QCA9377 FCC Certification User's Guide

Rev 1.0 20200227

Contents

1. Introduction

1.1 Purpose

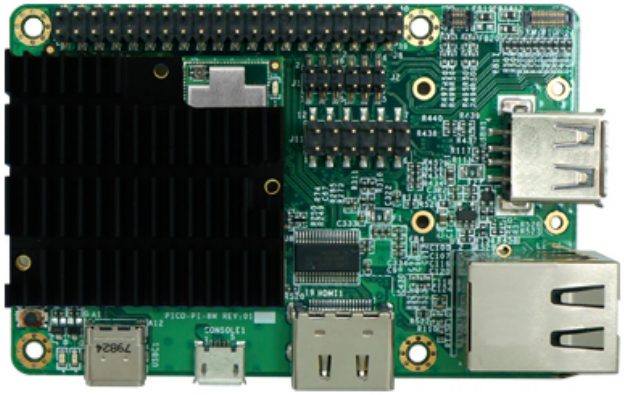
This document describes:

- how to build yocto for certification
- how to connect devices
- the steps for connect to DUT on WLAN and BT

2. Preparing for hardware device

1. Host PC used to build yocto operating system for certification.
2. TechNexion TEK3-BSW Box PC with Qualcomm QDART software.
3. TechNexion SoM with QCA9377 WiFi module.
4. Monitor(HDMI or miniDP), USB Keyboard, USB Mouse and Ethernet cable.

TEK3-BSW



TechNexion SoM + Baseboard

3. Build yocto operating system for certification

3.1 Set up Linux build environment

Method 1:

Set up build environment on host PC:

Our build environment is under ubuntu 16.04.

Install required packages:

```
$: sudo apt-get install gawk wget git git-core diffstat unzip texinfo gcc-multilib build-essential \
chrpath socat cpio python python3 python3-pip python3-pexpect \
xz-utils debianutils iputils-ping libsdl1.2-dev xterm \
language-pack-en coreutils texi2html file docbook-utils \
python-pysqlite2 help2man desktop-file-utils \
libgl1-mesa-dev libglu1-mesa-dev mercurial autoconf automake \
groff curl lzop asciidoc u-boot-tools libreoffice-writer \
sshpass ssh-askpass zip xz-utils kpartx vim screen
```

Method 2:

Download virtual machine with pre-installed Ubuntu 16.04 and packages.

ftp://ftp.technexion.net/development_resources/development_tools/vm

This virtual machine is validated to build Yocto 2.5.

Method 3:

Use the dockerfile to setup the build environment:

Install docker on ubuntu 16.04, please refer to:

https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-16-04

After fetch yocto source code, create a docker image from a dockerfile.

```
$: cd sources/meta-tn-imx-bsp/tools/container
$: docker build -t tn_ubuntu1604 .
$: docker run -it -u jenkins -v ${directory_in_host_machine}:${directory_in_docker} tn_ubuntu1604 /bin/bash
(-v: use to bind volume to the directory in host machine)
(password: jenkins)
```

3.2 Get yocto BSP

This BSP is a TechNexion release providing support NXP i.mx series processors:

```
$: mkdir ~/bin
$: curl http://commondatastorage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
$: chmod a+x ~/bin/repo
```

Download the BSP source:

```
$: PATH=${PATH}:~/bin
$: mkdir edm_yocto
$: cd edm_yocto
$: repo init -u https://github.com/TechNexion/tn-imx-yocto-manifest.git -b sumo_4.14.y_GA-next -m imx-4.14.98-2.0.1_patch.xml
$: repo sync -j8
```

3.3 Build image for TechNexion target platform

3.3.1 Configurations for setup script

Please visit TechNexion yocto github for more details:
https://github.com/TechNexion/tn-imx-yocto-manifest/tree/sumo_4.14.y_GA-next

For PICO-IMX8MQ:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-wayland MACHINE=pico-imx8mq source edm-setup-release.sh -b build-imx8mq
```

For EDM-IMX8MQ:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-wayland MACHINE=edm-imx8mq source edm-setup-release.sh -b build-imx8mq
```

For PICO-IMX8MM:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-wayland MACHINE=pico-imx8mm source edm-setup-release.sh -b build-imx8mm
```

For FLEX-IMX8MM:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-wayland MACHINE=flex-imx8mm source edm-setup-release.sh -b build-imx8mm
```

For AXON-IMX8MM:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-wayland MACHINE=axon-imx8mm source edm-setup-release.sh -b build-imx8mm
```

For XORE-IMX8MM:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-wayland
MACHINE=xore-imx8mm source edm-setup-release.sh -b build-imx8mm
```

For PICO-IMX7:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-x11
MACHINE=pico-imx7 BASEBOARD=pi source edm-setup-release.sh -b build-imx7
```

For EDM-IMX7:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-x11
MACHINE=edm-imx7 BASEBOARD=gnome source edm-setup-release.sh -b build-imx7
```

For PICO-IMX6:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-x11
MACHINE=pico-imx6 BASEBOARD=pi source edm-setup-release.sh -b build-imx6
```

For EDM-IMX6:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-x11
MACHINE=edm-imx6 BASEBOARD=fairy source edm-setup-release.sh -b build-imx6
```

For AXON-IMX6:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-x11
MACHINE=axon-imx6 source edm-setup-release.sh -b build-imx6
```

For PICO-IMX6UL/ULL:

```
$: TOKEN=SbtQ_mC4fvJRA88_9jB7 WIFI_FIRMWARE=y DISTRO=fsl-imx-x11
MACHINE=pico-imx6ul BASEBOARD=pi source edm-setup-release.sh -b build-imx6ul
```

Note: You need to read and accept the EULA.

```
Do you accept the EULA you just read? (y/n)
```

3.3.2 Build instructions

Set a specific DISTRO for FCC certification and build image:

```
$: echo "DISTRO_FEATURES_append = \" fcc\"" >>conf/local.conf
$: bitbake tn-image-fcc-qca
```

When build completes, the generated release image is under "${BUILD-TYPE}/tmp/deploy/images/${MACHINE}":

To decompress the .bz2:

```
$: bzip2 -fdk tn-image-fcc-qca-<MACHINE>.sdcard.bz2
```

Note: The special drivers and special firmwares in this image file are only used on certification and cannot be used in general mode.

3.4 Image deployment

3.4.1 Use mfgtool "uuu" to flash eMMC under Linux

Download the image deploy tool(uuu) from TechNexion FTP:
ftp://ftp.technexion.net/development_resources/development_tools/installer/imx-mfg-uuu-tool_20200212.zip
Please visit TechNexion yocto github wiki for more details:
https://github.com/TechNexion/u-boot-tn-imx/wiki/Use-mfgtool-%22uuu%22-to-flash-eMMC

1. Install required packages for executing mfgtool uuu:

```
$: sudo apt-get install libusb-1.0.0-dev libzip-dev libbz2-dev
```

2. First, Set the boot jumpers to Boot from serial download. Then, attach a USB Type-C peripheral cable to the board, and the other end to the host PC. Boot configuration settings:
https://www.technexion.com/support/knowledgebase/boot-configuration-settings-for-pico-baseboards/

3. Excute uuu to start flashing process.

Different instructions to flash image into eMMC:

For iMX6UL/6ULL:

```
$: sudo uuu/linux64/uuu -b emmc_imx6ul_img imx6ul/imx6ul-SPL imx6ul/imx6ul-u-boot.img tn-image-fcc-qca-pico-imx6ul.sdcard
```

For iMX6DL and iMX6Q:

```
$: sudo uuu/linux64/uuu -b emmc_imx6_img imx6/imx6-SPL imx6/imx6-u-boot.img tn-image-fcc-qca-pico-imx6.sdcard
```

For iMX7D:

```
$: sudo uuu/linux64/uuu -b emmc_imx7_img imx7/imx7-SPL imx7/imx7-u-boot.img tn-image-fcc-qca-pico-imx7.sdcard
```

For iMX8MM:

```
$: sudo uuu/linux64/uuu -b emmc_img imx8mm/pico imx8mm-flash.bin tn-image-fcc-qca-pico-imx8mm.sdcard
```

For iMX8M(Q):

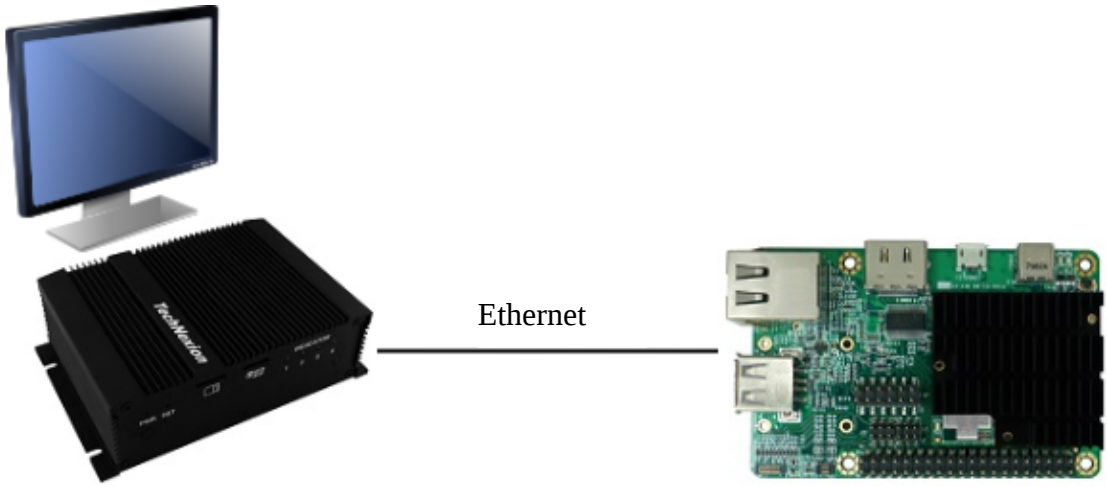
```
$: sudo uuu/linux64/uuu -b emmc_img imx8mq/pico imx8mq-flash.bin tn-image-fcc-qca-pico-imx8mq.sdcard
```

4. Once the flash process completes. Set the boot jumpers to Boot from eMMC and reset the board to boot from eMMC..

4. QDART software operating instructions

4.1 Connect a Box PC and DUT using an Ethernet network

Qualcomm QDART software is already installed on TEK3-BSW Box PC.



1. Plug in an Ethernet cable and power supply on the DUT.

2. Connect a monitor, keyboard, mouse, Ethernet(LAN1) and power supply to TEK3-BSW Box PC.

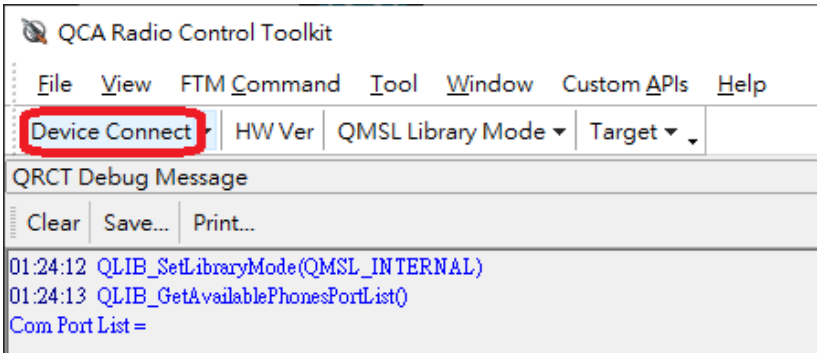
Note: Please replug the DUT power if there is an operation error or network disconnect.

4.2 Using QDART

4.2.1 QDART preaction

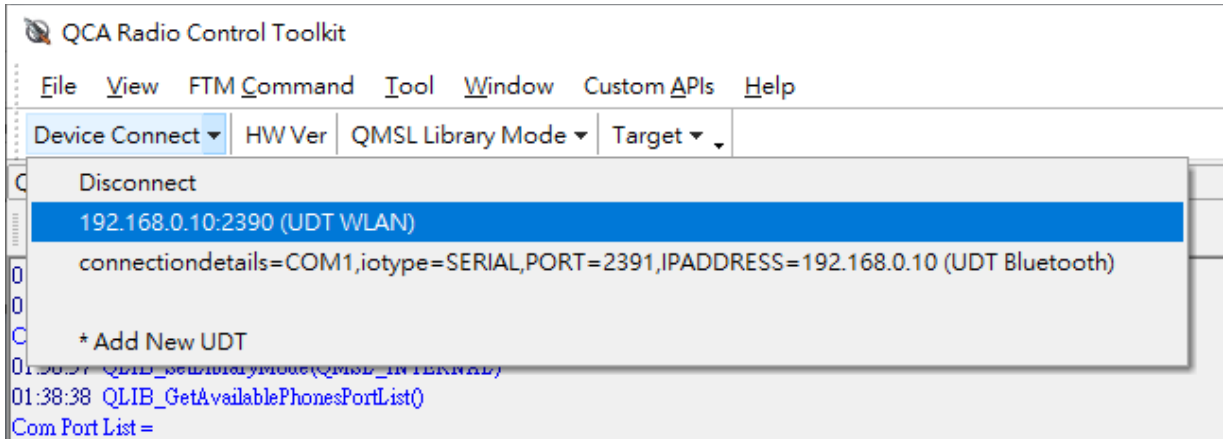
1. Open QRCT.

2. First click on the “Device Connect” to make the connection smoother.

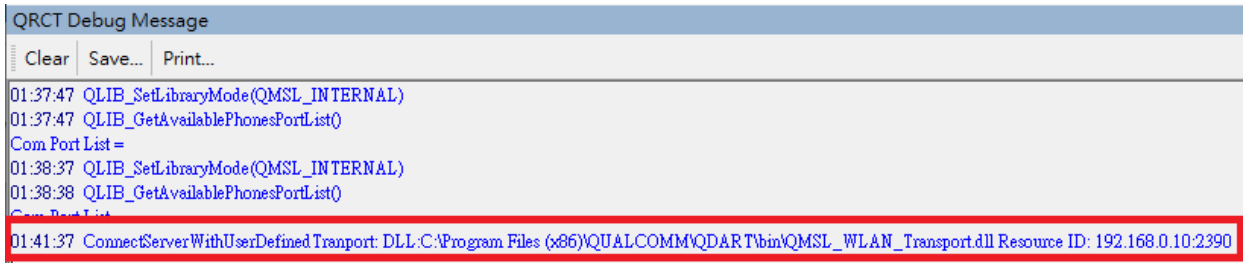


4.2.2 WLAN Certification

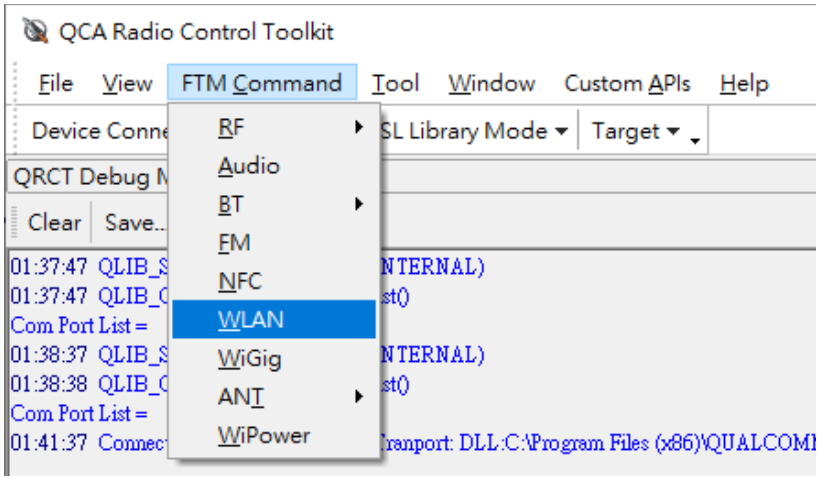
1. Connect to DUT using remote device.



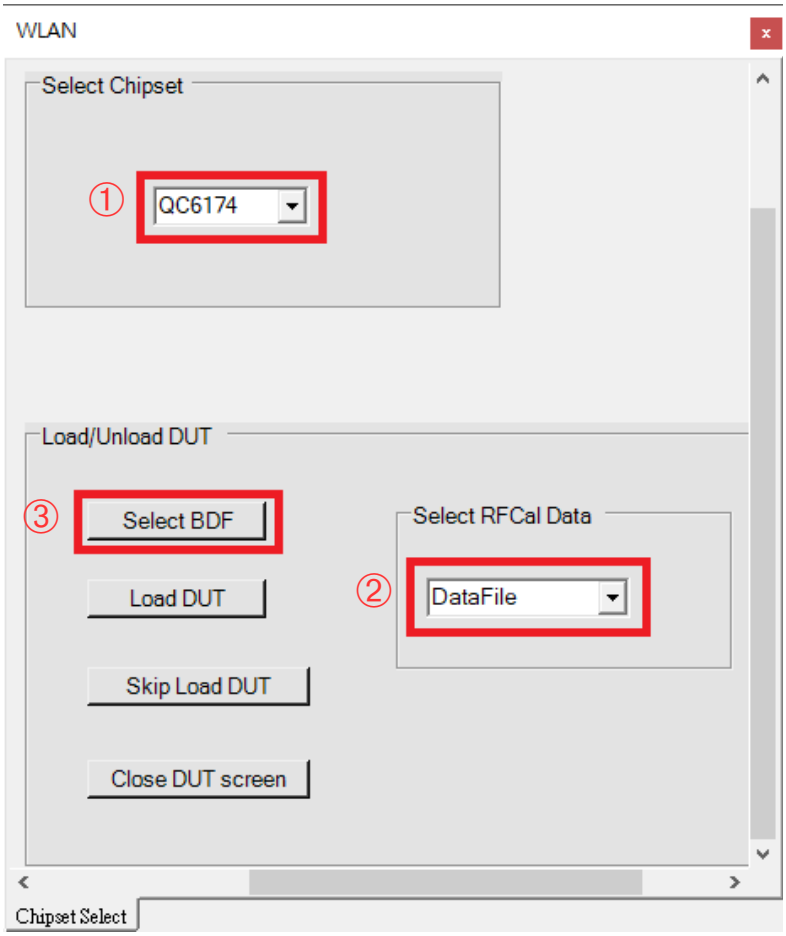
2. If successful you can see the connection message in “QRCT Debug Message”.



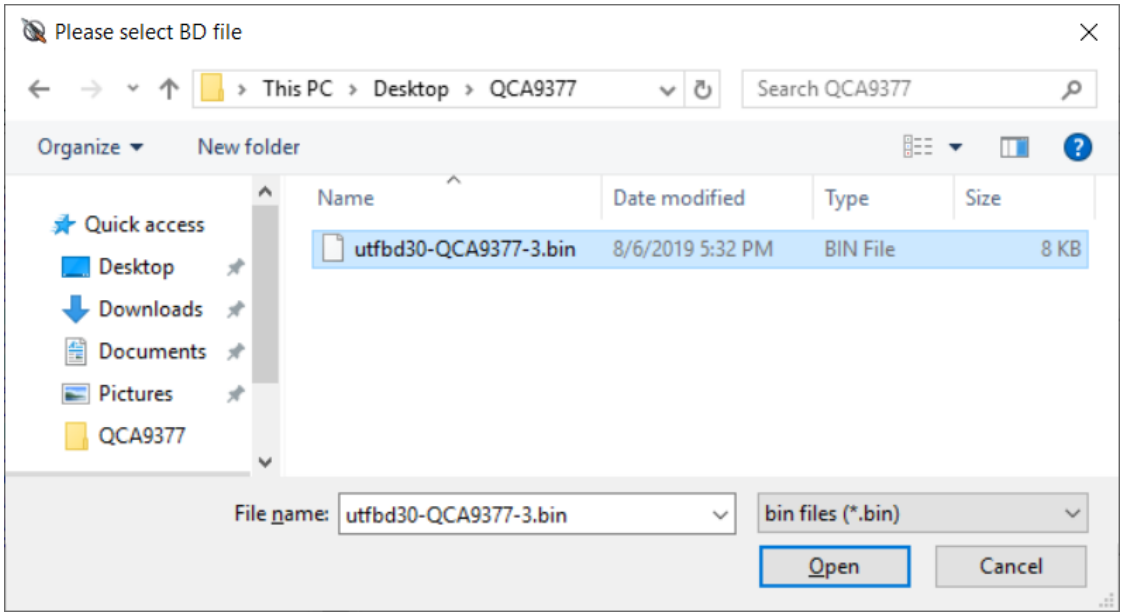
3. Open WLAN window in “FTM Command”.



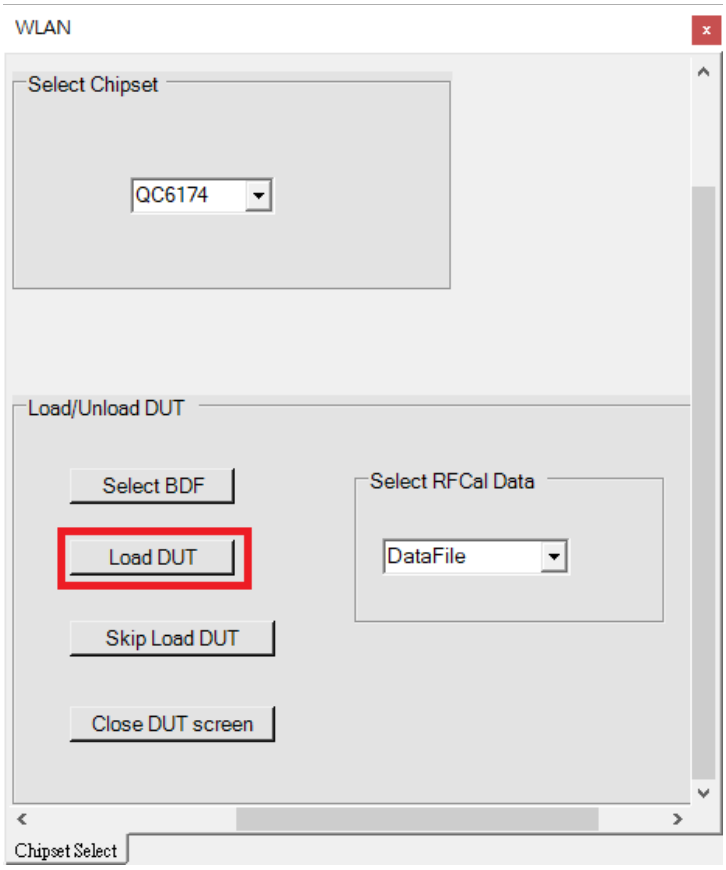
4. Select Chipset and Board Data File(BDF).



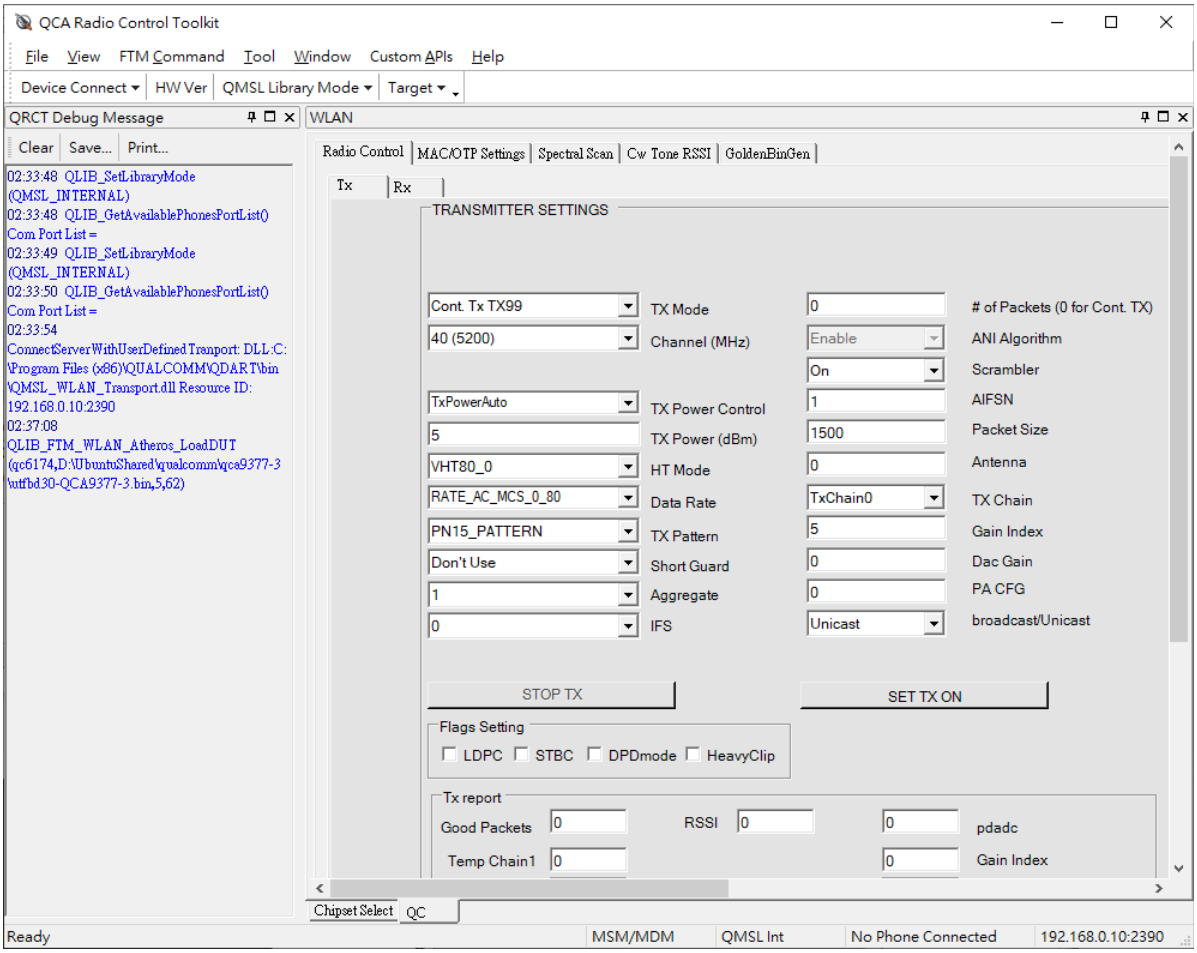
5. Select QCA9377 BDF.



6. Load DUT.

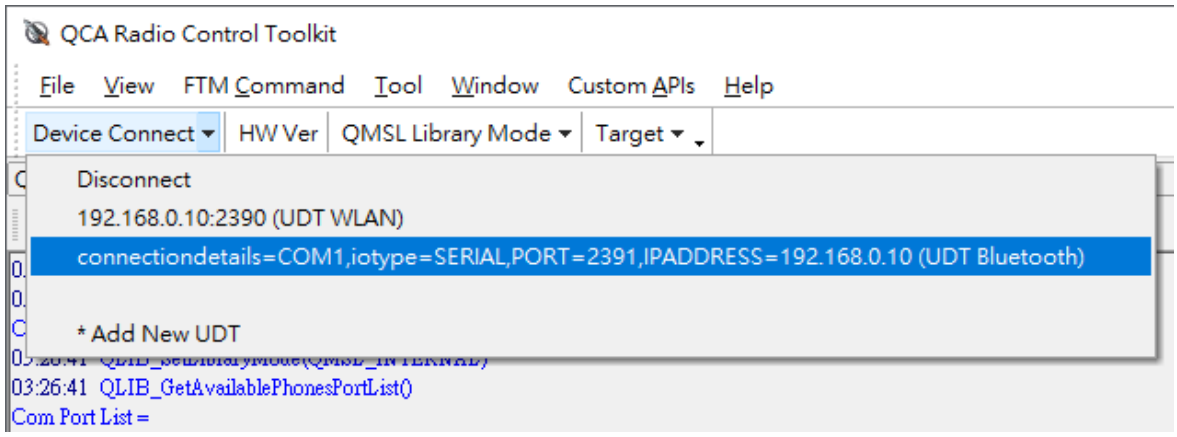


7. Start certification.

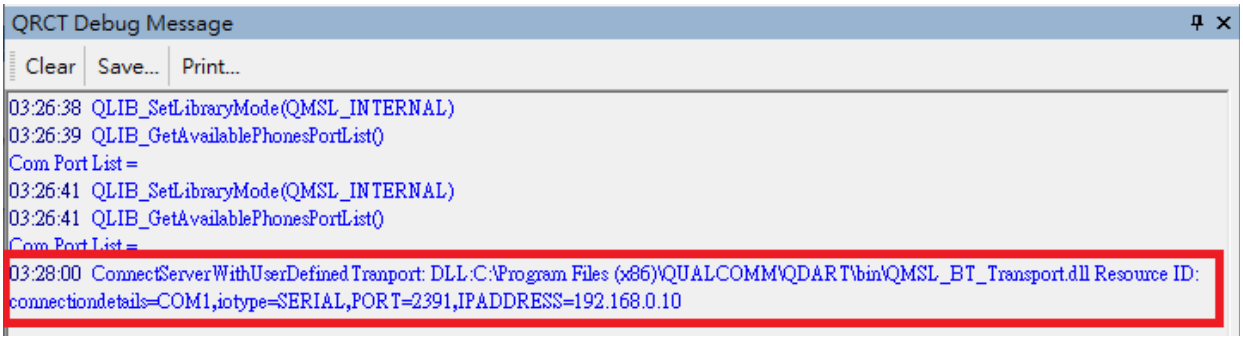


4.2.3 Bluetooth Certification

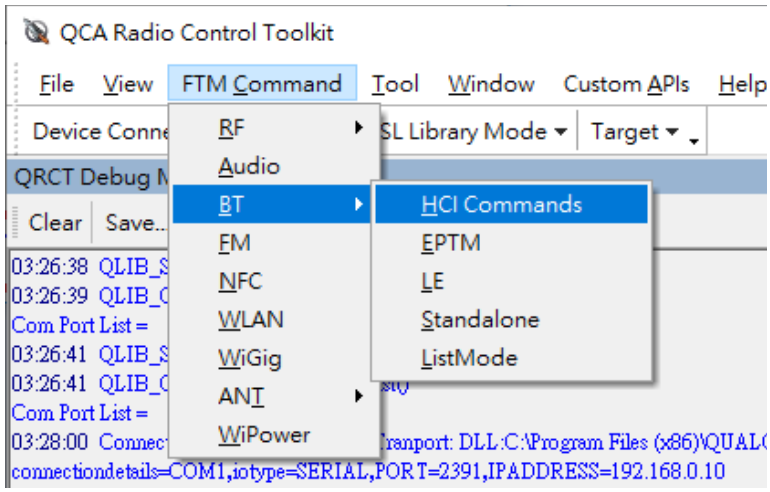
1. Connect to DUT using remote device.



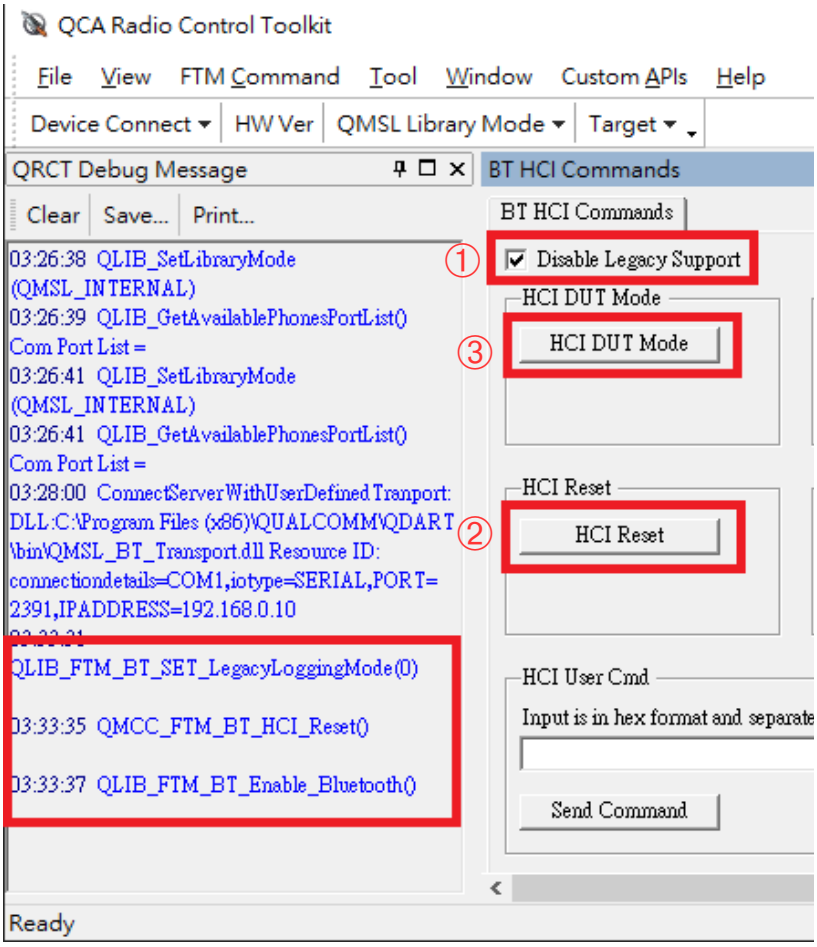
2. If successful you can see the connection message in “QRCT Debug Message”.



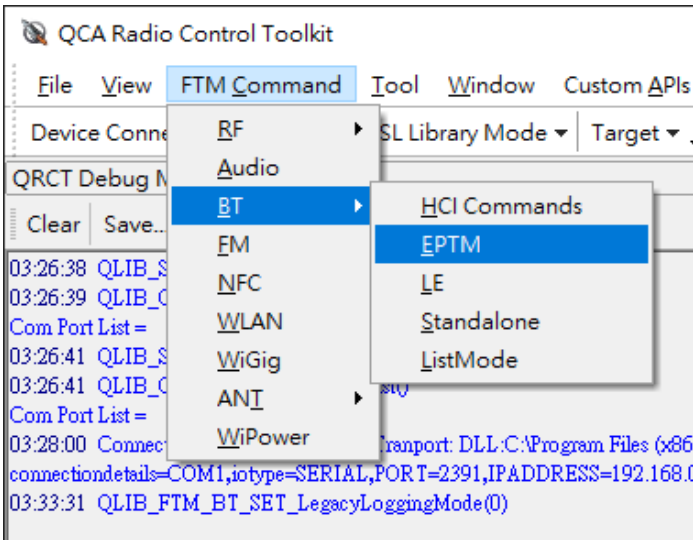
3. Open HCI Commands window in “FTM Command → BT”.



4. Click Disable Legacy Support → HCI Reset → HCI DUT Mode.



5. Open EPTM window in “FTM Command → BT”.



6. Start certification.