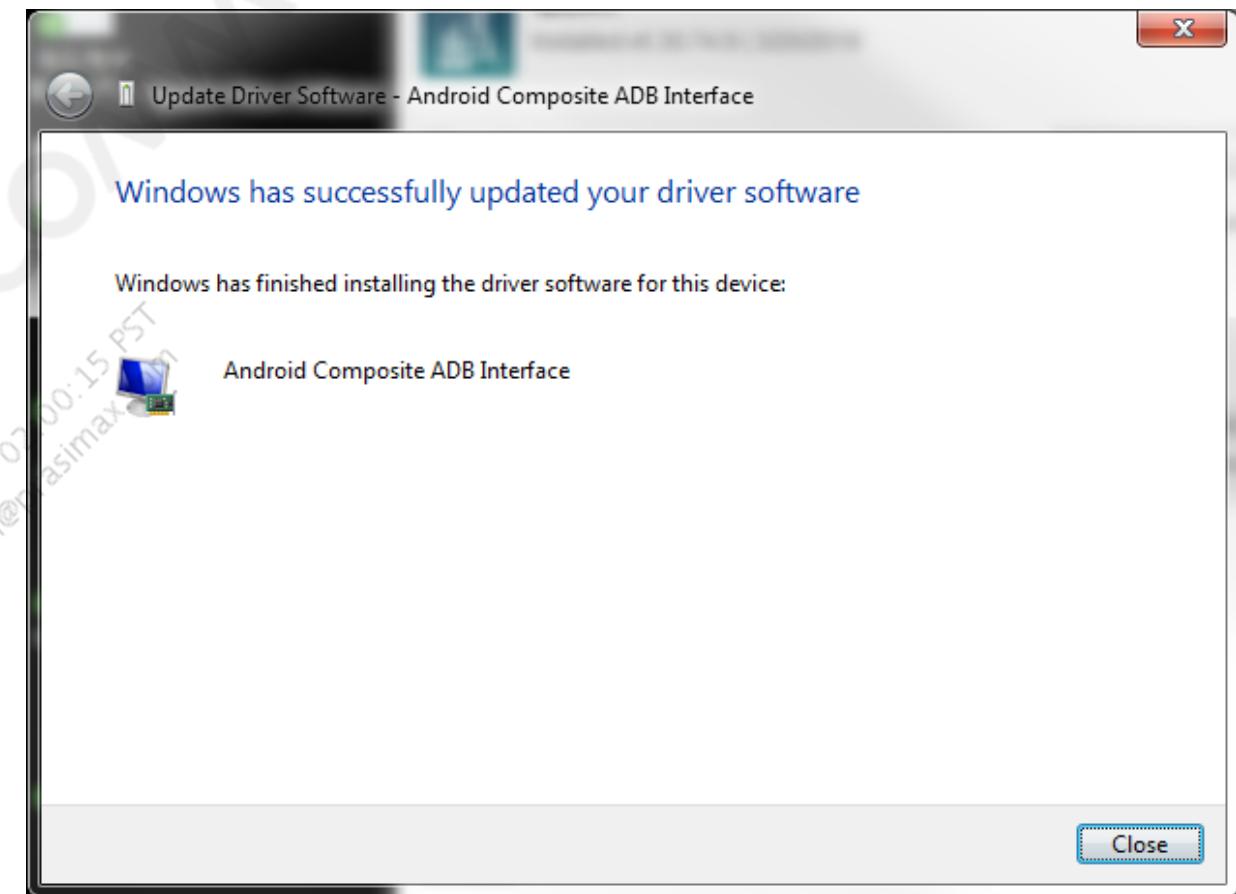
Installing Android SDK Tools for Windows – Install USB Driver (adb)

- After the device is plugged in, open Windows ‘Device Manager’ to install USB Driver for adb interface
- Right-click on ‘ADB Interface’ and select ‘Update Driver Software…’
- Click on ‘Browse my computer for driver software’
- Select ‘Show All Devices’ and Click on ‘Next’ button
- Click on ‘Have Disk…’ button
- Select ‘android_winusb.inf’ file on ‘<Android SDK Location>\extras\google\usb_driver\’ directory and then click on ‘Open’ button
- Click on ‘OK’ button
- Select ‘Android Composite ADB Interface’ and Click on ‘Next’



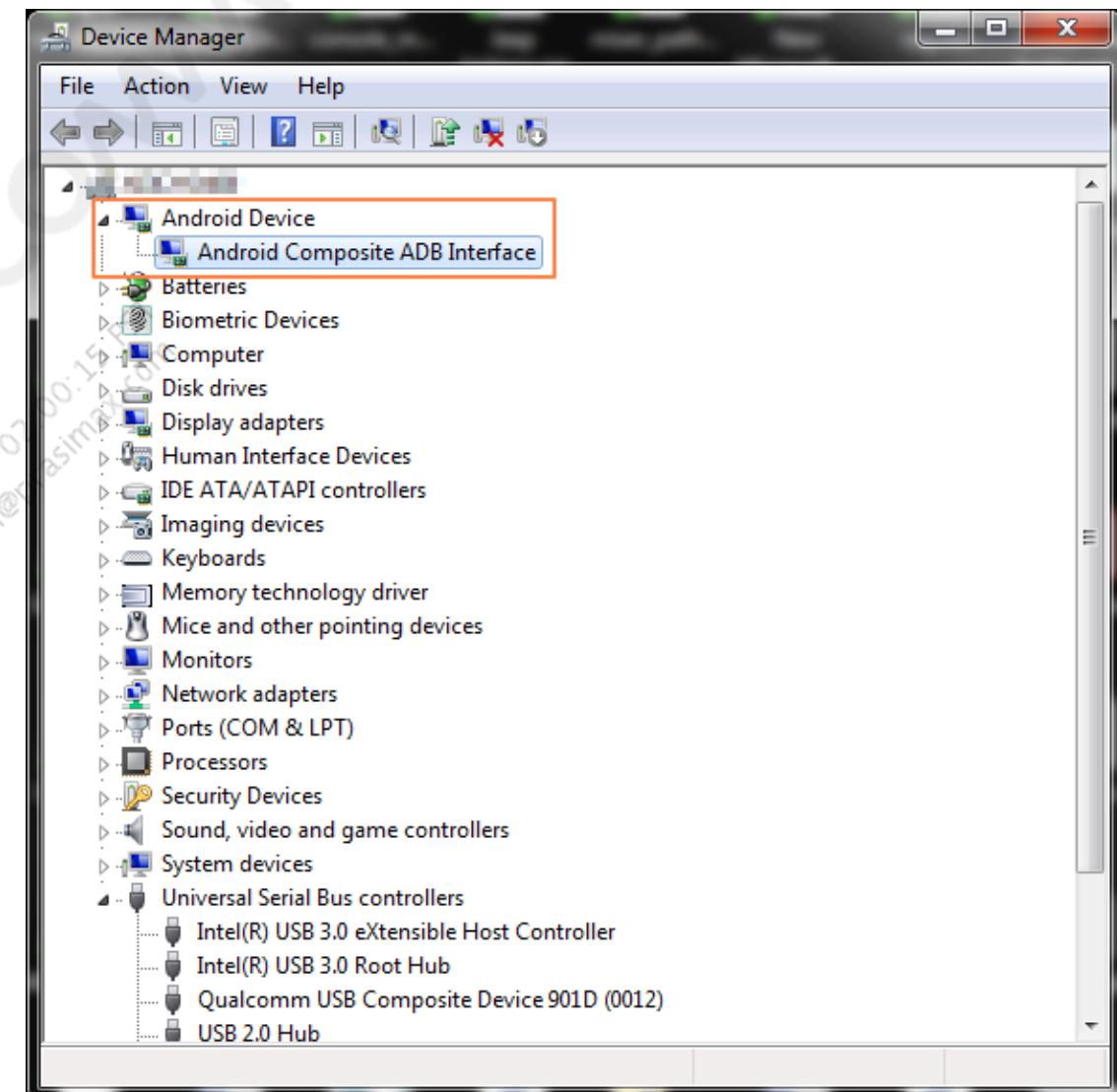
Installing Android SDK Tools for Windows – Install USB Driver (adb)

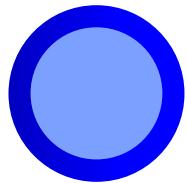
- After the device is plugged in, open Windows ‘Device Manager’ to install USB Driver for adb interface
- Right-click on ‘ADB Interface’ and select ‘Update Driver Software…’
- Click on ‘Browse my computer for driver software’
- Select ‘Show All Devices’ and Click on ‘Next’ button
- Click on ‘Have Disk…’ button
- Select ‘android_winusb.inf’ file on ‘<Android SDK Location>\extras\google\usb_driver\’ directory and then click on ‘Open’ button
- Click on ‘OK’ button
- Select ‘Android Composite ADB Interface’ and Click on ‘Next’
- Make sure the installation success message



Installing Android SDK Tools for Windows – Install USB Driver (adb)

- After the device is plugged in, open Windows ‘Device Manager’ to install USB Driver for adb interface
- Right-click on ‘ADB Interface’ and select ‘Update Driver Software…’
- Click on ‘Browse my computer for driver software’
- Select ‘Show All Devices’ and Click on ‘Next’ button
- Click on ‘Have Disk…’ button
- Select ‘android_winusb.inf’ file on ‘<Android SDK Location>\extras\google\usb_driver\’ directory and then click on ‘Open’ button
- Click on ‘OK’ button
- Select ‘Android Composite ADB Interface’ and Click on ‘Next’
- Make sure the installation success message
- Make sure the **ADB Interface** on ‘Device Manager’ window
- **DONE !**



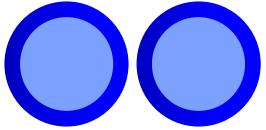


Section 4

Obtain Software

Qualcomm
Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

4.1 Software Components	79
4.2 Release Packages	83
4.3 QTI Proprietary Software	88
4.4 Open Source Software	101
4.5 Merge / Reconstruct Software	104
4.5.1 Merge Software	106
4.5.2 Reconstruct Software	110
4.6 3rd Party Software Integration	116



Section 4.1

Software Components

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Software Components

- QCS40x software consists of following categories
 - HLOS software
 - non-HLOS software

	Category	SW Modules
QCS40x software	HLOS	
	non-HLOS	

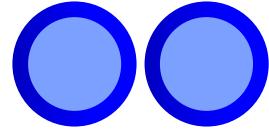
Software Components

- QCS40x software consists of following categories
 - HLOS software
 - non-HLOS software
 - HLOS software includes the following SW module
 - APPS_PROC

Software Components

- QCS40x software consists of following categories
 - HLOS software
 - non-HLOS software
- HLOS software includes the following SW module
 - APPS_PROC
- non-HLOS software includes the following SW modules
 - COMMON : Build environment to generate final images
 - BTFM_PROC : Bluetooth / FM SoC firmware and NVM files
 - ADSP_PROC : Audio DSP
 - CDSP_PROC : Computational DSP
 - RPM_PROC : Resource Power Management
 - WLAN_PROC : Wi-Fi firmware image and board profile
 - BOOT_IMAGES : Boot loader
 - TRUSTZONE_IMAGES : For more details, refer to ‘Qualcomm® Secure Execution Environment (QSEE)’.

	Category	SW Modules
QCS40x software	HLOS	APPS_PROC
	non-HLOS	COMMON
		BTFM_PROC
		ADSP_PROC
		CDSP_PROC
		RPM_PROC
		WLAN_PROC
		BOOT_IMAGES
		TRUSTZONE_IMAGES



Section 4.2

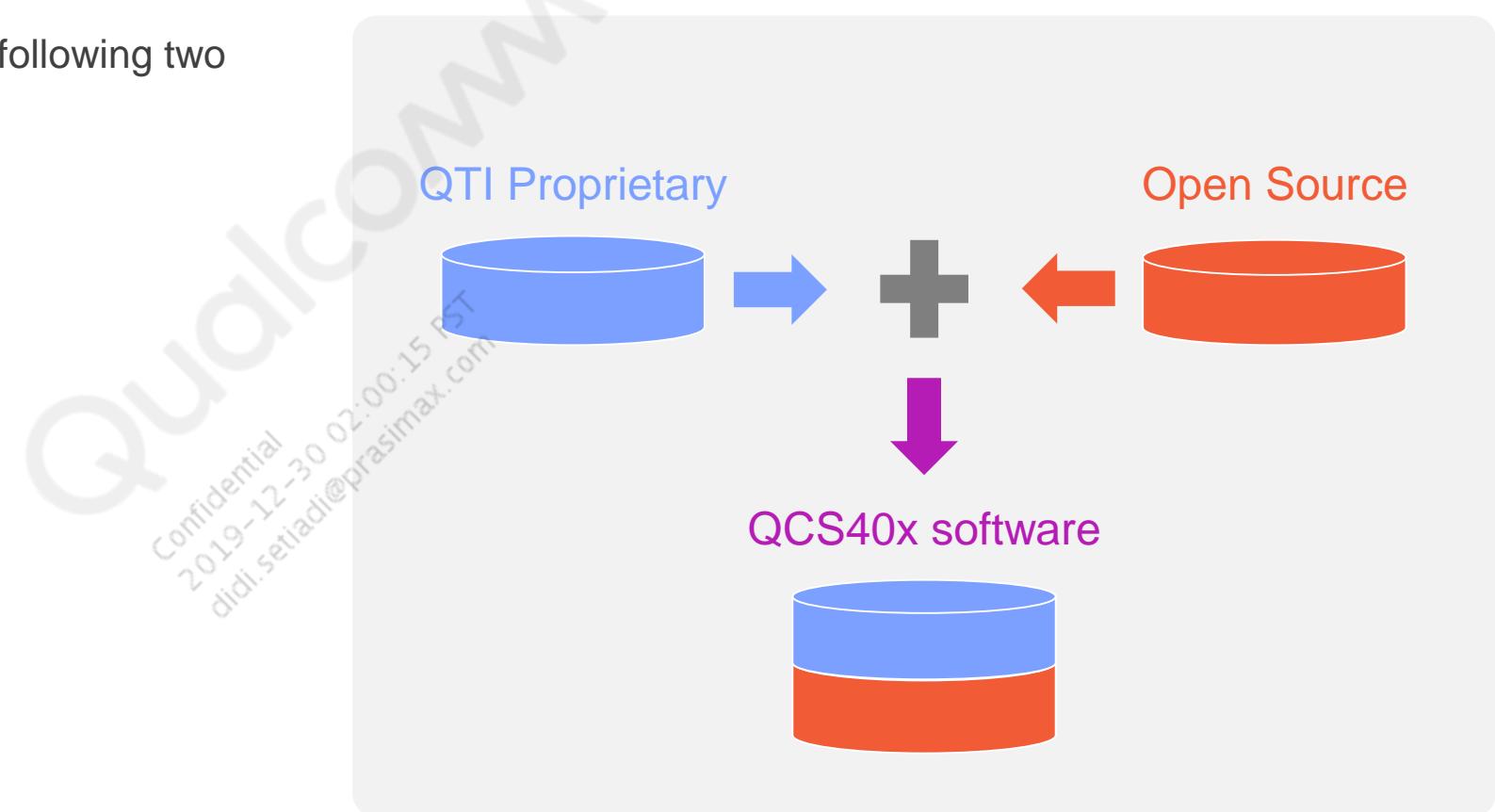
Release Packages

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Release Packages

- QCS40x software release consists of the following two packages
 - QTI Proprietary Software
 - Qualcomm ChipCode™ portal
 - <https://chipcode.qti.qualcomm.com>
 - Download with WEB browser
 - Open Source Software
 - Code Aurora Foundation (CAF)
 - <https://www.codeaurora.org>
 - Sync to source server with ‘repo’ command



Release Packages

- QCS40x software release consists of the following two packages
 - QTI Proprietary Software
 - ChipCode portal
 - <https://chipcode.qti.qualcomm.com>
 - Download with WEB browser
 - Open Source Software
 - Code Aurora Foundation (CAF)
 - <https://www.codeaurora.org>
 - Sync to source server with 'repo' command
- Detail components for each release package

Release Package	Category	Components
ChipCode	HLOS	<input type="checkbox"/> Proprietary source and firmware images for the applications processor HLOS
	non-HLOS	<input type="checkbox"/> Proprietary source and firmware images for all the non applications processors (non-apps) <input type="checkbox"/> An umbrella package built from a combined set of integrated individual component releases
CAF	HLOS	<input type="checkbox"/> Open source for the applications processor HLOS
	non-HLOS	-

Release Packages

- QCS40x software release consists of the following two packages

- QTI Proprietary Software
 - ChipCode portal
 - <https://chipcode.qti.qualcomm.com>
 - Download with WEB browser
 - Open Source Software
 - Code Aurora Foundation (CAF)
 - <https://www.codeaurora.org>
 - Sync to source server with 'repo' command

- Detail components for each release package
- Release package for each SW Module

Qualcomm Confidential
2019-12-30 02:00:15 PST
didi.setiadi@pradaidc.com

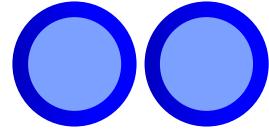
	Category	SW Modules	Release Package
QCS40x software	HLOS	APPS_PROC	ChipCode
		COMMON	CAF
	non-HLOS	BTFM_PROC	ChipCode
		ADSP_PROC	
		CDSP_PROC	
		RPM_PROC	
		WLAN_PROC	
		BOOT_IMAGES	
		TRUSTZONE_IMAGES	

Release Packages

- There are three types of packages as below
 - ap_standard_oem: for system software developer
 - hlos_dev: for HLOS software developer
 - test_device: for tester (demo, board bring-up, factory test, etc)
- Each package consists as shown in the table right
 - Source: Build is required
 - Binary: Build is NOT required

Confidential
2019-12-30 02:00 UTC
didi.setiadi@praktikum.ugm.ac.id

SW Modules	ap_standard_oem	hlos_dev	test_device
APPS_PROC	Source	Source	Source & Library
COMMON	Source	Source	Source
BTFM_PROC	Binary	Binary	Binary
ADSP_PROC	Source	Binary	Binary
CDSP_PROC	Source	Binary	Binary
RPM_PROC	Source	Binary	Binary
WLAN_PROC	Binary	Binary	Binary
BOOT_IMAGES	Source	Binary	Binary
TRUSTZONE_IMAGES	Source	Binary	Binary



Section 4.3

QTI Proprietary Software

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

QTI Proprietary Software

- If user wants to access ChipCode for the first time,
 - Refer to <https://chipcode.qti.qualcomm.com/helpki>
- NOTE: This document will explain how to obtain software based on getting r1.0_00007.0

Confidential
2019-12-30 or earlier
didi.setiadi@prashantx.com

The screenshot shows the Qualcomm ChipCode help page. The top navigation bar includes a star icon, a question mark, a download icon, a user profile icon, and a 'Help' link. Below the header, there are tabs for 'Projects' and 'Activities'. A 'Topics' section contains a search bar with 'Filter Wikis...' placeholder text. The main content area is organized into several sections: 'Training' (with 'Getting Started'), 'Installing Git' (with 'Installing Git on Linux' and 'Installing Git on Windows'), 'Upgrading Git' (with 'Upgrading Git on Linux' and 'Upgrading Git on Windows'), 'Notifications' (with 'ChipCode Email Notifications'), and 'Downloading' (with 'Shallow Cloning', 'Cloning Code from a Repository', and 'Download Delta'). Each topic item includes a timestamp indicating it was last edited almost 5 years ago.

QTI Proprietary Software

- To download QTI proprietary software,
 - Go: <https://createpoint.qti.qualcomm.com/dashboard/>

Confidential
2019-12-30 or earlier
didi.setiadi@qualcomm.com

The screenshot shows the Qualcomm CreatePoint interface. At the top, there's a navigation bar with links for Home, Documents, Hardware Components, Software Code, Tools, Support, and Hardware Purchase. Below that is a sub-navigation bar with links for All Products and Edit. The main content area is titled "Product Kits". On the left, there's a sidebar with a dropdown menu for "All Categories" and a list of "All Product Types" including 60GHz/WiGig, Access Point, Amplifier, Audio, Audio Player, Basic Phone, Bluetooth, Bluetooth/Wi-Fi, Cellular Modem, Charging, Computer Vision, and Computing. The main right-hand panel is titled "Product Kits: All Product Types" and includes filters for "Filter List by Product" and "Kit Level". It lists various product kits with their details and status indicators (e.g., APQ8009 Android Smart Speaker (Developer), APQ8009 Android Things Smart Speaker (Developer)).

QTI Proprietary Software

- To download QTI proprietary software,
 - Go: <https://createpoint.qti.qualcomm.com/dashboard/>
- Click on ‘Software Code’ tab

Qualcomm
createpoint

Home Documents Hardware Components **Software Code** Tools Support Hardware Purchase

All Products | Edit

Software Code: Download Code from ChipCode

ChipCode Software Packages Release History Release Notes Release Planner Tracker Multi-Product Planner

Specify a Software Product

Customer Software Product

Filter Results Reset all Filters

Your Filters Clear All

License Type amss (23418)

Security Level standard (80803) restricted (787) secret (2)

81593 Software Product Distros in View

Download This Page Download All Results

Customer	Distro
qualcomm	whs9410.qurt.1.0.test.device
sony-mobile-communications-inc	whs9410.qurt.1.0.test.device
hisense-group-co-ltd	whs9410.qurt.1.0.test.device
microsoft-corporation	whs9410.qurt.1.0.test.device
asustek-computer-inc	whs9410.qurt.1.0.test.device
huaqin-telecom-technology-co-ltd-shanghai	whs9410.qurt.1.0.test.device
high-tech-computer-corporation	whs9410.qurt.1.0.test.device
he-yuan-he-yuan-electronics-co-ltd	whs9410.qurt.1.0.test.device
universal-global-technology-kunshan-co-ltd	whs9410.qurt.1.0.test.device
qiku-internet-network-scientific-shenzhen-co-ltd	whs9410.qurt.1.0.test.device
fih-mobile-limited	whs9410.qurt.1.0.test.device
shanghai-wind-science-and-technologies-co-ltd	whs9410.qurt.1.0.test.device
smartisan-technology-co-ltd	whs9410.qurt.1.0.test.device
shenzhen-unicair-communication-technology-co-ltd	whs9410.qurt.1.0.test.device

QTI Proprietary Software

- To download QTI proprietary software,
 - Go: <https://createpoint.qti.qualcomm.com/dashboard/>
- Click on ‘Software Code’ tab
- Click on ‘Software Product’ combo box

The screenshot shows the Qualcomm createpoint interface for managing software code. The top navigation bar includes Home, Documents, Hardware Components, Software Code, Tools, Support, and Hardware Purchase. The Software Code tab is selected. Below it, the 'Download Code from ChipCode' section is visible. On the left, there's a sidebar with a 'Specify a Software Product' dropdown set to 'Customer', another dropdown for 'Software Product' (which is highlighted with a red box), and sections for 'Filter Results', 'Your Filters', 'License Type' (with 'amss (23418)' selected), and 'Security Level' (with 'standard (80803)', 'restricted (787)', and 'secret (2)' options). The main content area displays a table titled '81593 Software Product Distros in View'. The table has two columns: 'Customer' and 'Distro'. Numerous entries in the 'Customer' column are followed by a blue link 'whs9410.qurt.1.0.test.device'. Examples include 'qualcomm', 'sony-mobile-communications-inc', 'hisense-group-co-ltd', 'microsoft-corporation', 'asustek-computer-inc', 'huaqin-telecom-technology-co-ltd-shanghai', 'high-tech-computer-corporation', 'he-yuan-he-yuan-electronics-co-ltd', 'universal-global-technology-kunshan-co-ltd', 'qiku-internet-network-scientific-shenzhen-co-ltd', 'fih-mobile-limited', 'shanghai-wind-science-and-technologies-co-ltd', 'smartisan-technology-co-ltd', and 'shenzhen-unicair-communication-technology-co-ltd'.

QTI Proprietary Software

- To download QTI proprietary software,
 - Go: <https://createpoint.qti.qualcomm.com/dashboard/>
- Click on ‘Software Code’ tab
- Click on ‘Software Product’ combo box
- Input ‘QCS40X’ into the search box
- Click on ‘QCS40X_2018.SPF.1.0’ item

Software Code: Download Code from ChipCode

81593 Software Product Distros in View

Customer	Distro
qualcomm	whs9410.qurt.1.0.test.device
sony-mobile-communications-inc	whs9410.qurt.1.0.test.device
hisense-group-co-ltd	whs9410.qurt.1.0.test.device
microsoft-corporation	whs9410.qurt.1.0.test.device
asustek-computer-inc	whs9410.qurt.1.0.test.device
huaqin-telecom-technology-co-ltd-shanghai	whs9410.qurt.1.0.test.device
high-tech-computer-corporation	whs9410.qurt.1.0.test.device
he-yuan-he-yuan-electronics-co-ltd	whs9410.qurt.1.0.test.device
universal-global-technology-kunshan-co-ltd	whs9410.qurt.1.0.test.device
qiku-internet-network-scientific-shenzhen-co-ltd	whs9410.qurt.1.0.test.device
fih-mobile-limited	whs9410.qurt.1.0.test.device
shanghai-wind-science-and-technologies-co-ltd	whs9410.qurt.1.0.test.device
smartisan-technology-co-ltd	whs9410.qurt.1.0.test.device
shenzhen-unicair-communication-technology-co-ltd	whs9410.qurt.1.0.test.device

QTI Proprietary Software

- To download QTI proprietary software,
 - Go: <https://createpoint.qti.qualcomm.com/dashboard/>
- Click on ‘Software Code’ tab
- Click on ‘Software Product’ combo box
- Input ‘QCS40X’ into the search box
- Click on ‘QCS40X_2018.SPF.1.0’ item
- All available QCS40x software packages will be shown

Confidential
2019-12-30 or earlier
didi.setiadi@praktikant.com

The screenshot shows the Qualcomm CreatePoint interface for downloading software code. The top navigation bar includes Home, Documents, Hardware Components, Software Code (which is selected), Tools, Support, and Hardware Purchase. Below the navigation is a search bar with 'All Products' and 'Edit' buttons. The main title is 'Software Code: Download Code from ChipCode'. On the left, there's a sidebar with tabs for ChipCode and Software Packages, along with buttons for Release History, Release Notes, Release Planner, Tracker, and Multi-Product Planner. The sidebar also contains filters for Customer (Qualcomm), Software Product (QCS40X_2018.SPF.1.0), and License Type (standard (10)). The main content area displays a table titled '10 Software Product Distros in View' with two download buttons at the top: 'Download This Page' and 'Download All Results'. The table lists ten entries, each with a customer name and a distribution name, followed by a download link icon.

10 Software Product Distros in View	
Download This Page	Download All Results
Customer	Distro
qualcomm	qcs40x.2018.spf.1.0.test.device
qualcomm	qcs40x.2018.spf.1.0.hlos.dev
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.virtualx
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.trumpetdsp
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dtsx
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbydsp
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbyddp
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbydap
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbyatmos
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem

QTI Proprietary Software

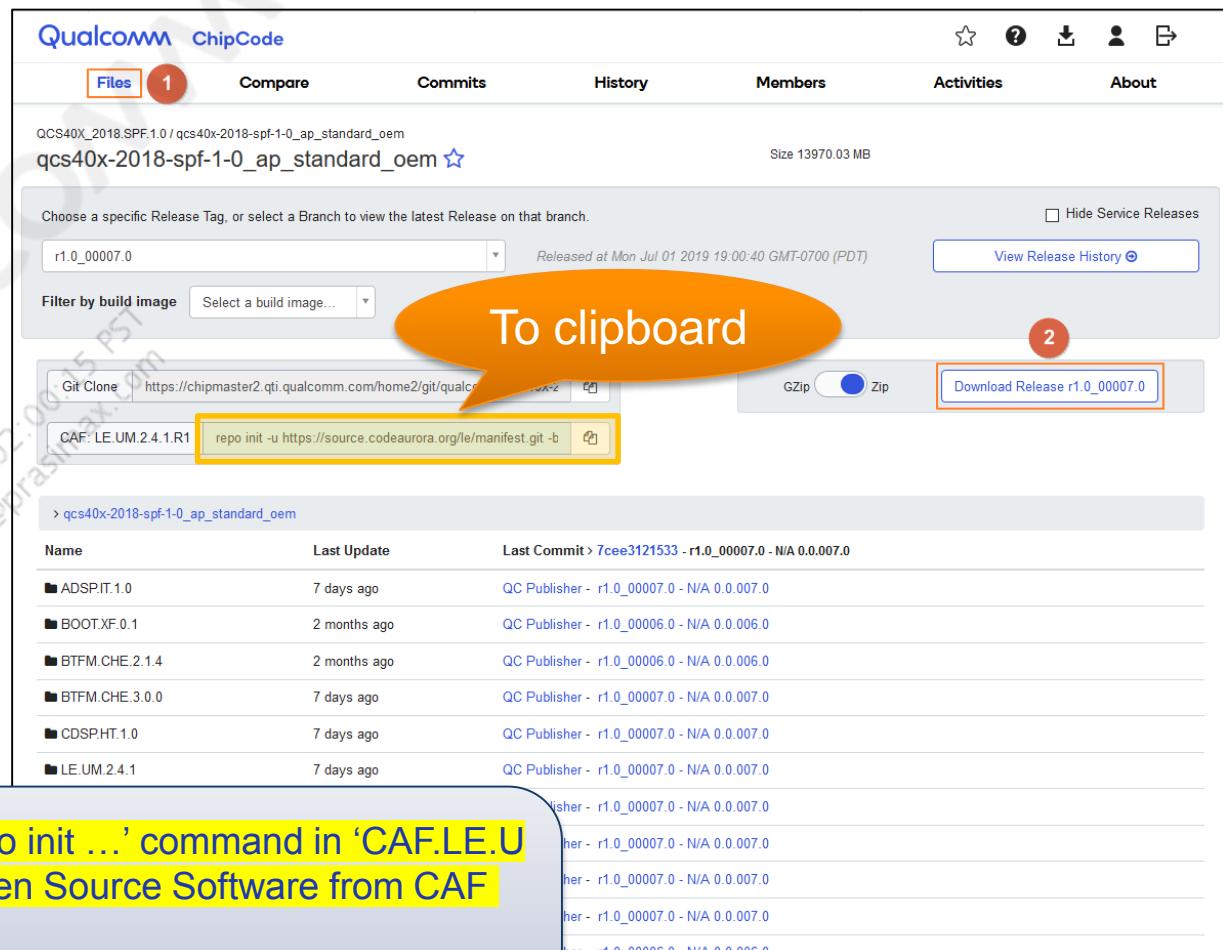
- To download QTI proprietary software,
 - Go: <https://createpoint.qti.qualcomm.com/dashboard/>
- Click on ‘Software Code’ tab
- Click on ‘Software Product’ combo box
- Input ‘QCS40X’ into the search box
- Click on ‘QCS40X_2018.SPF.1.0’ item
- All available QCS40x software packages will be shown
- Click on ‘qcs40x.2018.spf.1.0.ap.standard.oem’ item

The screenshot shows the Qualcomm CreatePoint Software Code Download interface. The top navigation bar includes Home, Documents, Hardware Components, Software Code (which is selected), Tools, Support, and Hardware Purchase. Below the navigation is a search bar with 'All Products' and 'Edit' buttons. The main title is 'Software Code: Download Code from ChipCode'. On the left, there's a sidebar with 'Specify a Software Product' dropdown set to 'Qualcomm' and 'QCS40X_2018.SPF.1.0' selected. It also has 'Filter Results' and 'Reset all Filters' buttons, and sections for 'Your Filters' (Software Product: QCS40X_2018.SPF.1.0, Customer: Qualcomm), 'Customer' (Qualcomm), 'License Type', and 'Security Level' (standard (10)). The main content area displays '10 Software Product Distributions in View' with two download buttons: 'Download This Page' and 'Download All Results'. The list of distributions is as follows:

Customer	Distro
qualcomm	qcs40x.2018.spf.1.0.test.device
qualcomm	qcs40x.2018.spf.1.0.hlos.dev
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.virtualx
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.trumpetdsp
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dtsx
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbydsp
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbyddp
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbydap
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbyatmos
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem

QTI Proprietary Software

- To download QTI proprietary software,
 - Go: <https://createpoint.qti.qualcomm.com/dashboard/>
- Click on ‘Software Code’ tab
- Click on ‘Software Product’ combo box
- Input ‘QCS40X’ into the search box
- Click on ‘QCS40X_2018.SPF.1.0’ item
- All available QCS40x software packages will be shown
- Click on ‘qcs40x.2018.spf.1.0.ap.standard.oem’ item
- Click on ‘Files’ tab (1)
- Click on ‘Download Release <version>’ button (2)



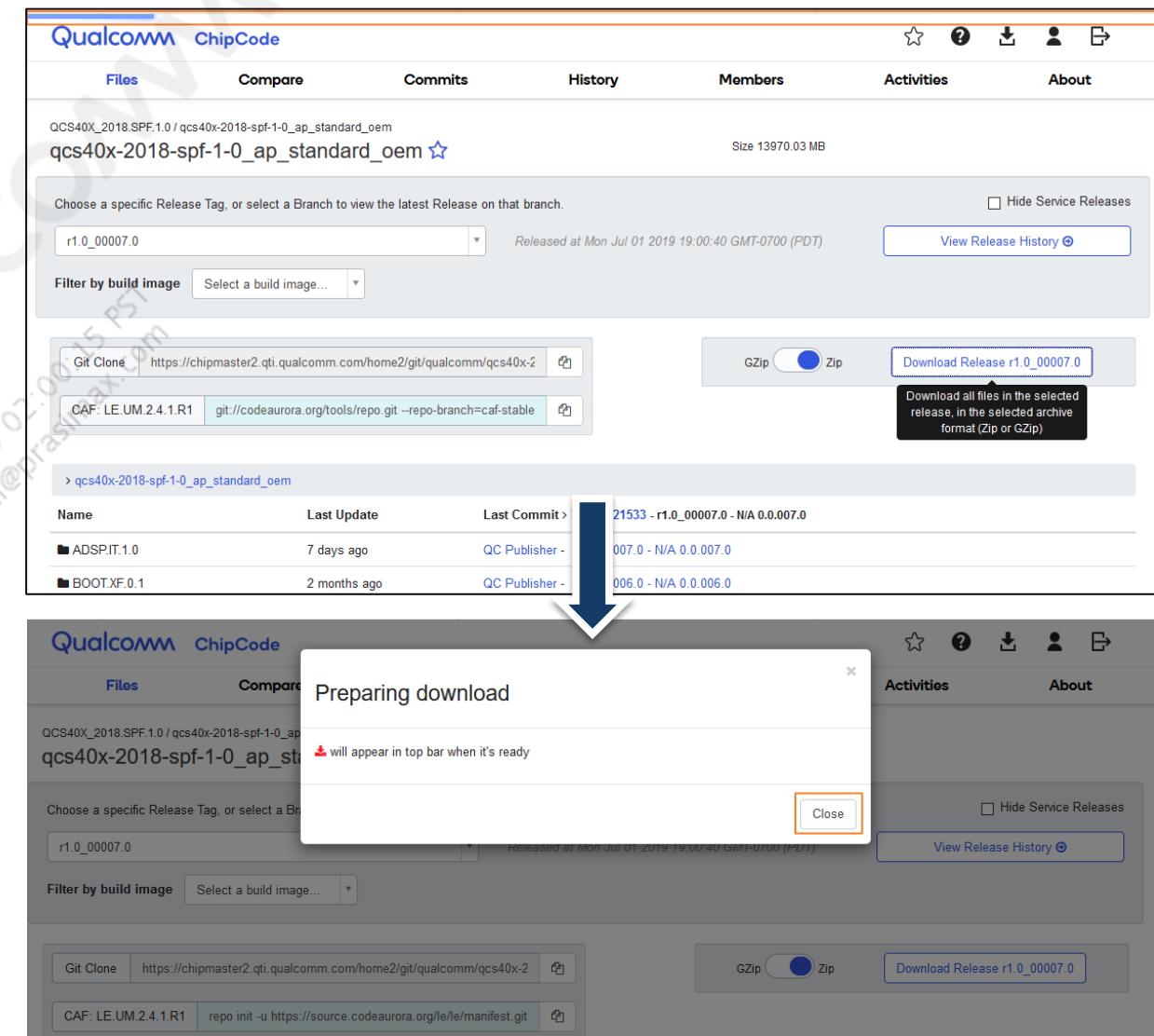
NOTE : Copy and backup ‘repo init ...’ command in ‘CAF.LE.U M.2.4.1.R1’ edit box to get Open Source Software from CAF

e.g. for r1.0_00007.0>

```
repo init -u https://source.codeaurora.org/le/le/manifest.git -b release -m LE.UM.2.4.1.r1-09900-qcs405.0.xml --repo-url=git://codeaurora.org/tools/repo.git --repo-branch=caf-stable
```

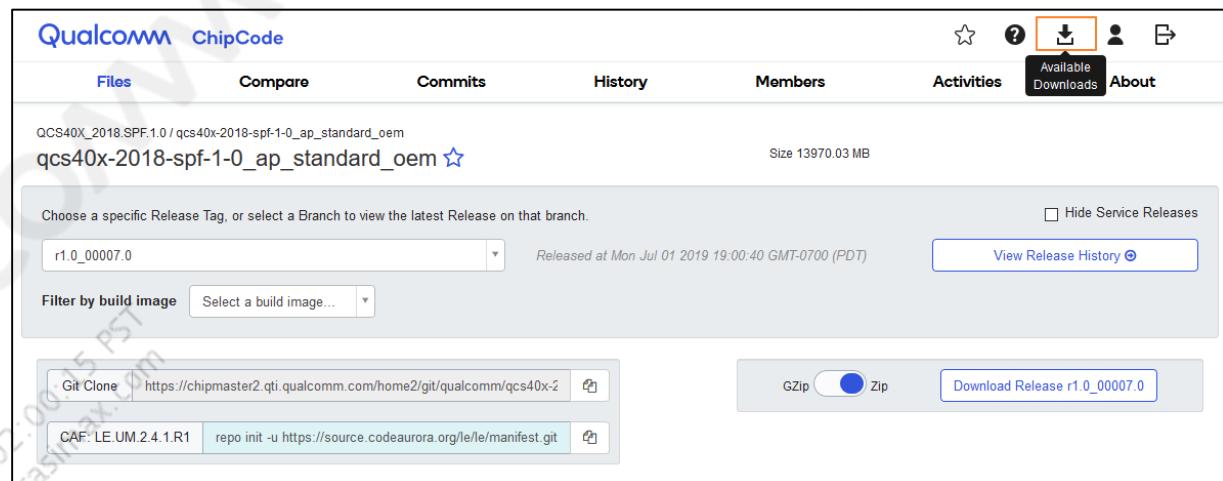
QTI Proprietary Software

- To download QTI proprietary software,
 - Go: <https://createpoint.qti.qualcomm.com/dashboard/>
- Click on ‘Software Code’ tab
- Click on ‘Software Product’ combo box
- Input ‘QCS40X’ into the search box
- Click on ‘QCS40X_2018.SPF.1.0’ item
- All available QCS40x software packages will be shown
- Click on ‘qcs40x.2018.spf.1.0.ap.standard.oem’ item
- Click on ‘Files’ tab
- Click on ‘Download Release <version>’ button
- Wait for the progress at the top until ‘Preparing download’ window appears
- Click on ‘Close’ button on ‘Preparing download’ window



QTI Proprietary Software

- To download QTI proprietary software,
 - Go: <https://createpoint.qti.qualcomm.com/dashboard/>
- Click on ‘Software Code’ tab
- Click on ‘Software Product’ combo box
- Input ‘QCS40X’ into the search box
- Click on ‘QCS40X_2018.SPF.1.0’ item
- All available QCS40x software packages will be shown
- Click on ‘qcs40x.2018.spf.1.0.ap.standard.oem’ item
- Click on ‘Files’ tab
- Click on ‘Download Release <version>’ button
- Wait for the progress at the top until ‘Preparing download’ window appears
- Click on ‘Close’ button on ‘Preparing download’ window
- Click on ‘Available Downloads’ button



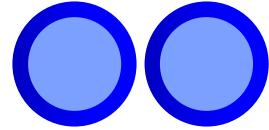
QTI Proprietary Software

- To download QTI proprietary software,
 - Go: <https://createpoint.qti.qualcomm.com/dashboard/>
- Click on ‘Software Code’ tab
- Click on ‘Software Product’ combo box
- Input ‘QCS40X’ into the search box
- Click on ‘QCS40X_2018.SPF.1.0’ item
- All available QCS40x software packages will be shown
- Click on ‘qcs40x.2018.spf.1.0.ap.standard.oem’ item
- Click on ‘Files’ tab
- Click on ‘Download Release <version>’ button
- Wait for the progress at the top until ‘Preparing download’ window appears
- Click on ‘Close’ button on ‘Preparing download’ window
- Click on ‘Available Downloads’ button
- Click on ‘zip’ button to start download – DONE !



QTI Proprietary Software – Links

- Below are the direct links for each package
 - Replace the <Company ID> with own Company ID of CreatePoint
 - For example for ‘qualcomm’: https://chipcode.qti.qualcomm.com/projects/qualcomm/qcs40x-2018-spf-1-0_ap_standard_oem
- SSRD & SBRD without 3rd party codices
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_test_device
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_hlos_dev
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_ap_standard_oem ↵ superset of above two
- SBRD with 3rd party codices (Dolby Atmos/DTS:X)
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_test_device
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_hlos_dev
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_ap_standard_oem ↵ superset of above two
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_ap_standard_oem_virtualx
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_ap_standard_oem_trumpetdsp
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_ap_standard_oem_dtsx
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_ap_standard_oem_dolbydsp
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_ap_standard_oem_dolbyddp
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_ap_standard_oem_dolbydap
 - https://chipcode.qti.qualcomm.com/projects/<Company ID>/qcs40x-2018-spf-1-0_ap_standard_oem_dolbyatmos



Section 4.4

Open Source Software

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Open Source Software

- Starting with r1.0_00007.0 software, Open Source Software moves to public CAF area
- So, approval process is not required for downloading the Open Source Software
- If the organization is not registered yet, contact the QTI regional sales team

Qualcomm Confidential
2019-12-30 02:00:15 PS
didi.setiadi@prasimax.com

Open Source Software is in private CAF area.

To get the private CAF area access,

- Register at <https://identity.codeaurora.org>
- Email the following details:
 - **CAF User ID**
 - **SSH key** : To create SSH key, follow the instruction in the link below.
https://wiki.codeaurora.org/xwiki/bin/Support/gitolite_ssh_setup
 - **CAF area** : For QCS40x, the CAF area is **prv_CAF_2.4.1**
 - **QCS Support Engineer** will forward this request to access provider
- **QCS Support Engineer** handles
- **CAF area** : For QCS40x, the CAF area is **prv_CAF_2.4.1**

NOT REQUIRED

Open Source Software

- To get Open Source Software, run commands on the right side in your Linux Build Machine
 - ‘repo init ...’ command is copied from [QTI Proprietary Source downloading step](#)
 - <work_root> is base folder to build QCS403 or QCS405 software
- DONE !

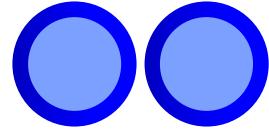
Example commands for <r1.0_00007.0> version

```
<work_root> $ mkdir CAF
```

```
<work_root> $ cd CAF
```

```
<work_root>/CAF $ repo init -u https://source.codeaurora.org/le/le/manifest.git -b release -m LE.UM.2.4.1.r1-09900-qcs405.0.xml --repo-url=git://codeaurora.org/tools/repo.git --repo-branch=caf-stable
```

```
<work_root>/CAF $ repo sync
```



Section 4.5

Merge / Reconstruct Software

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

4.5.1 Merge Software

[106](#)

4.5.2 Reconstruct Software

[110](#)

Merge / Reconstruct Software

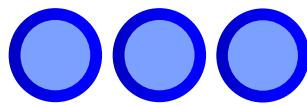
- The following steps are required before attempting to build software images
 - Merge the software downloaded from CAF and ChipCode
 - Reconstruct folder structure

Merge software downloaded from CAF and ChipCode

COPY from CAF source to ChipCode/APPS_PROC

Goal of Reconstruct folder structure

<QCS40x_WS>/contents.xml
<QCS40x_WS>/common
<QCS40x_WS>/apps_proc
<QCS40x_WS>/adsp_proc
<QCS40x_WS>/boot_images
<QCS40x_WS>/btfm_proc
<QCS40x_WS>/cdsp_proc
<QCS40x_WS>/rpm_proc
<QCS40x_WS>/trustzone_images
<QCS40x_WS>/wlan_proc



Section 4.5.1

Merge Software

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Merge Software

- 1. Copy QTI proprietary software .zip file to <work_root> folder and unzip the .zip file
 - Make sure 'qcs40x-2018-spf-1-0_ap_standard_oem.git' folder

Merge CAF and ChipCode commands

```
<work_root>$ unzip qcs40x-2018-spf-1-0_ap_standard_oem-....zip  
<work_root>$ ls  
qcs40x-2018-spf-1-0_ap_standard_oem.git
```

Confidential
2019-12-30 02:00:15 PT
didi.setiadi@prasimax.com

Merge Software

- 1. Copy QTI proprietary software .zip file to <work_root> folder and unzip the .zip file
 - Make sure 'qcs40x-2018-spf-1-0_ap_standard_oem.git' folder
- 2. Rename 'qcs40x-2018-spf-1-0_ap_standard_oem.git' to 'chipcode_root'

Merge CAF and ChipCode commands

```
<work_root>$ unzip qcs40x-2018-spf-1-0_ap_standard_oem-....zip  
  
<work_root>$ ls  
qcs40x-2018-spf-1-0_ap_standard_oem.git  
  
<work_root>$ mv qcs40x-2018-spf-1-0_ap_standard_oem.git chipcode_root
```

Confidential
2019-12-30 02:00:15
didi.setiadi@prasimax.com

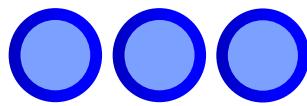
Merge Software

- 1. Copy QTI proprietary software .zip file to <work_root> folder and unzip the .zip file
 - Make sure 'qcs40x-2018-spf-1-0_ap_standard_oem.git' folder
- 2. Rename 'qcs40x-2018-spf-1-0_ap_standard_oem.git' to 'chipcode_root'
- 3. Make sure 'CAF' folder is in <work_root> folder and copy CAF code to ChipCode/APPS_PROC
- DONE !

Merge CAF and ChipCode commands

```
<work_root>$ unzip qcs40x-2018-spf-1-0_ap_standard_oem-....zip  
  
<work_root>$ ls  
qcs40x-2018-spf-1-0_ap_standard_oem.git  
  
<work_root>$ mv qcs40x-2018-spf-1-0_ap_standard_oem.git chipcode_root  
  
<work_root>$ ls  
CAF  
  
<work_root>$ cp -Rf CAF/* chipcode_root/LE.UM.2.4.1/apps_proc/
```

Confidential
2019-12-30
didi.setiadi@praktik.com



Section 4.5.2

Reconstruct Software

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Reconstruct Software

- QTI Proprietary Software contains both QCS403 and QCS405 software
- The table on the right side shows the corresponding chipset for each folder in QTI Proprietary Software
- NOTE: Starting with **r1.0_00007.0** software, **TZ.XF.5.1.2** folder is used for TrustZone instead of TZ.XF.5.1 folder

Folder	SW Module	Chipset
LE.UM.2.4.1	APPS_PROC	Both
QCS405.LE.1.0.1	COMMON	QCS403
BTFM.CHE.2.1.4	BTFM_PROC	QCS403
ADSP.IT.1.0	ADSP_PROC	Both
CDSP.HT.1.0	CDSP_PROC	Both
RPM.BF.1.9	RPM_PROC	Both
WLAN.HL.3.1	WLAN_PROC	Both
BOOT.XF.0.1	BOOT_IMAGES	Both
TZ.XF.5.1.2	TRUSTZONE_IMAGES	Both
QCS405.LE.1.0	COMMON	QCS405
BTFM.CHE.3.0.0	BTFM_PROC	QCS405

Reconstruct Software

- QTI Proprietary Software contains both QCS403 and QCS405 software
- The table on the right side shows the corresponding chipset for each folder in QTI Proprietary Software
- If you need to build both QCS403 and QCS405 on the same build machine, the following modules must be physically separated before build starts
 - APPS_PROC
 - ADSP_PROC
- If you don't want to separate the source code, you should do 'build clean' before build starts for the two modules above

Folder	SW Module	Chipset
LE.UM.2.4.1	APPS_PROC	Both
QCS405.LE.1.0.1	COMMON	QCS403
BTFM.CHE.2.1.4	BTFM_PROC	QCS403
ADSP.IT.1.0	ADSP_PROC	Both
CDSP.HT.1.0	CDSP_PROC	Both
RPM.BF.1.9	RPM_PROC	Both
WLAN.HL.3.1	WLAN_PROC	Both
BOOT.XF.0.1	BOOT_IMAGES	Both
TZ.XF.5.1.2	TRUSTZONE_IMAGES	Both
QCS405.LE.1.0	COMMON	QCS405
BTFM.CHE.3.0.0	BTFM_PROC	QCS405

Reconstruct Software – QCS403 reconstruct for only QCS403 user

- If you build only QCS403, follow the steps in below table

Step	Commands
1. Create work space folder	<work_root>\$ mkdir QCS403_WS <work_root>\$ cd QCS403_WS
2. Copy COMMON - <QCS403_WS> is '<work_root>/QCS403' folder	<QCS403_WS>\$ cp .../chipcode_root/QCS405.LE.1.0.1/contents.xml ./contents.xml <QCS403_WS>\$ cp -Rf .../chipcode_root/QCS405.LE.1.0.1/common/ common/
3. Create symbolic links for others	<QCS403_WS>\$ ln -s .../chipcode_root/LE.UM.2.4.1/apps_proc <QCS403_WS>\$ ln -s .../chipcode_root/ADSP.IT.1.0/adsp_proc <QCS403_WS>\$ ln -s .../chipcode_root/BTFM.CHE.2.1.4/btfm_proc <QCS403_WS>\$ ln -s .../chipcode_root/BOOT.XF.0.1/boot_images <QCS403_WS>\$ ln -s .../chipcode_root/CDSP.HT.1.0/cdsp_proc <QCS403_WS>\$ ln -s .../chipcode_root/RPM.BF.1.9/rpm_proc <QCS403_WS>\$ ln -s .../chipcode_root/TZ.XF.5.1.2/trustzone_images <QCS403_WS>\$ ln -s .../chipcode_root/WLAN.HL.3.1/wlan_proc

Reconstruct Software – QCS403 reconstruct for both QCS403 and QCS405 user

- If you build both QCS403 and QCS405, follow the steps in below table

Step	Commands
1. Create work space folder	<pre><work_root>\$ mkdir QCS403_WS <work_root>\$ cd QCS403_WS</pre>
2. Copy COMMON, APPS_PROC and ADSP_PROC - <QCS403_WS> is '<work_root>/QCS403' folder	<pre><QCS403_WS>\$ cp ..//chipcode_root/QCS405.LE.1.0.1/contents.xml ./contents.xml <QCS403_WS>\$ cp -Rf ..//chipcode_root/QCS405.LE.1.0.1/common/ common/ <QCS403_WS>\$ cp -Rf ..//chipcode_root/LE.UM.2.4.1/apps_proc/ apps_proc/ <QCS403_WS>\$ cp -Rf ..//chipcode_root/ADSP.IT.1.0/adsp_proc/ adsp_proc/</pre>
3. Create symbolic links for others	<pre><QCS403_WS>\$ ln -s ..//chipcode_root/BTFM.CHE.2.1.4/btfm_proc <QCS403_WS>\$ ln -s ..//chipcode_root/BOOT.XF.0.1/boot_images <QCS403_WS>\$ ln -s ..//chipcode_root/CDSP.HT.1.0/cdsp_proc <QCS403_WS>\$ ln -s ..//chipcode_root/RPM.BF.1.9/rpm_proc <QCS403_WS>\$ ln -s ..//chipcode_root/TZ.XF.5.1.2/trustzone_images <QCS403_WS>\$ ln -s ..//chipcode_root/WLAN.HL.3.1/wlan_proc</pre>

Reconstruct Software

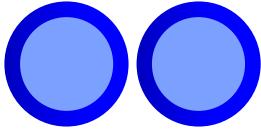
- Make sure the reconstruction result in <QCS40x_WS> folder
- The figure on the right shows the results for QCS403_WS
- DONE !

Confidential
2019-12-30
didi.setadi@.../QCS403_WS



QCS403_WS

```
.  
..  
adsp_proc  
apps_proc  
boot_images  
btfm_proc -> ../../chipcode_root/[BTFM.CHE.2.1.4]/btfm_proc  
cdsp_proc -> ../../chipcode_root/CDSP.HT.1.0/cdsp_proc  
common  
contents.xml  
rpm_proc -> ../../chipcode_root/RPM.BF.1.9/rpm_proc  
trustzone_images -> ../../chipcode_root/[T2.XF.5.1.2]/trustzone_images  
wlan_proc -> ../../chipcode_root/WLAN.HL.3.1/wlan_proc
```



Section 4.6

3rd Party Software Integration

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

3rd Party Software Integration

- 3rd Party Software consists:
 - Dolby
 - DTS
 - VirtualX
- The default QTI Proprietary Software does not contain 3rd Party Software
- In order to include the 3rd Party Software in the built image, separate library integration is required
- 3rd Party Software is provided as a Library through ChipCode portal

Qualcomm Confidential and Proprietary
2019-12-30 02:05 PT
didi.setiadi@prajnax.com

3rd Party Software Integration

- To download 3rd Party Software, refer to [QTI Proprietary Software downloading step](#) first
- In this chapter, DolbyAtmos will be explained as an example
- Download 'qcs40x.2018.spf.1.0.ap.standard.oem.dolbyatmos' package

Confidential
2019-12-30
didi.setiadi@praktikum.itb.ac.id

Software Code: Download Code from ChipCode

Customer	Distro
qualcomm	qcs40x.2018.spf.1.0.test.device
qualcomm	qcs40x.2018.spf.1.0.hlos.dev
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.virtualx
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.trumpetdsp
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dtsx
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbydsp
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbyddp
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbydap
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem.dolbyatmos
qualcomm	qcs40x.2018.spf.1.0.ap.standard.oem

3rd Party Software Integration

- To download 3rd Party Software, refer to [QTI Proprietary Software downloading step](#) first
- In this chapter, DolbyAtmos will be explained as an example
- Download 'qcs40x.2018.spf.1.0.ap.standard.oem.dolbyatmos' package
- Move downloaded .zip file to <work_root> folder and unzip the .zip file
 - Make sure 'qcs40x-2018-spf-1-0_ap_standard_oem_dolbyatmos.git' folder on the <work_root> folder

```
<work_root>$ unzip qcs40x-2018-spf-1-0_ap_standard_oem_dolbyatmos-xxx.zip
```

```
<work_root>$ ls -al
```



```
.  
..  
CAF  
CAF.zip  
chipcode_root  
QCS405_WS  
qcs40x-2018-spf-1-0_ap_standard_oem-536b4b090a981fe3ac8df3437984f630d3313c5d.zip  
qcs40x-2018-spf-1-0_ap_standard_oem_dolbyatmos-3caec3fa501b9fbce17a229ca9872f8a527536ac.zip  
qcs40x-2018-spf-1-0_ap_standard_oem_dolbyatmos.git
```

3rd Party Software Integration

- To download 3rd Party Software, refer to [QTI Proprietary Software downloading step](#) first
- In this chapter, DolbyAtmos will be explained as an example
- Download 'qcs40x.2018.spf.1.0.ap.standard.oem.dolbyatmos' package
- Move downloaded .zip file to <work_root> folder and unzip the .zip file
 - Make sure 'qcs40x-2018-spf-1-0_ap_standard_oem_dolbyatmos.git' folder on the <work_root> folder
- Copy 'prebuilt-FEAT-BIN-DOLBY_ATMOS' folder into <QCS40x_WS>/apps_proc/
 - Make sure 'prebuilt-FEAT-BIN-DOLBY_ATMOS' folder on the <QCS40x_WS>/apps_proc folder
- Integration DONE !
- Now DolbyAtmos will be included in the built image

```
<work_root>$ cp -nr qcs40x-2018-spf-1-0_ap_standard_oem_dolbyatmos.git/LE.U  
M.2.4.1/apps_proc/prebuilt_FEAT-BIN-DOLBY-ATMOS QCS405_WS/apps_proc/pr  
ebuilt_FEAT-BIN-DOLBY-ATMOS
```

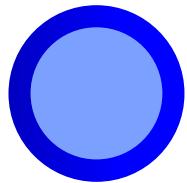
```
<work_root>$ cd QCS405_WS/apps_proc
```

```
<work_root>/QCS405_WS/apps_proc $ ls -al
```



Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

```
qti-pv6  
poky  
prebuilt_FEAT-BIN-DOLBY-ATMOS  
prebuilt_HY11  
qcom-opensource  
qmi  
qmi-framework  
remotesfs
```

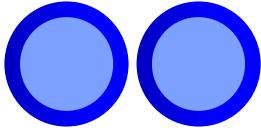


Section 5

Build Software

Qualcomm
Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

5.1 Build HLOS Software	122
5.2 Build non-HLOS software	127
5.2.1 Build ADSP	129
5.2.2 Build CDSP	131
5.2.3 Build RPM	133
5.2.4 Build Bootloader	135
5.2.5 Build TrustZone	137
5.3 Generate Final Image	140



Section 5.1

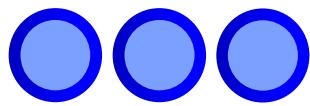
Build HLOS Software

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Build HLOS Software – QCS403

HLOS software image build commands for QCS403	
Build setup	<QCS403_WS>\$ cd apps_proc/poky <QCS403_WS>/apps_proc/poky\$ export SHELL=/bin/bash <QCS403_WS>/apps_proc/poky\$ source build/conf/set_bb_env.sh <QCS403_WS>/apps_proc/poky/build\$ list-build-commands
Build all (debug)	<QCS403_WS>/apps_proc/poky/build\$ build-qcs403-som2-qsap-image
Build all (perf)	<QCS403_WS>/apps_proc/poky/build\$ build-qcs403-som2-qsap-perf-image
Clean all	<QCS403_WS>/apps_proc/poky/build\$ buildclean all
Build log	NOTE: Tasks Summary: Attempted 5907 tasks of which 5262 didn't need to be rerun and all succeeded. NOTE: Writing buildhistory Summary: There were 21 WARNING messages shown. /home/donald/r1.0_00006.0/chipcode_root/LE.UM.2.4.1/apps_proc/poky/build
Build log file	✓ There are multiple build logs in the folder below <QCS403_WS>/apps_proc/poky/build/tmp-glibc/work/qcs403_som2-oe-linux-gnueabi/machine-image/1.0-r0/temp/
Build result	✓ All result files are <QCS403_WS>/apps_proc/poky/build/tmp-glibc/deploy/images/<Build_Variant>/ ✓ <Build_Variant> is 'qcs403-som2-qsap' or 'qcs403-som2-qsap-perf' <QCS403_WS>/apps_proc/poky/build/tmp-glibc/deploy/images/<Build_Variant>/abl-squashfs.elf <QCS403_WS>/apps_proc/poky/build/tmp-glibc/deploy/images/<Build_Variant>/qcs40x-boot.img <QCS403_WS>/apps_proc/poky/build/tmp-glibc/deploy/images/<Build_Variant>/qcs40x-squashfs-sysfs.ubi



Build HLOS Software - Trouble Shooting

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Build HLOS Software – Trouble shooting

- [ERROR]: Often unexpected build error or Linux reboot may occur due to the physical performance of your build machine (Memory, CPU)
- [SOLUTION]: You can adjust the number of build threads by modifying the file below and then try to build again
 - <QCS40x_WS>/apps_proc/poky/build/conf/local.conf
 - BB_NUMBER_THREADS ?= "20"
 - PARALLEL_MAKE ?= "-j 20"
 - 20 is default value of the build thread count
 - Change the value from 20 to 8 or 4
 - **NOTE #1:** KEEP the question mark and space before the count value
 - BB_NUMBER_THREADS ?= "20"
 - **NOTE #2:** This unexpected error can cause either object or archive files broken which is extremely hard to catch, therefore '**buildclean all**' first and then try to build again

Error message example

ERROR: Worker process (21511) exited unexpectedly (-9), shutting down...

<QCS40x_WS>/apps_proc/poky/build/conf/local.conf

```
#  
# Parallelism Options  
#  
# These two options control how much parallelism BitBake should use. The first  
# option determines how many tasks bitbake should run in parallel:  
#  
BB_NUMBER_THREADS ?= "20"  
#  
# The second option controls how many processes make should run in parallel when  
# running compile tasks:  
#  
PARALLEL_MAKE ?= "-j 20"  
#  
# For a quadcore, BB_NUMBER_THREADS = "4", PARALLEL_MAKE = "-j 4" would  
# be appropriate for example.
```

Build HLOS Software – Trouble shooting

- [ERROR]: 'opensslv.h' file not found error

- [SOLUTION]: [Install 'libssl-dev' package](#)

- \$ sudo apt-get install libssl-dev

- If you failed to install 'libssl-dev' package, resolve as follow

- \$ apt-cache policy libssl-dev
 - \$ sudo apt-get install libssl1.0.0=1.0.2g-1ubuntu4.1
 - \$ sudo apt-get install libssl-dev

Error message example

```
/poky/build/tmp-glibc/work-shared/qcs405-som1/kernel-source/scripts/sign-file.c:25:30:  
fatal error: openssl/opensslv.h: No such file or directory
```

Install libssl-dev

```
$ sudo apt-get install libssl-dev  
...  
E: Unable to correct problems, you have held broken packages.
```

```
$ apt-cache policy libssl-dev  
libssl-dev:  
  Installed: (none)  
  Candidate: 1.0.2g-1ubuntu4.1  
  Version table:  
    1.0.2g-1ubuntu4.1 500  
      500 http://mirror.uoregon.edu/ubuntu xenial-updates/main amd64 Packages  
      500 http://mirror.uoregon.edu/ubuntu xenial-security/main amd64 Packages  
    1.0.2g-1ubuntu4 500  
      500 http://mirror.uoregon.edu/ubuntu xenial/main amd64 Packages
```

```
$ sudo apt-get install libssl1.0.0=1.0.2g-1ubuntu4.1  
$ sudo apt-get install libssl-dev
```

Build HLOS Software – Trouble shooting

- [ERROR]: 'bouncycastle' recipe build error

Error message example

ERROR: Task (/home/donald/work_root/chipcode_root/LE.UM.2.4.1/apps_proc/poky/meta-qti-bsp/recipes-devtools/bouncycastle/bouncycastle_git.bb:do_compile) failed with exit code '1'

- [SOLUTION]: [Install Java v1.7.0 JDK](#)

- \$ sudo add-apt-repository ppa:openjdk-r/ppa
- \$ sudo apt-get update
- \$ sudo apt-get install openjdk-7-jdk
- \$ update-java-alternatives --list
- \$ sudo update-java-alternatives -s java-1.7.0-openjdk-amd64
- \$ java -version

Install Java v1.7

```
$ sudo add-apt-repository ppa:openjdk-r/ppa
gpg: key 86F44E2A: public key "Launchpad OpenJDK builds (all archs)" imported
gpg: Total number processed: 1
gpg:          imported: 1 (RSA: 1)
OK
```

```
$ sudo apt-get update
$ sudo apt-get install openjdk-7-jdk
$ update-java-alternatives --list
java-1.7.0-openjdk-amd64      1071  /usr/lib/jvm/java-1.7.0-openjdk-amd64
java-1.8.0-openjdk-amd64      1081  /usr/lib/jvm/java-1.8.0-openjdk-amd64
```

```
$ sudo update-java-alternatives -s java-1.7.0-openjdk-amd64
```

```
$ java -version
java version "1.7.0_95"
OpenJDK Runtime Environment (IcedTea 2.6.4) (7u95-2.6.4-3)
OpenJDK 64-Bit Server VM (build 24.95-b01, mixed mode)
```

Build HLOS Software – Trouble shooting

- [ERROR]: MD5 and SHA256 Checksum mismatch! error

- [REASON]:

- Some open source packages will be downloaded in build time
 - e.g. gstreamer, iotivity, AVS SDK, and so on...
 - The downloaded file's expected checksum is saved into the related recipe file
 - If the downloaded file from open source package is changed, then the checksum in recipe file must be changed too
 - If the open source package changes after the software release, this problem will occur

- [SOLUTION]: Modify recipe file

- The error log guide us the solution. So modify recipe file according to error log's guide first
 - NOTE: Maybe this modification will be reflected in the next SW release

Error message example

a63d757390b80d44f8c2.patch;name=he-aac-rate-fix'. [Checksum mismatch!](#)

...

If this change is expected (e.g. you have upgraded to a new version without updating the checksums) then you can use these lines within the recipe:

SRC_URI[he-aac-rate-fix.[md5sum](#)] = "[204111a4f2a5cf5c36bdb65ceeb14c0](#)"
SRC_URI[he-aac-rate-fix.[sha256sum](#)] = "[f6a22569f460a9efcc67822a5078f32cdbf89cd76e58026b85be185398ca798f](#)"

...

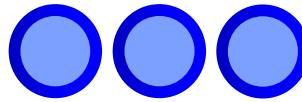
ERROR: Logfile of failure stored in: /local2/mnt/workspace/QCS40x/r1.0_00006.4/[QCS403 WS/apps_proc/poky/build/tmp-glibc/work/armv7ahf-neon-oe-linux-gnueabi/gstreamer1.0-plugins-base/1.12.2-r0/temp/log.do_fetch.24678](#)

...

Modify recipe(gstreamer1.0-plugins-base_1.12.2.bbappend) file

1. Check build log (log.do_fetch.24678) file and find the exact recipe file for error
- In this case '[<QCS403 WS>/apps_proc/poky/meta-qt4/recipes-multimedia/gstreamer/gstreamer1.0-plugins-base_1.12.2.bbappend](#)' is the **recipe** file for gstreamer

2. Replace MD5 and SHA256 checksum values like below



Build HLOS Software - Tips

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Build HLOS Software – Tips: How to enable apps debug log in perf image

- In perf image, apps debug log is disabled by default
- If you want to enable apps debug log, follow the steps below:
 - ✓ Modify '[/apps_proc/kernel/msm-4.14/arch/arm/configs/vendor/qcs405-perf_defconfig](#)' file
 - Following three configurations are required to enable apps debug log

```
CONFIG_SERIAL_MSM=y  
CONFIG_SERIAL_MSM_CONSOLE=y  
CONFIG_SERIAL_MSM_HS=y
```

- If the configuration exists, change value to **y**
- If the configuration does not exist, add it

- ✓ Rebuild Kernel

```
<QCS403_WS>/apps_proc/poky/build$ MACHINE=qcs403-som2 DISTRO=qsap VARIANT=perf rebake virtual/kernel
```

- ✓ Build all apps image

```
<QCS403_WS>/apps_proc/poky/build$ build-qcs403-som2-qsap-perf-image
```

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasinax.com

Build HLOS Software – Tips: How to enable Login Function in perf image

- In perf image, Login Function is disabled by default
- If you want to enable Login Function, follow the steps below:
 - ✓ Modify '[/apps_proc/poky/meta-qtibsp/conf/qcs40x.inc](#)' file

BEFORE	SERIAL_CONSOLE ?= \${@["'115200 ttyMSM0'"] [dgetVar('VARIANT', True) == (" or 'debug')]}
AFTER	SERIAL_CONSOLE ?= "ttyMSM0"

- ✓ Rebuild Kernel

```
<QCS403_WS>/apps_proc/poky/build$ MACHINE=qcs403-som2 DISTRO=qsap VARIANT=perf rebake virtual/kernel
```

- ✓ Build all apps image

```
<QCS403_WS>/apps_proc/poky/build$ build-qcs403-som2-qsap-perf-image
```

Build HLOS Software – Tips: How to make ubifs image (1/2)

- Starting with r1.0_00007.0 software, ubifs images(abl.elf, qcs40x-sysfs.ubi) do not generated by default
- If you want to build ubifs images, follow the steps below:
 - ✓ Modify '/apps_proc/poky/meta-qtibsp/recipes-kernel/edk2/edk2_git.bb' file

* Remove following two lines

```
NAND_SQUASHFS_SUPPORT = ${@bb.utils.contains('DISTRO_FEATURES', 'nand-squashfs', '1', '0', d)}  
EXTRA_OEMAKE_append = "NAND_SQUASHFS_SUPPORT=${NAND_SQUASHFS_SUPPORT}"
```

- ✓ Modify '/apps_proc/poky/meta-qtibsp-prop/recipes-products/images/qcs40x/qcs40x-qi-image.inc' file

* Replace 'qcs40x-sysfs.squash' with 'qcs40x-sysfs.ubifs'

* Replace 'create_squash_ubi_img()' function with:

```
create_squash_ubi_img() {  
    create_squashfs_ubinize_config  
    mkfs.ubifs -r ${USR_IMAGE_ROOTFS} -o ${OUTPUT_FILE_USR_UBIFS} ${MKUBIFS_ARGS}  
    mkfs.ubifs -r ${IMAGE_ROOTFS} -o ${DEPLOY_DIR_IMAGE}/qcs40x-sysfs.ubifs ${MKUBIFS_ARGS}  
    ubinize -o ${OUTPUT_FILE_FINAL_SQUASHFS_UBI} ${UBINIZE_ARGS} ${UBINIZE_SQUASHFS_CFG}  
    chmod 644 ${OUTPUT_FILE_FINAL_SQUASHFS_UBI}  
}
```

* Modify 'SYSTEM_SQUASHFS_VOLUME_SIZE' value to fit the generated image size

- If you build after reflecting the above fix, then the following error will occur:

ubinize: error! error in section "sysfs volume": size of the image file ".../apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap/qcs40x-sysfs.ubifs" is 122150912, which is larger than volume size 90725120

- SYSTEM_SQUASHFS_VOLUME_SIZE is a value that limits the size of the sysfs. Therefore, the value should be modified to more than image size

- SYSTEM_SQUASHFS_VOLUME_SIZE value should be multiple of LEB, LEB is 253952 bytes

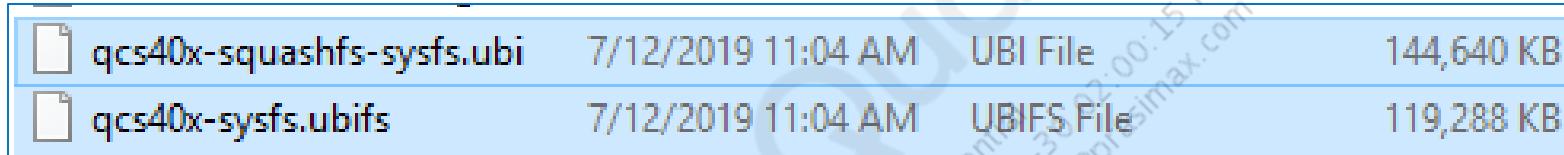
- For example, SYSTEM_SQUASHFS_VOLUME_SIZE = "146784256"

Build HLOS Software – Tips: How to make ubifs image (2/2)

- ✓ Build all apps image

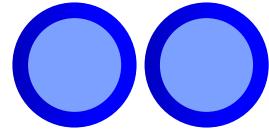
Debug	<QCS403_WS>/apps_proc/poky/build\$ build-qcs403-som2-qsap-image
Perf	<QCS403_WS>/apps_proc/poky/build\$ build-qcs403-som2-qsap-perf-image

- ✓ Check the build result in '[`/apps_proc/poky/build/tmp-glibc/deploy/images/<Build_Variant>/`](#)' folder



 <code>qcs40x-squashfs-sysfs.ubi</code>	7/12/2019 11:04 AM	UBI File	144,640 KB
 <code>qcs40x-sysfs.ubifs</code>	7/12/2019 11:04 AM	UBIFS File	119,288 KB

- ✓ Note that, the final image file name (`qcs40x_squashfs_sysfs.ubi`) include squashfs, but it is made up of ubifs



Section 5.2

Build non-HLOS software

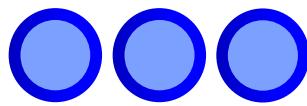
Qualcomm
Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

5.2.1 Build ADSP	129
5.2.2 Build CDSP	131
5.2.3 Build RPM	133
5.2.4 Build Bootloader	135
5.2.5 Build TrustZone	137

Build non-HLOS software

- Based on 'ap_standard_oem' package:
- Following modules must be built before to generate final image
 - ADSP_PROC
 - CDSP_PROC
 - RPM_PROC
 - BOOT_IMAGES
 - TRUSTZONE_IMAGES
- Following two modules are provided as binary, so build is not required
 - WLAN_PROC
 - BTFM_PROC

Qualcomm Confidential
2019-12-30 02:00:15 PST
ddi.setadi@prasimax.com



Section 5.2.1

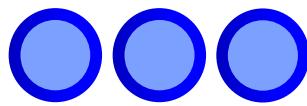
Build ADSP

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Build ADSP

ADSP build commands and prerequisite for QCS403	
Prerequisite	<ul style="list-style-type: none">✓ Hexagon Tool v8.2.02 installation
Build setup	<ul style="list-style-type: none">✓ HEXAGON_ROOT depends on where the tools are installed <pre><QCS403_WS>\$ cd adsp_proc <QCS403_WS>/adsp_proc\$ export HEXAGON_ROOT=~/Qualcomm/HEXAGON_Tools <QCS403_WS>/adsp_proc\$ export HEXAGON_RTOS_RELEASE=8.2.02 <QCS403_WS>/adsp_proc\$ export HEXAGON_Q6VERSION=v66 <QCS403_WS>/adsp_proc\$ export HEXAGON_IMAGE_ENTRY=0x8AC00000 <QCS403_WS>/adsp_proc\$ env grep HEXA</pre>
Build	<pre><QCS403_WS>/adsp_proc\$ python ./build/build.py -c qcs405 -o all -f ADSP,USES_VT_AUDIO</pre>
Clean	<pre><QCS403_WS>/adsp_proc\$ python ./build/build.py -c qcs405 -o clean -f ADSP,USES_VT_AUDIO</pre>
Build log	<pre>scons: done reading SConscript files. scons: Building targets ... scons: `.' is up to date. scons: done building targets. Compilation SUCCESS</pre>
Build log file	<pre><QCS403_WS>/adsp_proc/build/ms/build-log.txt</pre>
Build result file	<pre><QCS403_WS>/adsp_proc/build/ms/adsp.bin</pre>



Section 5.2.2

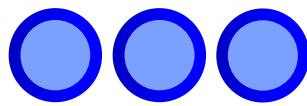
Build CDSP

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Build CDSP

CDSP build commands and prerequisite for QCS40x	
Prerequisite	<ul style="list-style-type: none">✓ Hexagon Tool v8.2.02 installation
Build setup	<ul style="list-style-type: none">✓ HEXAGON_ROOT depends on where the tools are installed <pre><QCS40x_WS>\$ cd cdsp_proc <QCS40x_WS>/cdsp_proc\$ export HEXAGON_ROOT=~/Qualcomm/HEXAGON_Tools <QCS40x_WS>/cdsp_proc\$ export HEXAGON_RTOS_RELEASE=8.2.02 <QCS40x_WS>/cdsp_proc\$ export HEXAGON_Q6VERSION=v66 <QCS40x_WS>/cdsp_proc\$ export HEXAGON_IMAGE_ENTRY=0x8AC00000 <QCS40x_WS>/cdsp_proc\$ env grep HEXA</pre>
Build	<pre><QCS40x_WS>/cdsp_proc\$ python ./build/build.py -c qcs405 -o all -f CDSP</pre>
Clean	<pre><QCS40x_WS>/cdsp_proc\$ python ./build/build.py -c qcs405 -o clean -f CDSP</pre>
Build log	<pre>scons: done reading SConscript files. scons: Building targets ... scons: `.' is up to date. scons: done building targets. Compilation SUCCESS</pre>
Build log file	<pre><QCS40x_WS>/cdsp_proc/build/ms/build-log.txt</pre>
Build result file	<pre><QCS40x_WS>/cdsp_proc/build/ms/cdsp.bin</pre>



Section 5.2.3

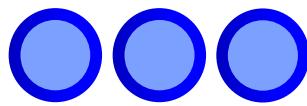
Build RPM

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Build RPM

RPM build commands and prerequisite for QCS40x	
Prerequisite	✓ SD LLVM Toolchain v3.9.2
Build setup	<QCS40x_WS>\$ cd rpm_proc/build
Build	<QCS40x_WS>/rpm_proc/build\$./build_405.sh
Clean	<QCS40x_WS>/rpm_proc/build\$./build_405.sh -c
Build log	scons: done building targets. ===== SCons build summary ===== ** Build time... Build start : Thu Mar 28 14:48:31 2019 Build end : Thu Mar 28 14:52:10 2019 Elapsed time : 0:03:40 Cleaning up any old build artifacts Copying build artifacts to .../core/bsp/rpm/build/405/ Build started: 2019-03-28 14:34:33.741437 Build ended: 2019-03-28 14:52:10.906696 Elapsed time: 0:17:37.165259
Build log file	<QCS40x_WS>/rpm_proc/build/ build-log.txt
Build result file	<QCS40x_WS>/rpm_proc/build/ms/bin/AAAAANAZR/ rpm.mbn



Section 5.2.4

Build Bootloader

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

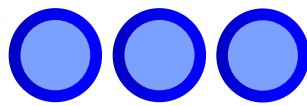
Build Bootloader – Check Configuration Data Table

- In the following case, you should use hardcoded CDT:
 - Target device does not have a storage for CDT
 - The storage for CDT does not have CDT information
- SBRD with QCS405 based SOM stores CDT into eMMC
- SSRD with QCS403 based SOM stores CDT into EPROM
- For more details about using hardcoded CDT, refer to *[80-YB420-3] Bootloader training slide*

Qualcomm Confidential
2019-12-30 02:15 PST
didi.setiadi@prasimajaya.com

Build Bootloader

Bootloader build commands and prerequisite for QCS40x	
Prerequisite	<ul style="list-style-type: none">✓ SD LLVM Toolchain v4.0.2✓ GCC Linaro Toolchain v4.8
Build setup	<QCS40x_WS>\$ cd boot_images/QcomPkg/Qcs405Pkg
Build	<QCS40x_WS>/boot_images/QcomPkg/Qcs405Pkg\$ python ..//buildit.py --variant LA -r RELEASE -t Qcs405Pkg,QcomToolsPkg
Clean	<QCS40x_WS>/boot_images/QcomPkg/Qcs405Pkg\$ python ..//buildit.py --variant LA -r RELEASE -t Qcs405Pkg,QcomToolsPkg -b clean
Build log	<pre>***** [buildit.py] Completed building Target: QcomToolsPkg Subtarget: ---- Variant: -- Single: -- Release: RELEASE [buildit.py] Successfully built following flavors: ##### Target: Qcs405Pkg Subtarget: ---- Varaint: LA Single: -- release: RELEASE Target: QcomToolsPkg Subtarget: ---- Variant: -- Single: -- release: RELEASE [buildit.py] Now exiting...</pre>
Build log file	<code><QCS40x_WS>/boot_images/QcomPkg/Qcs405Pkg/LA/build_Core.log</code> <code><QCS40x_WS>/boot_images/QcomPkg/Qcs405Pkg/LA/build_Loader.log</code>
Build result	<code><QCS40x_WS>/boot_images/QcomPkg/Qcs405Pkg/Bin/405/LA/RELEASE/xbl_nand.elf</code> <code><QCS40x_WS>/boot_images/QcomPkg/Qcs405Pkg/Bin/405/LA/RELEASE/pmic.elf</code>



Section 5.2.5

Build TrustZone

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Build TrustZone – Modify ‘build_config_deploy.xml’

- Before TZ build, the build configuration should be corrected
- File : <QCS40x_WS>/trustzone_images/build/ms/build_config_deploy.xml
 - Line #51
 - Parameter name: “PYTHON_PATH”
 - Default value: /pkg/qct/software/python/2.7/bin
- Modify with your python2.7 binary installed folder path
 - Command to check python2.7 binary installed path
\$ whereis python2.7
- For example, if the installed folder is /usr/bin

BEFORE	AFTER
/pkg/qct/software/python/2.7/bin	/usr/bin

<QCS40x_WS>/trustzone_images/build/ms/build_config_deploy.xml

```
<build>
  <branch aliases="tz51.5.1,tz.xf.5.1" default_target="qcs405 ... >
    <chipset build_asic="405" build_id="OAPAANAA" ... >
      <environment>
        ...
        ...
          <var description="Bin directory ..." ... name="PYTHON_PATH" value="/usr/bin"/>
        ...
        ...
      ...
    ...
  ...
<intermediate="false" name="CRMPERL" value="/usr/bin"/>
  name="LLVMCLANG" value="/pkg/qct/software/llvm/release/arm/4.0.11/tools" ...
  " intermediate="false" name="CCBIN" value="/pkg/qct/software/qct/software/llvm/release/arm/4.0.11/aarch64-none-elf/libc"/>
  E" value="/pkg/asw/compilers/arm/ADS1.2"/>
  LPATH" value="/pkg/qct/software/llvm/release/arm/4.0.11/tools" ...
  " value="/pkg/qct/software/bullseyecoverage/BullseyeCoverage-8" ...
  " intermediate="false" name="PYTHON_PATH" value="/usr/bin"/>
  ediate="false" name="ARMTTOOLS" value="ARMCCT6"/>
```

Build TrustZone – Check build branch name in ‘build_config_deploy.xml’

- TZ build command requires the build branch name argument
- The build branch name argument can be found in ‘build_config_deploy.xml’ file
- File : <QCS40x_WS>/trustzone_images/build/ms/build_config_deploy.xml
 - <Build>
 - <Branch ... name=“build_branch_name” ... >
- e.g. For the r1.0_00007.0 software, the build branch name is **TZ.XF.5.1**

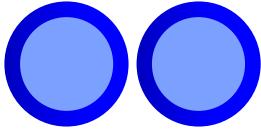
```
<QCS40x_WS>/trustzone_images/build/ms/build_config_deploy.xml

<build>
  <branch aliases="tz51,5.1,tz.xf.5.1" default_target="qcs405,nicobar,mdm9205"
  name="TZ.XF.5.1" virtual="false">
  ...
  ...

<build>
  <branch aliases="tz51,5.1,tz.xf.5.1" default_target="qcs405,nicobar,mdm9205" name="TZ.XF.5.1" virtual="false">
```

Build TrustZone

TZ build commands and prerequisite for QCS40x	
Prerequisite	<ul style="list-style-type: none">✓ SD LLVM Toolchain v4.0.11✓ GCC Linaro Toolchain v4.9
Build setup	<QCS40x_WS>\$ cd trustzone_images/build/ms
Build for TZ.XF.5.1.2	<QCS40x_WS>/trustzone_images/build/ms\$ python build_all.py -b TZ.XF.5.1 CHIPSET=qcs405 config=build_config_deploy.xml --recompile ✓ TZ.XF.5.1 is the build branch name
Clean for TZ.XF.5.1.2	<QCS40x_WS>/trustzone_images/build/ms\$ python build_all.py -b TZ.XF.5.1 CHIPSET=qcs405 config=build_config_deploy.xml --clean ✓ TZ.XF.5.1 is the build branch name
Build log	Start Time = Thu Mar 28 15:28:13 KST 2019 - End Time = Thu Mar 28 15:28:59 KST 2019 Elapsed Time = 46 seconds WARNING:build_all:Extra artifacts: devcfg.mbn, gpsample.mbn, seccamdemo.mbn, seccamdemo64.mbn, secure_ui_sample.mbn, secure_ui_sample64.mbn, secureindicator.mbn, smplap32.mbn, smplap64.mbn, smplcert.mbn, smplserv.mbn, teetest.mbn INFO:build_all:Build completed successfully INFO:build_all:Total elapsed time: 1 minutes, 33 seconds
Build log file	<QCS40x_WS>/trustzone_images/build/ms/ build-log.txt
Build result	<QCS40x_WS>/trustzone_images/build/ms/bin/OAPAANAA/ devcfg.mbn <QCS40x_WS>/trustzone_images/build/ms/bin/OAPAANAA/ tz.mbn



Section 5.3

Generate Final Image

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Generate Final Image

- Before ‘Generate Final Image’, the **apps image location** should be modified according to APPS_PROC built variant
- The apps image location information is in ‘<QCS40x>/contents.xml’ file
- Below is the apps image location for each build variant

Build Variant	APPS Image Location <apps_image_base>: apps_proc/poky/build/tmp-glibc/deploy/images	<QCS40x>/contents.xml
QCS403 (debug)	<apps_image_base>/qcs403-som2-qsap/	<pre>Line 94: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap/</file_path> Line 110: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap/</file_path> Line 114: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap/</file_path> Line 118: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap/</file_path></pre>
QCS403 (perf)	<apps_image_base>/qcs403-som2-qsap-perf/	<pre>Line 94: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap-perf/</file_path> Line 110: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap-perf/</file_path> Line 114: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap-perf/</file_path> Line 118: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap-perf/</file_path></pre>
QCS405 (debug)	<apps_image_base>/qcs405-som1-qsap/	<pre>Line 127: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap/</file_path> Line 131: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap/</file_path> Line 135: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap/</file_path> Line 139: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap/</file_path> Line 143: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap/</file_path> Line 147: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap/</file_path> Line 151: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap/</file_path> Line 155: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap/</file_path> Line 719: <sparse_dir>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap/</sparse_dir></pre>
QCS405 (perf)	<apps_image_base>/qcs405-som1-qsap-perf/	<pre>Line 127: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap-perf/</file_path> Line 131: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap-perf/</file_path> Line 135: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap-perf/</file_path> Line 139: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap-perf/</file_path> Line 143: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap-perf/</file_path> Line 147: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap-perf/</file_path> Line 151: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap-perf/</file_path> Line 155: <file_path>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap-perf/</file_path> Line 719: <sparse_dir>apps_proc/poky/build/tmp-glibc/deploy/images/qcs405-som1-qsap-perf/</sparse_dir></pre>

Generate Final Image

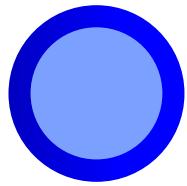
Generate Final Image commands for QCS40x	
Setup	<QCS40x_WS>\$ cd common/build
Generate	<QCS40x_WS>/common/build\$ python build.py
Build log	[22:40:43] - update_meta.py: Validating File paths in contents.xml [22:40:43] - ===== [22:40:43] - update_meta.py: Total number of file paths found in contents.xml = 93 [22:40:43] - update_meta.py: file path validation passed. [22:40:43] - ===== CHECK COMPLETE ===== [22:40:43] - update_common_info.py:===== UPDATE COMMON INFO COMPLETE=====
Build log file	<QCS403_WS>/common/build/ update_common.log

Generate Final Image

- The following errors may be logged in the QCS405 build log
- This is because we are currently using unsigned images so this error can be ignored for now

Build errors example

```
...  
[31mERROR: Following validations failed for the image:  
Image /local2/mnt/workspace/QCS40x/r1.0_00007.0/QCS405_WS/trustzone_images/build/ms/bin/OAPAANAA/smplap32.mbn is not encrypted  
Image /local2/mnt/workspace/QCS40x/r1.0_00007.0/QCS405_WS/trustzone_images/build/ms/bin/OAPAANAA/smplap64.mbn is not encrypted  
Image /local2/mnt/workspace/QCS40x/r1.0_00007.0/QCS405_WS/trustzone_images/build/ms/bin/OAPAANAA/securemm.mbn is not encrypted  
Image /local2/mnt/workspace/QCS40x/r1.0_00007.0/QCS405_WS/trustzone_images/build/ms/bin/OAPAANAA/cmnlib.mbn is not encrypted  
Image /local2/mnt/workspace/QCS40x/r1.0_00007.0/QCS405_WS/trustzone_images/build/ms/bin/OAPAANAA/cmnlib64.mbn is not encrypted  
Image /local2/mnt/workspace/QCS40x/r1.0_00007.0/QCS405_WS/trustzone_images/build/ms/bin/OAPAANAA/cmnlib.mbn is not encrypted  
Image /local2/mnt/workspace/QCS40x/r1.0_00007.0/QCS405_WS/trustzone_images/build/ms/bin/OAPAANAA/cmnlib64.mbn is not encrypted  
Image /local2/mnt/workspace/QCS40x/r1.0_00007.0/QCS405_WS/trustzone_images/build/ms/bin/OAPAANAA/keymaster64.mbn is not encrypted  
Image /local2/mnt/workspace/QCS40x/r1.0_00007.0/QCS405_WS/wlan_proc/build/ms/bin/QCAHNALAVDL/qdsp6sw.mbn is not encrypted  
...
```



Section 6

Device Boot Configuration for SSRD

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

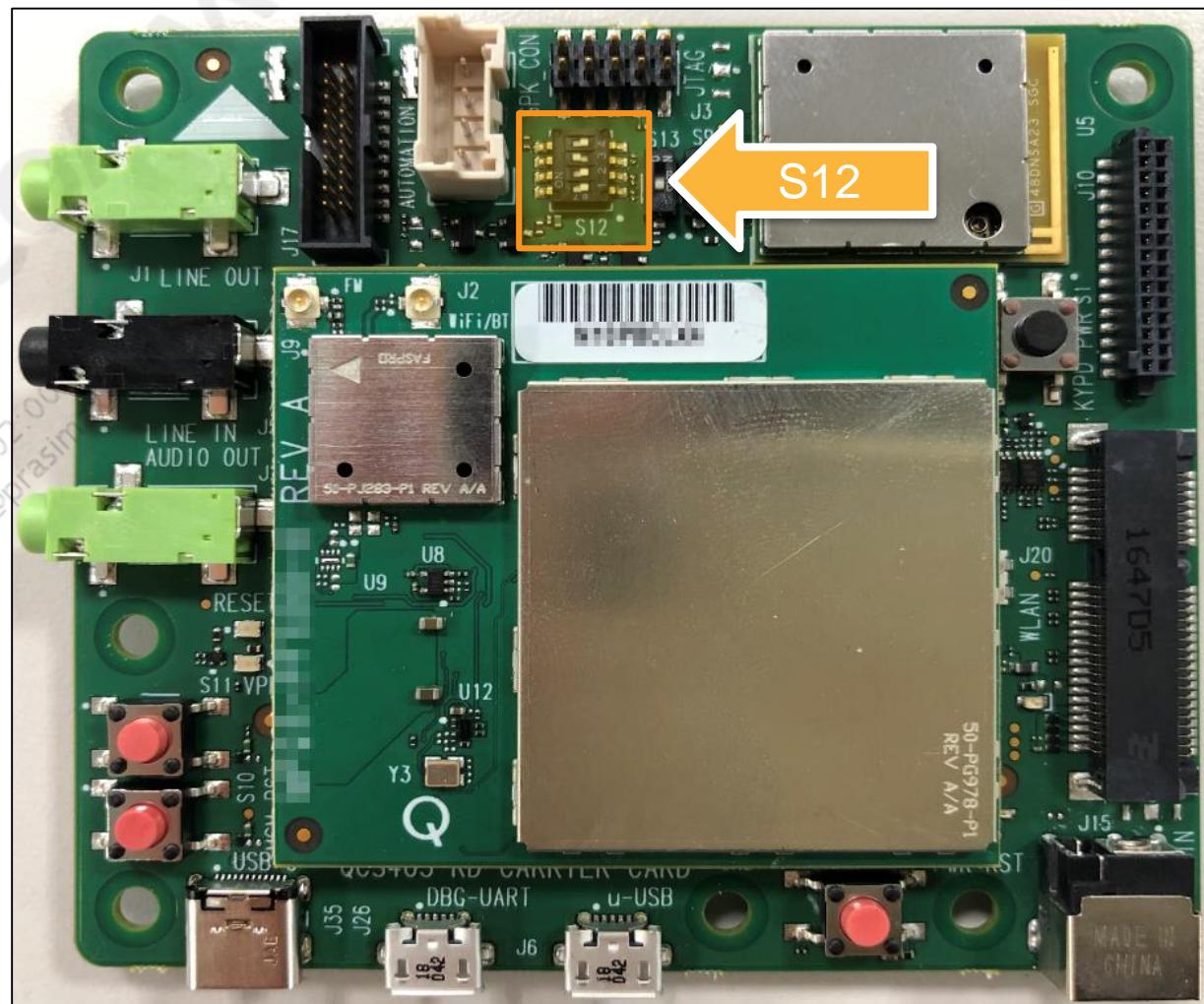
Device Boot Configuration for SSRD

- QCS403 has the following boot options using S12 circuit

Boot option	GPIO_45 (PIN #4)	GPIO_49 (PIN #3)	GPIO_57 (PIN #2)	GPIO_56 (PIN #1)
eMMC > SD > HS USB	0	0	0	0
SD > eMMC > EDL	0	0	0	1
eMMC > EDL	0	0	1	0
HS USB	0	0	1	1
NAND > EDL (default)	0	1	0	0
Forced USB BOOT	1	x	x	x

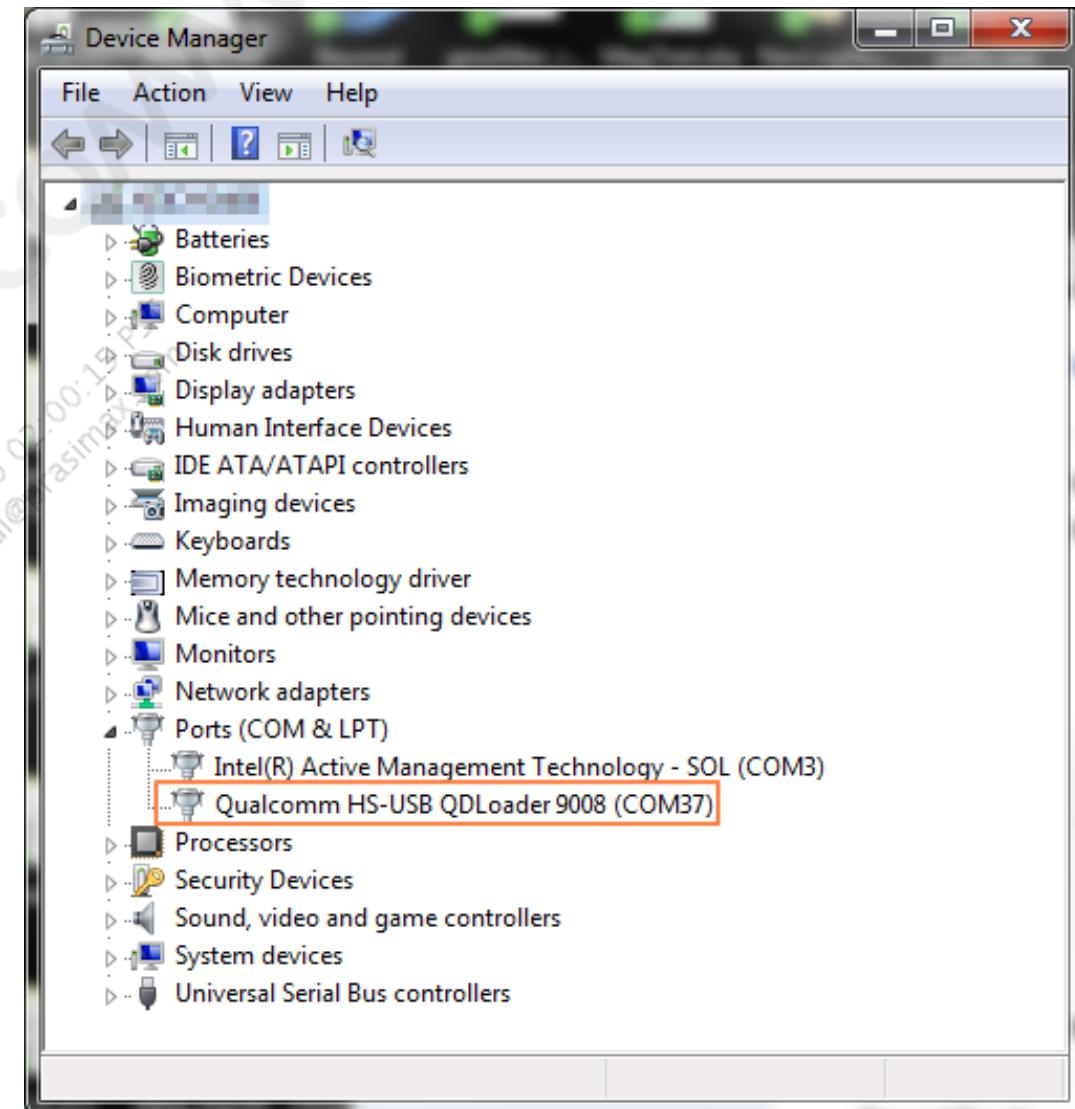
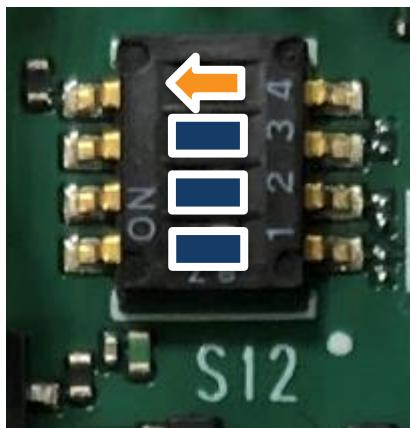
- Below is the pin assignment of GPIOs for S12

Pin #	GPIO
1	GPIO_56 / BOOT_CONFIG[1]
2	GPIO_57 / FAST_BOOT_CONFIG[2]
3	GPIO_49 / FAST_BOOT_CONFIG[3]
4	GPIO_45 / Forced USB BOOT



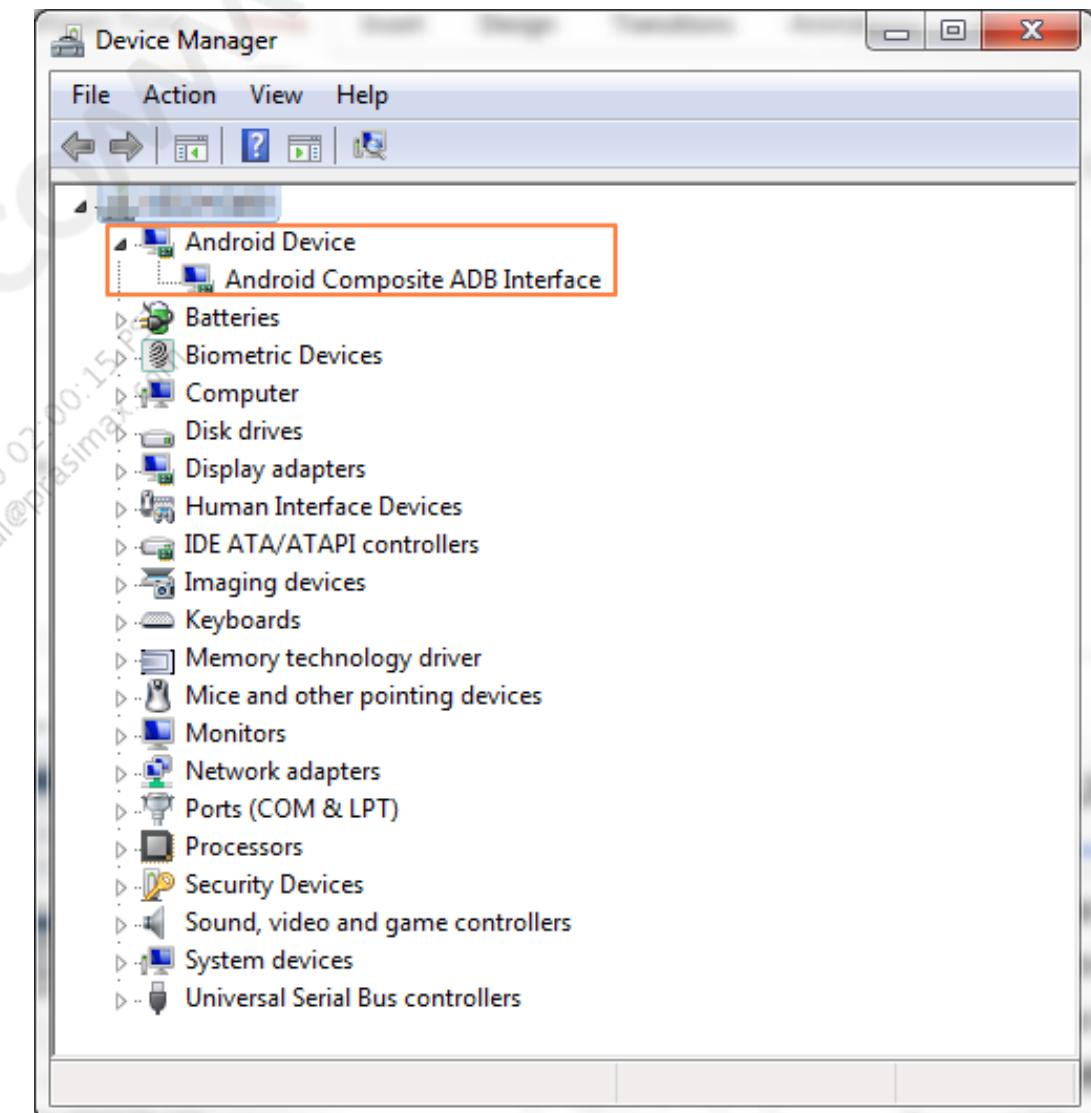
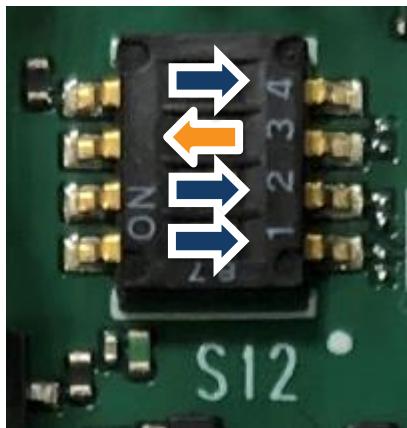
Device Boot Configuration for SSRD – EDL mode

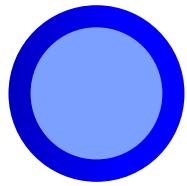
- To configure device to EDL mode, follow the instruction below
 - 1) Remove Main Power from target board
 - 2) Turn ON PIN#4 in S12, Don't care other PINs
 - 3) Connect Micro USB Cable with Host PC
 - 4) Restore Main Power to target board
 - 5) Check Device Manager > **QDLoader 9008** port is enumerated



Device Boot Configuration for SSRD – Normal boot mode

- To configure device to operation mode, follow the instruction below
 - Remove Main Power from target board
 - Turn ON PIN#3 and Turn OFF other PINs in S12
 - Connect Micro USB Cable with Host PC
 - Restore Main Power to target board
 - Check Device Manager > **ADB Interface** is enumerated





Section 7

Writing Images

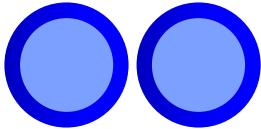
Qualcomm
Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

7.1 Writing Images using QFIL	150
7.2 Writing Images using fastboot	160
7.3 Packaging Images	167
7.3.1 Packaging Images using QFIL	169
7.3.2 Writing Flat Build Images using QFIL	179

Writing Images

- In this chapter, only basic procedures will be explained
 - Writing Images using:
 - Qualcomm Flash Image Loader (QFIL) which is included in QPST
 - Fastboot
 - Packaging Images using [QFIL](#)
- For more details, refer to [80-YB405-129] *QCS40x Software User Guide* document

Qualcomm Confidential and Proprietary
2019-12-30 00:15 PST
didi.setiadi@prajakta.com



Section 7.1

Writing Images using QFIL

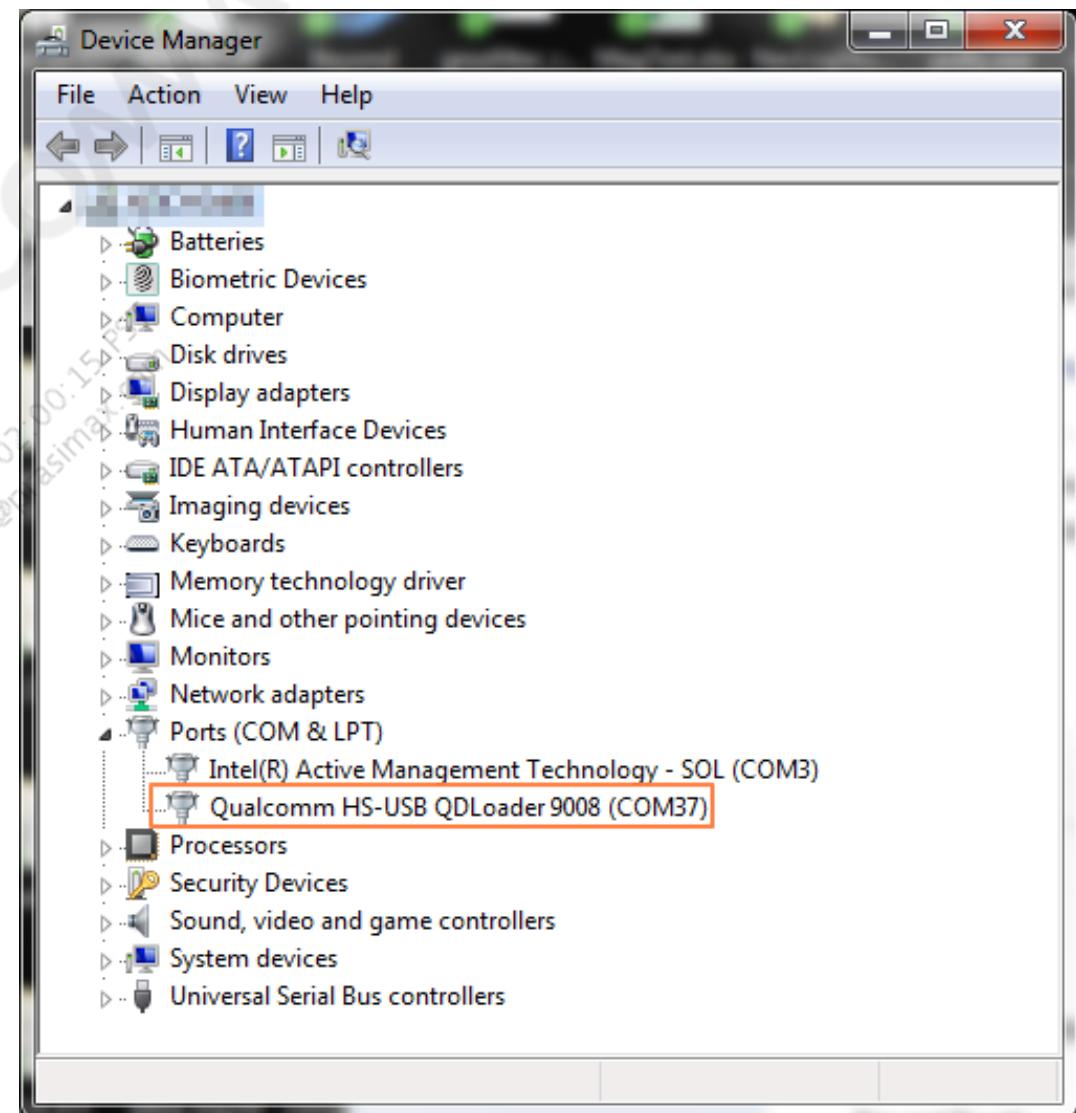
Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Writing Images using QFIL

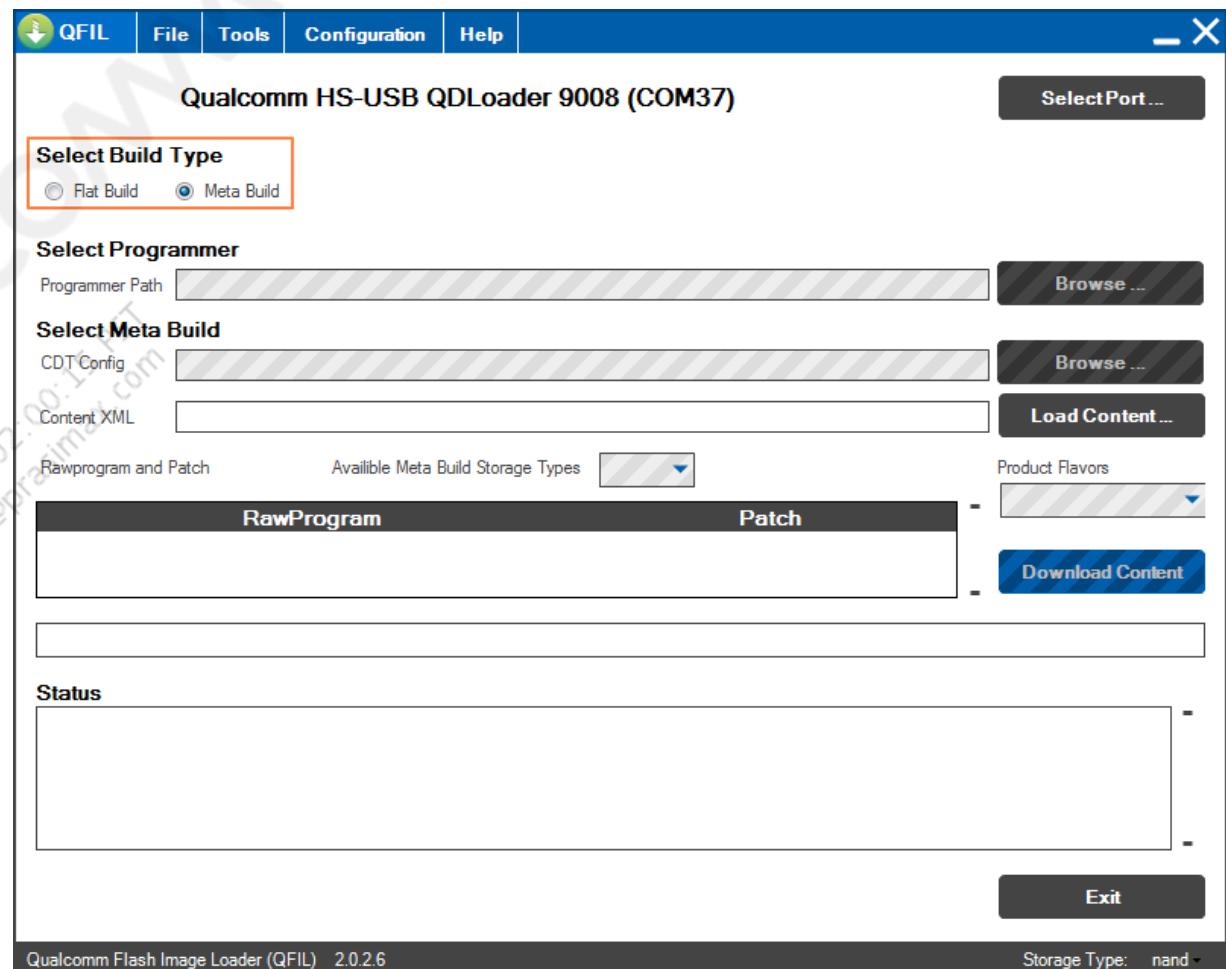
- To write images using QFIL, the device must be configured to EDL mode
- Refer to:
 - [Device Boot Configuration for SSRD - EDL mode](#)

Confidential
2019-12-30 01:00:15
didi.setiadi@rasimahaxi



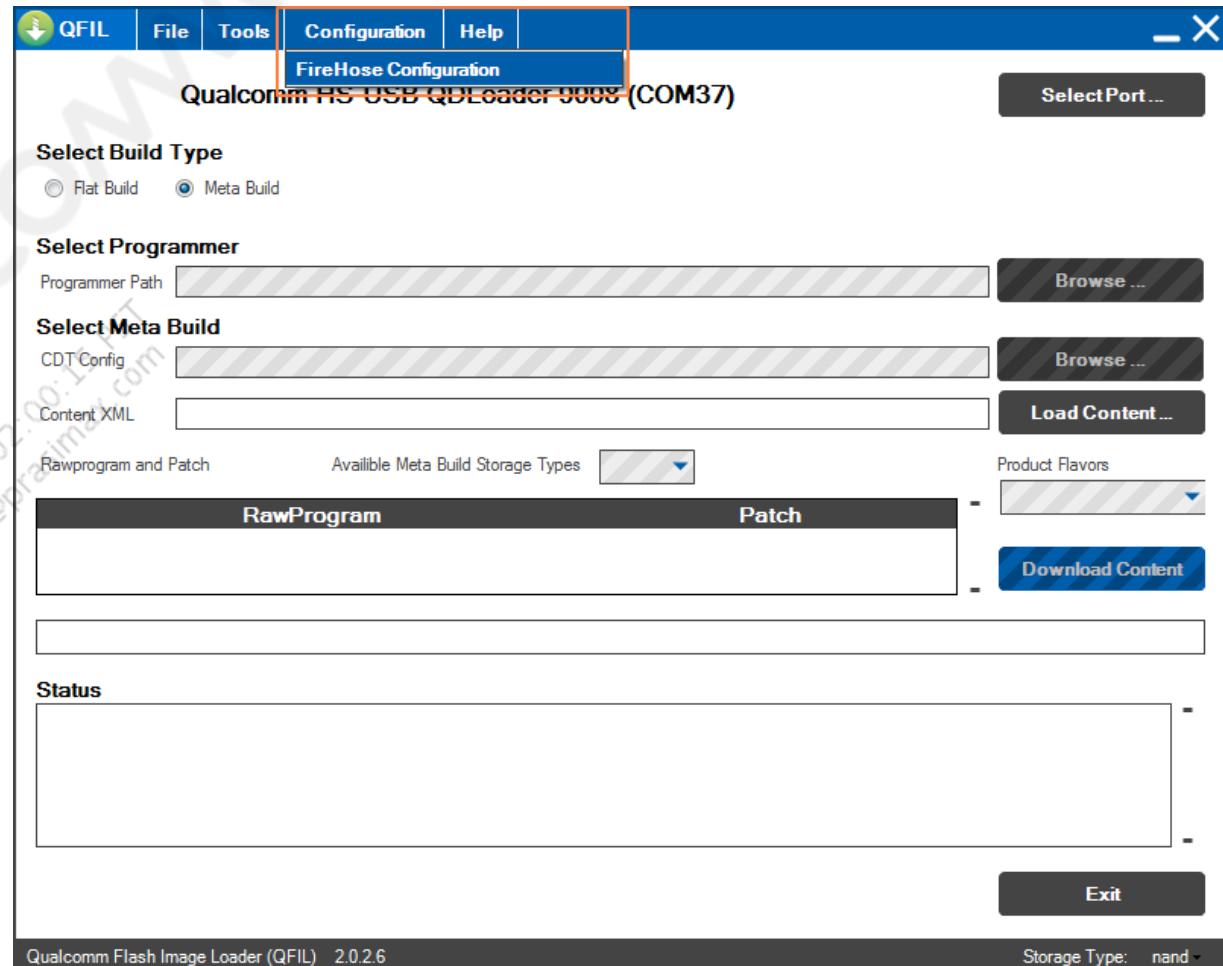
Writing Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'



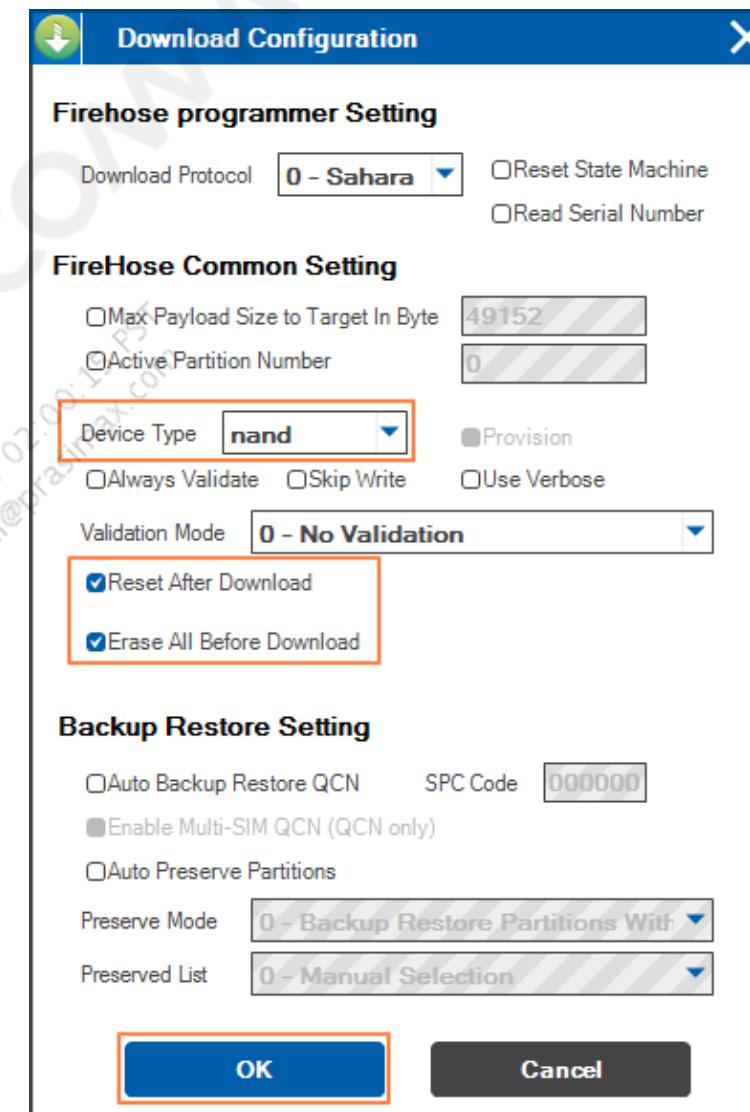
Writing Images using QFIL

- Launch QFIL and Select Build Type to ‘Meta Build’
- Click on ‘Configuration > FireHose Configuration’ to check or modify settings for QCS403 (NAND)



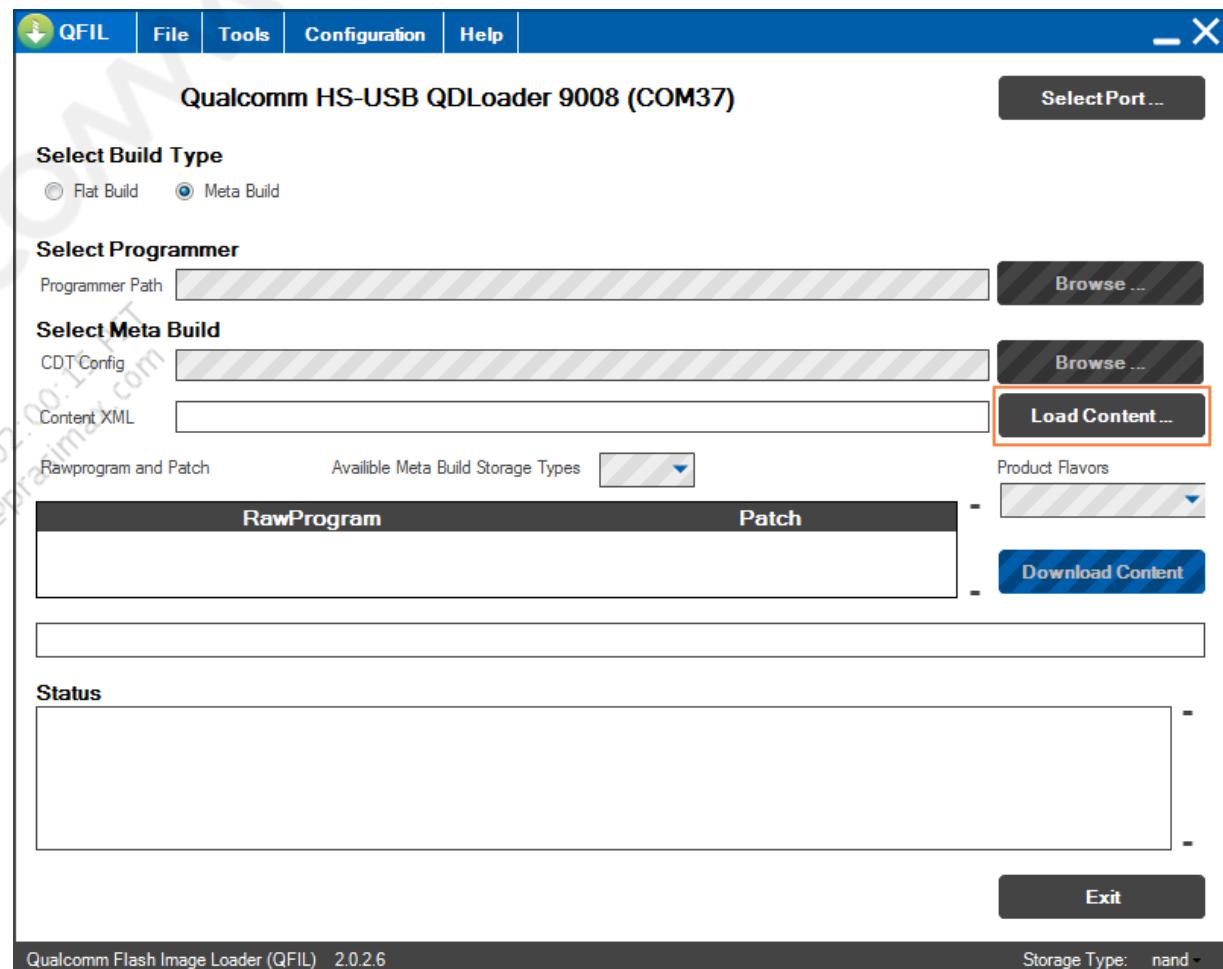
Writing Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON



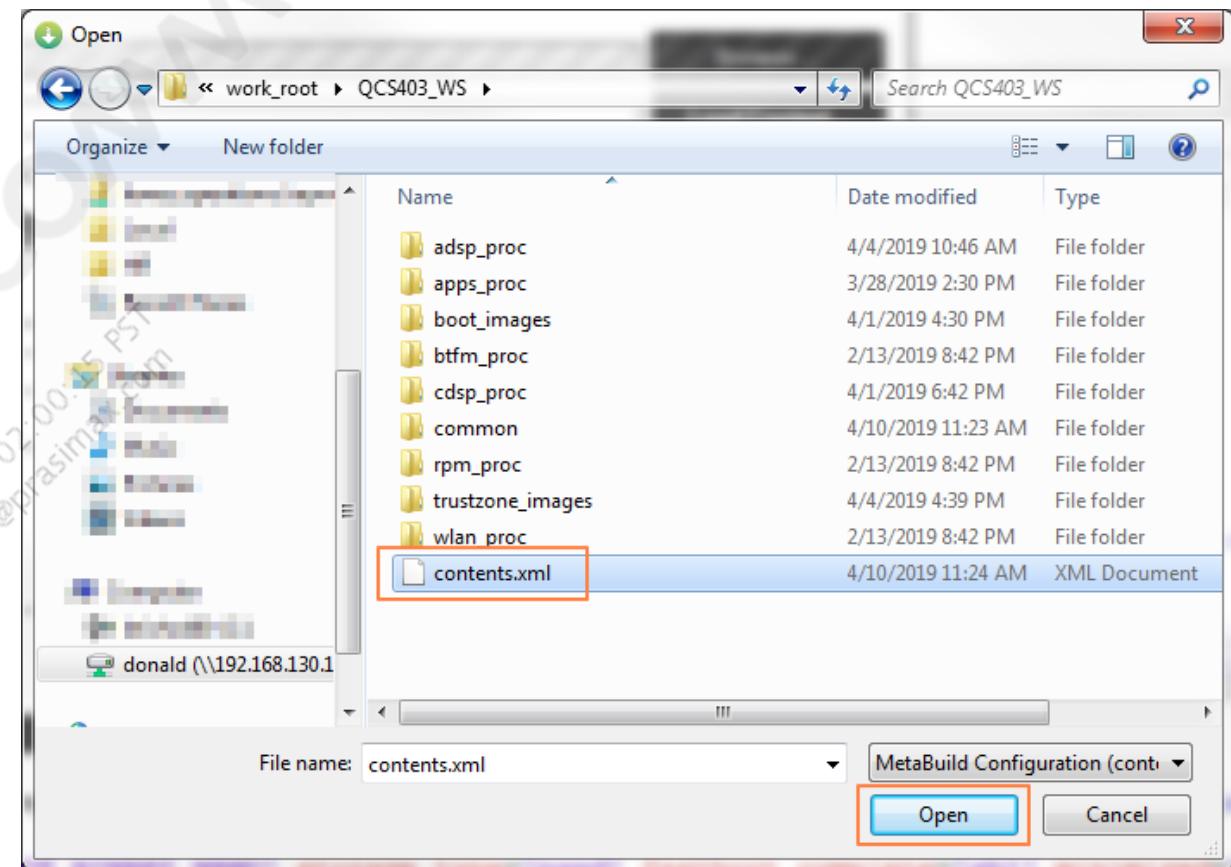
Writing Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on 'Load Content...' to load contents.xml file



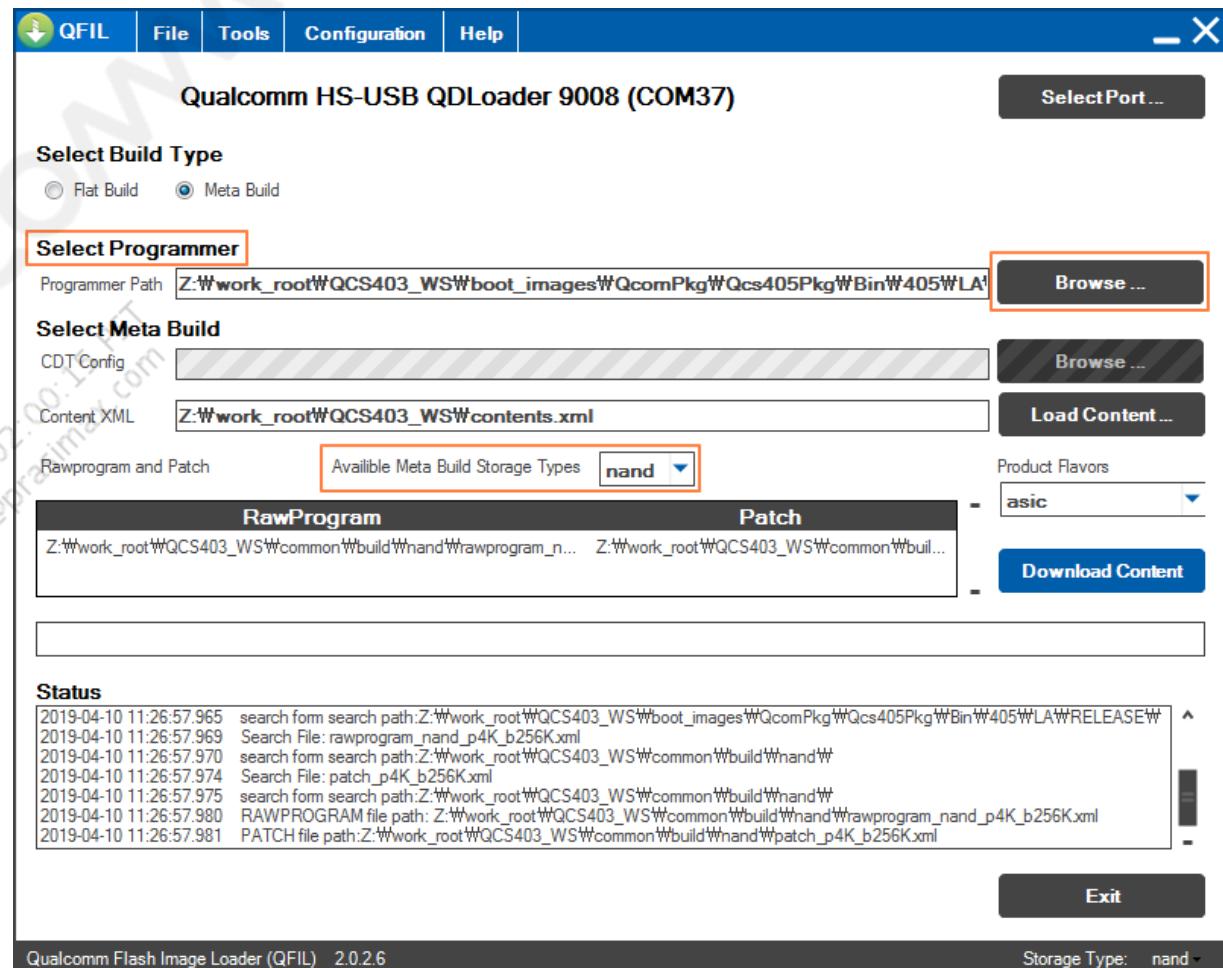
Writing Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on 'Load Content...' to load contents.xml file
- Select <QCS403_WS>/contents.xml file and Click on 'Open'



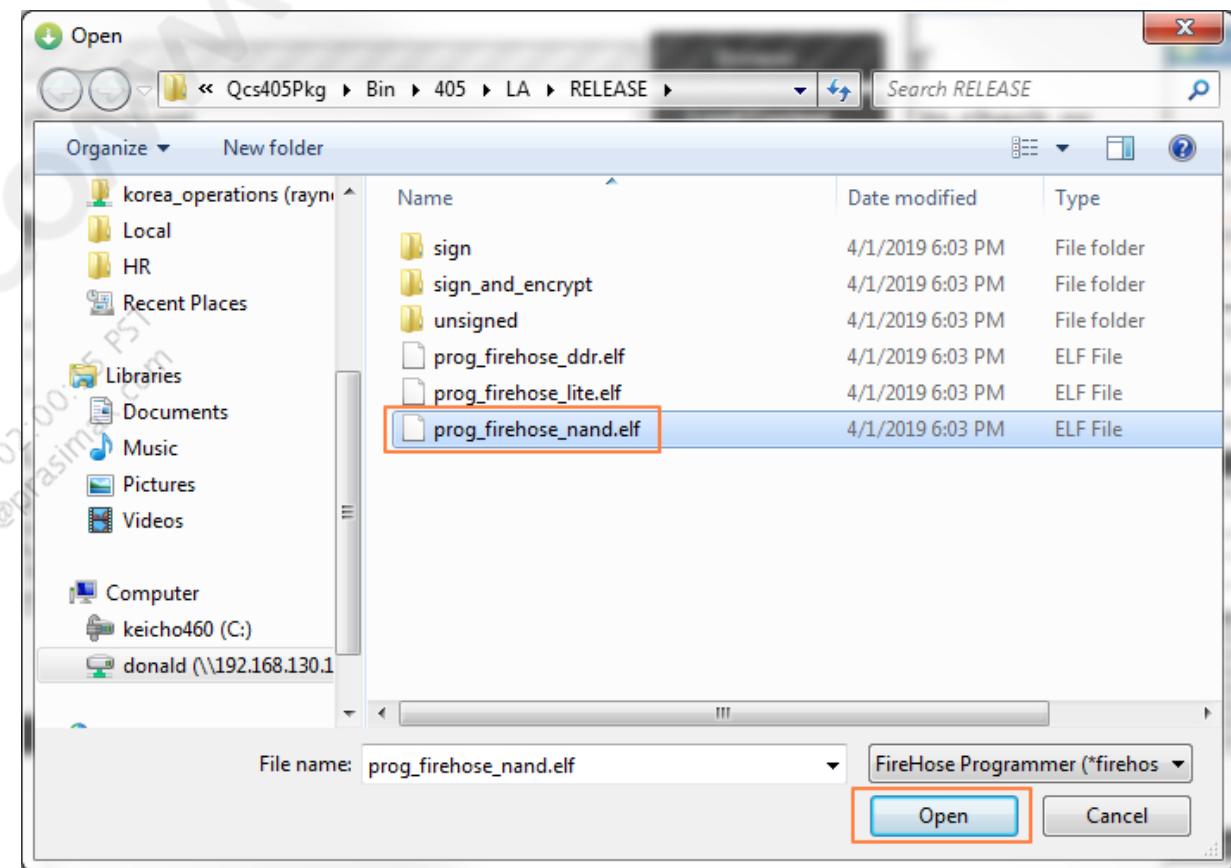
Writing Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on 'Load Content...' to load contents.xml file
- Select <QCS403_WS>/contents.xml file and Click on 'Open'
- Make sure 'Available Meta Build Storage Types' is 'nand'
- Click on 'Select Programmer > Browse...' button



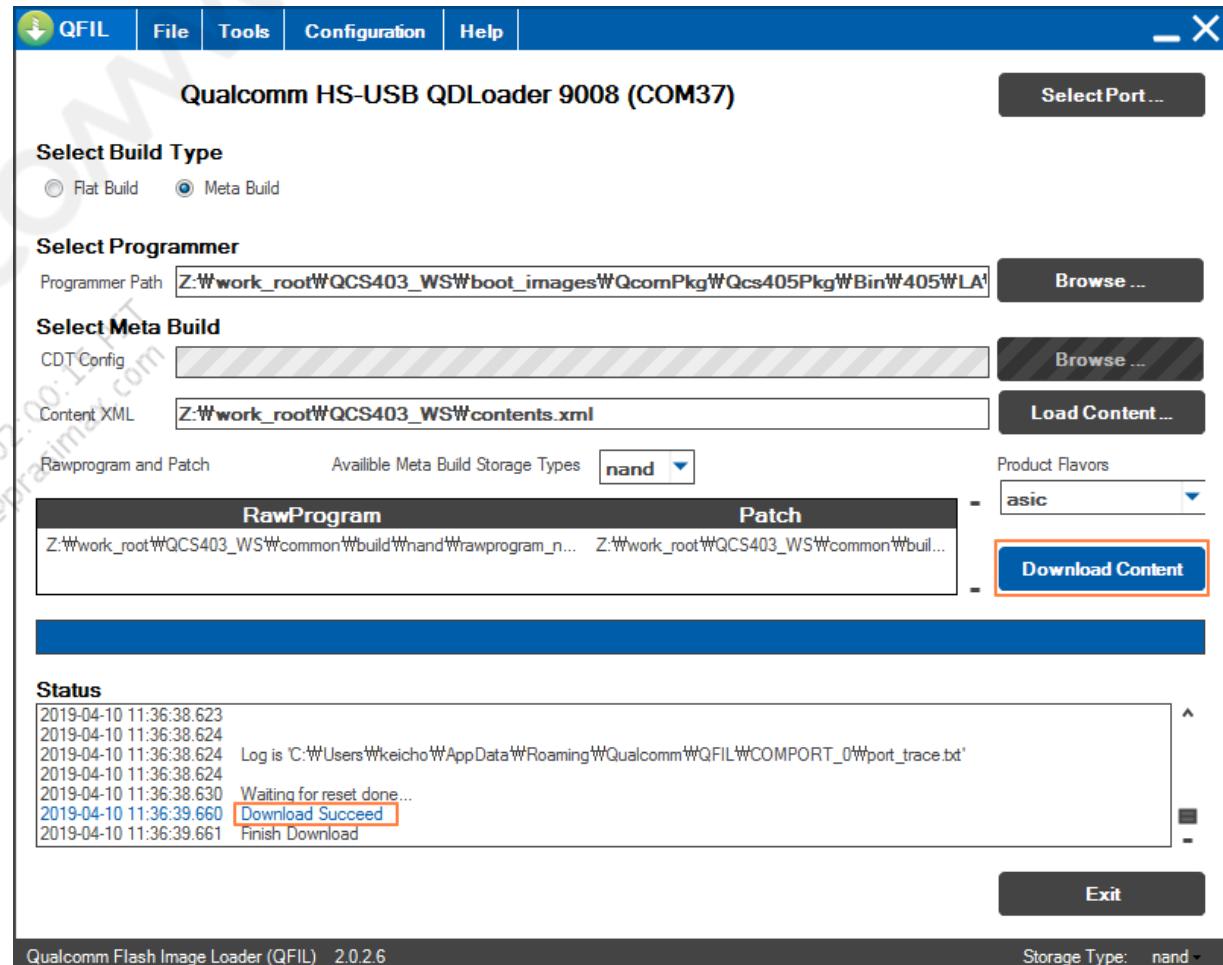
Writing Images using QFIL

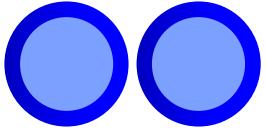
- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on 'Load Content...' to load contents.xml file
- Select <QCS403_WS>/contents.xml file and Click on 'Open'
- Make sure 'Available Meta Build Storage Types' is 'nand'
- Click on 'Select Programmer > Browse...' button
- Select 'prog_firehose_nand.elf' file and Click on 'Open'



Writing Images using QFIL

- Launch QFIL and Select Build Type to ‘Meta Build’
- Click on ‘Configuration > FireHose Configuration’ to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on ‘OK’
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on ‘Load Content...’ to load contents.xml file
- Select <QCS403_WS>/contents.xml file and Click on ‘Open’
- Make sure ‘Available Meta Build Storage Types’ is ‘nand’
- Click on ‘Select Programmer > Browse...’ button
- Select ‘prog_firehose_nand.elf’ file and Click on ‘Open’
- Click on ‘Download Content’ button
- Make sure “Download Succeed” message
- DONE !





Section 7.2

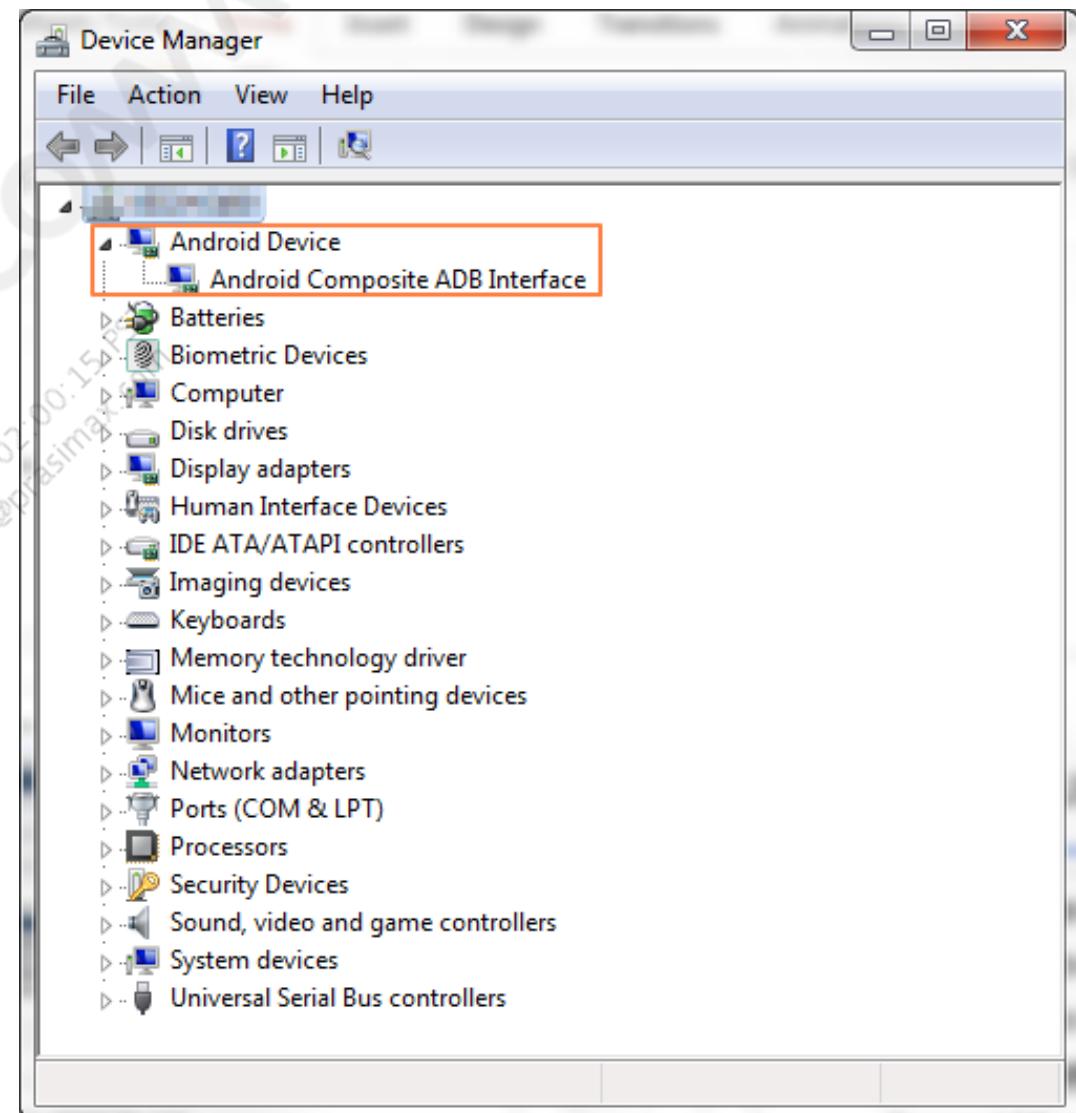
Writing Images using fastboot

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Writing Images using fastboot – Windows

- To write images using fastboot, the device must be configured to normal boot mode
- Refer to:
 - [Device Boot Configuration for SSRD – Normal boot mode](#)

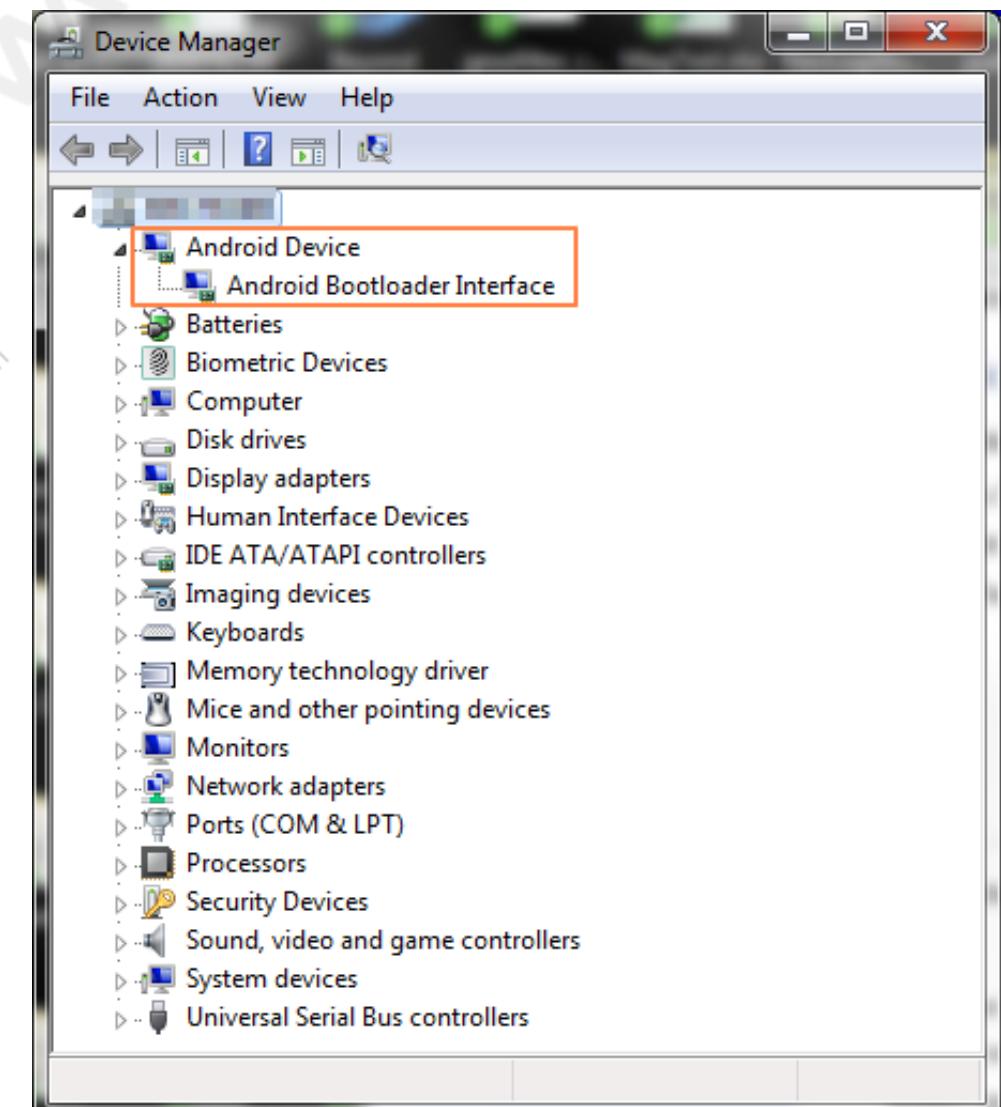
Confidential
2019-12-30 02:00:15
didi.setiadi@rasinamax



Writing Images using fastboot – Windows

- To write images using fastboot, the device must be configured to normal boot mode
- Refer to:
 - [Device Boot Configuration for SSRD – Normal boot mode](#)
- After the device boots into adb mode, reboot device from adb mode to fastboot mode

Command	Description
> <code>adb devices</code>	Check adb interface
> <code>adb reboot bootloader</code>	Switch to fastboot mode
> <code>fastboot devices</code>	Check fastboot interface



Writing Images using fastboot – Windows

	Commands for Writing Images with python script
Prerequisite	✓ Python 2.7
Setup	> pushd <QCS403_WS>/common/build
Run writing script	<QCS403_WS>/common/build> python fastboot_complete.py --st=nand
Log	<pre>... ===== fastboot_complete.py: Processing input arguments ===== ... fastboot.exe flash abl Z:\work_root\QCS403_WS\apps_proc\poky\build\tmp-glibc\deploy\images\qcs403-som2-qsap\abl-squashfs.elf fastboot.exe flash boot Z:\work_root\QCS403_WS\apps_proc\poky\build\tmp-glibc\deploy\images\qcs403-som2-qsap\qcs40x-boot.img fastboot.exe flash logfs Z:\work_root\QCS403_WS\boot_images\QcomPkg\Tools/binaries/logfs_ufs_8mb.bin fastboot.exe flash modem Z:\work_root\QCS403_WS\common\build\flash\modem\NON-HLOS.ubi fastboot.exe flash pmic Z:\work_root\QCS403_WS\boot_images\QcomPkg\Qcs405Pkg\Bin\405/LA/RELEASE/pmic.elf fastboot.exe flash rpm Z:\work_root\QCS403_WS\rpm_proc\build\ms\bin\AAAAANAZR/rpm.mbn fastboot.exe flash sbl Z:\work_root\QCS403_WS\boot_images\QcomPkg\Qcs405Pkg\Bin\405/LA/RELEASE/xbl_nand.elf fastboot.exe flash system Z:\work_root\QCS403_WS\apps_proc\poky\build\tmp-glibc\deploy\images\qcs403-som2-qsap\qcs40x-squashfs-sysfs.ubi fastboot.exe flash tz Z:\work_root\QCS403_WS\trustzone_images\build\ms\bin\OAPAANAA/tz.mbn fastboot.exe flash tz_devcfg Z:\work_root\QCS403_WS\trustzone_images\build\ms\bin\OAPAANAA/devcfg.mbn ... ===== FLASHING SUCCESSFUL! ===== fastboot_complete.py: Loading complete.</pre>

Writing Images using fastboot – Linux

- After the device boots into adb mode, reboot device from adb mode to fastboot mode

Command	Description
\$ lsusb	Check connected USB devices
\$ sudo adb start-server	Start adb server with sudo permission
\$ adb devices	Check adb interface
\$ adb reboot bootloader	Switch to fastboot mode
\$ sudo fastboot devices	Check fastboot interface

- If ‘no permissions’ error has occurred on adb or fastboot then USE ‘sudo’ command
- OR, add rules for the two USB devices to ‘51-android.rules’ file
 - File path: /etc/udev/rules.d/51-android.rules
 - Add rules and Save the file:

```
# For QUALCOMM USB device
SUBSYSTEM=="usb",ATTR{idVendor}=="05c6",MODE=="0666",GROUP="plugdev"
# For ANDROID USB device
SUBSYSTEM=="usb",ATTR{idVendor}=="18d1",MODE=="0666",GROUP="plugdev"
```
 - Restart service: \$ sudo service udev restart

```
donald@ubuntu:~$ lsusb
Bus 001 Device 002: ID 05c6:901d Qualcomm, Inc.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 003: ID 0e0f:0002 VMware, Inc. Virtual USB Hub
Bus 002 Device 002: ID 0e0f:0003 VMware, Inc. Virtual Mouse
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
donald@ubuntu:~$ 
donald@ubuntu:~$ sudo adb start-server
* daemon not running. starting it now on port 5037 *
* daemon started successfully *
donald@ubuntu:~$ 
donald@ubuntu:~$ adb devices
List of devices attached
8bbd0412          device
donald@ubuntu:~$ 
donald@ubuntu:~$ adb reboot bootloader
donald@ubuntu:~$ fastboot devices
no permissions    fastboot
donald@ubuntu:~$ sudo fastboot devices
8bbd0412          fastboot
donald@ubuntu:~$
```

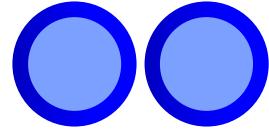
Writing Images using fastboot – Linux

	Commands for Writing Images with python script
Prerequisite	✓ Python 2.7
Setup	<QCS403_WS>\$ cd common/build
Run writing script	<QCS403_WS>/common/build\$ sudo python fastboot_complete.py --st=nand
Log	<pre>... ===== fastboot_complete.py: Processing input arguments ===== ... fastboot flash abl /home/donald/work_root/QCS403_WS/apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap/abl-squashfs.elf fastboot flash boot /home/donald/work_root/QCS403_WS/apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap/qcs40x-boot.img fastboot flash logfs /home/donald/work_root/QCS403_WS/boot_images/QcomPkg/Tools/binaries/logfs_ufs_8mb.bin fastboot flash modem /home/donald/work_root/QCS403_WS/common/build/nand/NON-HLOS.ubi fastboot flash pmic /home/donald/work_root/QCS403_WS/boot_images/QcomPkg/Qcs405Pkg/Bin/405/LA/RELEASE/pmic.elf fastboot flash rpm /home/donald/work_root/QCS403_WS/rpm_proc/build/ms/bin/AAAAANAZR/rpm.mbn fastboot flash sbl /home/donald/work_root/QCS403_WS/boot_images/QcomPkg/Qcs405Pkg/Bin/405/LA/RELEASE/xbl_nand.elf fastboot flash system /home/donald/work_root/QCS403_WS/apps_proc/poky/build/tmp-glibc/deploy/images/qcs403-som2-qsap/qcs40x-squashfs-sysfs.ubi fastboot flash tz /home/donald/work_root/QCS403_WS/trustzone_images/build/ms/bin/OAPAANAA/tz.mbn fastboot flash tz_devcfg /home/donald/work_root/QCS403_WS/trustzone_images/build/ms/bin/OAPAANAA/devcfg.mbn ... ===== FLASHING SUCCESSFUL! ===== fastboot_complete.py: Loading complete.</pre>

Writing Images using fastboot

- Each binary can also be flashed using the following fastboot command

SW Module	Fastboot Command	Image File Location
APPS_PROC	> fastboot flash abl abl-squashfs.elf > fastboot flash boot qcs40x-boot.img > fastboot flash system qcs40x-squashfs-sysfs.ubi	<QCS403_WS>/apps_proc/poky/build/tmp-glibc/deploy/images/<build_variant>/ e.g. <build_variant> is qcs403-som2-qsap or qcs403-som2-qsap-perf
RPM_PROC	> fastboot flash rpm rpm.mbn	<QCS403_WS>/rpm_proc/build/ms/bin/AAAAANAZR/
ADSP_PROC CDSP_PROC WLAN_PROC BTFM_PROC	> fastboot flash modem NON-HLOS.ubi	<QCS403_WS>/common/build/nand/
BOOT_IMAGES	> fastboot flash sbl xbl_nand.elf > fastboot flash pmic pmic.elf	<QCS403_WS>/boot_images/QcomPkg/Qcs405Pkg/Bin/405/LA/RELEASE/
	> fastboot flash logfs logfs_ufs_8mb.bin	<QCS403_WS>/boot_images/QcomPkg/Tools/binaries/
TRUSTZONE	> fastboot flash tz tz.mbn > fastboot flash devcfg devcfg.mbn	<QCS403_WS>/trustzone_images/build/ms/bin/OAPAANAA/



Section 7.3

Packaging Images

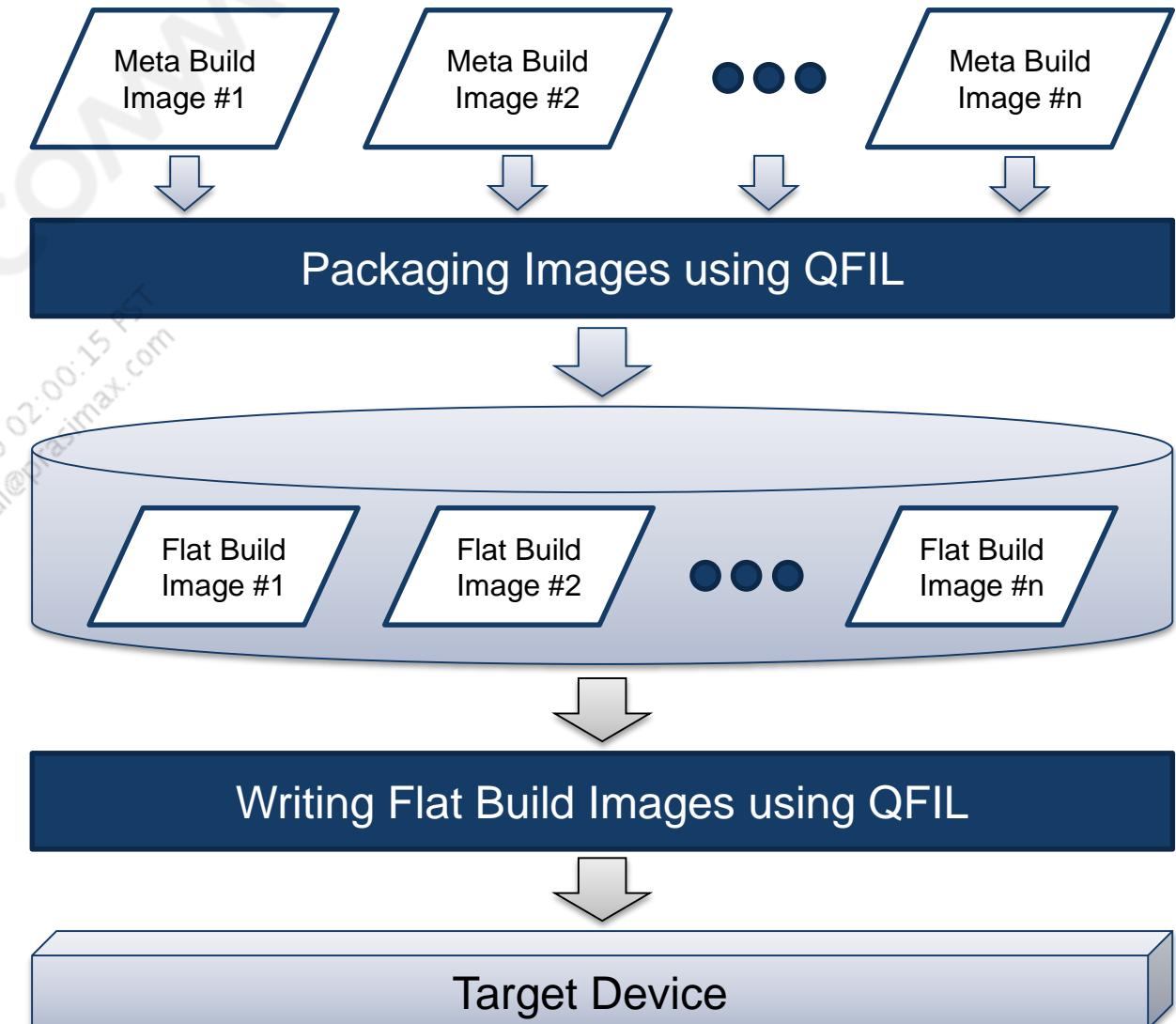
Qualcomm

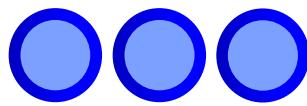
Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

- | | |
|--|---------------------|
| 7.3.1 Packaging Images using QFIL | 169 |
| 7.3.2 Writing Flat Build Images using QFIL | 179 |

Packaging Images

- Once build process is completed, images are located in multiple folders
- Packaging Images is the process of collecting these images in one folder
- This makes it easier to deploy to a machine other than the build machine
- In this chapter, following process will be explained
 - How to Package Images using QFIL
 - How to write the result of Packaging Images using QFIL
- In QFIL, Packaging Images process is called 'Flat Meta Build'
- The result of Packaging Images is also called 'Flat Build Images'





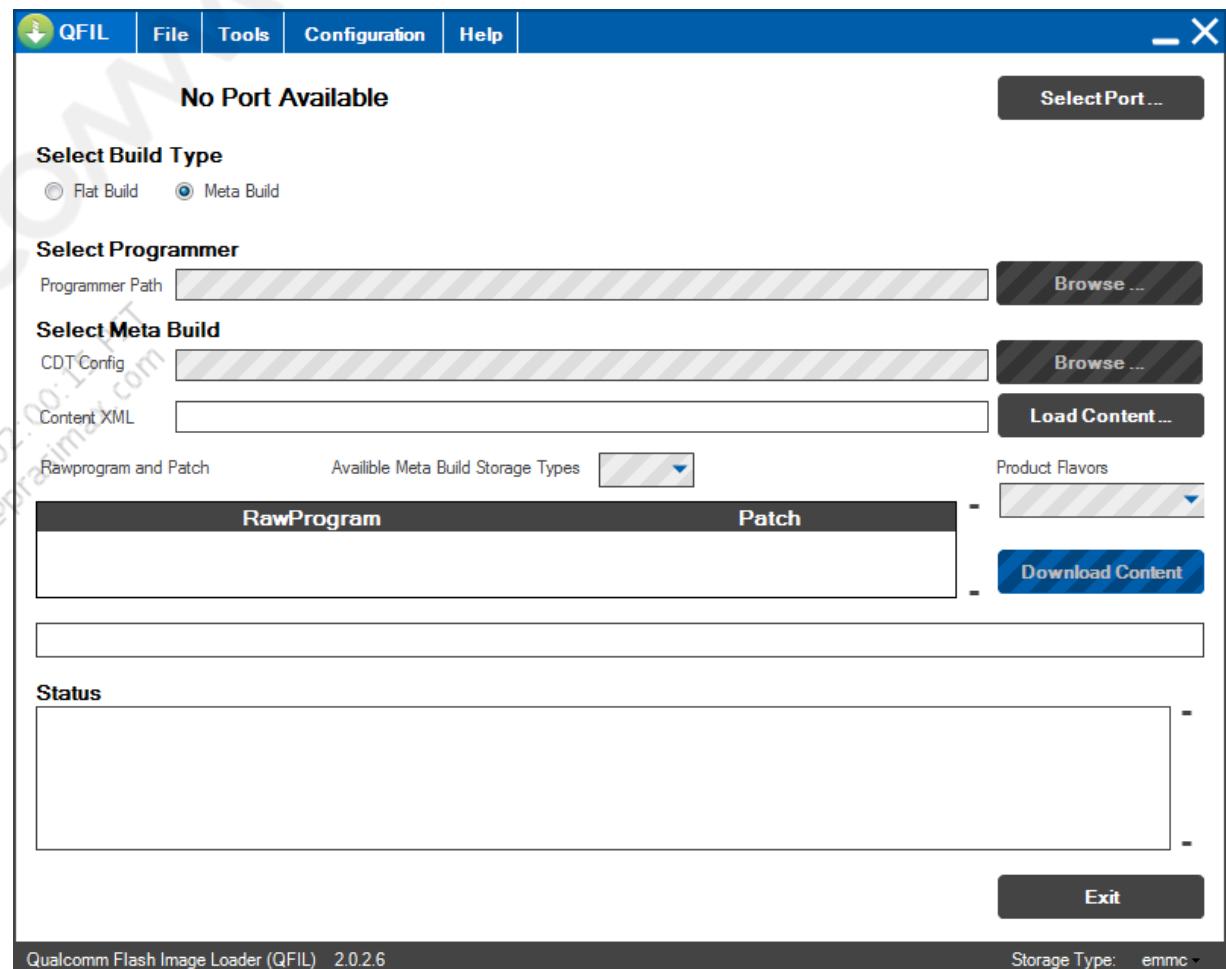
Section 7.3.1

Packaging Images using QFIL

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

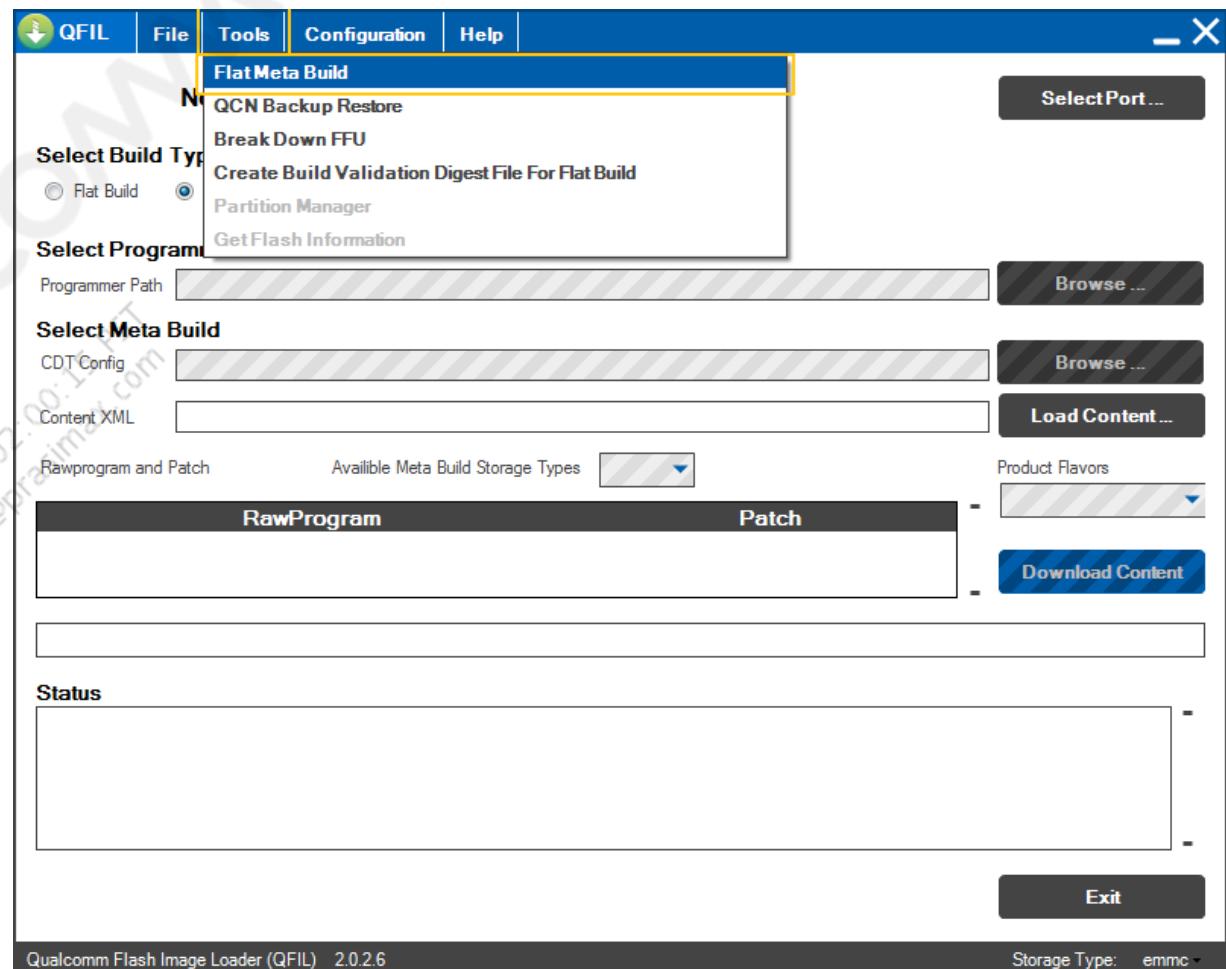
Packaging Images using QFIL

- Launch QFIL



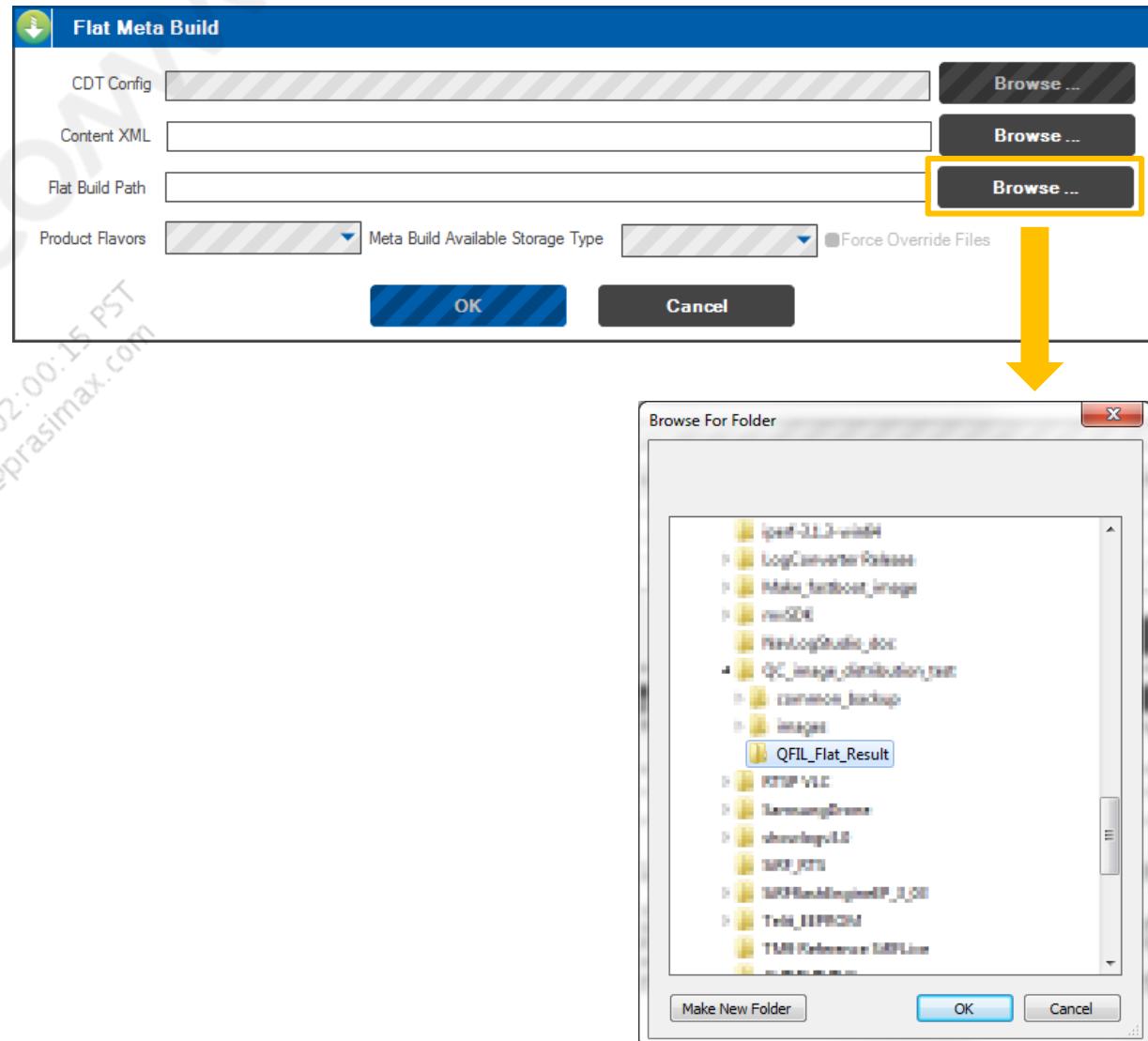
Packaging Images using QFIL

- Launch QFIL
- Click on ‘Tools > Flat Meta Build’ menu



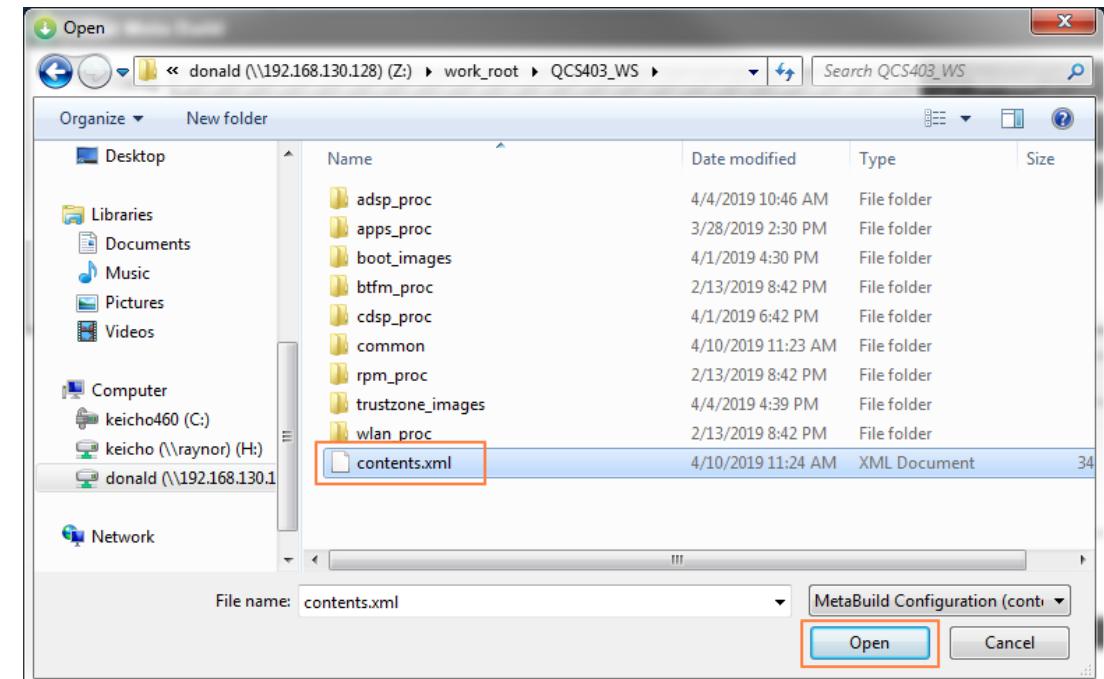
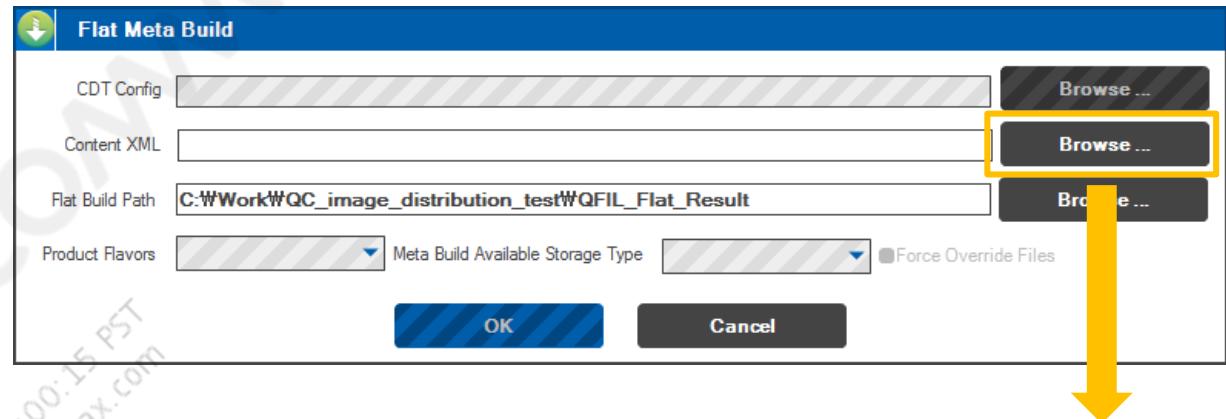
Packaging Images using QFIL

- Launch QFIL
- Click on 'Tools > Flat Meta Build' menu
- Click on 'Flat Build Path > Browse...' button
- Select <OUTPUT_FOLDER>



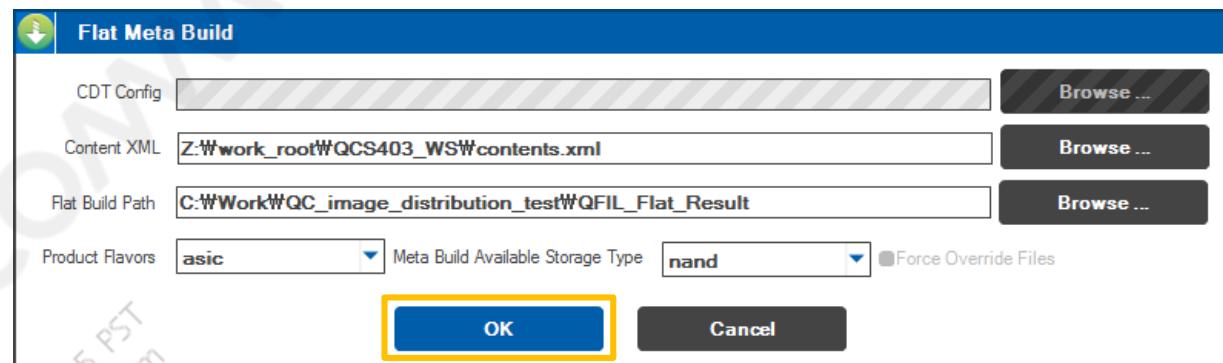
Packaging Images using QFIL

- Launch QFIL
- Click on ‘Tools > Flat Meta Build’ menu
- Click on ‘Flat Build Path > Browse...’ button
- Select <OUTPUT_FOLDER>
- Click on ‘Content XML > Browse...’ button
- Open ‘<QCS403_WS>/contents.xml’ file



Packaging Images using QFIL

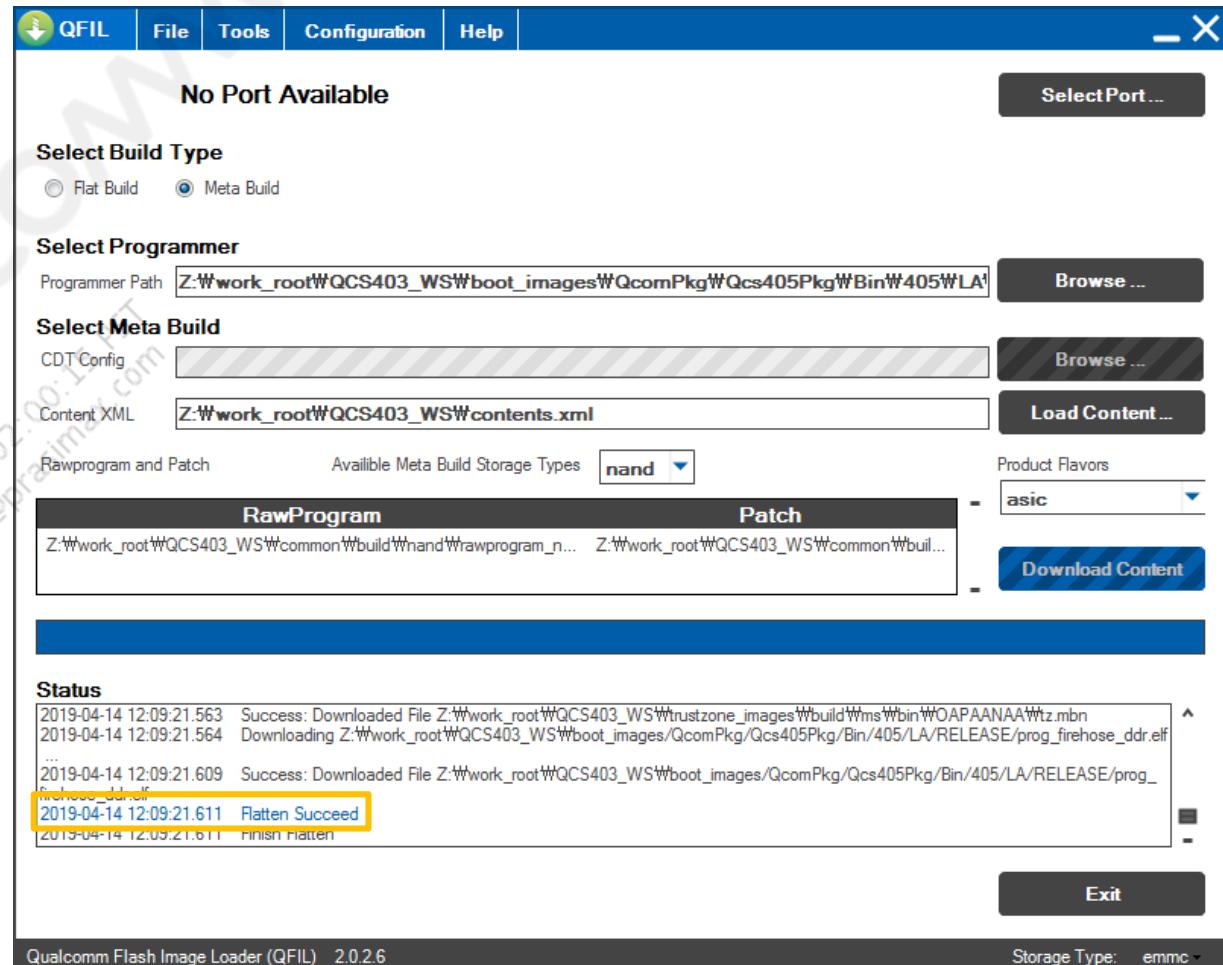
- Launch QFIL
- Click on ‘Tools > Flat Meta Build’ menu
- Click on ‘Flat Build Path > Browse...’ button
- Select <OUTPUT_FOLDER>
- Click on ‘Content XML > Browse...’ button
- Open ‘<QCS403_WS>/contents.xml’ file
- Click on ‘OK’ button to start Flat Meta Build



Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasmimax.com

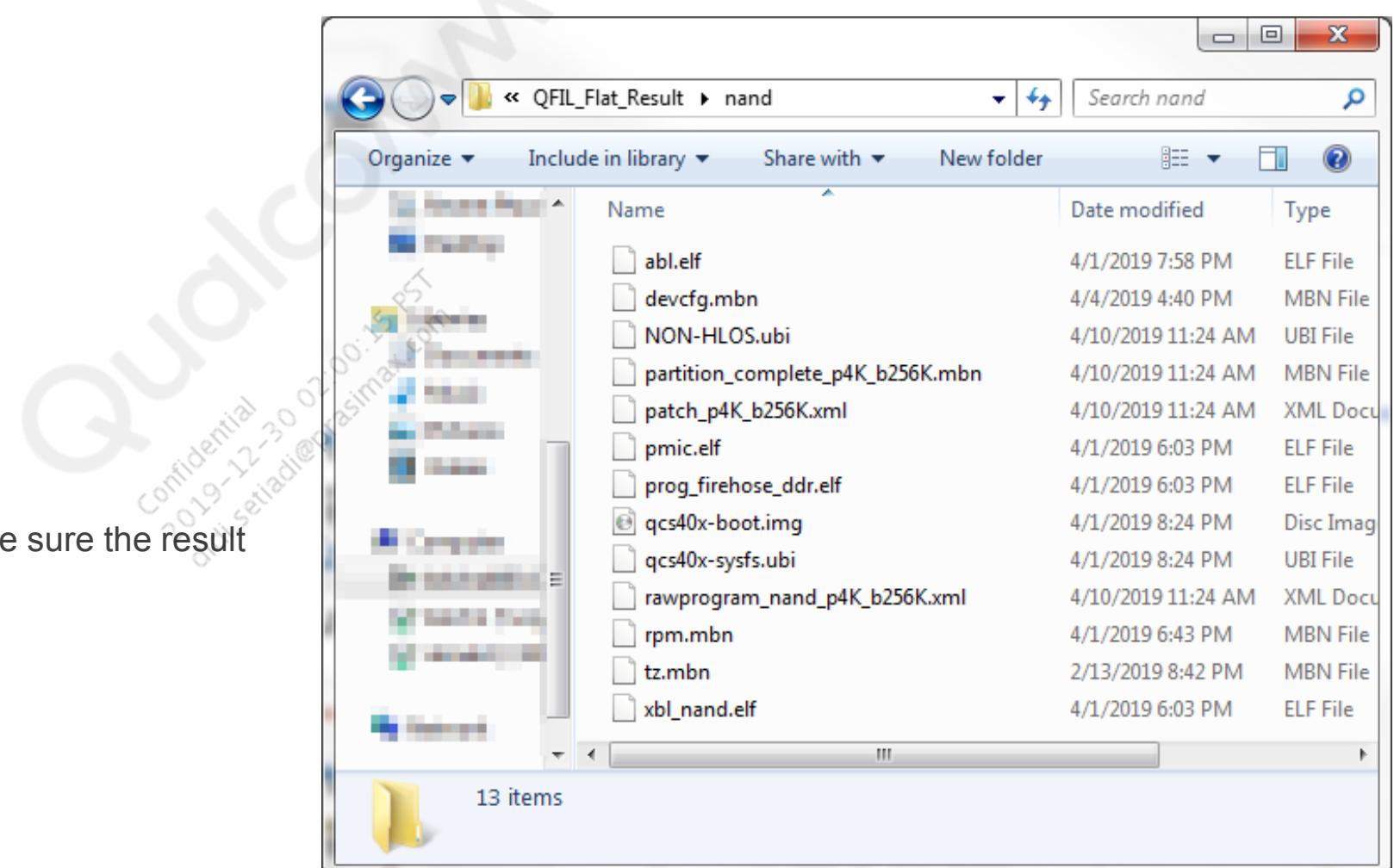
Packaging Images using QFIL

- Launch QFIL
- Click on ‘Tools > Flat Meta Build’ menu
- Click on ‘Flat Build Path > Browse...’ button
- Select <OUTPUT_FOLDER>
- Click on ‘Content XML > Browse...’ button
- Open ‘<QCS403_WS>/contents.xml’ file
- Click on ‘OK’ button to start Flat Meta Build
- Make sure “Flatten Succeed” message



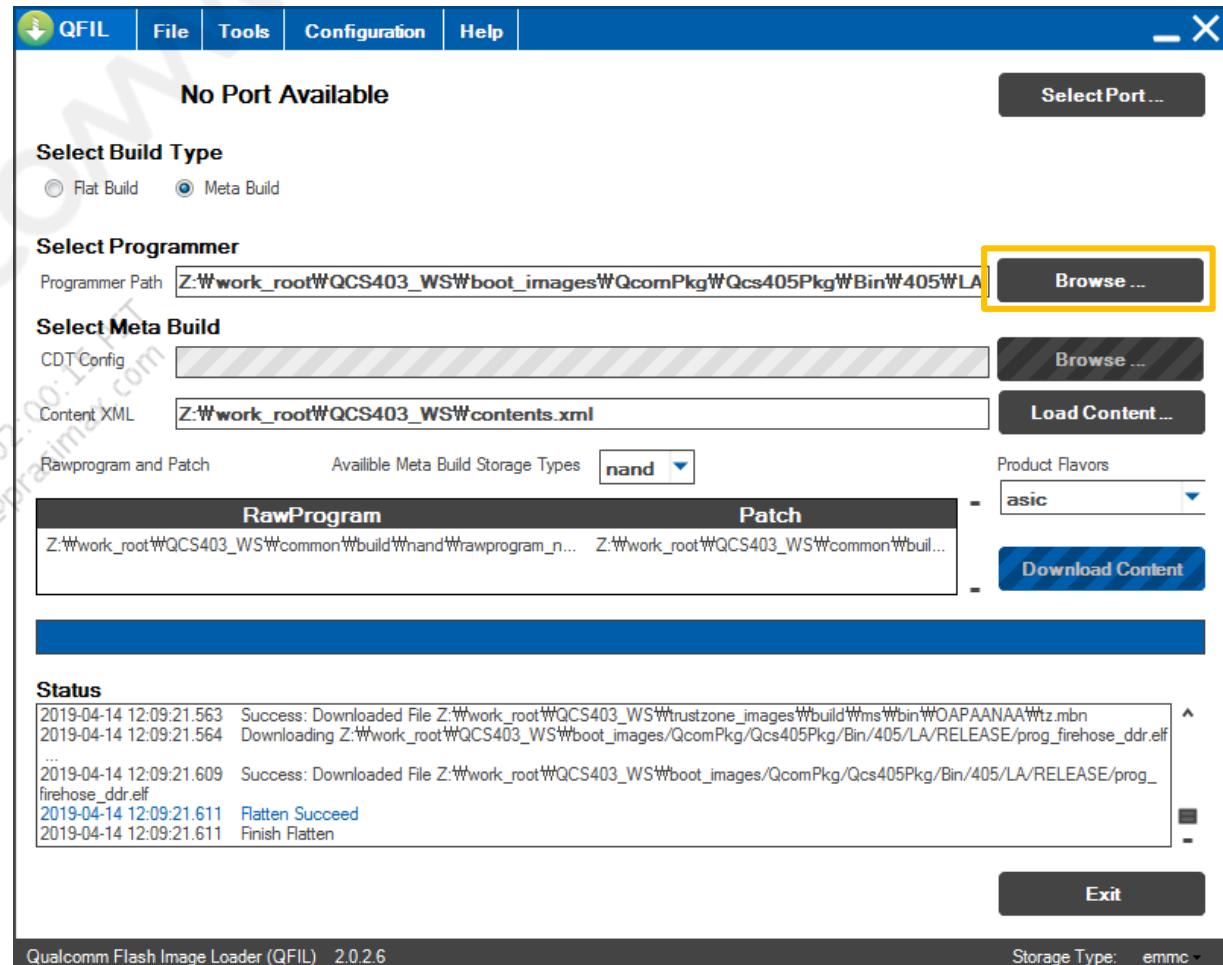
Packaging Images using QFIL

- Launch QFIL
- Click on ‘Tools > Flat Meta Build’ menu
- Click on ‘Flat Build Path > Browse...’ button
- Select <OUTPUT_FOLDER>
- Click on ‘Content XML > Browse...’ button
- Open ‘<QCS403_WS>/contents.xml’ file
- Click on ‘OK’ button to start Flat Meta Build
- Make sure “Flatten Succeed” message
- Open ‘<OUTPUT_FOLDER>/nand’ and make sure the result



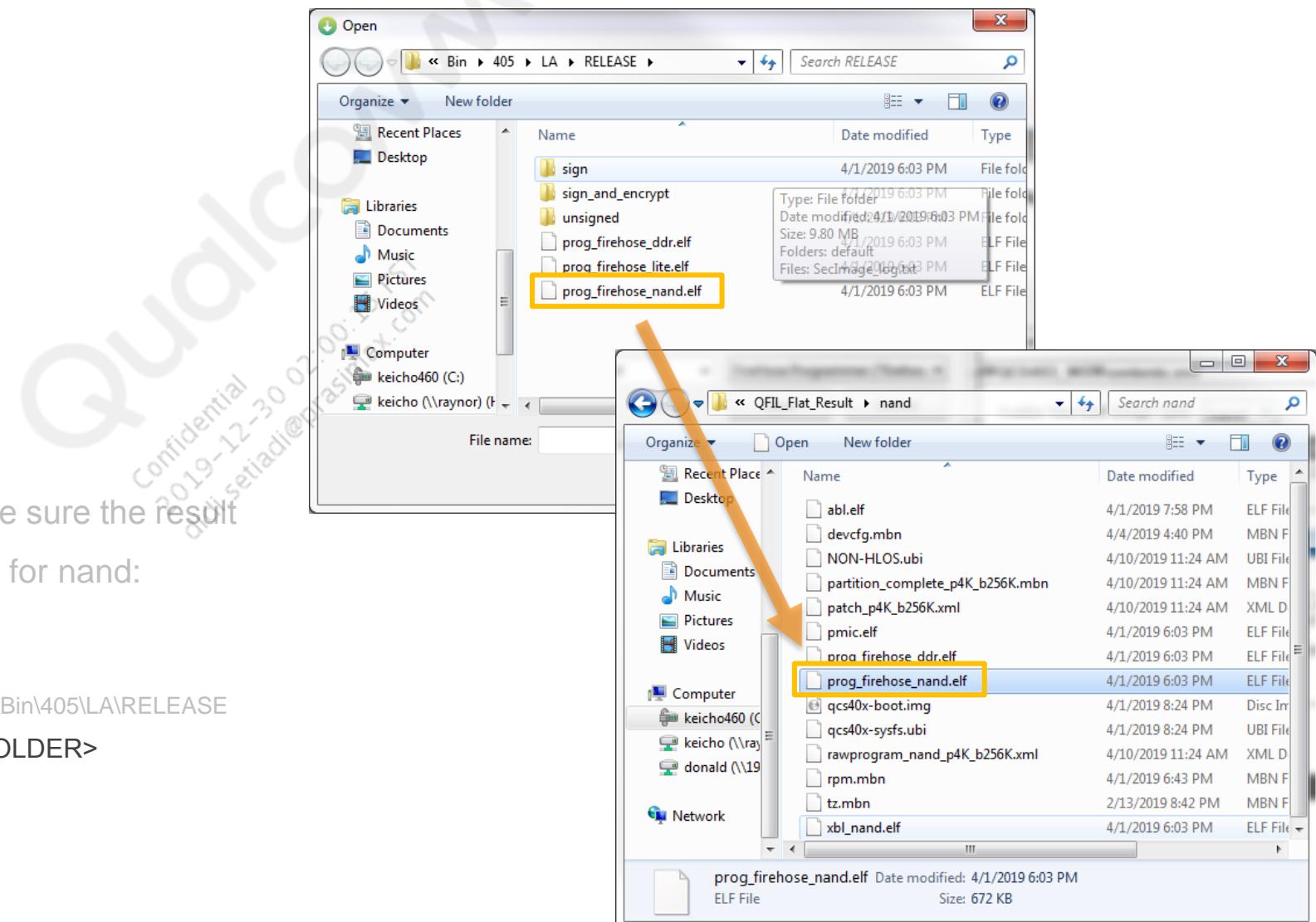
Packaging Images using QFIL

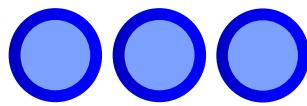
- Launch QFIL
- Click on ‘Tools > Flat Meta Build’ menu
- Click on ‘Flat Build Path > Browse...’ button
- Select <OUTPUT_FOLDER>
- Click on ‘Content XML > Browse...’ button
- Open ‘<QCS403_WS>/contents.xml’ file
- Click on ‘OK’ button to start Flat Meta Build
- Make sure “Flatten Succeed” message
- Open ‘<OUTPUT_FOLDER>/nand’ and make sure the result
- To manually copy additional Programmer file for nand:
 - Click on ‘Programmer Path > Browse...’ button
 - Then following folder will be opened:
 - <QCS403_WS>\boot_images\QcomPkg\Qcs405Pkg\Bin\405\LA\RELEASE



Packaging Images using QFIL

- Launch QFIL
- Click on ‘Tools > Flat Meta Build’ menu
- Click on ‘Flat Build Path > Browse...’ button
- Select <OUTPUT_FOLDER>
- Click on ‘Content XML > Browse...’ button
- Open ‘<QCS403_WS>/contents.xml’ file
- Click on ‘OK’ button to start Flat Meta Build
- Make sure “Flatten Succeed” message
- Open ‘<OUTPUT_FOLDER>/nand’ and make sure the result
- To manually copy additional Programmer file for nand:
 - Click on ‘Programmer Path > Browse...’ button
 - Then following folder will be opened:
 - <QCS403_WS>\boot_images\QcomPkg\Qcs405Pkg\Bin\405\LA\RELEASE
 - Copy ‘prog_firehose_nand.elf’ file to <OUTPUT_FOLDER>
- DONE !





Section 7.3.2

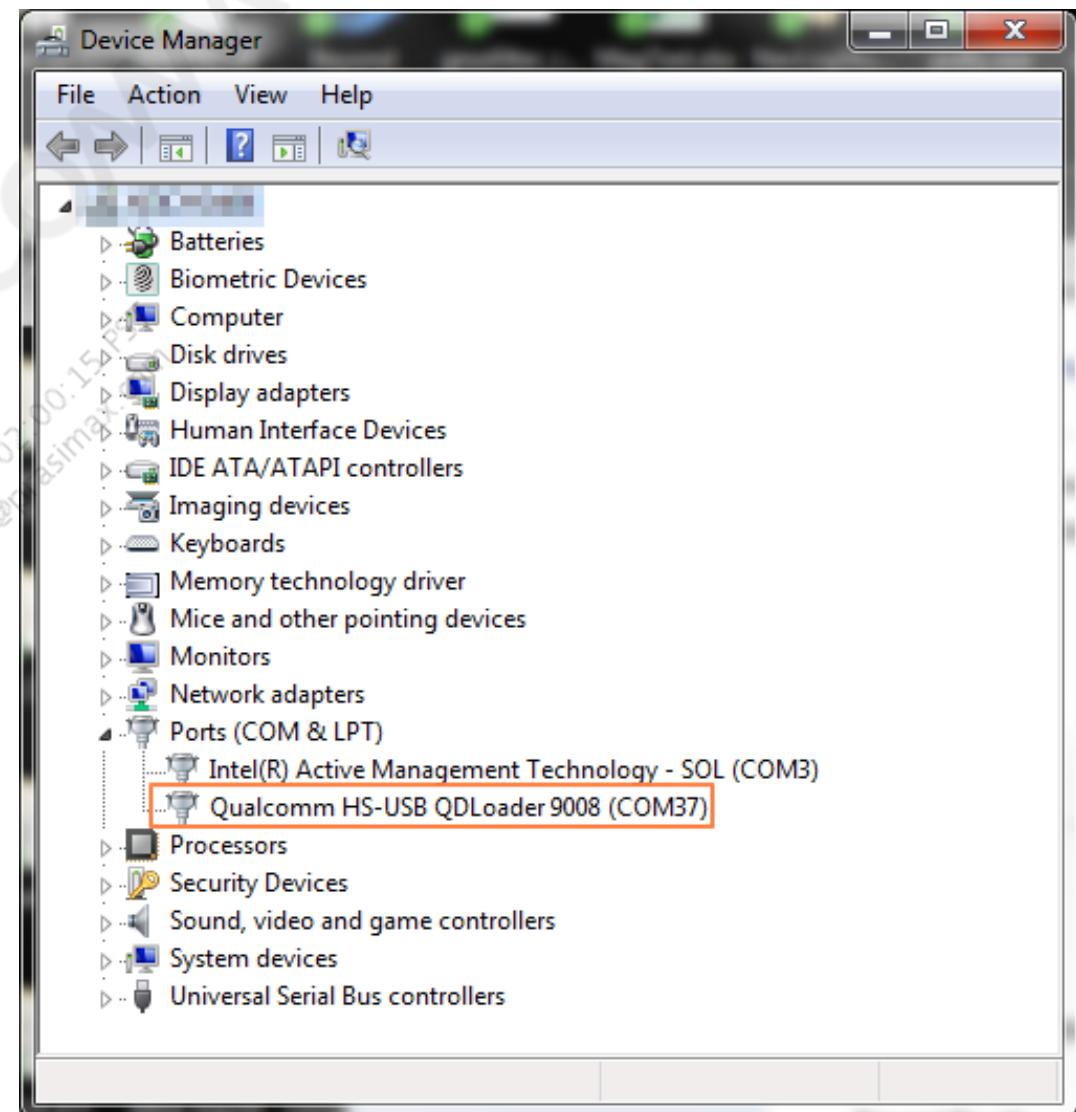
Writing Flat Build Images using QFIL

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Writing Flat Build Images using QFIL

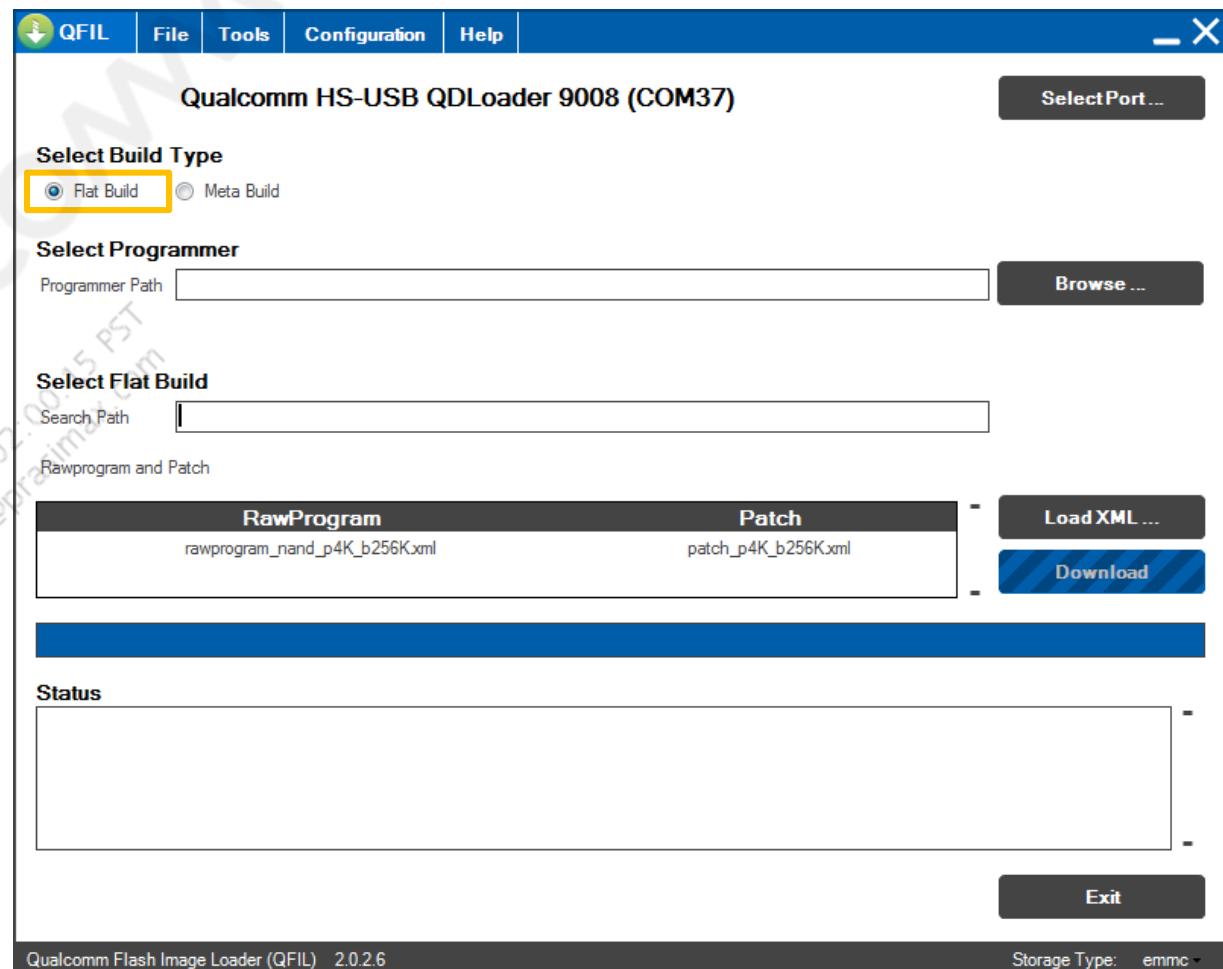
- To write images using QFIL, the device must be configured to EDL mode
- Refer to:
 - [Device Boot Configuration for SSRD - EDL mode](#)

Confidential
2019-12-30 01:00:15
didi.setiadi@rasimahaxi



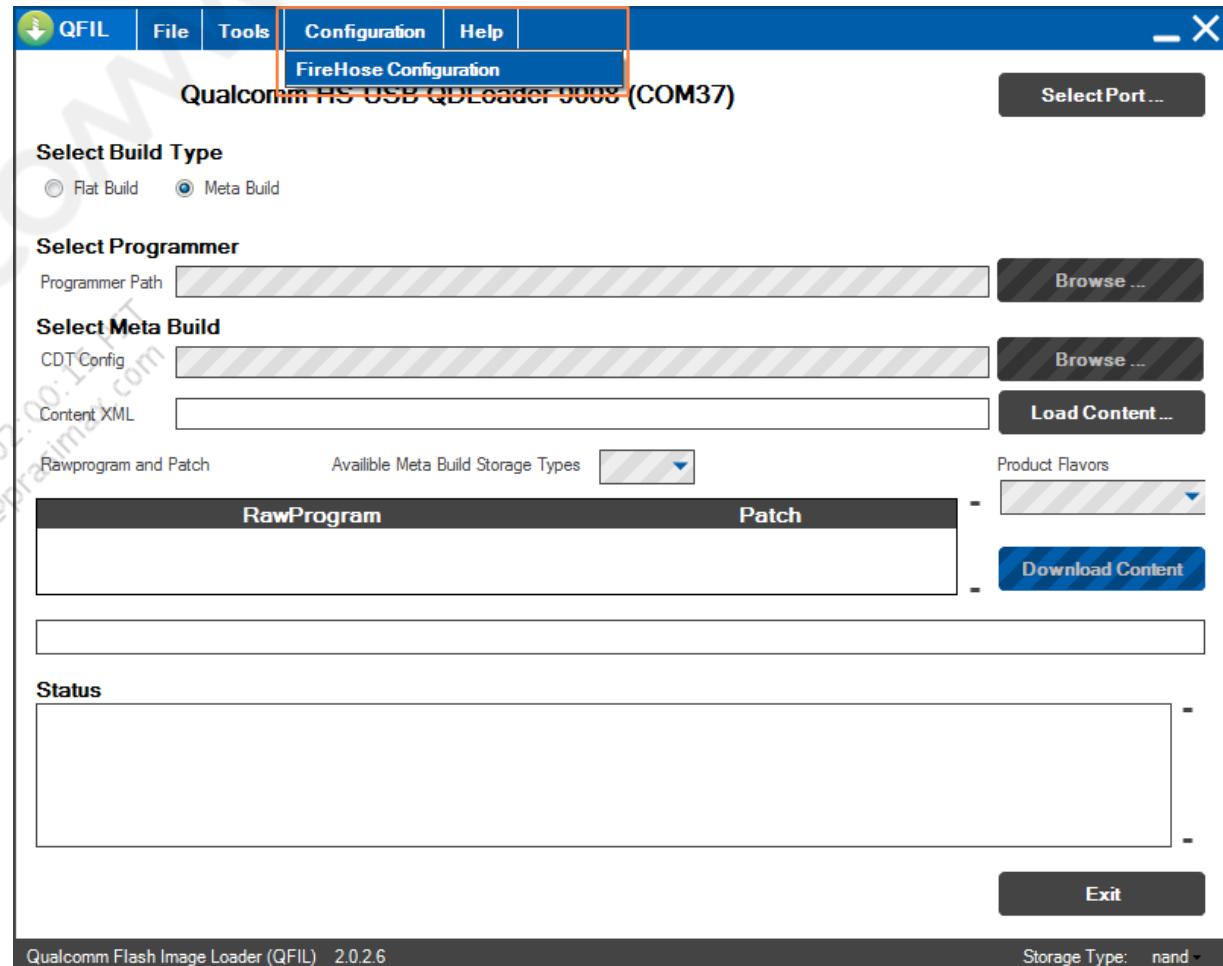
Writing Flat Build Images using QFIL

- Launch QFIL and Select Build Type to 'Flat Build'



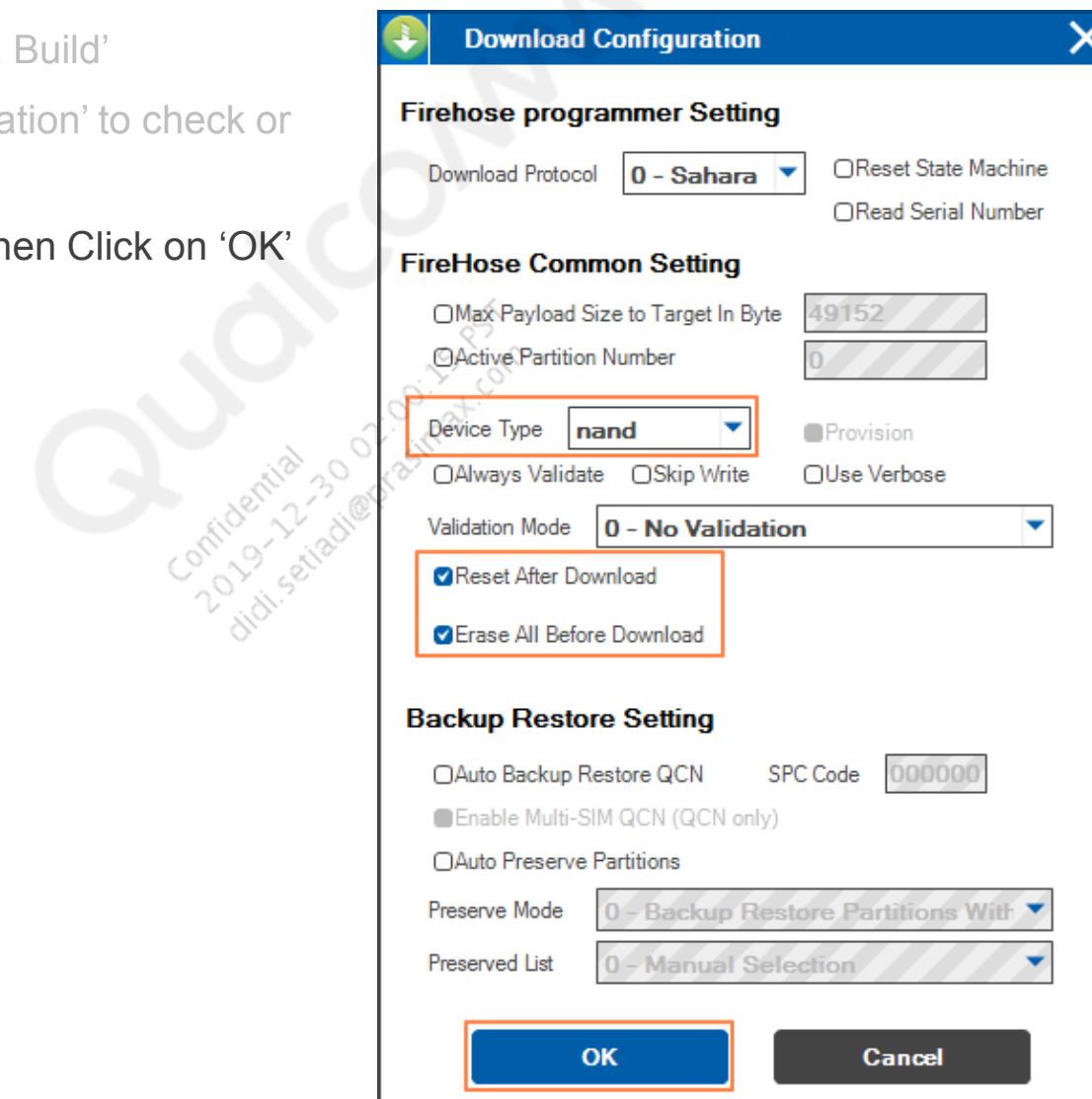
Writing Flat Build Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)



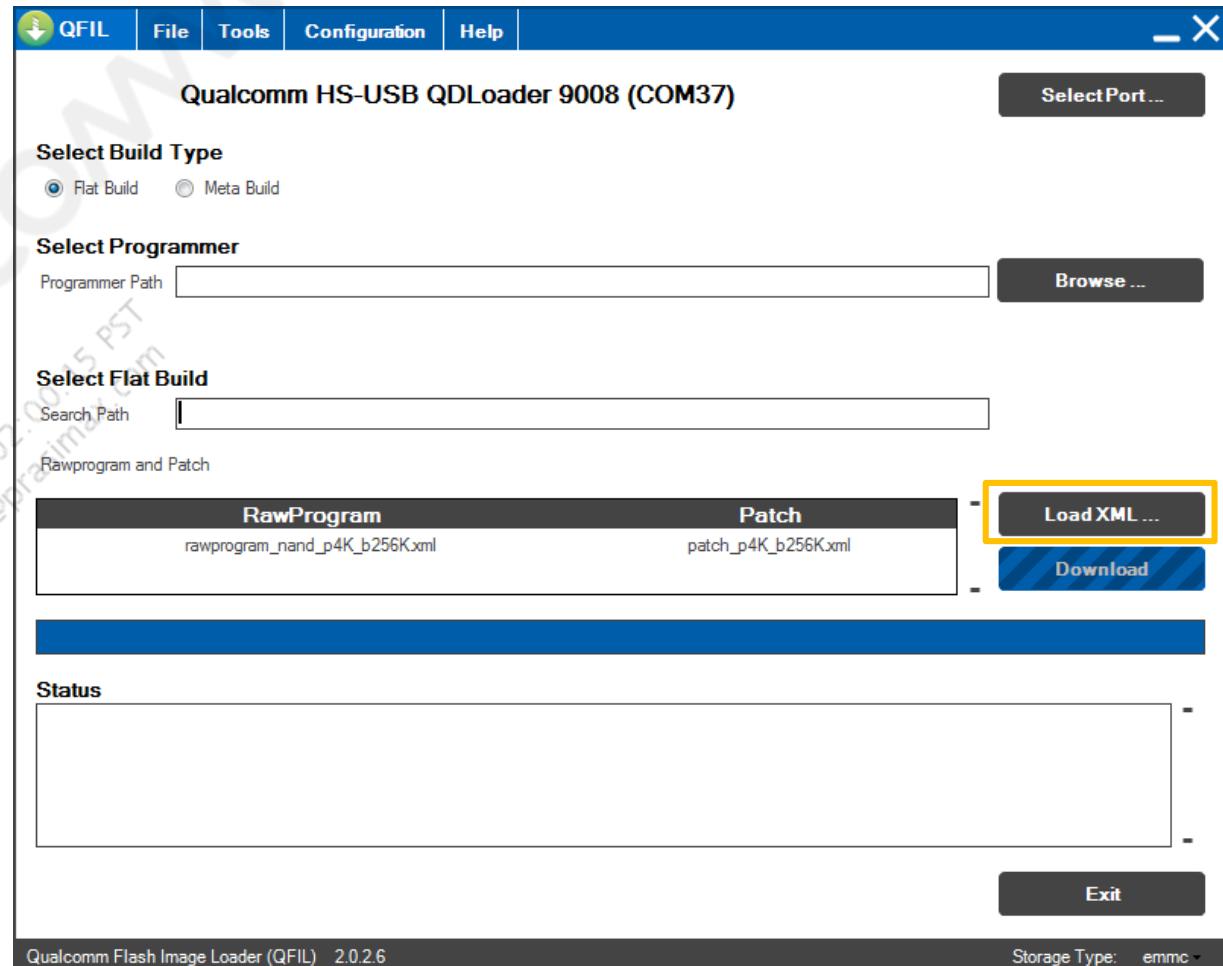
Writing Flat Build Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON



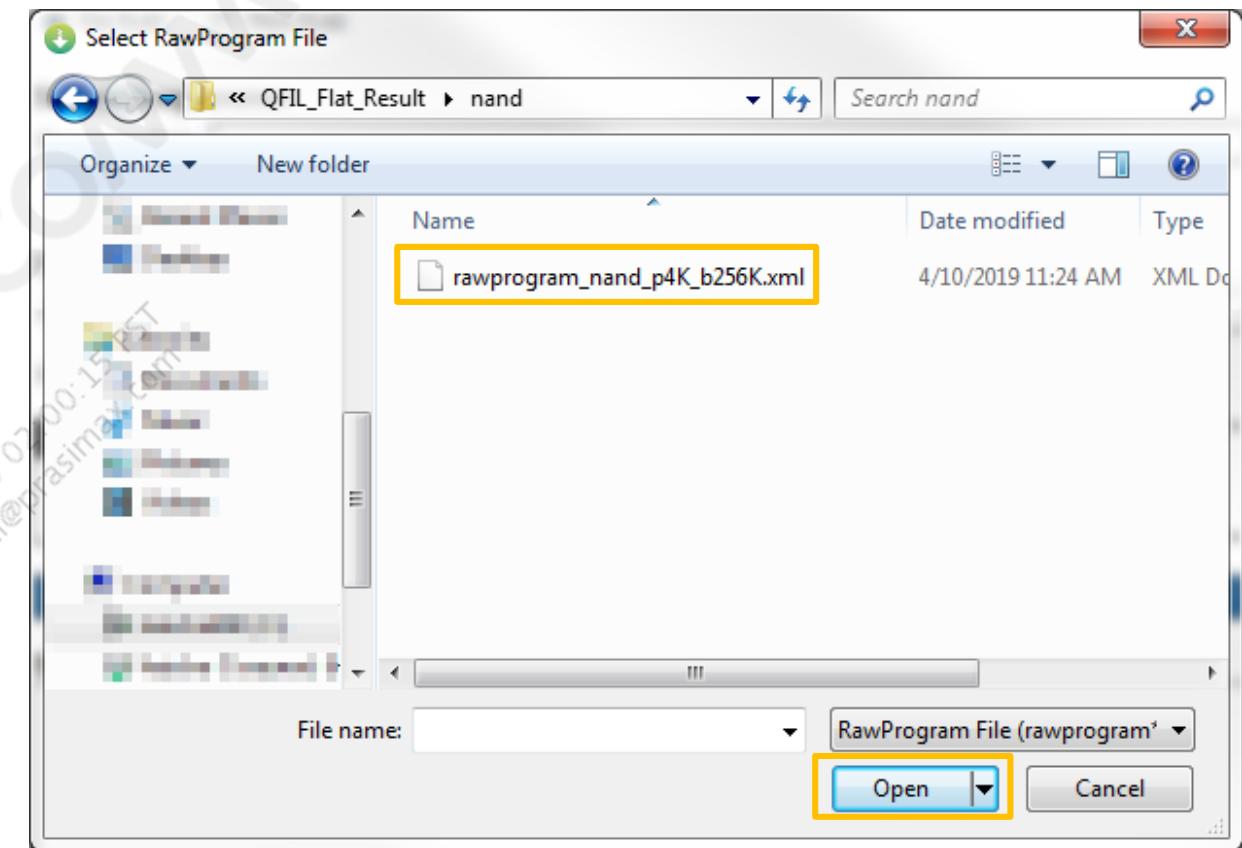
Writing Flat Build Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on 'Load XML...' button



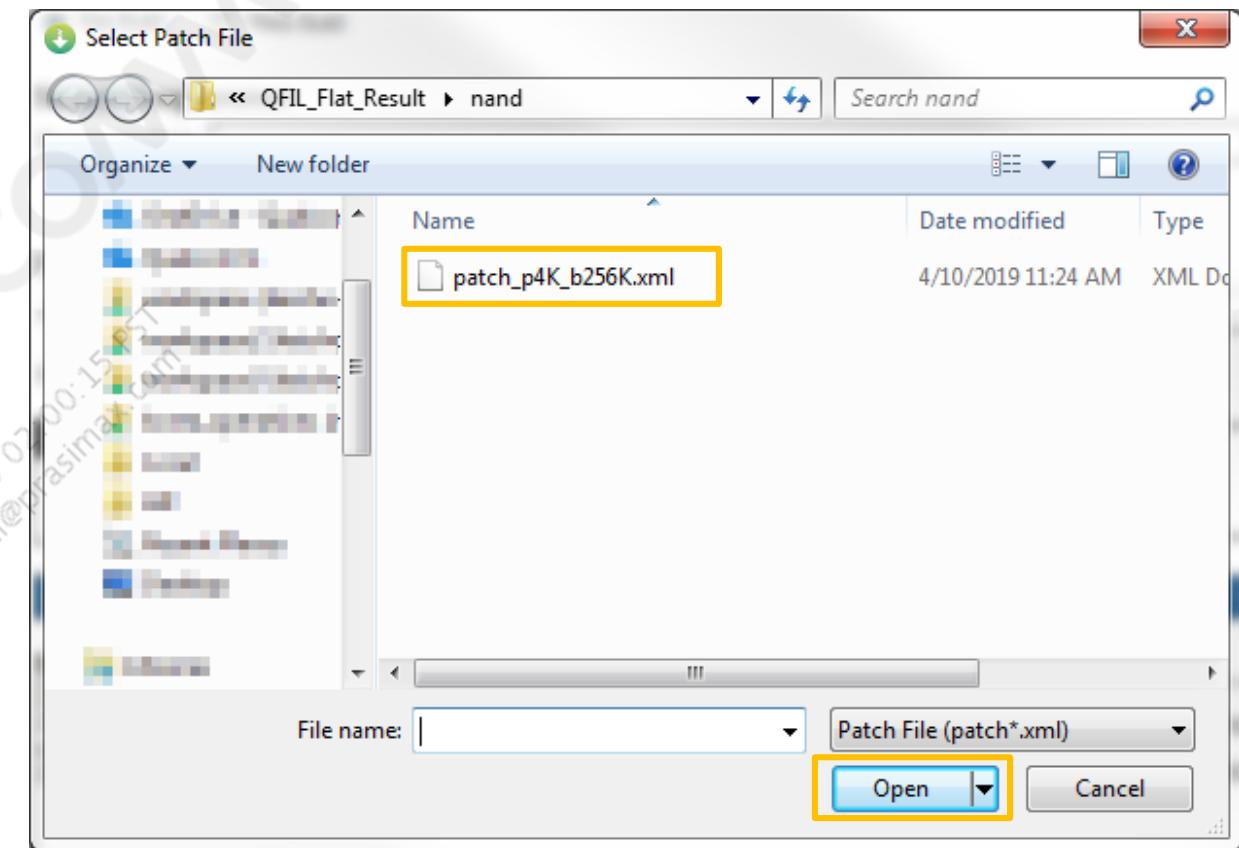
Writing Flat Build Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on 'Load XML...' button
- Open 'rawprogram_nand' xml file



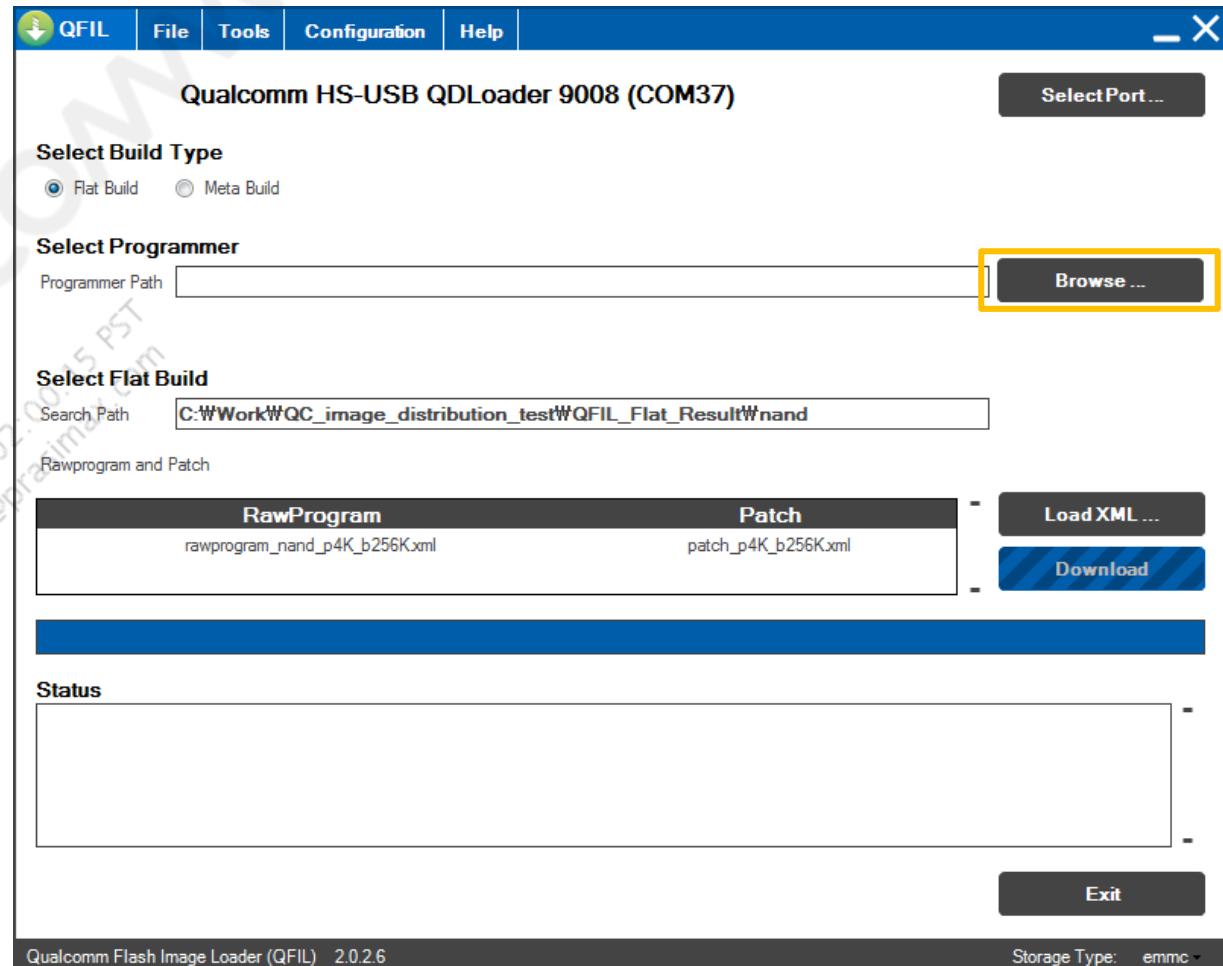
Writing Flat Build Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on 'Load XML...' button
- Open 'rawprogram_nand' xml file
- Open 'patch' xml file



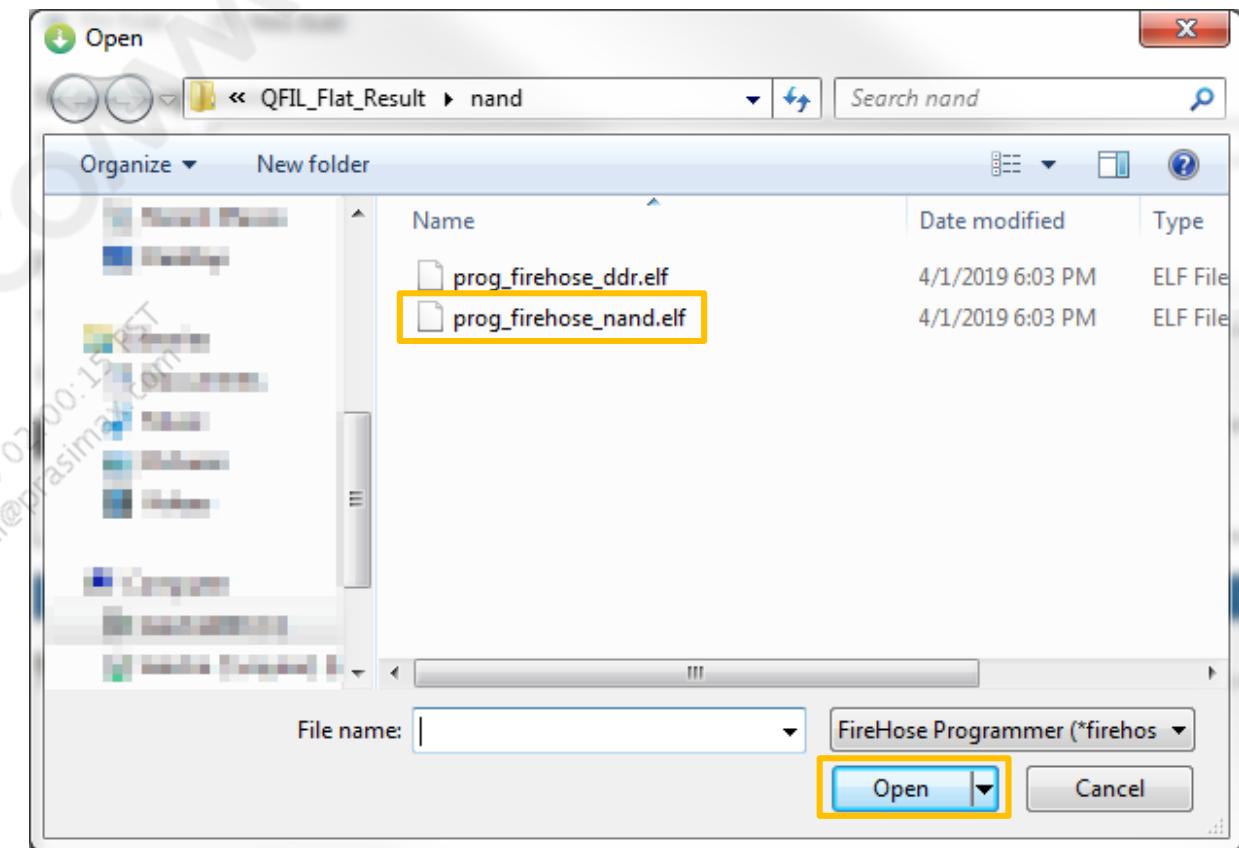
Writing Flat Build Images using QFIL

- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on 'Load XML...' button
- Open 'rawprogram_nand' xml file
- Open 'patch' xml file
- Click on 'Programmer Path > Browse...' button



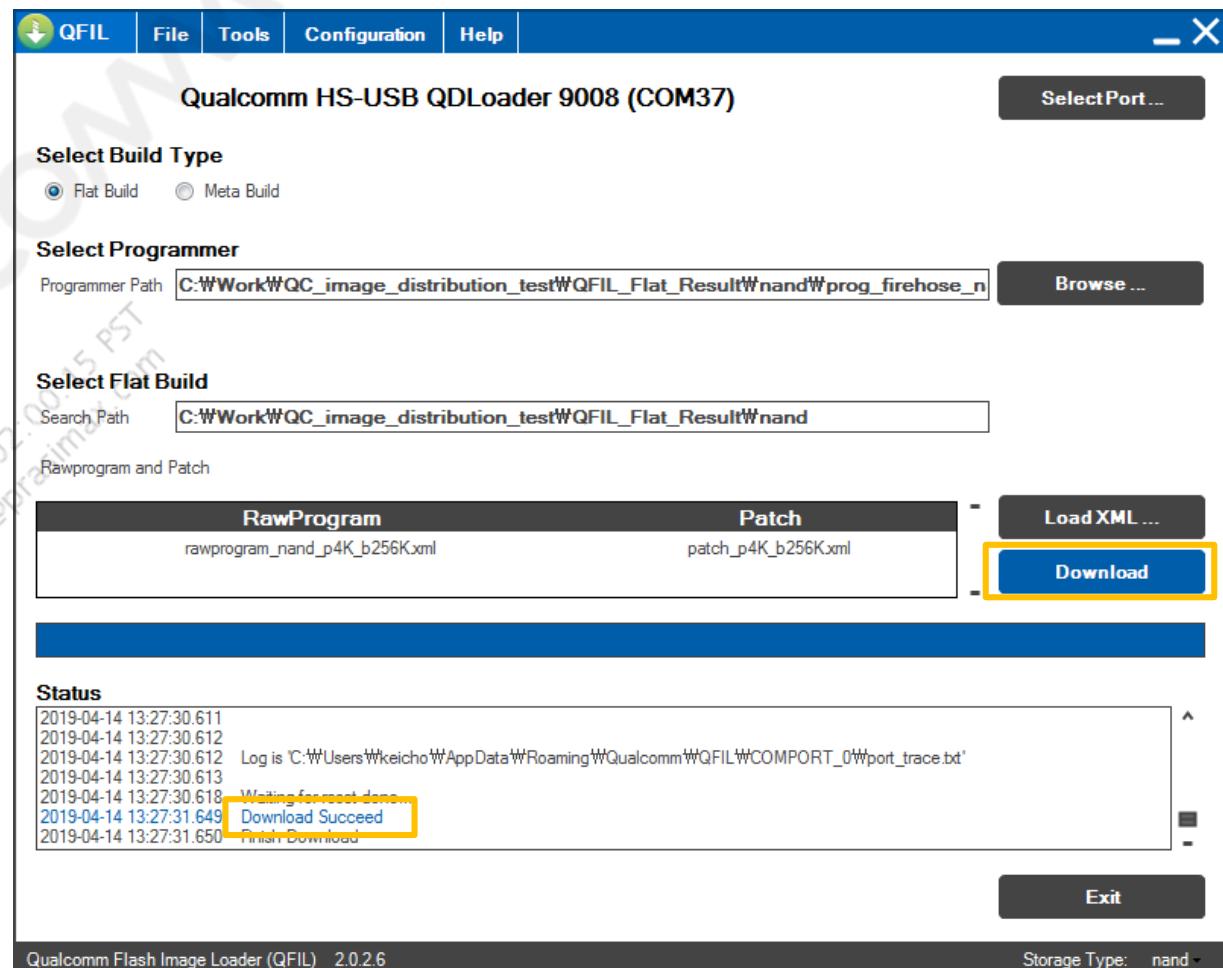
Writing Flat Build Images using QFIL

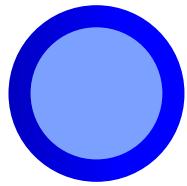
- Launch QFIL and Select Build Type to 'Meta Build'
- Click on 'Configuration > FireHose Configuration' to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on 'OK'
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on 'Load XML...' button
- Open 'rawprogram_nand' xml file
- Open 'patch' xml file
- Click on 'Programmer Path > Browse...' button
- Open 'prog_firehose_nand.elf' file



Writing Flat Build Images using QFIL

- Launch QFIL and Select Build Type to ‘Meta Build’
- Click on ‘Configuration > FireHose Configuration’ to check or modify settings for QCS403 (NAND)
- Set Download Configuration as below and then Click on ‘OK’
 - Device Type: nand
 - Reset After Download: Check ON
 - Erase All Before Download: Check ON
- Click on ‘Load XML...’ button
- Open ‘rawprogram_nand’ xml file
- Open ‘patch’ xml file
- Click on ‘Programmer Path > Browse...’ button
- Open ‘prog_firehose_nand.elf’ file
- Click on ‘Download’ button
- Make sure “Download Succeed” message
- DONE !





Section 8

Basic Operation Test

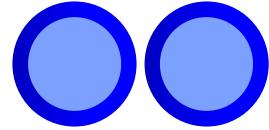
Qualcomm
Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

8.1 Bring up DUT	192
8.2 Writing Device Configuration	202
8.3 Setting up Haven License	214
8.4 Onboarding Wi-Fi	220
8.5 Onboarding AVS	227
8.6 Local ASR + AVS Test	235
8.7 Bluetooth Audio Streaming Test	241

Basic Operation Test

- In this chapter, following process will be explained for successful AVS and Bluetooth testing setup
 - Bring up DUT
 - Writing Device Configuration
 - Setting up Haven License
 - Onboarding Wi-Fi
 - Onboarding AVS
 - Bluetooth Audio Streaming Test

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com



Section 8.1

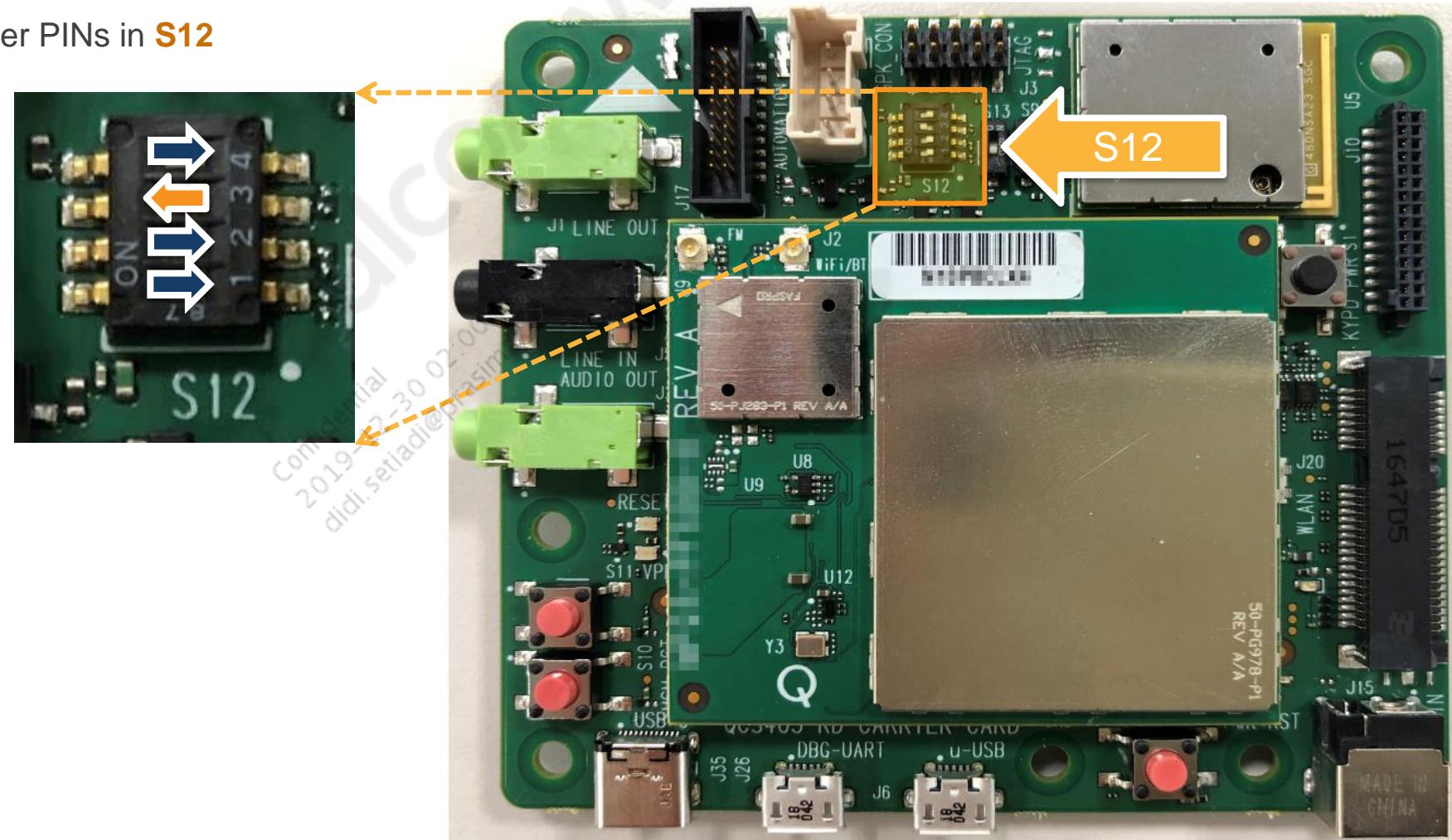
Bring up DUT

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

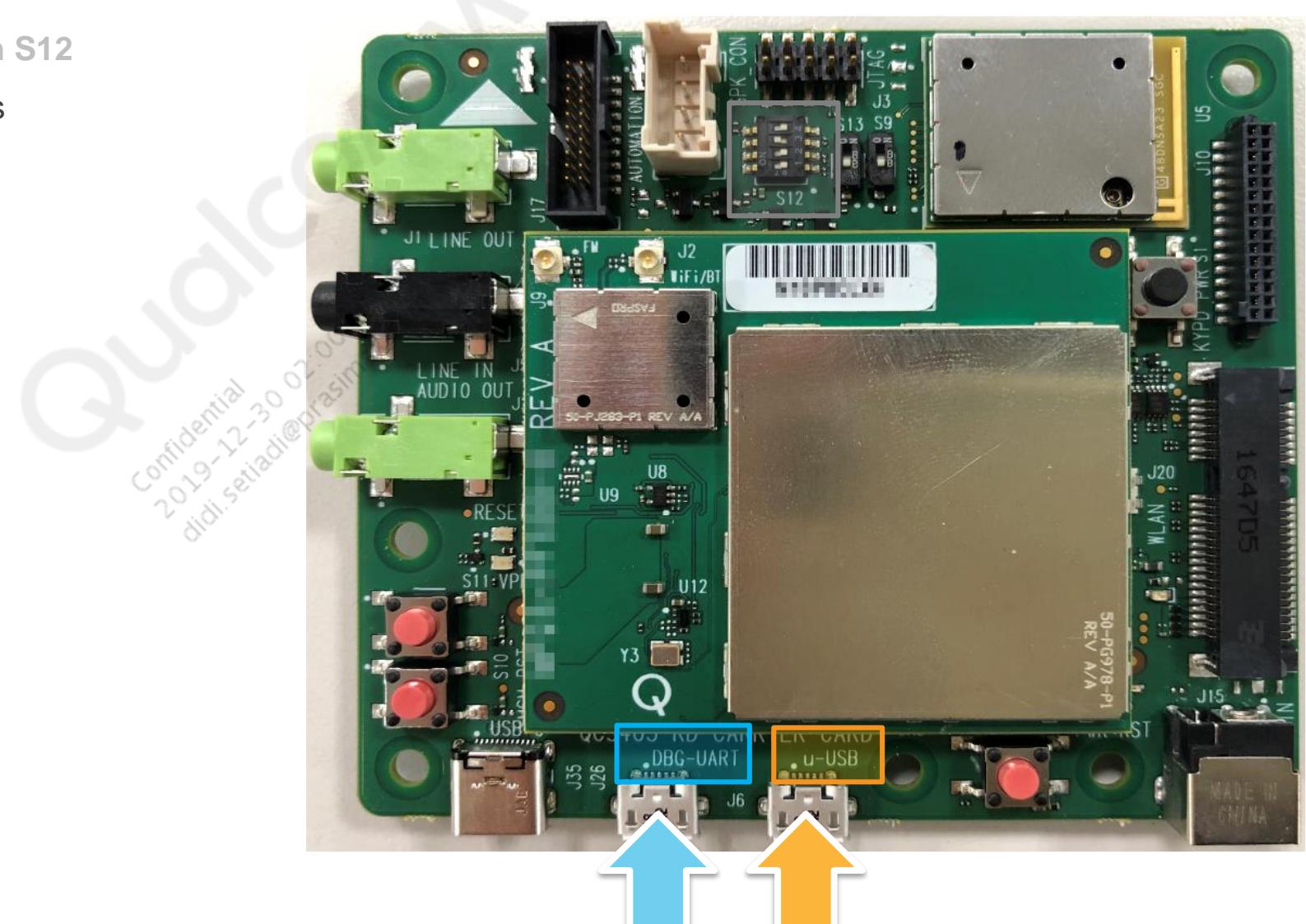
Bring up DUT

1. Turn ON PIN#3 and Turn OFF other PINs in **S12**



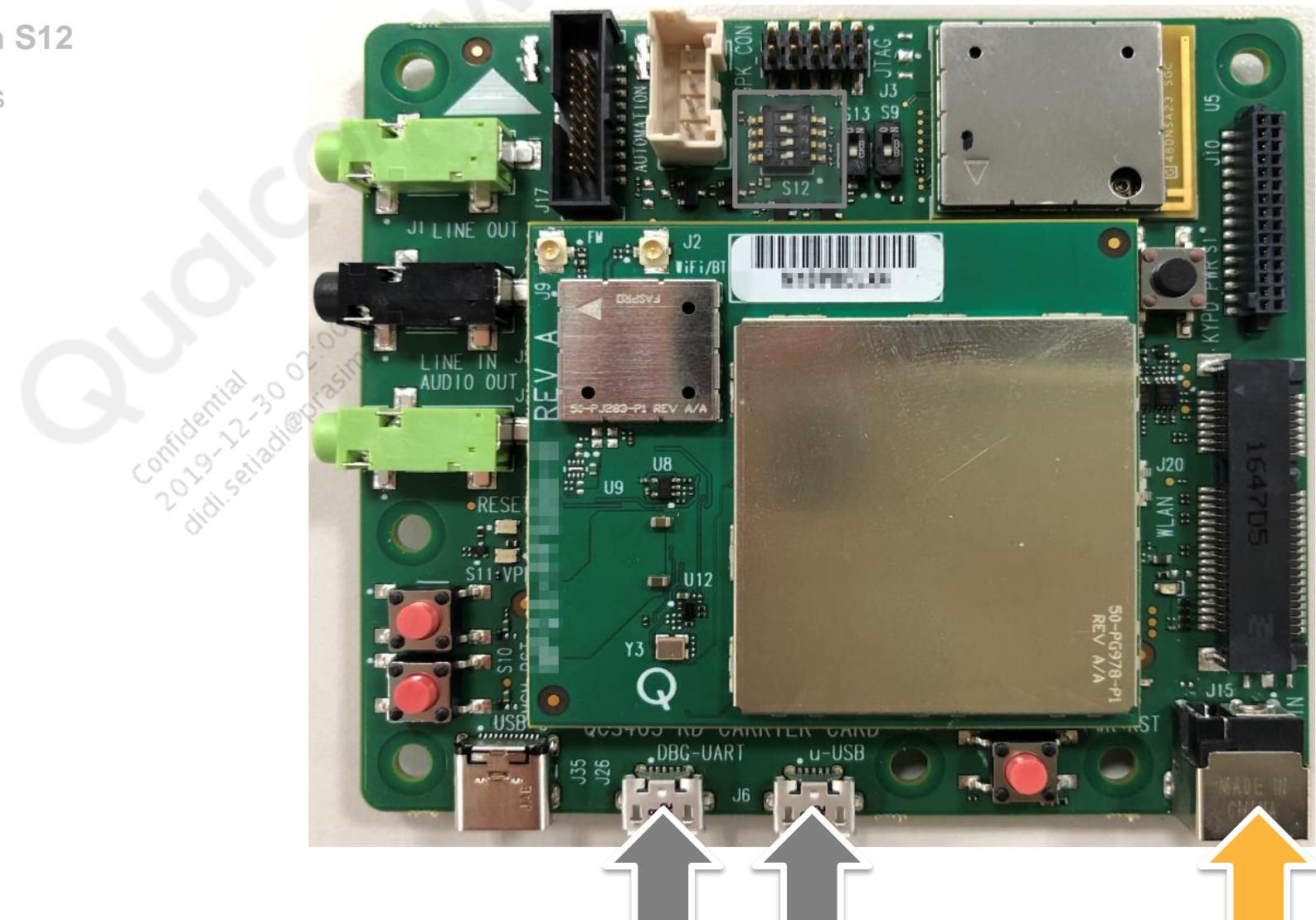
Bring up DUT

1. Turn ON PIN#3 and Turn OFF other PINs in S12
2. Connect device to host PC with USB cables
 - u-USB for adb interface
 - DBG-UART for debug log interface



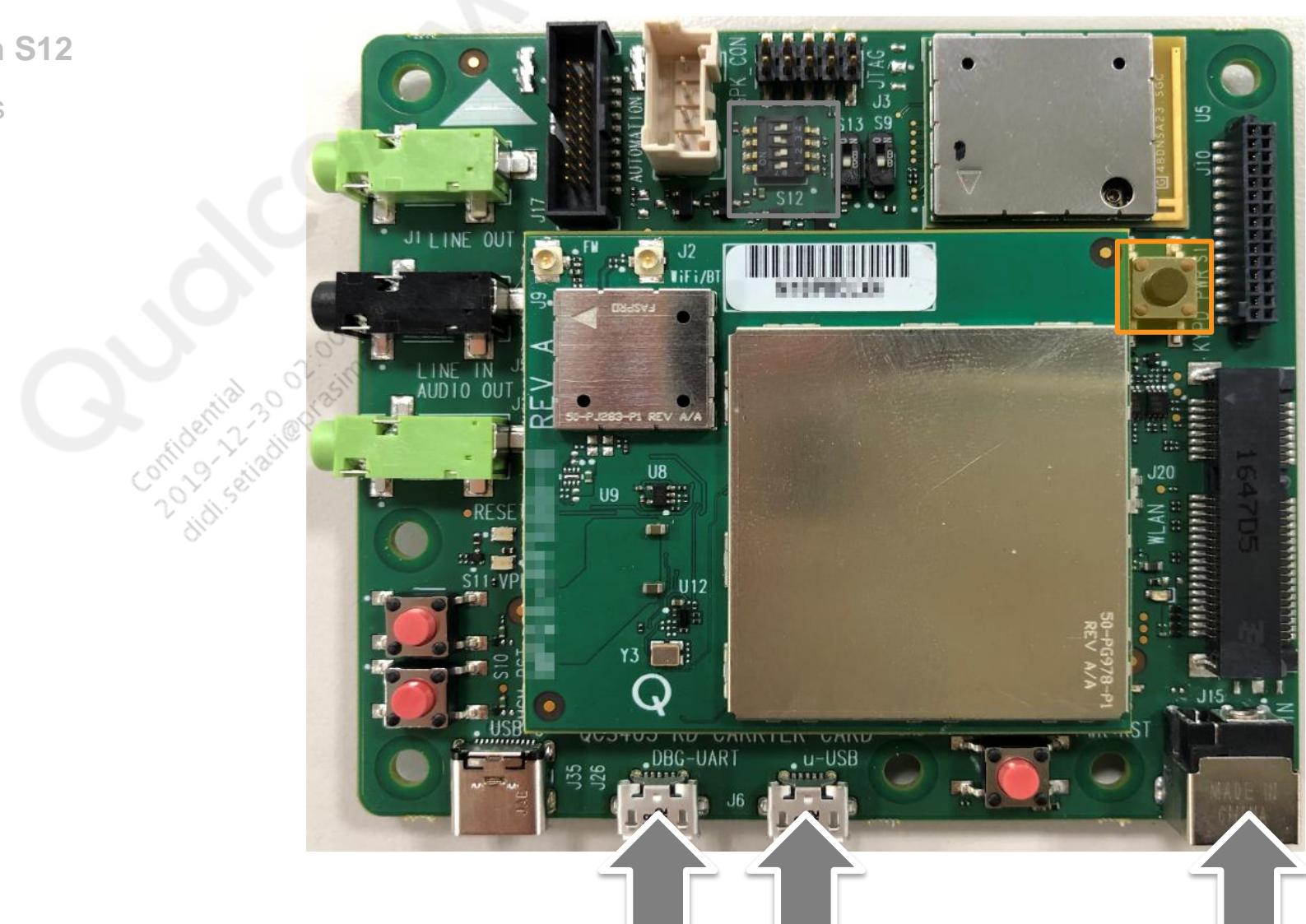
Bring up DUT

1. Turn ON PIN#3 and Turn OFF other PINs in S12
2. Connect device to host PC with USB cables
 - u-USB for adb interface
 - DBG-UART for debug log interface
3. Connect the Main Power cable



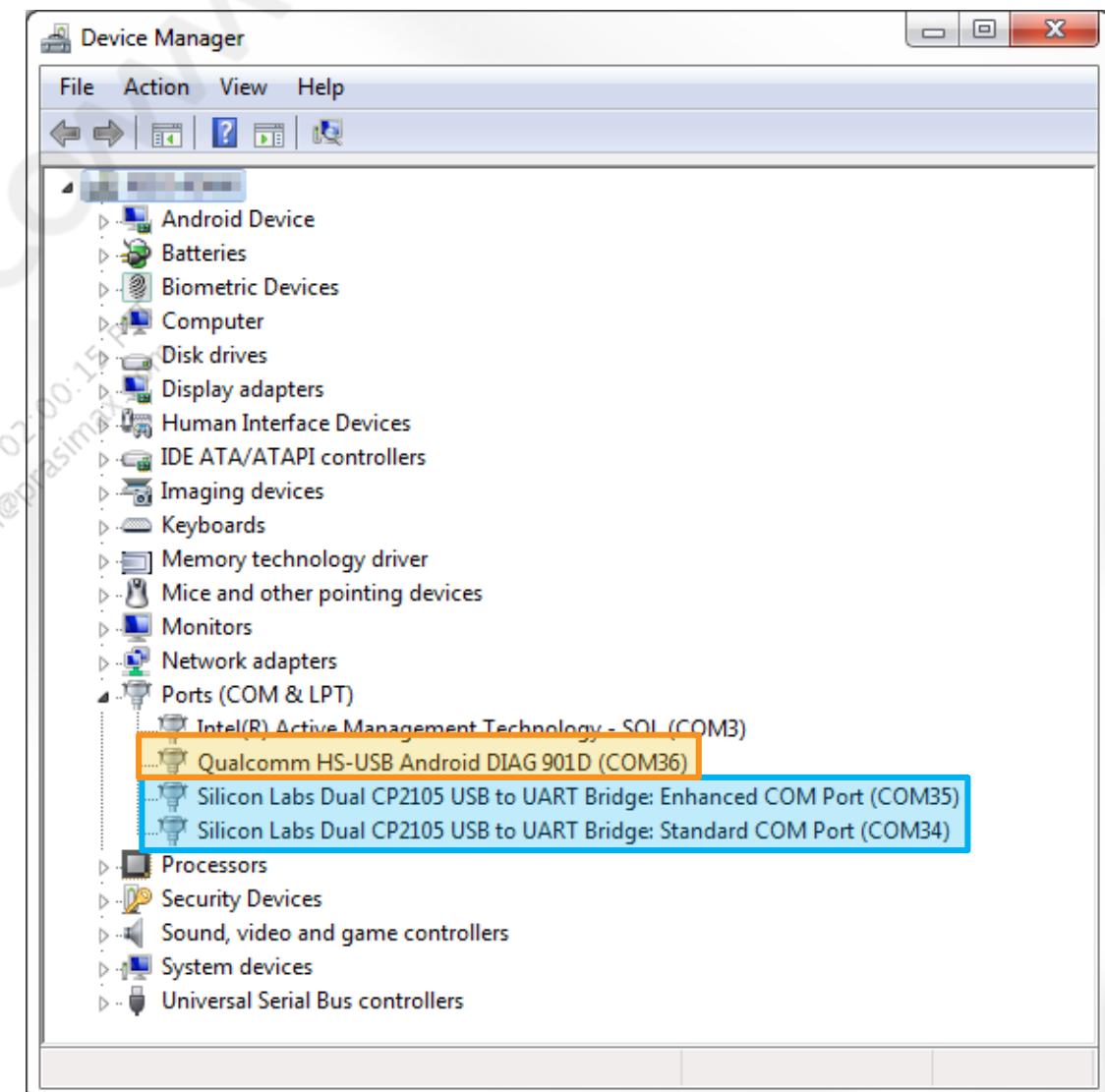
Bring up DUT

1. Turn ON PIN#3 and Turn OFF other PINs in S12
2. Connect device to host PC with USB cables
 - u-USB for adb interface
 - DBG-UART for debug log interface
3. Connect the Main Power cable
4. Push the KYPD_PWR button



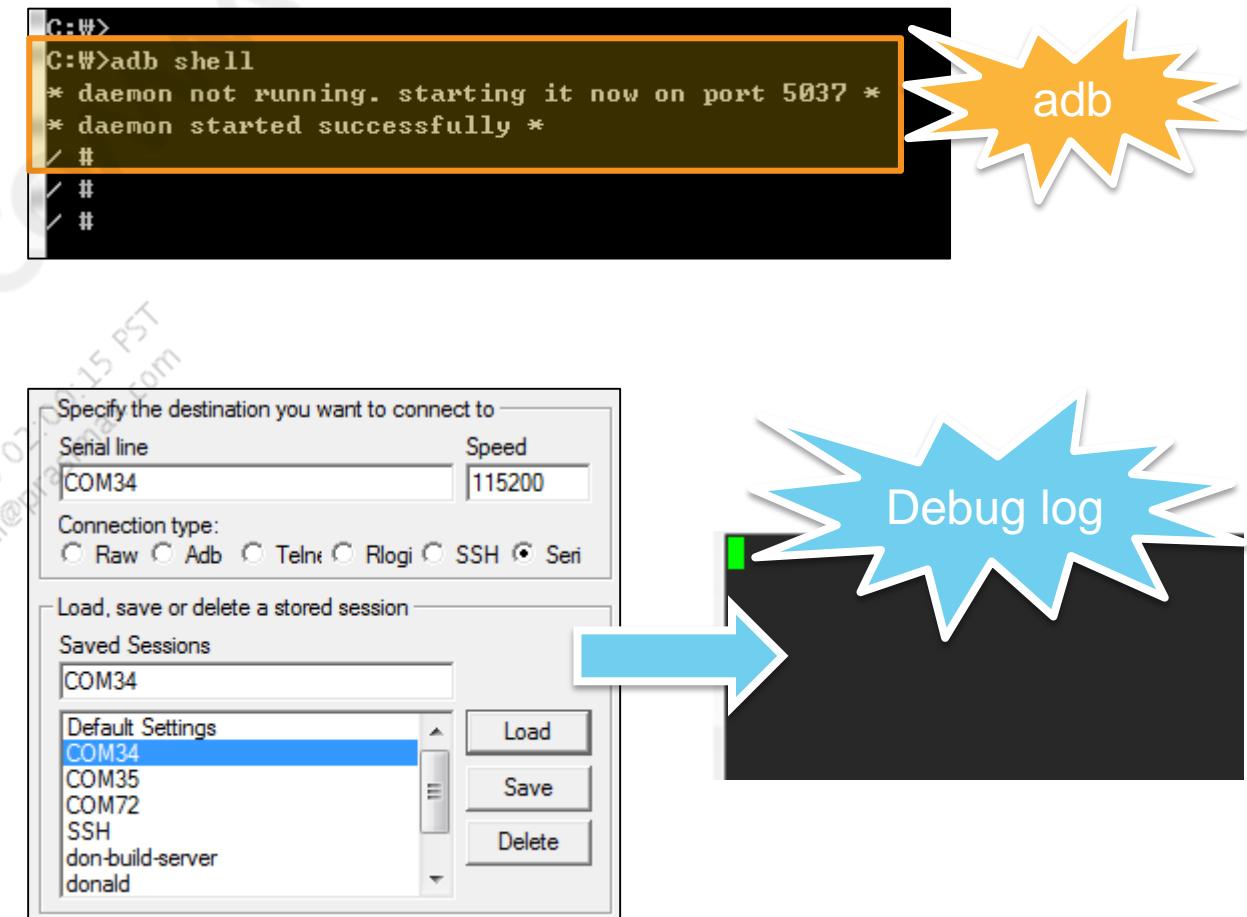
Bring up DUT

1. Turn ON PIN#3 and Turn OFF other PINs in S12
2. Connect device to host PC with USB cables
 - u-USB for adb interface
 - DBG-UART for debug log interface
3. Connect the Main Power cable
4. Push the KYPD_PWR button
5. Check the USB connection on host PC > Device Manager
 - ‘Qualcomm HS-USB Android DIAG...’ for **adb** interface
 - ‘**Silicon Labs...**’ for **debug log** interface
 - Normally, among the two ports, the **front number** port is selected as debug port



Bring up DUT

1. Turn ON PIN#3 and Turn OFF other PINs in S12
2. Connect device to host PC with USB cables
 - u-USB for adb interface
 - DBG-UART for debug log interface
3. Connect the Main Power cable
4. Push the KYPD_PWR button
5. Check the USB connection on host PC > Device Manager
 - ‘Qualcomm HS-USB Android DIAG...’ for adb interface
 - ‘Silicon Labs...’ for debug log interface
 - Normally, among the two ports, the front number port is selected as debug port
6. Open terminal windows for **adb** and **debug log**



Bring up DUT

1. Turn ON PIN#3 and Turn OFF other PINs in S12
2. Connect device to host PC with USB cables
 - u-USB for adb interface
 - DBG-UART for debug log interface
3. Connect the Main Power cable
4. Push the KYPD_PWR button
5. Check the USB connection on host PC > Device Manager
 - ‘Qualcomm HS-USB Android DIAG...’ for adb interface
 - ‘Silicon Labs...’ for debug log interface
 - Normally, among the two ports, the front number port is selected as debug port
6. Open terminal windows for adb and debug log
7. Reboot device and check the device booting log on debug log terminal



The screenshot shows a terminal window with the command `C:\W> adb reboot` entered. A large orange arrow points from the terminal window down to a separate window displaying the device booting log. The log output is as follows:

```
e -5 = ret
[ 58.432826] QCS405 ULL_NOIRQ: ASoC: prepare FE QCS405 ULL_NOIRQ failed
[ 58.525460] snd_pcm_hw_constraint_integer failed
[ 58.525497] msm_pcm_open: P buffer bytes minmax constraint ret 1
[ 58.544472] msm-pcm-dsp-noirq soc:qcom,msm-pcm-dsp-noirq: ASoC: platform prepare error: -5
[ 58.544518] soc_pcm_prepare: Issue stop stream for codec_dai due to op failure
e -5 = ret
[ 58.558387] QCS405 ULL_NOIRQ: ASoC: prepare FE QCS405 ULL_NOIRQ failed
[ 58.697582] snd_pcm_hw_constraint_integer failed
[ 58.697619] msm_pcm_open: P buffer bytes minmax constraint ret 1
[ 58.715164] msm-pcm-dsp-noirq soc:qcom,msm-pcm-dsp-noirq: ASoC: platform prepare error: -5
[ 58.715209] soc_pcm_prepare: Issue stop stream for codec_dai due to op failure
e -5 = ret
[ 58.719264] ep92 3-0064: ep92_sysfs_wta_power: device error
[ 58.735532] QCS405 ULL_NOIRQ: ASoC: prepare FE QCS405 ULL_NOIRQ failed
[ 79.452176] msm_rpm_flush_requests: Error: more than 24 requests are buffered
[ 88.918076] sh (1658) used greatest stack depth: 3872 bytes left
[ 89.153364] wpa_supplicant (1462) used greatest stack depth: 3616 bytes left
[ 89.254396] wlan: [1675:D:HDD] 00000000: 7f 05 00 00 0a 02 01 dd 08 8c fd f0
01 01 02 01
[ 89.254444] wlan: [1675:D:HDD] 00000010: 00
```

A yellow starburst graphic labeled "adb" is positioned to the right of the terminal window, and a blue starburst graphic labeled "Debug log" is positioned to the right of the log window.

Bring up DUT

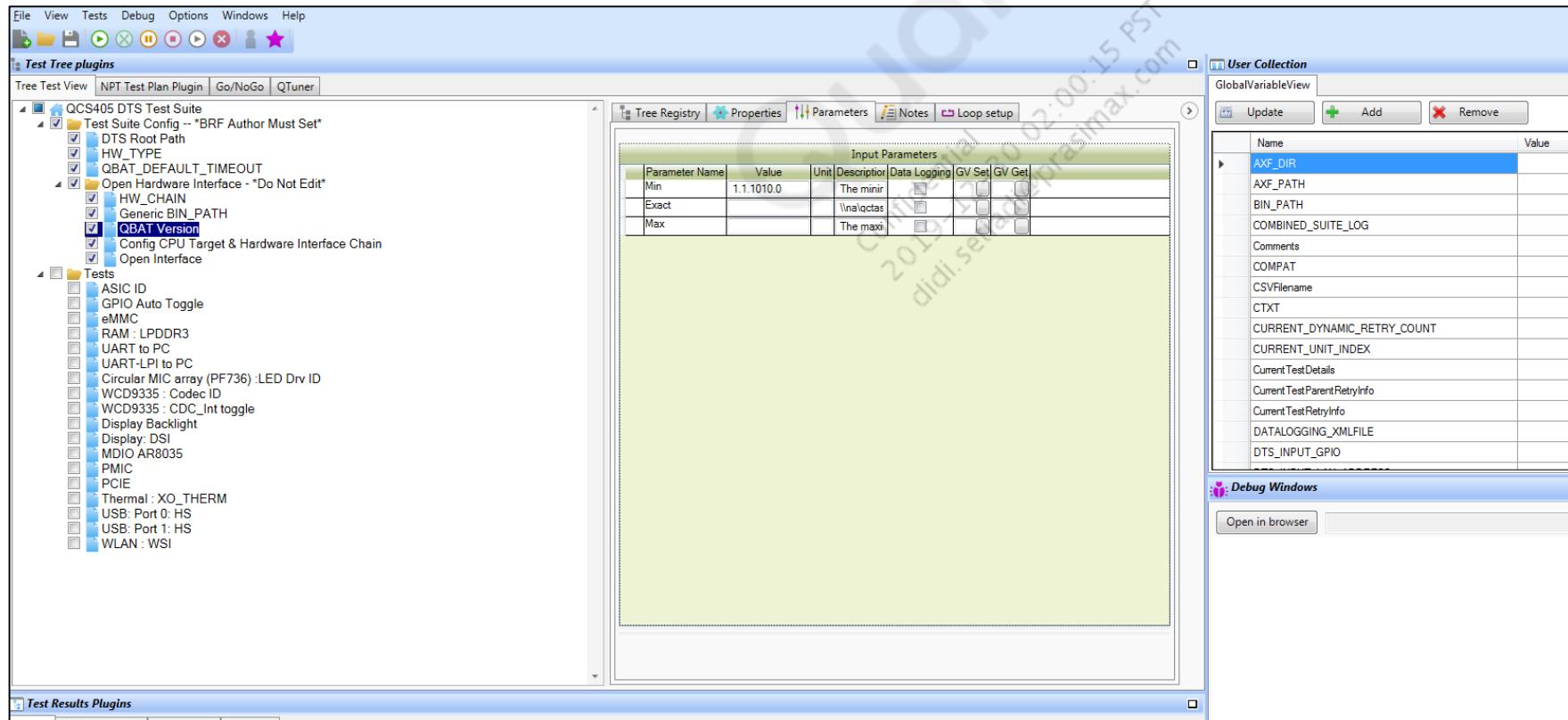
1. Turn ON PIN#3 and Turn OFF other PINs in S12
 2. Connect device to host PC with USB cables
 - u-USB for adb interface
 - DBG-UART for debug log interface
 3. Connect the Main Power cable
 4. Push the KYPD_PWR button
 5. Check the USB connection on host PC > Device Manager
 - ‘Qualcomm HS-USB Android DIAG...’ for adb interface
 - ‘Silicon Labs...’ for debug log interface
 - Normally, among the two ports, the front number port is decided as debug port
 6. Open terminal windows for adb and debug log
 7. Reboot device and check the device booting log on debug log terminal
 - In **debug apps image**, press **ENTER** key on the debug log window to login. Default password is ‘**oelinux123**’
 - In **perf apps image**, **Login Function** and **apps debug log** are **not supported**

Press ENTER

Debug log

Bring up DUT – Trouble shooting

- **DO NOT MIX USE** HLOS image with **different version** of non-HLOS image
- Otherwise, unexpected problems may occur even in the middle of operation
- If a problem occurs during the booting process, QBAT can be used to check the basic device status

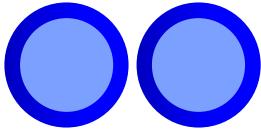


Bring up DUT – Trouble shooting

- If you have any problem during operation, check the firmware version information from '[/firmware/verinfo/Ver_info.txt](#)' file
- And then please report the issue with the firmware version information to QCS Support Engineer

Confidential
2019-12-30 09:00
didi.setadi@qcom.com

```
/ # cat /firmware/verinfo/Ver_Info.txt
{
    "Image_Build_IDs": {
        "adsp": "ADSP.IT.1.0-00370.3-QCS405-1",
        "apps": "LE.UM.2.4.1.rl-09304-qcs405.1-1",
        "boot": "BOOT.XF.0.1-00106-QCS405LAB-1",
        "btfm": "BTFM.CHE.2.1.4-00507-QCACHROM-1",
        "cdsp": "CDSP.IT.1.0-00249-QCS405-1",
        "common": "QCS405.LE.1.0.1-00293-INT.APC.NAND-1",
        "glue": "GLUE.QCS405.LE.1.0-00117-NOOP_TEST-1",
        "rpm": "RPM.BF.1.9-00030-QCS405AAAAANAAR-1",
        "tz": "TZ.XF.5.1-00200-Q405AAAAANAZT-1",
        "wcnss_qz": "CNSS_W.QZ.4.0-00023-QZHW4024-1",
        "wlan": "WLAN.HL.3.1-01018-QCAHLSWMTPL-1"
    },
    "Metabuild_Info": {
        "Meta_Build_ID": "QCS405.LE.1.0.1-00293-INT.APC.NAND-1",
        "Product_Flavor": "[asic]",
        "Time_Stamp": "2019-04-28 21:52:32"
    },
    "Version": "1.0"
}
/ #
/ #
```



Section 8.2

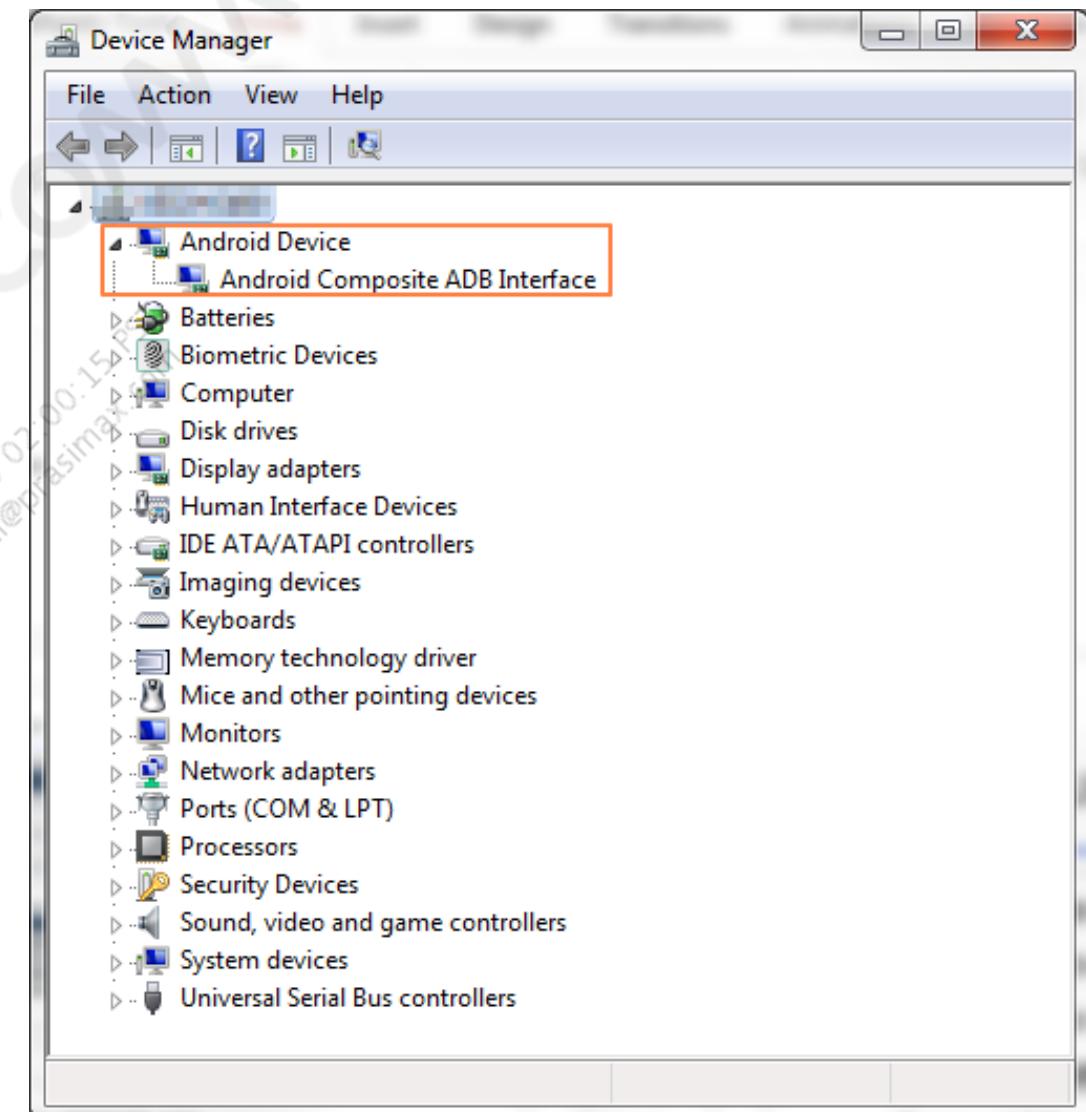
Writing Device Configuration

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

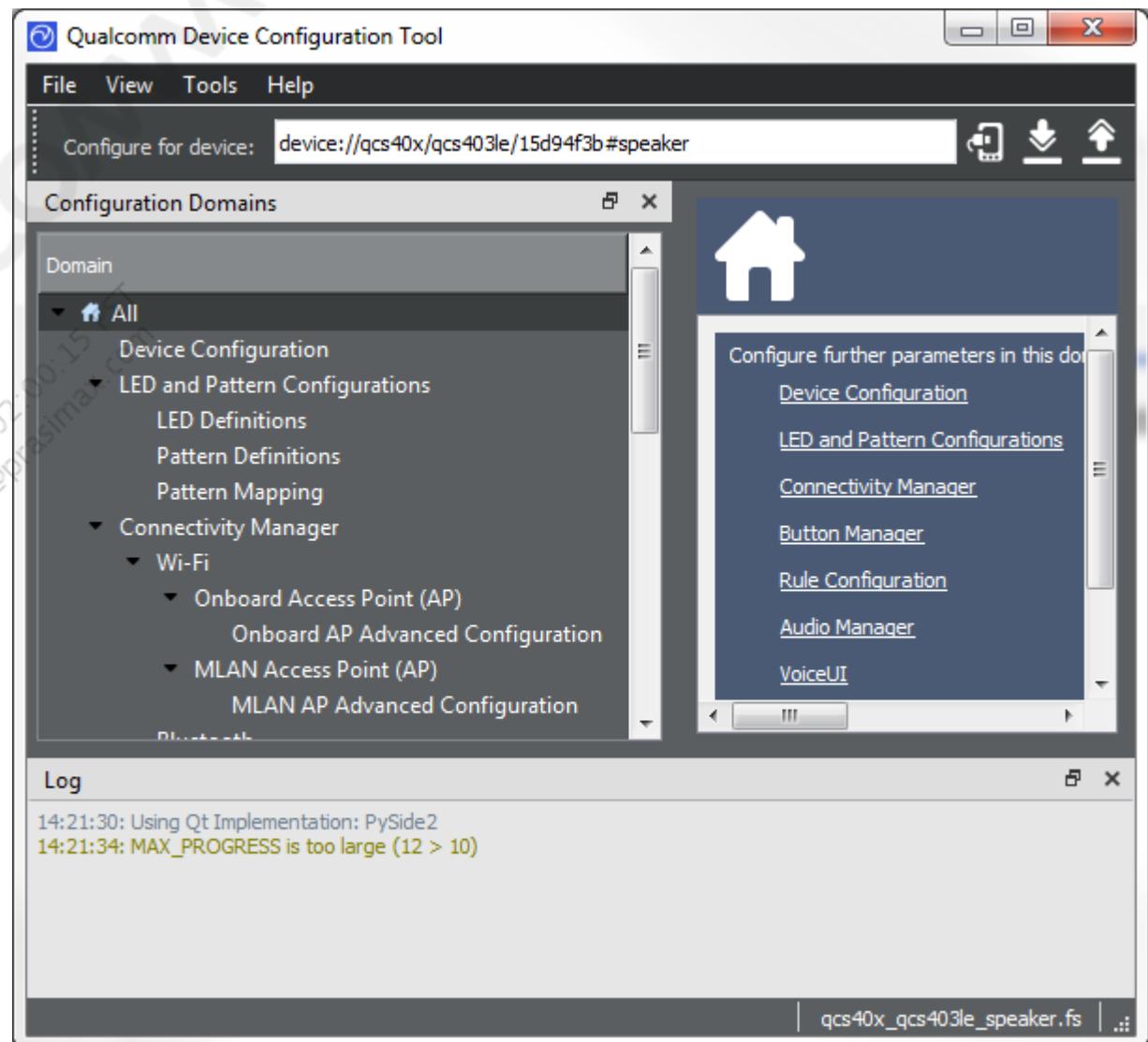
Writing Device Configuration

- To start many basic HLOS services and daemons, configuration data must be written to the device using Qualcomm Smart Audio Platform Configuration Tool
- To write configuration data, the device must be configured to normal boot mode
- Refer to:
 - [Device Boot Configuration for SSRD – Normal boot mode](#)



Writing Device Configuration

- Launch Qualcomm Smart Audio Platform Configuration Tool



Configure further parameters in this domain:
[Device Configuration](#)
[LED and Pattern Configurations](#)
[Connectivity Manager](#)
[Button Manager](#)
[Rule Configuration](#)
[Audio Manager](#)
[VoiceUI](#)

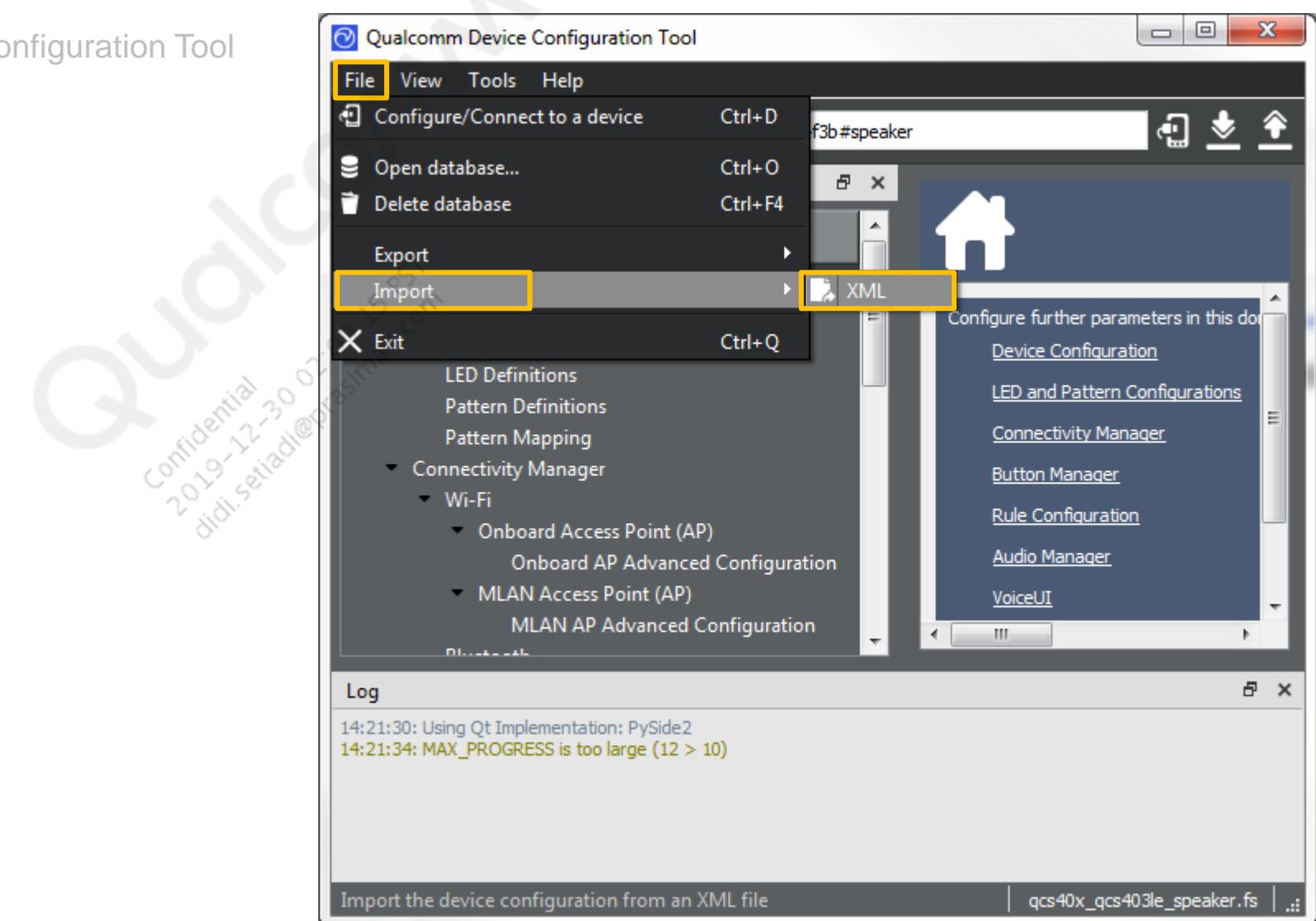
Log

14:21:30: Using Qt Implementation: PySide2
14:21:34: MAX_PROGRESS is too large (12 > 10)

qcs40x_qcs403le_speaker.fs

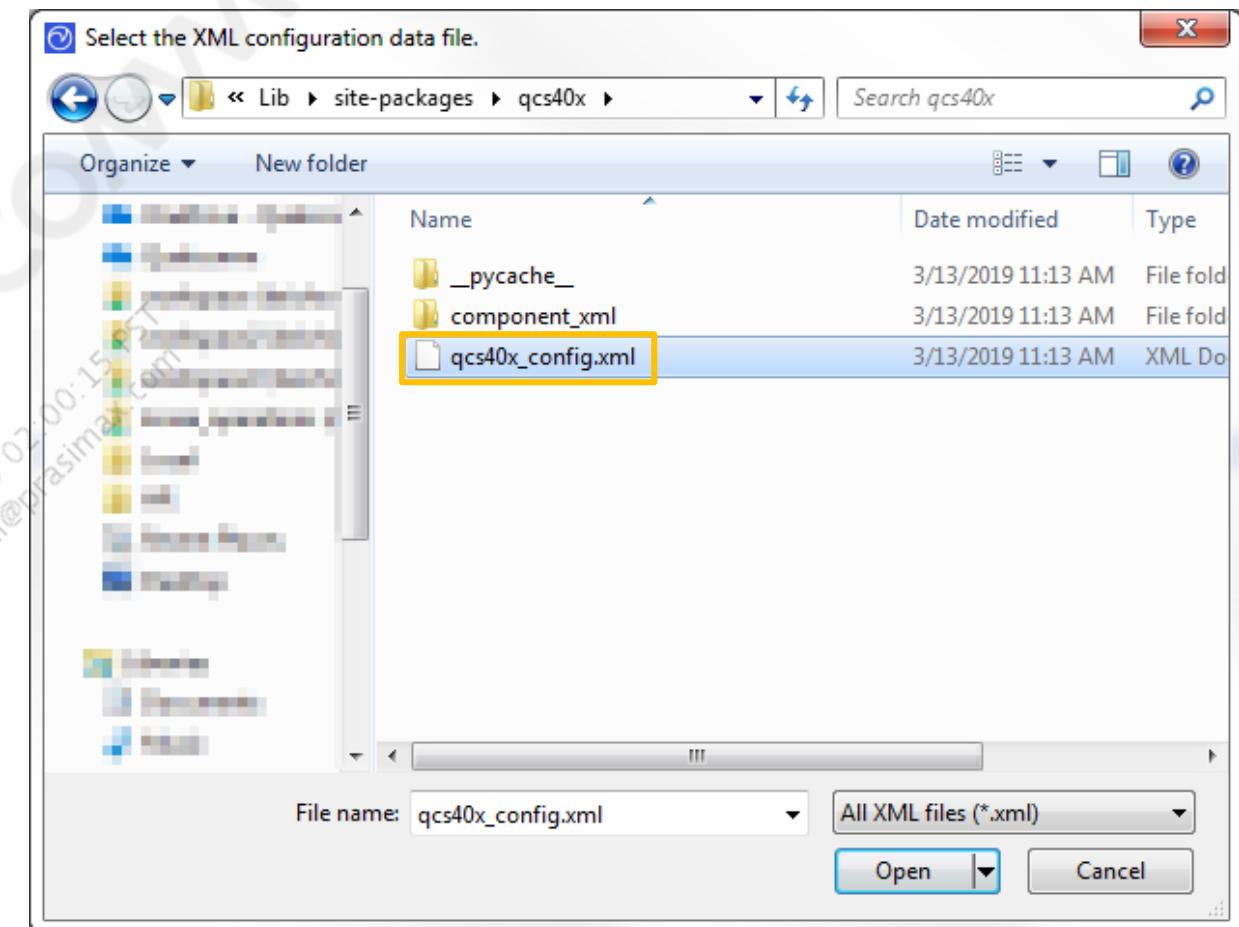
Writing Device Configuration

- Launch Qualcomm Smart Audio Platform Configuration Tool
- Click on ‘File > Import > XML’ menu



Writing Device Configuration

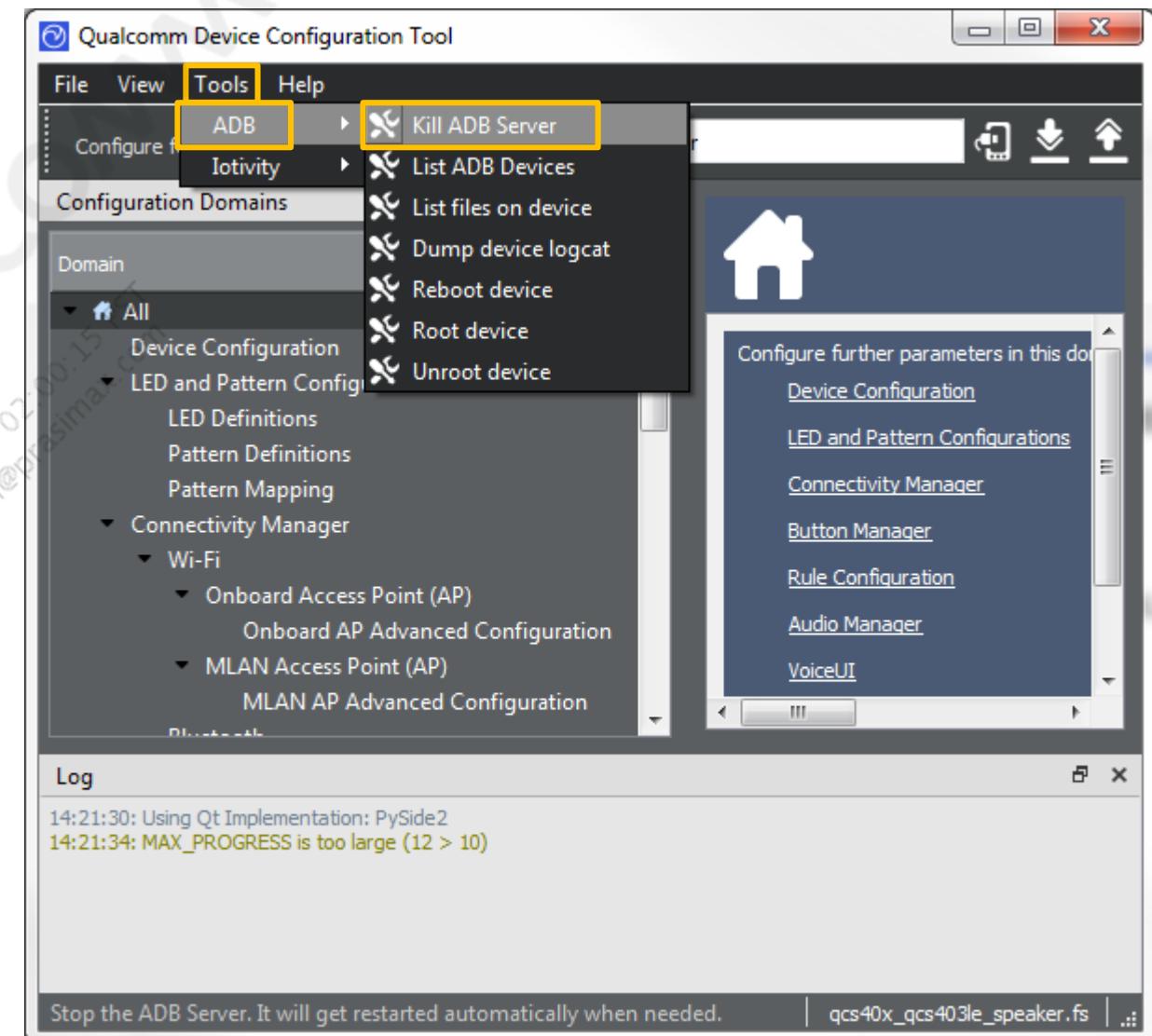
- Launch Qualcomm Smart Audio Platform Configuration Tool
- Click on 'File > Import > XML' menu
- Open 'qcs40x_config.xml' file on the following folder:
 - C:\Program Files (x86)\QualcommDeviceConfigTool\pyenv36\Lib\site-packages\qcs40x



Writing Device Configuration

- Launch Qualcomm Smart Audio Platform Configuration Tool
- Click on ‘File > Import > XML’ menu
- Open ‘qcs40x_config.xml’ file on the following folder:
 - C:\Program Files (x86)\QualcommDeviceConfigTool\pyenv36\Lib\site-packages\qcs40x
- Kill adb server with ‘Tools > ADB > Kill ADB Server’ menu

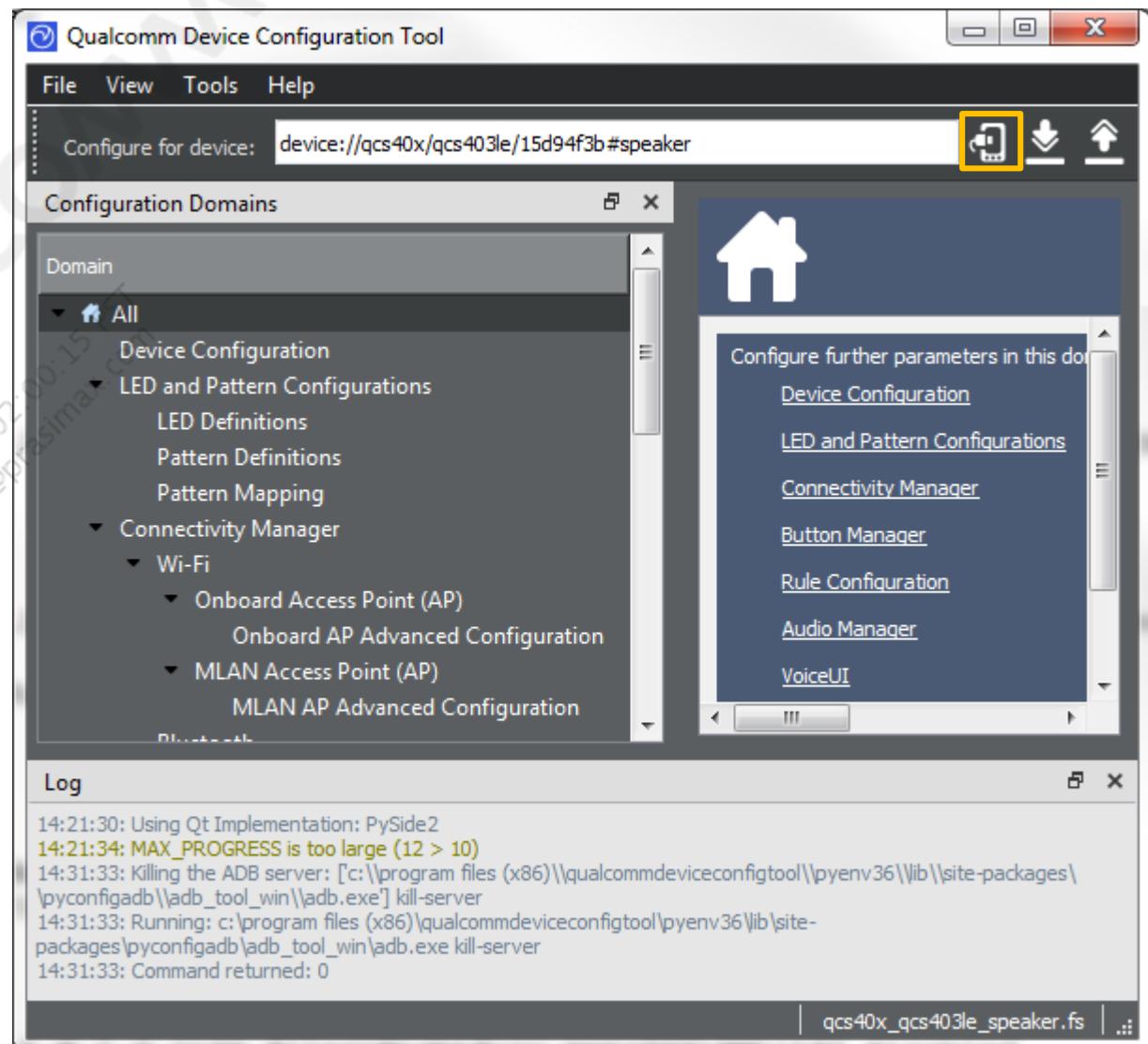
Confidential
2019-12-30 07:00:15
didi.setiadi@prismaqa.com



Writing Device Configuration

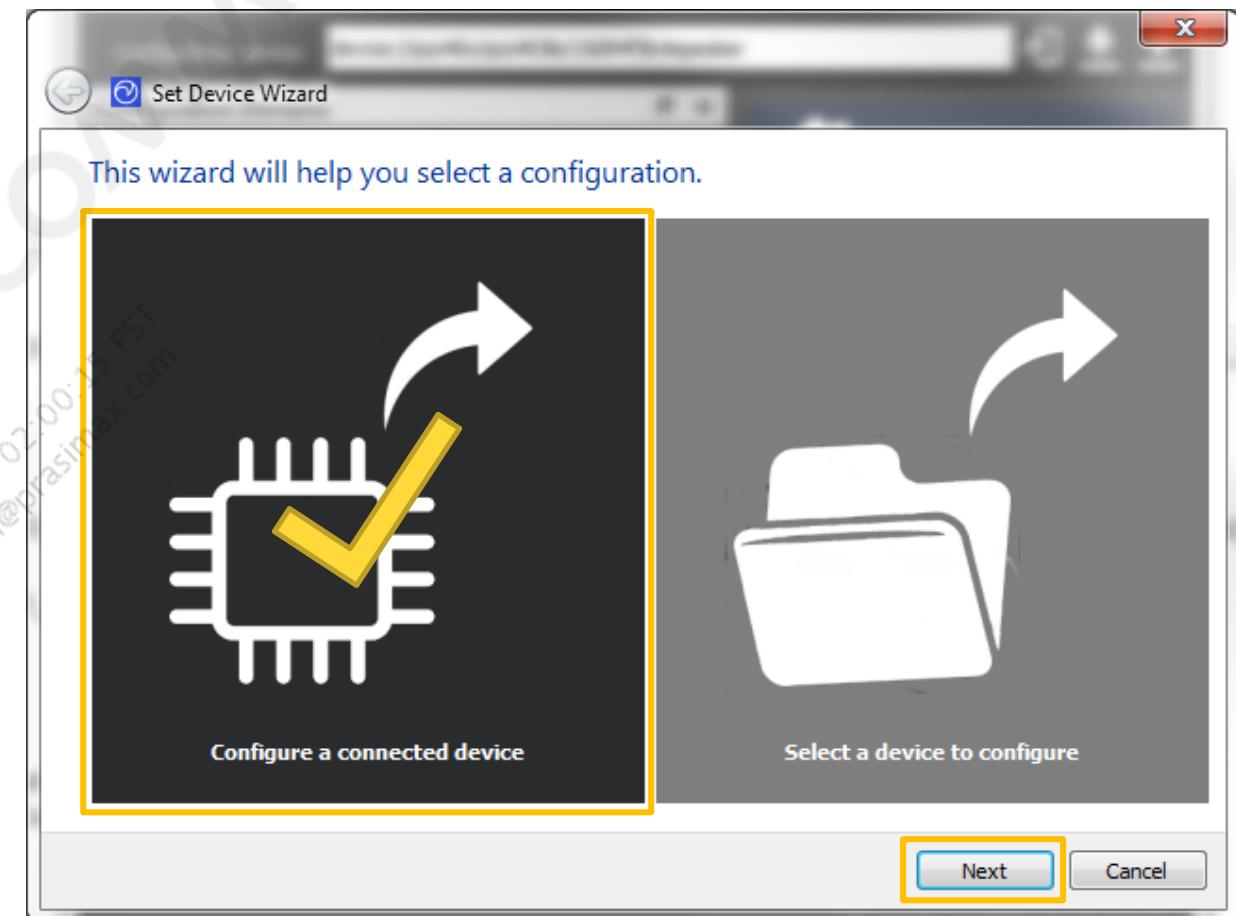
- Launch Qualcomm Smart Audio Platform Configuration Tool
 - Click on 'File > Import > XML' menu
 - Open 'qcs40x_config.xml' file on the following folder:
 - C:\Program Files (x86)\QualcommDeviceConfigTool\pyenv36\Lib\site-packages\qcs40x
 - Kill adb server with 'Tools > ADB > Kill ADB Server' menu
 - Click on 'Choose the device to configure' button

Q
Confidential
2019-12-30
didi setia



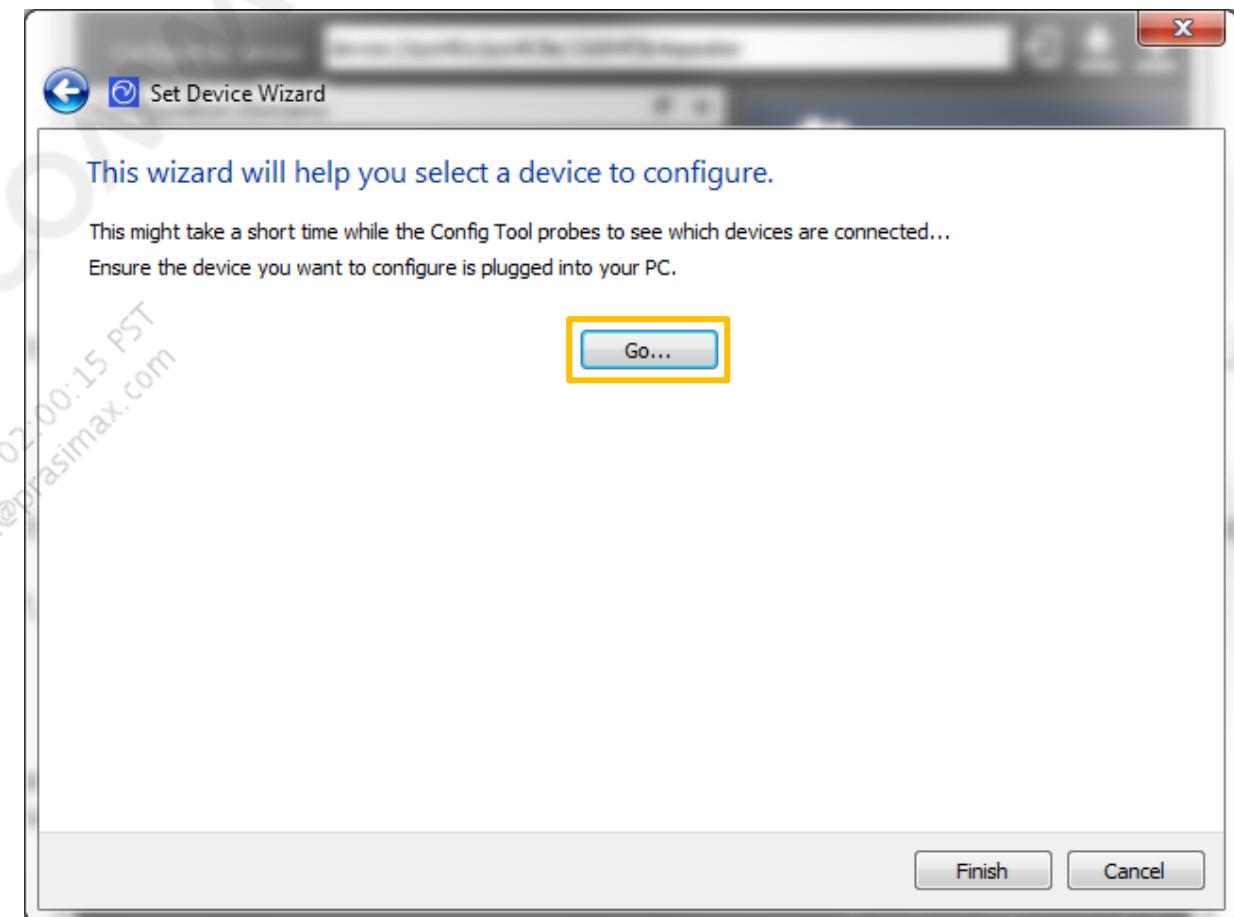
Writing Device Configuration

- Launch Qualcomm Smart Audio Platform Configuration Tool
- Click on ‘File > Import > XML’ menu
- Open ‘qcs40x_config.xml’ file on the following folder:
 - C:\Program Files (x86)\QualcommDeviceConfigTool\pyenv36\Lib\site-packages\qcs40x
- Kill adb server with ‘Tools > ADB > Kill ADB Server’ menu
- Click on ‘Choose the device to configure’ button
- Select ‘Configure a connected device’ and click on ‘Next’



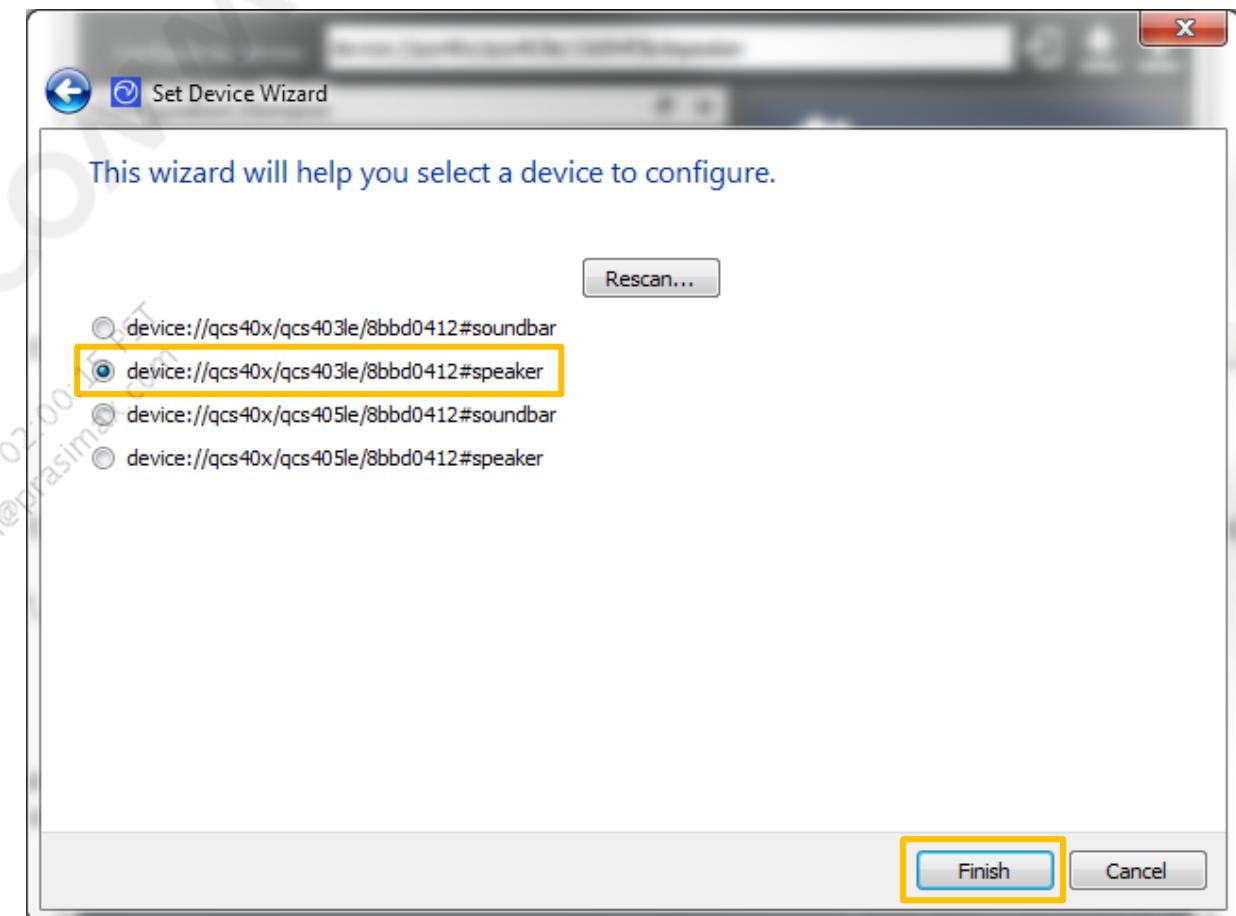
Writing Device Configuration

- Launch Qualcomm Smart Audio Platform Configuration Tool
- Click on ‘File > Import > XML’ menu
- Open ‘qcs40x_config.xml’ file on the following folder:
 - C:\Program Files (x86)\QualcommDeviceConfigTool\pyenv36\Lib\site-packages\qcs40x
- Kill adb server with ‘Tools > ADB > Kill ADB Server’ menu
- Click on ‘Choose the device to configure’ button
- Select ‘Configure a connected device’ and click on ‘Next’
- Click on ‘Go...’ button



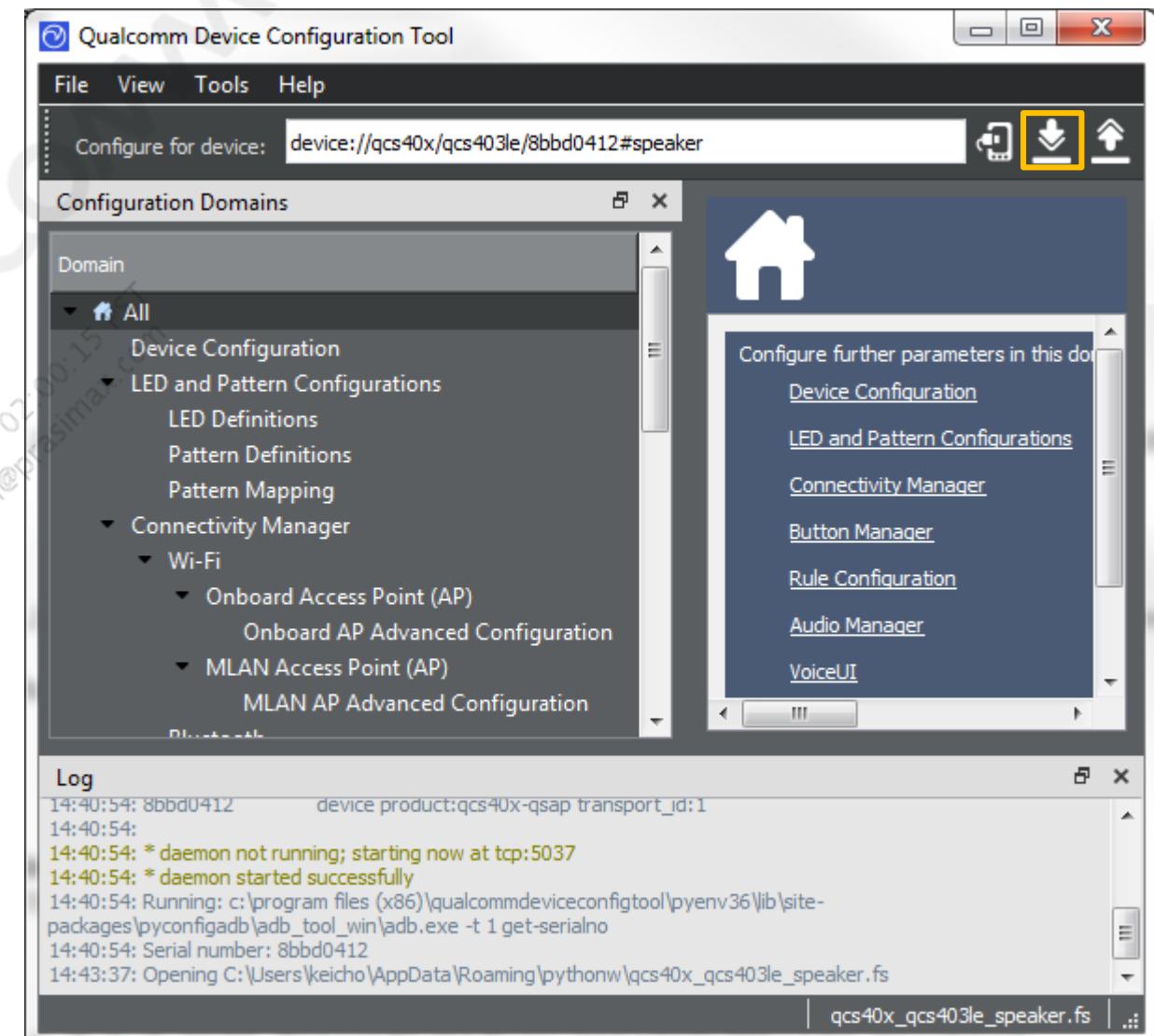
Writing Device Configuration

- Launch Qualcomm Smart Audio Platform Configuration Tool
- Click on ‘File > Import > XML’ menu
- Open ‘qcs40x_config.xml’ file on the following folder:
 - C:\Program Files (x86)\QualcommDeviceConfigTool\pyenv36\Lib\site-packages\qcs40x
- Kill adb server with ‘Tools > ADB > Kill ADB Server’ menu
- Click on ‘Choose the device to configure’ button
- Select ‘Configure a connected device’ and click on ‘Next’
- Click on ‘Go...’ button
- Select ‘device://qcs40x/qcs403le/8bbd0412#speaker’ and click on ‘Finish’ button



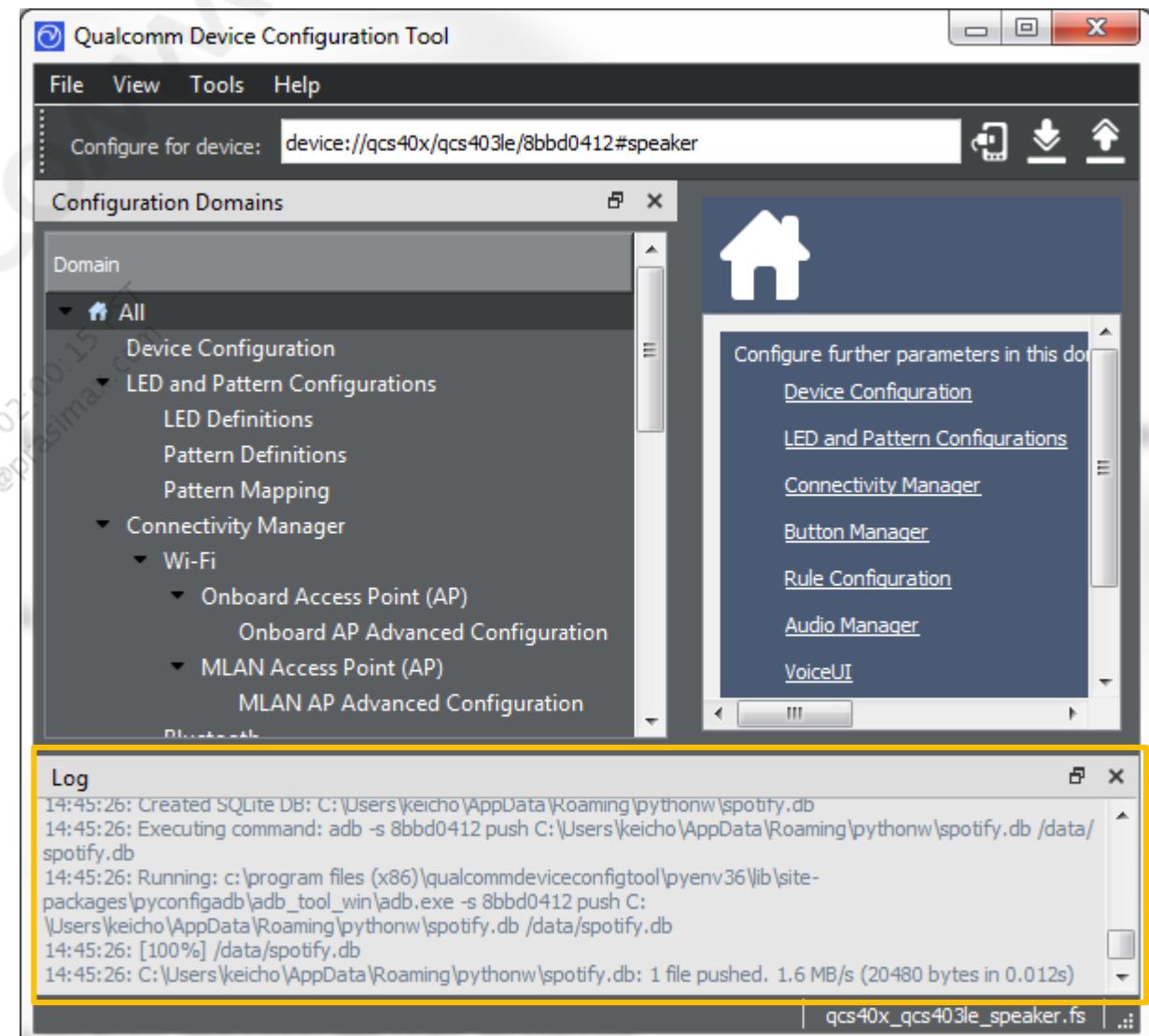
Writing Device Configuration

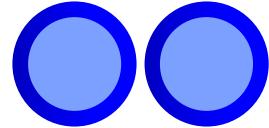
- Launch Qualcomm Smart Audio Platform Configuration Tool
- Click on ‘File > Import > XML’ menu
- Open ‘qcs40x_config.xml’ file on the following folder:
 - C:\Program Files (x86)\QualcommDeviceConfigTool\pyenv36\Lib\site-packages\qcs40x
- Kill adb server with ‘Tools > ADB > Kill ADB Server’ menu
- Click on ‘Choose the device to configure’ button
- Select ‘Configure a connected device’ and click on ‘Next’
- Click on ‘Go...’ button
- Select ‘device://qcs40x/qcs403le/8bbd0412#speaker’ and click on ‘Finish’ button
- Click on ‘Push configuration to device’ button



Writing Device Configuration

- Launch Qualcomm Smart Audio Platform Configuration Tool
- Click on ‘File > Import > XML’ menu
- Open ‘qcs40x_config.xml’ file on the following folder:
 - C:\Program Files (x86)\QualcommDeviceConfigTool\pyenv36\Lib\site-packages\qcs40x
- Kill adb server with ‘Tools > ADB > Kill ADB Server’ menu
- Click on ‘Choose the device to configure’ button
- Select ‘Configure a connected device’ and click on ‘Next’
- Click on ‘Go...’ button
- Select ‘device://qcs40x/qcs403le/8bbd0412#speaker’ and click on ‘Finish’ button
- Click on ‘Push configuration to device’ button
- Make sure the result log and then reboot device
 - > **adb reboot**
- **DONE !**





Section 8.3

Setting up Haven License

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Setting up Haven License

- In this chapter, it will be explained how to get Evaluation License
- Evaluation License allow to use the following audio features:
 - Far Field Voice (FFV)
 - Qualcomm® aptX™
 - Local ASR
 - Qualcomm® AllPlay™
- Below are the pre-defined Bluetooth Device Address for Evaluation License
 - 70:c9:4e:b7:f6:54
 - 70:c9:4e:7f:63:7a
 - 70:c9:4e:5b:b3:fe
 - 70:c9:4e:5b:b9:4e
 - 70:c9:4e:7f:6d:0e
- In this chapter, '**70:c9:4e:b7:f6:54**' will be used as an example
- For more details, refer to *[80-YB405-129] QCS40x Software User Guide*

Qualcomm Confidential and Proprietary
2019-12-30 10:00:15 PST
didi.setiadi@prajakta.com

Setting up Haven License

1) Modify 'bt_config.conf' file

Modify 'bt_config.conf' file

```
// Run adb shell  
> adb shell  
  
// Enable Bluetooth  
/ # btproperty &  
/ # btapp  
    gap_menu  
    enable  
    CTRL+C  
  
...  
/ # killall btproperty  
/ # exit  
  
// Change Bluetooth address in bt_config.conf file  
> adb pull /data/misc/bluetooth/bt_config.conf  
=====  
  
...  
[Adapter]  
Address = 70:c9:4e:b7:f6:54  
...  
=====  
> adb push bt_config.conf /data/misc/bluetooth  
> adb shell  
  
// Change bt_config.conf access permission  
/ # chmod 777 /data/misc/bluetooth/bt_config.conf
```

Setting up Haven License

- 1) Modify 'bt_config.conf' file
- 2) Change Bluetooth device address

Change device BT address

```
// Run adb shell  
> adb shell  
  
// Change Bluetooth device address  
/ # setprop persist.vendor.service.bdroid.bdaddr 70:c9:4e:b7:f6:54  
  
// Check Bluetooth device address is 70:c9:4e:b7:f6:54  
/ # getprop persist.vendor.service.bdroid.bdaddr  
70:c9:4e:b7:f6:54
```

Confidential
2019-12-30 02:00:00 PST
didi.setiadi@praktikum.psu.ac.id

Setting up Haven License

- 1) Modify 'bt_config.conf' file
- 2) Change Bluetooth device address
- 3) Check the license file is on the device, if not, push the license file to device and then reboot device
 - License file name: **vipertooth_test_evaluation_license.pfm**
 - File location: <QCS403_WS>/apps_proc/poky/meta-qt-audio-prop/recipes/alm/files/havenlicense/
 - License expiration: **June 30, 2019.**

Push license file to device

```
// Run adb shell  
> adb shell  
  
// Check the license file is in '/persist/data/pfm/licenses'  
/ # ls /persist/data/pfm/licenses  
900-930-jun-2019-770545770.pfm . . .  
e.g. In LE.1.0 CS, the license file already exist in device  
*****  
// If the license file does not exist, Run following steps  
*****  
// Create a folder where the license file is located to the device  
/ # mkdir -p /persist/data/pfm/licenses  
/ # exit  
  
// Push license file  
> adb push vipertooth_test_evaluation_license.pfm /persist/data/pfm/licenses  
> adb shell  
  
// Change access permission  
/ # chmod -R 777 /persist/data  
/ # exit  
  
// Reboot device,  
// NOTE: DO NOT USE 'adb shell reboot'  
> adb reboot
```

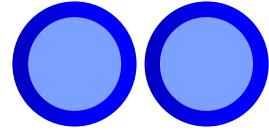
Setting up Haven License

- 1) Modify 'bt_config.conf' file
- 2) Change Bluetooth device address
- 3) Check the license file is on the device, if not, push the license file to device and then reboot device
 - License file name: `vipertooth_test_evaluation_license.pfm`
 - File location: `<QCS403_WS>/apps_proc/poky/meta-qt-audio-prop/recipes/alm/files/havenlicense/`
 - License expiration: **June 30, 2019.**
- 4) After reboot, check Bluetooth device address is
70:c9:4e:b7:f6:54
- DONE !

Check device BT address

```
// Check Bluetooth device address  
> adb shell getprop persist.vendor.service.bdroid.bdaddr  
70:c9:4e:b7:f6:54
```

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com



Section 8.4

Onboarding Wi-Fi

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Onboarding Wi-Fi

- There are several methods for Wi-Fi Onboarding:
 - Using QCMAPI_CLI application (adb)
 - Using wpa_supplicant application (adb)
 - Using ADK message (adb)
 - Using Qualcomm Smart Audio App (Android, iOS)
 - Using Web browser (PC, SmartPhone, and so on...)
- In this chapter, ‘Using ADK message’ will be explained

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Onboarding Wi-Fi

- If the default configuration was written successfully, the device's Wi-Fi is in 'STA + SoftAP' mode
 - / # ifconfig

Confidential
2019-12-30 07:00:15 PT
didi.setiadi@prashmax.com

```
/ # ifconfig
bridge0  Link encap:Ethernet HWaddr 82:A4:A0:15:7A:AB
          inet addr:192.168.225.1 Bcast:192.168.225.255 Mask:255.255.255.0
          inet6 addr: fe80::80a4:aOff:fe12:77a8/64 Scope:Link
            UP BROADCAST RUNNING PROMISC MULTICAST MTU:1500 Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:20 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:0 (0.0 B) TX bytes:3642 (3.5 KiB)

lo      Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING MTU:65536 Metric:1
            RX packets:24 errors:0 dropped:0 overruns:0 frame:0
            TX packets:24 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:1960 (1.9 KiB) TX bytes:1960 (1.9 KiB)

wlan0   Link encap:Ethernet HWaddr 00:0A:F5:BD:04:12
          UP BROADCAST MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3000
          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

wlan1   Link encap:Ethernet HWaddr 02:0A:F5:42:04:12
          inet addr:169.254.1.1 Bcast:169.254.1.255 Mask:255.255.255.0
          inet6 addr: fe80::a:f5ff:fe42:412/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:26 errors:0 dropped:7 overruns:0 carrier:0
            collisions:0 txqueuelen:3000
            RX bytes:0 (0.0 B) TX bytes:4853 (4.7 KiB)

/ #
```

Onboarding Wi-Fi

- If the default configuration was written successfully, the device's Wi-Fi is in 'STA + SoftAP' mode
 - / # ifconfig
- Start Wi-Fi Onboarding with following commands
 - / # adk-message-send 'connectivity_wifi_disable{}'
 - / # adk-message-send 'connectivity_wifi_enable{}'
 - / # adk-message-send 'connectivity_wifi_scan{}'
 - / # adk-message-send 'connectivity_wifi_connect {ssid:"**MY_AP_SSID**" password:"**MY_AP_PASSWORD**" homeap:true}'

```
/ # adk-message-send 'connectivity wifi disable{}'
Sending message:
connectivity_wifi_disable {
}
Attempted to unregister path (path[0] = com path[1] = qualcomm) which isn't registered
/ # adk-message-send 'connectivity_wifi_enable{}'
Sending message:
connectivity_wifi_enable {
}
Attempted to unregister path (path[0] = com path[1] = qualcomm) which isn't registered
/ # adk-message-send 'connectivity_wifi_scan{}'
Sending message:
connectivity_wifi_scan {
}
Attempted to unregister path (path[0] = com path[1] = qualcomm) which isn't registered
/ # adk-message-send 'connectivity wifi_connect {ssid:"V30" password:"11112222"
homeap:true}'
Sending message:
connectivity_wifi_connect {
    ssid: "V30"
    password: "11112222"
    homeap: true
}
Attempted to unregister path (path[0] = com path[1] = qualcomm) which isn't registered
/ #
```

Confidential
2019-12-30
didi.setiadi@q.com

Onboarding Wi-Fi

- If the default configuration was written successfully, the device's Wi-Fi is in 'STA + SoftAP' mode
 - / # ifconfig
- Start Wi-Fi Onboarding with following commands
 - / # adk-message-send 'connectivity_wifi_disable{}'
 - / # adk-message-send 'connectivity_wifi_enable{}'
 - / # adk-message-send 'connectivity_wifi_scan{}'
 - / # adk-message-send 'connectivity_wifi_connect {ssid:"MY_AP_SSID" password:"MY_AP_PASSWORD" homeap:true}'
- Check 'wlan0' gets the IP address
 - / # ifconfig
 - / # ping 8.8.8.8

```
/ # ifconfig
bridge0  Link encap:Ethernet HWaddr 82:A4:A0:15:7A:AB
          inet addr:192.168.225.1 Bcast:192.168.225.255 Mask:255.255.255.0
          inet6 addr: fe80::80a4:a0ff:fe12:77a8/64 Scope:Link
          UP BROADCAST RUNNING PROMISC ALLMULTI MULTICAST MTU:1500 Metric:1
          RX packets:1 errors:0 dropped:0 overruns:0 frame:0
          TX packets:966 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:282 (282.0 B) TX bytes:285517 (278.8 KiB)

lo      Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING MTU:65536 Metric:1
          RX packets:1736 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1736 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:145888 (142.4 KiB) TX bytes:145888 (142.4 KiB)

wlan0   Link encap:Ethernet HWaddr 00:0A:F5:BD:04:12
          inet addr:192.168.43.241 Bcast:192.168.43.255 Mask:255.255.255.0
          inet6 addr: fe80::20a:f5ff:febд:412/64 Scope:Link
          UP BROADCAST RUNNING PROMISC ALLMULTI MULTICAST MTU:1500 Metric:1
          RX packets:243 errors:0 dropped:0 overruns:0 frame:0
          TX packets:258 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3000
          RX bytes:32429 (31.6 KiB) TX bytes:32433 (31.6 KiB)

wlan1   Link encap:Ethernet HWaddr 02:0A:F5:42:04:12
          inet addr:169.254.1.1 Bcast:169.254.1.255 Mask:255.255.255.0
          inet6 addr: fe80::a:f5ff:fe42:412/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:98 errors:0 dropped:7 overruns:0 carrier:0
          collisions:0 txqueuelen:3000
          RX bytes:0 (0.0 B) TX bytes:25350 (24.7 KiB)

/ # ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=116 time=99.1 ms
```

Onboarding Wi-Fi

- If the default configuration was written successfully, the device's Wi-Fi is in 'STA + SoftAP' mode
 - / # ifconfig
- Start Wi-Fi Onboarding with following commands
 - / # adk-message-send 'connectivity_wifi_disable{}'
 - / # adk-message-send 'connectivity_wifi_enable{}'
 - / # adk-message-send 'connectivity_wifi_scan{}'
 - / # adk-message-send 'connectivity_wifi_connect {ssid:"MY_AP_SSID", password:"MY_AP_PASSWORD" homeap:true}'
- Check 'wlan0' gets the IP address
 - / # ifconfig
 - / # ping 8.8.8.8
- Complete Wi-Fi Onboarding with following command
 - / # adk-message-send 'connectivity_wifi_completeonboarding{}'
- Wi-Fi will be reset automatically

```
/ # adk-message-send 'connectivity wifi completeonboarding{}'
Sending message:
connectivity_wifi_completeonboarding {
}
Attempted to unregister path (path[0] = com path[1] = qualcomm) which isn't registered
/ # ifconfig
bridge0  Link encap:Ethernet  HWaddr 82:A4:A0:15:7A:AB
          inet addr:192.168.225.1  Bcast:192.168.225.255  Mask:255.255.255.0
          inet6 addr: fe80::80a4:a0ff:fe12:77a8/64 Scope:Link
                  UP BROADCAST PROMISC ALLMULTI MULTICAST  MTU:1500  Metric:1
                  RX packets:1 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:1152 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:282 (282.0 B)  TX bytes:343880 (335.8 KiB)
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
                  UP LOOPBACK RUNNING  MTU:65536  Metric:1
                  RX packets:1736 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:1736 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:145888 (142.4 KiB)  TX bytes:145888 (142.4 KiB)
```

'wlan' disappears for a few seconds

Onboarding Wi-Fi

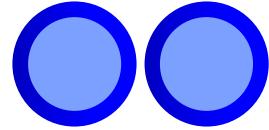
- If the default configuration was written successfully, the device's Wi-Fi is in 'STA + SoftAP' mode
 - / # ifconfig
- Start Wi-Fi Onboarding with following commands
 - / # adk-message-send 'connectivity_wifi_disable{}'
 - / # adk-message-send 'connectivity_wifi_enable{}'
 - / # adk-message-send 'connectivity_wifi_scan{}'
 - / # adk-message-send 'connectivity_wifi_connect {ssid:"MY_AP_SSID" password:"MY_AP_PASSWORD" homeap:true}'
- Check 'wlan0' gets the IP address
 - / # ifconfig
 - / # ping 8.8.8.8
- Complete Wi-Fi Onboarding with following command
 - / # adk-message-send 'connectivity_wifi_completeonboarding{}'
- Wi-Fi will be reset automatically
- Wait until Wi-Fi reconnects to MY_AP with 'STA only' mode
- DONE !

```
/ # ifconfig
bridge0  Link encap:Ethernet  HWaddr 82:A4:A0:15:7A:AB
          inet addr:192.168.225.1  Bcast:192.168.225.255  Mask:255.255.255.0
          inet6 addr: fe80::80a4:aOff:fe12:77a8/64 Scope:Link
                  UP BROADCAST PROMISC ALLMULTI MULTICAST  MTU:1500  Metric:1
                  RX packets:1 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:1152 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:282 (282.0 B)  TX bytes:343880 (335.8 KiB)

lo      Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
                  UP LOOPBACK RUNNING  MTU:65536  Metric:1
                  RX packets:1736 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:1736 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:145888 (142.4 KiB)  TX bytes:145888 (142.4 KiB)

wlan0   Link encap:Ethernet  HWaddr 00:0A:F5:BD:04:12
          inet addr:192.168.43.241  Bcast:192.168.43.255  Mask:255.255.255.0
          inet6 addr: 2001:2d8:e204:b77b:20a:f5ff:febд:412/64 Scope:Global
          inet6 addr: fe80::20a:f5ff:febд:412/64 Scope:Link
                  UP BROADCAST RUNNING PROMISC ALLMULTI MULTICAST  MTU:1500  Metric:1
                  RX packets:466 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:488 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:3000
                  RX bytes:69023 (67.4 KiB)  TX bytes:68527 (66.9 KiB)

/ #
```



Section 8.5

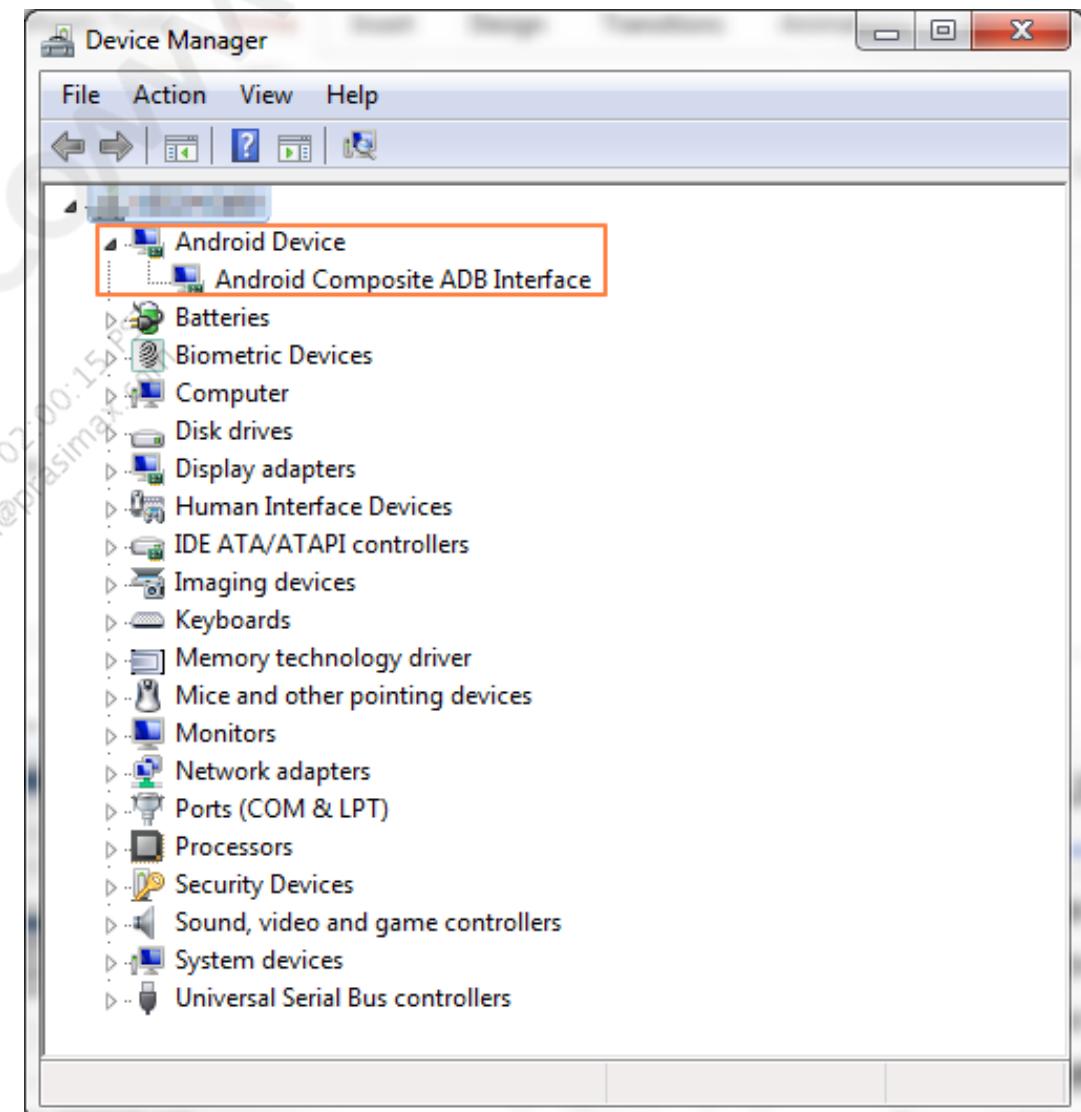
Onboarding AVS

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

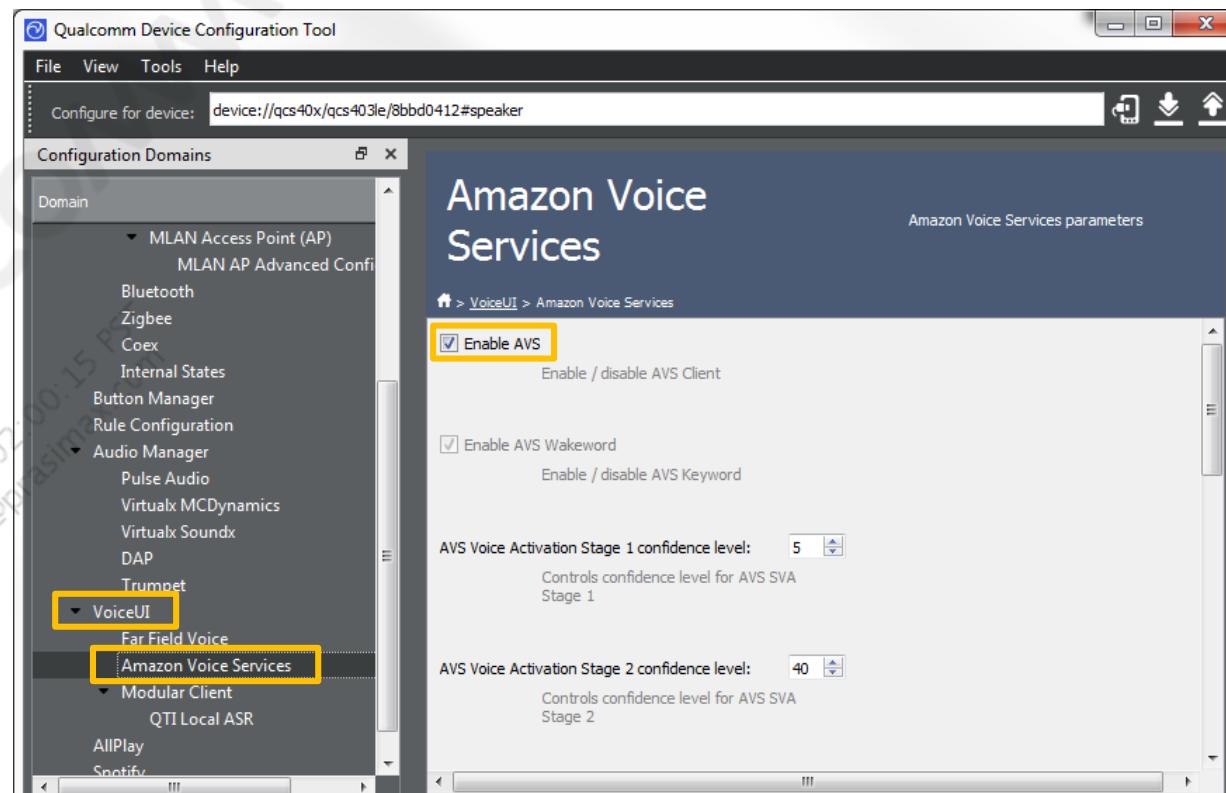
Onboarding AVS – Prerequisite

- Amazon User Account is required to onboard AVS
- The following below should have already been performed
 - [Writing Device Configuration](#)
 - [Haven License](#)
 - [Onboarding Wi-Fi](#)
- The device must be configured to normal boot mode
 - Refer to: [Device Boot Configuration for SSRD – Normal boot mode](#)



Onboarding AVS – Enable AVS Configuration

- AVS service is enabled in default configuration XML file, so AVS service would have already been enabled through the [Writing Device Configuration](#) step
- To check AVS service is enabled in default XML file:
 - Open Device Configuration Tool
 - Import default XML file
 - Click on ‘VoiceUI > Amazon Voice Services’ item
 - Make sure the ‘Enable AVS’ is check ON
- If the ‘Enable AVS’ is checked OFF then check ON the ‘Enable AVS’ and write again the updated configuration



Onboarding AVS

1. Check Wi-Fi connection and **VoiceUISampleApp** process is running

- / # ifconfig
- / # ps | grep Voice

```
/ # ifconfig
bridge0  Link encap:Ethernet  HWaddr 0A:AC:04:5C:EC:0A
          inet addr:192.168.225.1  Bcast:192.168.225.255  Mask:255.255.255.0
          inet6 addr: fe80::8ac:4ff:fe59:e907/64 Scope:Link
             UP BROADCAST PROMISC ALLMULTI MULTICAST  MTU:1500  Metric:1
             RX packets:0 errors:0 dropped:0 overruns:0 frame:0
             TX packets:1 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:0 (0.0 B)  TX bytes:90 (90.0 B)

lo      Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
             UP LOOPBACK RUNNING  MTU:65536  Metric:1
             RX packets:100 errors:0 dropped:0 overruns:0 frame:0
             TX packets:100 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:8368 (8.1 KiB)  TX bytes:8368 (8.1 KiB)

wlan0   Link encap:Ethernet  HWaddr 00:0A:F5:BD:04:12
          inet addr:192.168.43.241  Bcast:192.168.43.255  Mask:255.255.255.0
          inet6 addr: 2001:2d8:eb5c:ac2d:20a:f5ff:febд:412/64 Scope:Global
          inet6 addr: fe80::20a:f5ff:febд:412/64 Scope:Link
             UP BROADCAST RUNNING PROMISC ALLMULTI MULTICAST  MTU:1500  Metric:1
             RX packets:40 errors:0 dropped:0 overruns:0 frame:0
             TX packets:40 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:3000
             RX bytes:5090 (4.9 KiB)  TX bytes:5373 (5.2 KiB)

/ # ps | grep Voice
1656 root      0:00 /usr/bin/VoiceUISampleApp
2011 root      0:00 {grep} /bin/busybox /bin/grep Voice
/ #
```

Onboarding AVS

1. Check Wi-Fi connection and VoiceUISampleApp process is running

- / # ifconfig
- / # ps | grep Voice

2. **Terminal #1:** Run ADK Message Monitor Application

- / # adk-message-monitor -a

Confidential
2019-12-30
didi.setadi@qti.com

```
/ # adk-message-monitor -a
Subscribing to group: allplay, name: next
Subscribing to group: allplay, name: pause
StartServerService()
Subscribing to group: allplay, name: play
Subscribing to group: allplay, name:
Subscribing to group: allplay, name:
Subscribing to group: allplay, name:
Subscribing to group: allplay, name: toggle_shuttle
Subscribing to group: audio, name: mic_mute_set
Subscribing to group: audio, name: mic_mute_toggle
Subscribing to group: audio, name: mic_mute_updated
Subscribing to group: audio, name: mute_set
Subscribing to group: audio, name: mute_toggle
Subscribing to group: audio, name: mute_updated
Subscribing to group: audio, name: player_metadata_updated
Subscribing to group: audio, name: player_status_update
Subscribing to group: audio, name: player_updated
Subscribing to group: audio, name: prompt_play
Subscribing to group: audio, name: source_changed
Subscribing to group: audio, name: source_next
Subscribing to group: audio, name: source_prev
Subscribing to group: audio, name: source_select
Subscribing to group: audio, name: source_set
Subscribing to group: audio, name: track_next
Subscribing to group: audio, name: track_pause
Subscribing to group: audio, name: track_play
Subscribing to group: audio, name: track_play_pause_toggle
Subscribing to group: audio, name: track_previous
Subscribing to group: audio, name: track_stop
Subscribing to group: audio, name: volume_down
Subscribing to group: audio, name: volume_set
```



Terminal #1

Onboarding AVS

1. Check Wi-Fi connection and VoiceUISampleApp process is running

- / # ifconfig
- / # ps | grep Voice

2. Terminal #1: Run ADK Message Monitor Application

- / # adk-message-monitor -a

3. Open another terminal window for adb shell - **Terminal #2**

4. **Terminal #2:** Run Onboarding AVS start command

- / # adk-message-send 'voiceui_start_onboarding {client:"AVS"}'

5. **Terminal #1:** Check 'code' value

```
/ # adk-message-send 'voiceui_start_onboarding {client:"AVS"}'  
Sending message:  
voiceui_start_onboarding {  
    client: "AVS"  
}  
Attempted to unregister path (path[0] = com.path)  
stered  
/ #
```



```
AdkMessage received: voiceui_start_onboarding  
Message Received [ms:1999539191 Wed Mar 13 13:12:13 2019]  
voiceui_start_onboarding {  
}  
AdkMessage received: display_notification  
Message Received [ms:1999539194 Wed Mar 13 13:12:13 2019]  
display_notification {  
    group: "voiceui"  
    name: "AVS"  
    message: "FLRVGA"  
    time_ms: "120000"  
}  
AdkMessage received: voiceui_authenticate_avs  
Message Received [ms:1999539198 Wed Mar 13 13:12:13 2019]  
voiceui_authenticate_avs {  
    url: "https://amazon.com/us/code"  
    code: "FLRVGA"  
}
```



Onboarding AVS

1. Check Wi-Fi connection and VoiceUISampleApp process is running
 - / # ifconfig
 - / # ps | grep Voice
2. Terminal #1: Run ADK Message Monitor Application
 - / # adk-message-monitor -a
3. Open another terminal window for adb shell - Terminal #2
4. Terminal #2: Run Onboarding AVS start command
 - / # adk-message-send 'voiceui_start_onboarding {client:"AVS"}'
5. Terminal #1: Check 'code' value
6. Open Web Browser and Go: <https://amazon.com/us/code>
 - Login with User's Amazon Account
 - Input the 'code' value and click on 'Continue' button

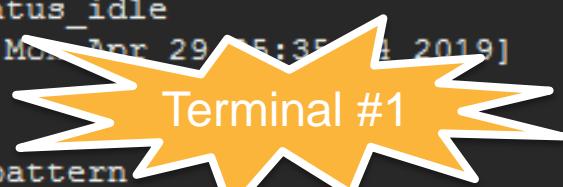


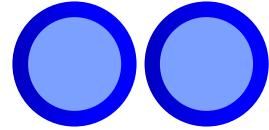
Onboarding AVS

1. Check Wi-Fi connection and VoiceUISampleApp process is running
 - / # ifconfig
 - / # ps | grep Voice
2. Terminal #1: Run ADK Message Monitor Application
 - / # adk-message-monitor -a
3. Open another terminal window for adb shell - Terminal #2
4. Terminal #2: Run Onboarding AVS start command
 - / # adk-message-send 'voiceui_start_onboarding {client:"AVS"}'
5. Terminal #1: Check 'code' value
6. Open Web Browser and Go: <https://amazon.com/us/code>
 - Login with User's Amazon Account
 - Input the 'code' value and click on 'Continue' button
7. Make sure 'voiceui.avs.onboarded' message on Terminal #1
8. DONE ! Say "Alexa"

Confidential
2019-12-30 or later
didi.setiadi@qualcomm.com

AdkMessage received: voiceui status_idle
Message Received [ms:1737943315 Mon Apr 29 05:35:04 2019]
voiceui_status_idle {
}
AdkMessage received: led start_pattern
Message Received [ms:1737943322 Mon Apr 29 05:35:04 2019]
led_start_pattern {
 pattern: 15
}
AdkMessage received: voiceui database_updated
Message Received [ms:1737943331 Mon Apr 29 05:35:04 2019]
voiceui_database_updated {
 key: "voiceui.avs.onboarded"
}





Section 8.6

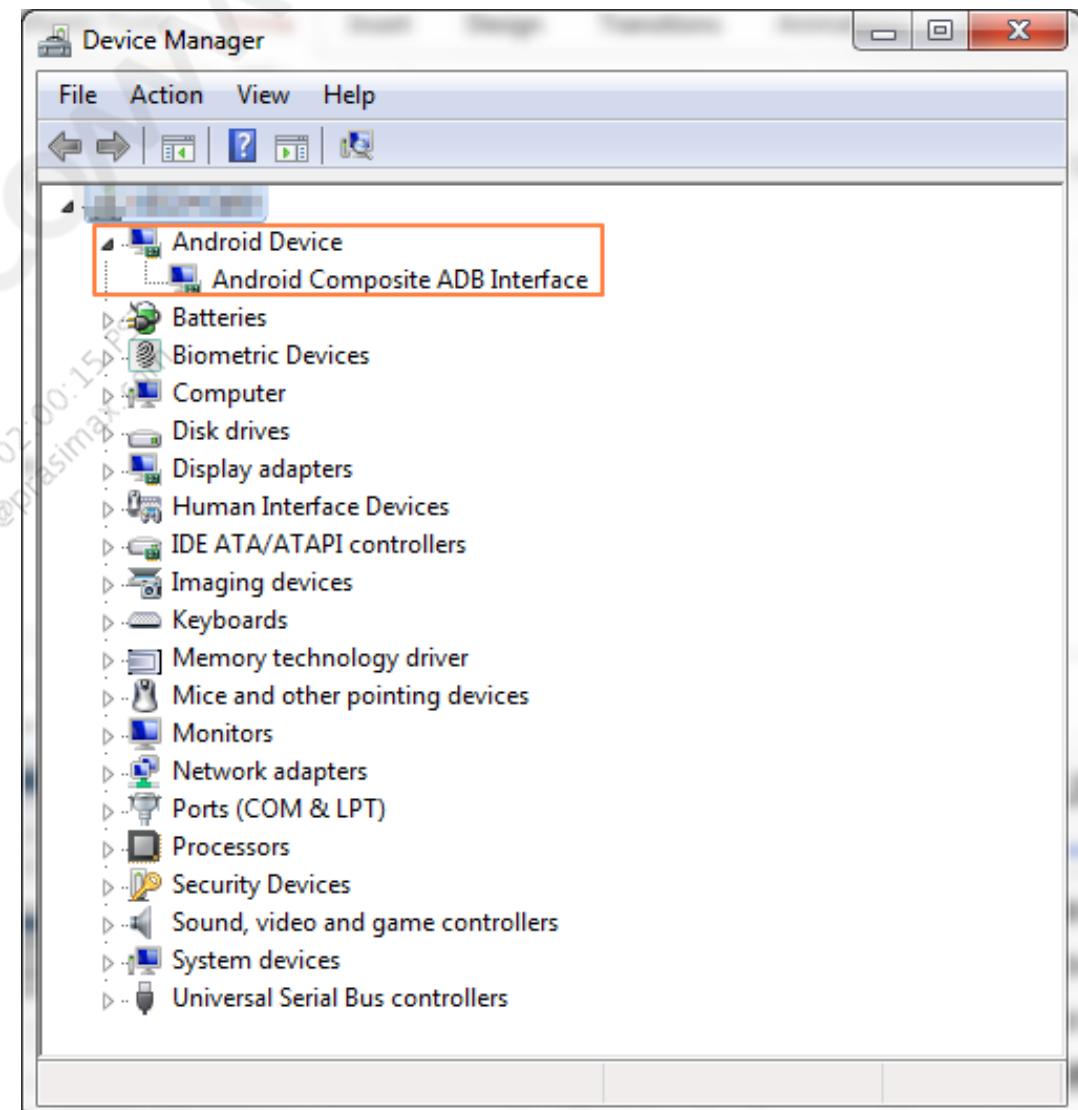
Local ASR + AVS Test

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

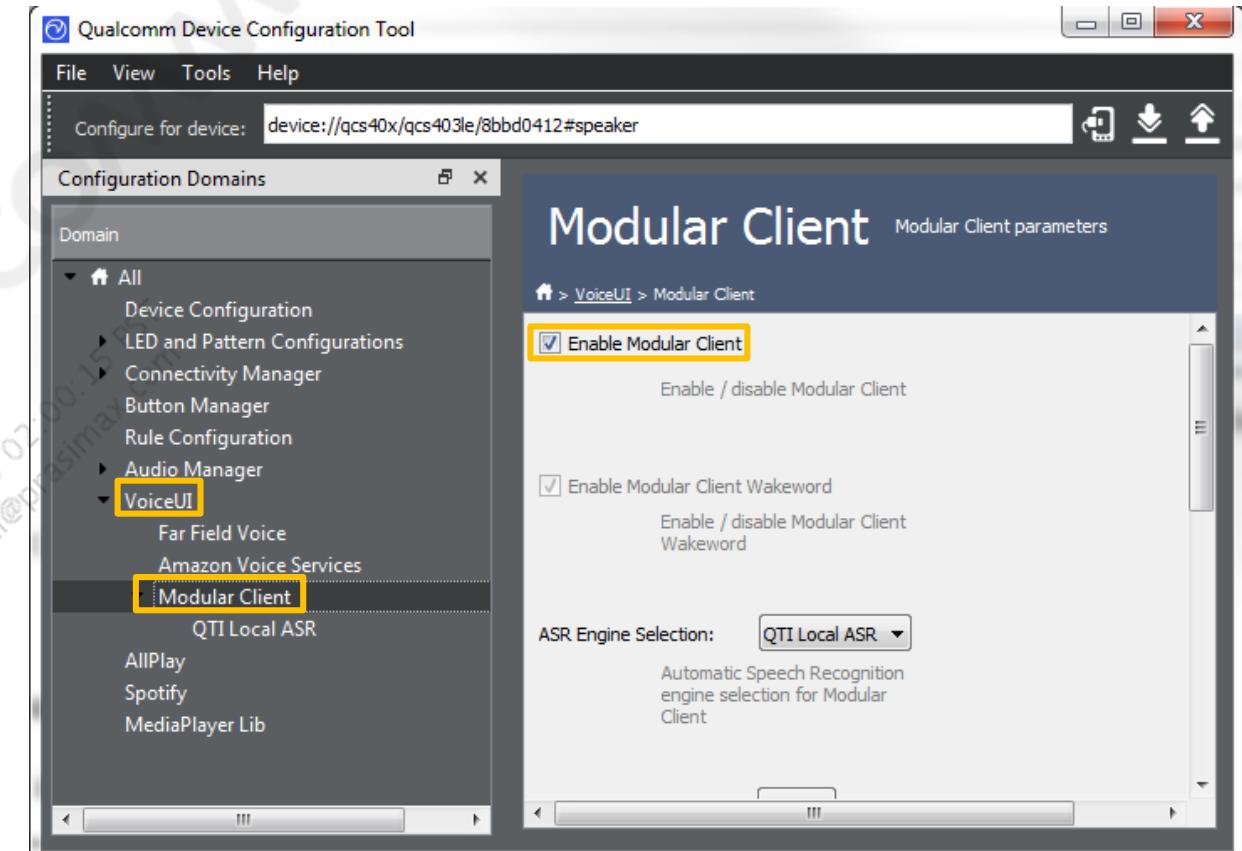
Local ASR + AVS Test – Prerequisite

- AVS is already enabled on [Onboarding AVS](#) chapter
- Local ASR is enabled by default function of VoiceUI
- The following below should have already been performed
 - [Writing Device Configuration](#)
 - [Haven License](#)
- The device must be configured to normal boot mode
 - Refer to: [Device Boot Configuration for SSRD – Normal boot mode](#)



Local ASR + AVS Test – Enable Local ASR Configuration

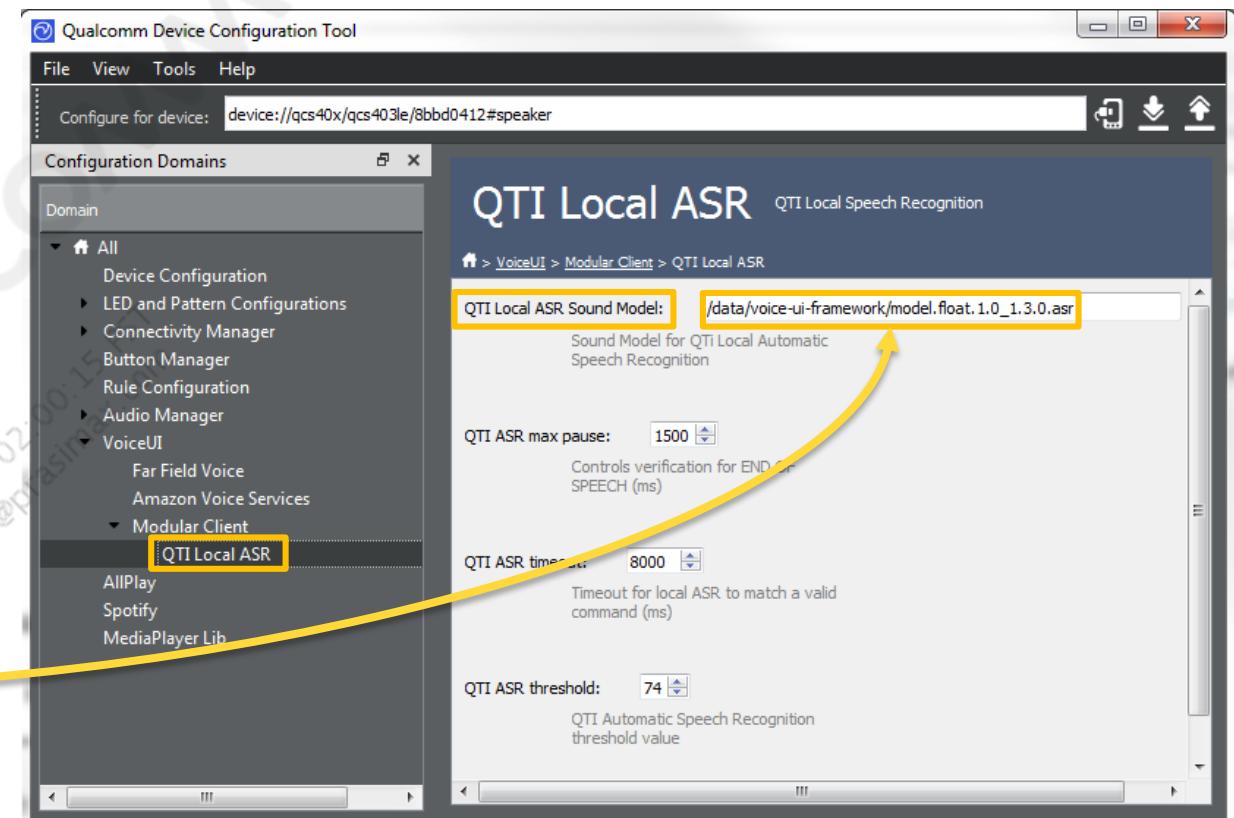
- Local ASR service is enabled in default configuration XML file, so Local ASR service would have already been enabled through the [Writing Device Configuration](#) step
- To check Local ASR service is enabled in default XML file:
 - Open Device Configuration Tool
 - Import default XML file
 - Click on ‘VoiceUI > Modular Client’ item
 - Make sure the ‘Modular Client’ is checked ON



Local ASR + AVS Test – Enable Local ASR Configuration

- In addition, make sure the QTI Local ASR Sound model file is in device
 - Click on 'VoiceUI > Modular Client > QTI Local ASR' item
 - Check 'QTI Local ASR Sound Model' file path
 - Check the file name in device

```
/data/voice-ui-framework # ls
Bedroom_Lights_OFF.mp3      Lights_OFF.mp3
Bedroom_Lights_ON.mp3       Lights_ON.mp3
Call_Ended.mp3
Device_has_been-muted.mp3
Device_has_been-unmuted.mp3
Front_Door_Closed.mp3
Front_Door_Opened.mp3
Garage_Door_Closed.mp3
Garage_Door_Opened.mp3
HeySnapdragon.uim
Incoming_call_Rejected.mp3
Incoming_call_accepted.mp3
Kitchen_Lights_ON.mp3
Kitchen_lights_OFF.mp3
/data/voice-ui-framework #
```



- If the file name on ConfigTool is not match with the file name in device, then modify the ConfigTool's file name and write again the updated configuration

Local ASR + AVS Test

- Now AVS and Local ASR are available
- Reboot device and wait until Wi-Fi is connected
- Run ADK monitor application
 - adk-message-monitor -a
- Run Local ASR monitor
 - watch -n 2 "grep -i jston_str /var/log/messages | tail -1"
- Voice activation keywords are
 - For AVS: "Alexa"
 - For Local ASR: "Hey SnapDragon"

For example, Say
"Turn on the Kitchen Light"

AdkMessage received: connectivity zigbee_set_group_on_off_state
Message Received [ms:-2126565190 Sat May 4 05:09:23 2019]
connectivity_zigbee_set_group_on_off_state {
group: "kitchen"
state: true
}
AdkMessage received: led start_pattern
Message Received [ms:-2126565184 Sat May 4 05:09:23 2019]
led_start_pattern {
pattern: 16
}
AdkMessage received: display notification
Message Received [ms:-2126565179 Sat May 4 05:09:23 2019]
display_notification {
group: "voiceui"
name: "LocalASR"
message: "LightIntent"
time_ms: "3000"
}

ADK Monitor

Local ASR
Monitor

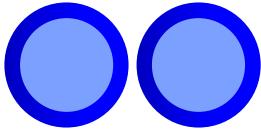
Every 2s: grep -i jston_str /var/log/messages | tail -1 2019-05-04 05:10:29
May 4 05:09:23 qc8403-som2 user.info VoiceUISampleApp::ModularClientManager::ProcessJSON -jston_str = {"slots":[{"value":"on","name":"State"}, {"name":"Room","value":"kitchen"}], "intent": "LightIntent"}

Local ASR + AVS Test

- To simply check the available Local ASR command list in current software, check the mp3 files in '/data/voice-ui-framework' folder

```
/data/voice-ui-framework # ls
Bedroom_Lights_OFF.mp3           Lights_OFF.mp3
Bedroom_Lights_ON.mp3            Lights_ON.mp3
Call_Ended.mp3                  Living_Room_Lights_OFF.mp3
Device_has_been-muted.mp3        Living_Room_Lights_ON.mp3
Device_has_been-unmuted.mp3      Thermostat_OFF.mp3
Front_Door_Closed.mp3           Thermostat_ON.mp3
Front_Door_Opened.mp3           Thermostat_temperature_has_changed.mp3
Garage_Door_Closed.mp3          Volume_Decreased.mp3
Garage_Door_Opened.mp3          Volume_Increased.mp3
HeySnapdragon.uim               Volume_changed.mp3
Incoming_call_Rejected.mp3       model.float.1.0_1.3.0.asr
Incoming_call_accepted.mp3       sm3_gmm_0703_cnn_jan08_Alexa.uim
Kitchen_Lights_ON.mp3           voiceDumpAVS
Kitchen_lights_OFF.mp3          voiceDumpLocalASR
/data/voice-ui-framework #
```

Confidential
2019-12-30 00:00:00
didi.setiadi@qcsm.com



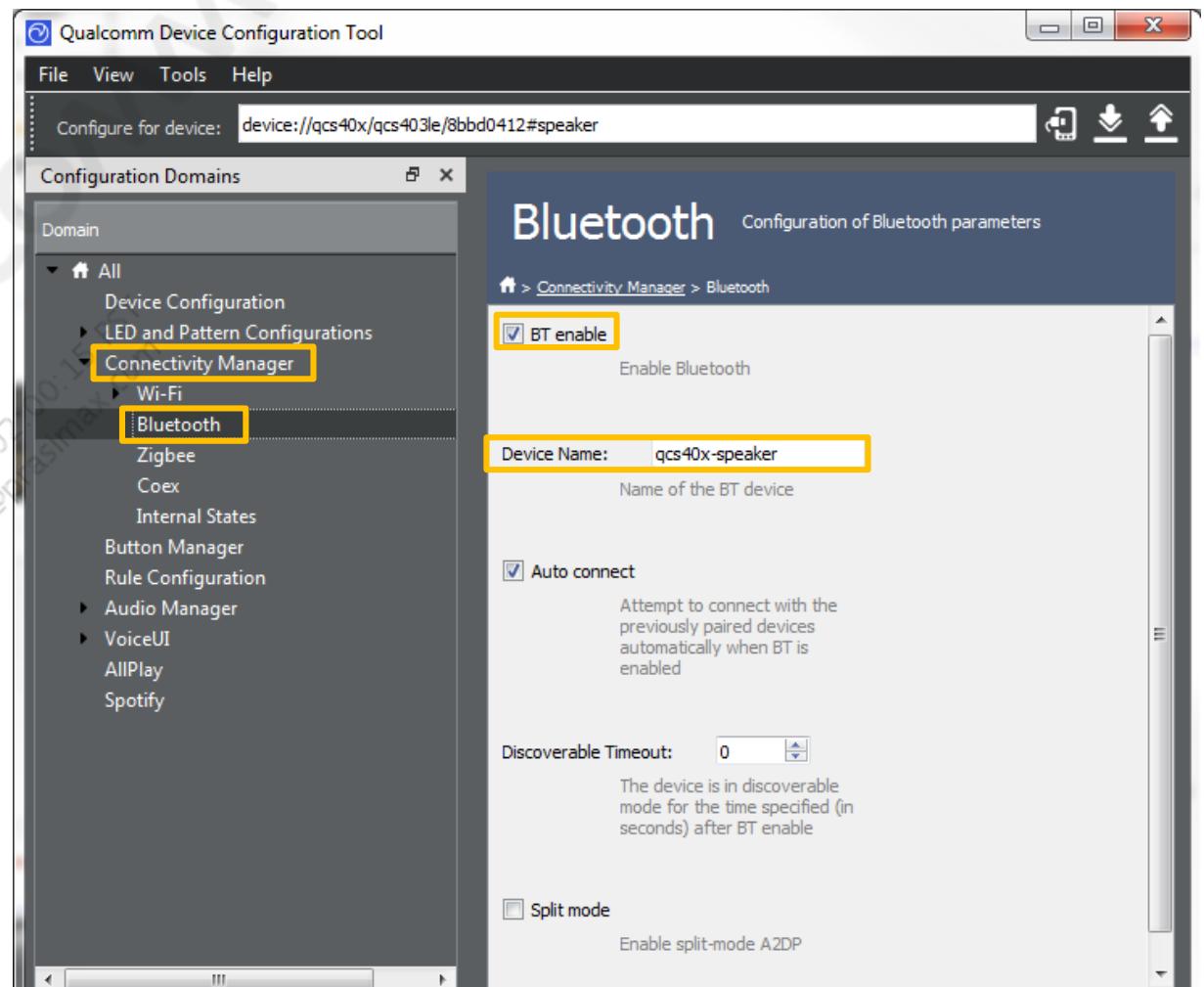
Section 8.7

Bluetooth Audio Streaming Test

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Bluetooth Audio Streaming Test

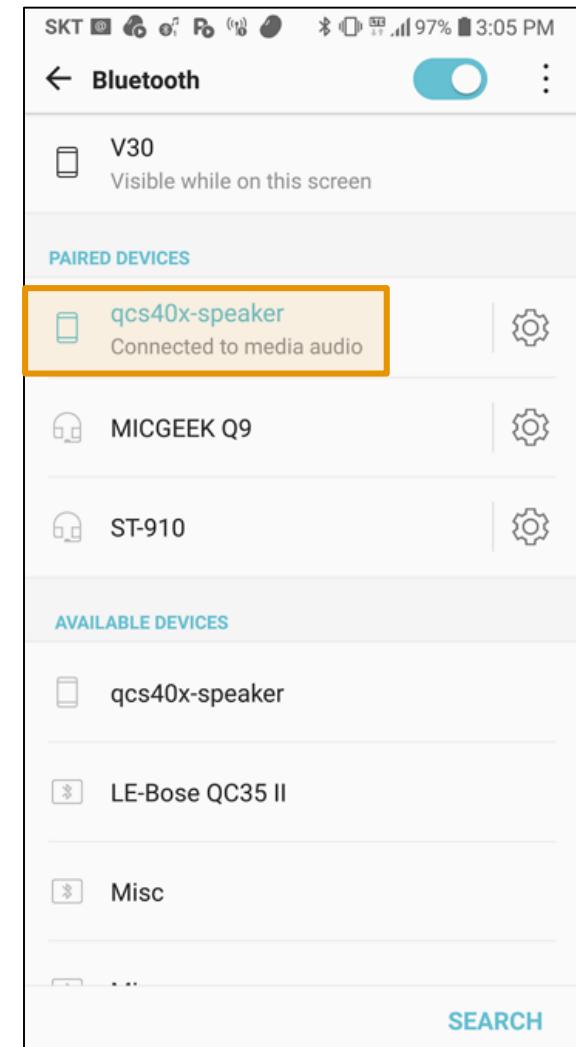
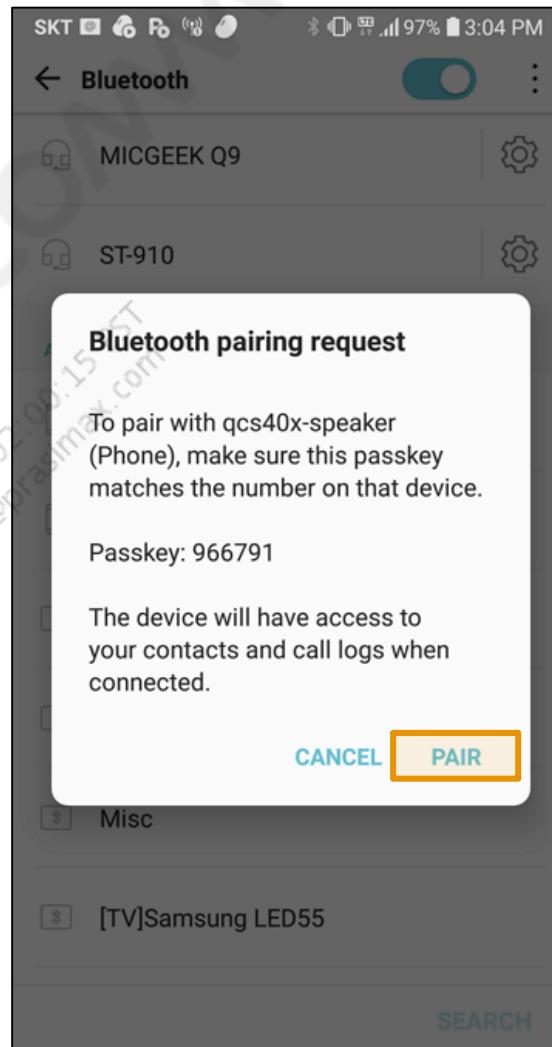
- Bluetooth feature is enabled in default configuration XML file
- To check Bluetooth feature is enabled in default XML file:
 - Open Device Configuration Tool
 - Import default XML file
 - Click on ‘Connectivity Manager > Bluetooth’ item
 - Make sure the ‘BT enable’ is check ON
- Make sure the ‘Device Name’ is set to something unique, default device name is ‘qcs40x-speaker’
- If the ‘BT enable’ is checked OFF then check ON the ‘BT enable’ and write again the updated configuration

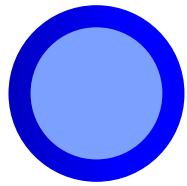


Bluetooth Audio Streaming Test

1. Open 'Bluetooth' setting on your Smart Phone
2. Search 'qcs40x-speaker' device
3. Pair and Bond with the device
 - When the 'Bluetooth paring request' window appears, just click 'PAIR' button, Passkey input is not required.
4. After connected to device, play some music with your Smart Phone
5. DONE !

Confidential
2019-12-30 02:15:15
didi.setiadi@pramana.com





Section 9

References

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

References

- [80-YB405-129] QCS40x Software User Guide
- [80-V1400-3] QUALCOMM PRODUCT SUPPORT TOOL (QPST) 2.7 USER GUIDE
- [80-NN120-1] QUALCOMM FLASH IMAGE LOADER (QFIL) USER GUIDE
- [80-YB420-4] BOOTLOADER

Qualcomm
Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com

Questions?

For additional information or to submit technical questions, go to <https://createpoint.qti.qualcomm.com>

Thank You

Qualcomm

Confidential
2019-12-30 02:00:15 PST
didi.setiadi@prasimax.com