UM11490

Feature Configuration Guide for NXP-based Wireless Modules on i.MX 8M Quad EVK

Rev. 2 — 29 January 2021 User manual

Document information

Information	Content
Keywords	NXP-based wireless modules, Wi-Fi features, Bluetooth features, guidelines, driver debug information
Abstract	Details the Wi-Fi and Bluetooth features and configurations for NXP-based wireless modules on i.MX 8M Quad EVK

Revision history

Rev	Date	Description
v.1	20200929	Initial version
v.2	20210129	**Modifications:** • Extended the scope to 88W8997-based wireless modules • Section 1 "About this document": updated • Section 2.2.4 "Set up udhcp server": added • Section 2.6 "Configure and test Wi-Fi direct": updated • Section 2.8.3 "Set the RF bandwidth": added • Section 2.8.4 "Set the RF channel": added • Section 2.8.5 "Set Tx power": added • Section 2.8.6 "Set Tx continuous mode": added • Section 2.8.7 "Set Tx frame": added • Section 2.10 "Set the Wi-Fi device in suspend state": added • Section 4.3 "Hands-free profile (HFP)": added • Section 4.9 "RF test mode": updated

1 About this document

1.1 Purpose and scope

This document details the Wi-Fi and Bluetooth features and configurations for NXP-based wireless modules on i.MX 8M Quad EVK. The Wi-Fi features include scanning for nearby access points, connecting to an access point, configuring the device as an access point, Wi-Fi security, Wi-Fi Direct, and throughput testing using the iPerf utility. The Bluetooth features comprise scan, pair, connect to a Bluetooth or Bluetooth Low Energy (LE) device, A2DP profile, handsfree profile and Bluetooth LE device as a GATT server. Guidelines to enable the driver debug logging are also provided.

This document specifies the Wi-Fi/Bluetooth features and configurations on i.MX 8M Quad EVK with Azurewave AW-CM358MA (88W8987) and Azurewave AW-CM276MA (88W8997) wireless modules. It covers the initialization and configuration of the Wi-Fi and Bluetooth interfaces. The same instructions apply to other i.MX 8 families and NXP-based wireless modules.

The Wi-Fi features demonstrated in this document are configured with the open source supplicant and the Linux utilities.

The host platform setup and build of the Wi-Fi/Bluetooth software for the platform interfaces is not covered in this document. The Release Notes included in NXP driver software package provides the list of the features supported by NXP-based wireless module.

The user should first read the reference document *Getting Started with NXP-based Wi-Fi modules on i.MX 8M Quad EVK Running Linux OS (UM11483)* before using this user manual, as it details:

- The board support package build configurations
- The setup for the hardware interconnections
- How to enable the Wi-Fi solutions

1.2 References

Table 1. References

Reference type	Description
User manual	NXP - UM11483 - Getting Started with NXP-based Wireless Modules on i.MX 8M Quad EVK Running Linux OS (link)
Desktop tool	Nordic Semiconductor - nRF Connect for Desktop (link)
Specification	Bluetooth Special Interest Group - Bluetooth Core Specification v5.2 (link)

2 Wi-Fi features and configurations

2.1 Scan the visible Access Points

This section describes the scanning of the visible Access Points using the NXP-based wireless module. Linux operating system provides the standard utilities to show/ manipulate wireless devices and their configuration.

2.1.1 Using `iw` command

`iw` command is one of the methods used to scan visible access points.

Execute the command to start the scan

```
root@imx8mqevk:~# iw dev mlan0 scan
```

Command output example:

```
[11204.740176] wlan: mlan0 START SCAN
[11207.573646] wlan: SCAN COMPLETED: scanned AP count=1
BSS 44:82:e5:10:2c:38(on mlan0)
    TSF: 11207445411 usec (0d, 03:06:47)
    freq: 2427
    beacon interval: 100 TUs
    capability: ESS Privacy ShortPreamble ShortSlotTime RadioMeasure
 (0x1431)
    signal: -56.00 dBm
    last seen: 0 ms ago
    SSID: destin
    Supported rates: 1.0* 2.0* 5.5* 6.0 9.0 11.0* 12.0 18.0
    DS Parameter set: channel 4
    Country: in   Environment: Indoor/Outdoor
                  Channels [1 - 13] @ 21 dBm
    TPC report: TX power: 20 dBm
    ERP: NonERP_Present Use_Protection
    RSN:     * Version: 1
             * Group cipher: TKIP
             * Pairwise ciphers: CCMP TKIP
             * Authentication suites: PSK
             * Capabilities: 1-PTKSA-RC 1-GTKSA-RC (0x0000)
    WPA:     * Version: 1
             * Group cipher: TKIP
             * Pairwise ciphers: CCMP TKIP
             * Authentication suites: PSK
    WMM:     * Parameter version 1
             * BE: CW 15-1023, AIFSN 3
             * BK: CW 15-1023, AIFSN 7
             * VI: CW 7-15, AIFSN 2, TXOP 3008 usec
             * VO: CW 3-7, AIFSN 2, TXOP 1504 usec
    Extended supported rates: 24.0 36.0 48.0 54.0
    BSS Load:
              * station count: 4
              * channel utilisation: 52/255
              * available admission capacity: 0 [*32us]
    HT capabilities:
              Capabilities: 0x11ed
                  RX LDPC
                  HT20
                  SM Power Save disabled
                  RX HT20 SGI
                  RX HT40 SGI
```

```
                TX STBC
                RX STBC 1-stream
                Max AMSDU length: 3839 bytes
                DSSS/CCK HT40
        Maximum RX AMPDU length 65535 bytes (exponent: 0x003)
        Minimum RX AMPDU time spacing: 4 usec (0x05)
        HT TX/RX MCS rate indexes supported: 0-15
    HT operation:
              * primary channel: 4
              * secondary channel offset: no secondary
              * STA channel width: 20 MHz
              * RIFS: 0
              * HT protection: nonmember
              * non-GF present: 1
              * OBSS non-GF present: 1
    * dual beacon: 0
              * dual CTS protection: 0
              * STBC beacon: 0
              * L-SIG TXOP Prot: 0
              * PCO active: 0
              * PCO phase: 0
     Overlapping BSS scan params:
               * passive dwell: 20 TUs
               * active dwell: 10 TUs
               * channel width trigger scan interval: 300 s
               * scan passive total per channel: 200 TUs
               * scan active total per channel: 20 TUs
               * BSS width channel transition delay factor: 5
               * OBSS Scan Activity Threshold: 0.25 %
     Extended capabilities:
               * HT Information Exchange Supported
               * BSS Transition
```

2.1.2 Using `wpa_supplicant` and `wpa_cli` commands

You can also use `wpa_supplicant` and `wpa_cli` to scan the visible access points.

Edit the configuration file:

```
root@imx8mqevk:~# nano /etc/wpa_supplicant.conf
```

Content of the configuration file:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1
network={
      key_mgmt=NONE
}
```

Start `wpa_supplicant` in the background

```
root@imx8mqevk:~# wpa_supplicant -B -i mlan0 -c /etc/
wpa_supplicant.conf
```

Start `wpa_cli` for the scan

```
root@imx8mqevk:~# wpa_cli
```

Command output:

```
wpa_cli v2.9
    Copyright (c) 2004-2019, Jouni Malinen <j@w1.fi> and contributors
    This software may be distributed under the terms of the BSD
 license.
    See README for more details.
    Selected interface 'mlan0'
    Interactive mode
```

Start the scanning:

```
 > scan
```

Command output example showing the scanned AP count and the SCAN COMPLETE event:

```
OK
[  253.835605] wlan: mlan0 START SCAN
 <3>CTRL-EVENT-SCAN-STARTED
 > [  258.910760] wlan: SCAN COMPLETED: scanned AP count=1
 <3>CTRL-EVENT-SCAN-RESULTS
 <3>CTRL-EVENT-NETWORK-NOT-FOUND
```

Get the scan results:

```
> scan_results
```

Command output example showing one SSID:

```
bssid / frequency / signal level / flags / ssid
44:82:e5:10:2c:38  2442  -93  [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]
  destin
```

2.2 Configure and start the Access Point

This section shows how to configure and start the Access Point from the NXP-based wireless module. Linux operating system provides the hostapd user space utility for the access point and authentication servers.

The configurations for hostapd are stored in the *hostapd.conf* file located in */etc/* directory.

2.2.1 Get the hostpad version

Execute the command:

```
root@imx8mqevk:~# hostapd -v
```

Command output example:

```
hostapd v2.9
User space daemon for IEEE 802.11 AP management,
IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticator
Copyright (c) 2002-2019, Jouni Malinen <j@w1.fi> and contributors
```

2.2.2 Configure the 2.4 GHz Access Point

Edit the configuration file:

```
root@imx8mqevk:~# nano /etc/hostapd-2.4g.conf
```

Parameter values in the configuration file:

```
interface=uap0
hw_mode=g
channel=0
country_code=US
ssid=NXP_Demo
ieee80211n=1
```

2.2.3 Configure the 5 GHz Access Point

Edit the configuration file:

```
root@imx8mqevk:~# nano /etc/hostapd-5g.conf
```

Parameter values in the configuration file:

```
interface=uap0
hw_mode=a
channel=0
country_code=US
ssid=NXP_Demo
ieee80211n=1
```

2.2.4 Set up udhcp server

The udhcp server is used to assign the IP to the client devices that are associated with the access point.

Step 1 - Create the configuration file for udhcp server

- Create the configuration file *udhcpd.conf* in *etc* repository

```
root@imx8mqevk:~# nano /etc/udhcpd.conf
```

- Add the following content to *udhcpd.conf* file

```
start 192.168.1.10
end 192.168.1.20
interface uap0
```

Step 2 - Create udhcp client lease database file

udhcp server uses the database file to store all the leases that is has assigned.

- Create the lease database file *udhcpd.leases* in */var/lib/misc/* repository so udhcp server can start
- The newly created file does not require any content.

```
root@imx8mqevk:~# touch /var/lib/misc/udhcpd.leases
```

2.2.5 Start the Access Point

Run the following command to start the Access Point:

```
root@imx8mqevk:~# killall wpa_supplicant
root@imx8mqevk:~# killall hostapd
root@imx8mqevk:~# ifconfig uap0 192.168.1.2 netmask 255.255.255.0 up
root@imx8mqevk:~# route add default gw 192.168.1.1
```

Start udhcp server to assign the IP address to the client devices that are connected to the access point

```
root@imx8mqevk:~# udhcpd /etc/udhcpd.conf
```

Command to start the 2.4 GHz Access Point:

```
root@imx8mqevk:~# hostapd /etc/hostapd-2.4g.conf
```

Command output example:

```
    Configuration file: /etc/hostapd-2.4g.conf
    rfkill: Cannot open RFKILL control device
uap0: interface state UNINITIALIZED->COUNTRY_UPDATE
    ACS: Automatic channel selection started, this may take a bit
    [ 9618.998539] wlan: SCAN COMPLETED: scanned AP count=1
uap0: interface state COUNTRY_UPDATE->ACS
uap0: ACS-STARTED
    [ 9619.479337] wlan: SCAN COMPLETED: scanned AP count=1
    [ 9619.959539] wlan: SCAN COMPLETED: scanned AP count=0
    [ 9620.439382] wlan: SCAN COMPLETED: scanned AP count=0
    [ 9620.919360] wlan: SCAN COMPLETED: scanned AP count=1
uap0: ACS-COMPLETED freq=2412 channel=1[ 9620.933098] wlan: Starting
 AP
    Using interface uap0 with hwaddr 00:50:43:24:84:c4 and ssid
 "NX[ 9620.939164] Get ht_cap from beacon ies: 0xc
    P_Demo"
    [ 9620.960100] wlan: AP started
    [ 9620.964230] Set AC=3, txop=47 cwmin=3, cwmax=7 aifs=1
    [ 9620.971787] Set AC=2, txop=94 cwmin=7, cwmax=15 aifs=1
    [ 9620.979077] Set AC=0, txop=0 cwmin=15, cwmax=63 aifs=3
    [ 9620.986466] Set AC=1, txop=0 cwmin=15, cwmax=1023 aifs=7
uap0: interface state ACS->ENABLED
uap0: AP-ENABLED
```

Command to start the 5 GHz Access Point:

```
root@imx8mqevk:~# hostapd /etc/hostapd-5g.conf
```

Command output example:

```
    Configuration file: /etc/hostapd-5g.conf
    rfkill: Cannot open RFKILL control device
uap0: interface state UNINITIALIZED->COUNTRY_UPDATE
    ACS: Automatic channel selection started, this may take a bit
    [ 9816.085716] wlan: SCAN COMPLETED: scanned AP count=0
uap0: interface state COUNTRY_UPDATE->ACS
uap0: ACS-STARTED
    [ 9816.483935] wlan: SCAN COMPLETED: scanned AP count=0
    [ 9816.883357] wlan: SCAN COMPLETED: scanned AP count=0
    [ 9817.279225] wlan: SCAN COMPLETED: scanned AP count=0
    [ 9817.675171] wlan: SCAN COMPLETED: scanned AP count=0
uap0: ACS-COMPLETED freq=5180 channel=36[ 9817.686492] wlan: Starting
 AP
    Using interface uap0 with hwaddr 00:50:43:24:84:c4 and ssid
 "NX[ 9817.692595] Get ht_cap from beacon ies: 0xc
    P_Demo"
    [ 9817.702468] fw doesn't support 11ax
    [ 9817.717811] wlan: AP started
    [ 9817.722028] Set AC=3, txop=47 cwmin=3, cwmax=7 aifs=1
    [ 9817.729606] Set AC=2, txop=94 cwmin=7, cwmax=15 aifs=1
    [ 9817.736887] Set AC=0, txop=0 cwmin=15, cwmax=63 aifs=3
    [ 9817.744417] Set AC=1, txop=0 cwmin=15, cwmax=1023 aifs=7
uap0: interface state ACS->ENABLED
uap0: AP-ENABLED
```

2.3 Connect with the Access Point

This section shows how to define the configurations for wpa_supplicant to connect with the Access Point using NXP-based wireless module. Linux operating system provides the wpa_supplicant user space utility to connect to the access point using IEEE 802.1X.

2.3.1 Create the configuration file

Create and edit *wpa_supplicant.conf* configuration file in *etc* directory.

```
root@imx8mqevk:~# nano /etc/wpa_supplicant.conf
```

Configuration file content

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1
network={
      ssid="NXP_Demo"
      key_mgmt=NONE
}
```

2.3.2 Get wpa_supplicant version

Command to get wpa_supplicant version

```
root@imx8mqevk:~# wpa_supplicant -v
```

Command output example:

```
wpa_supplicant v2.9
Copyright (c) 2003-2019, Jouni Malinen <j@w1.fi> and contributors
```

2.3.3 Use wpa_supplicant to connect with the AP

Connect the device with the Access Point:

```
root@imx8mqevk:~# killall wpa_supplicant
root@imx8mqevk:~# killall hostapd
root@imx8mqevk:~# wpa_supplicant -i mlan0 -Dnl80211 -c /etc/
wpa_supplicant.conf
```

Command output example:

```
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL contro[25602.796098] IPv6:
 ADDRCONF(NETDEV_UP): mlan0: link is not ready
    [25602.922052] wlan: mlan0 START SCAN
    [25607.566402] wlan: SCAN COMPLETED: scanned AP count=18
mlan0: Trying to associate with 1a:2d:8c:55:73:56 (SSID='NXP_Demo'
 freq=2427 MHz)
    [25607.598754] wlan: Connected to bssid 1a:XX:XX:XX:73:56
 successfully
mlan0: Associated with 1a:2d:8c:5[25607.606587] IPv6:
 ADDRCONF(NETDEV_CHANGE): mlan0: link becomes ready
    5:73:56
mlan0: CTRL-EVENT-CONNECTED - Connection to 1a:2d:8c:55:73:56
 completed [id=0 id_str=]
```

2.4 Wi-Fi security

This section describe the Wi-Fi security features including the WPA, WPA2 and WPA3 SAE security modes. Both the configuration and setup use the open source supplicants.

2.4.1 Configure WPA for the AP using the open source supplicant

This section describes the hostapd configuration used to configure WPA security and start the Access Point using the open source supplicant.

Create the configuration file

Configure WPA for open source supplicant:

```
root@imx8mqevk:~# nano /etc/hostapd-wpa.conf
```

Configuration file content:

```
interface=uap0
hw_mode=g
channel=0
country_code=US
ssid=NXP_Demo
auth_algs=1
ieee80211n=1
wpa=1
rsn_pairwise=CCMP
wpa_passphrase=123456789
```

Start the Access Point

```
root@imx8mqevk:~# killall wpa_supplicant
root@imx8mqevk:~# killall hostapd
root@imx8mqevk:~# ifconfig uap0 192.168.1.2 netmask 255.255.255.0 up
root@imx8mqevk:~# route add default gw 192.168.1.1
root@imx8mqevk:~# hostapd /etc/hostapd-wpa.conf
```

Command output example:

```
    Configuration file: /etc/hostapd-wpa.conf
    rfkill: Cannot open RFKILL control device
uap0: interface state UNINITIALIZED->COUNTRY_UPDATE
    ACS: Automatic channel selection started, this may take a bit
    [19645.656230] wlan: SCAN COMPLETED: scanned AP count=1
uap0: interface state COUNTRY_UPDATE->ACS
uap0: ACS-STARTED
    [19646.137152] wlan: SCAN COMPLETED: scanned AP count=0
    [19646.617233] wlan: SCAN COMPLETED: scanned AP count=0
    [19647.097138] wlan: SCAN COMPLETED: scanned AP count=1
    [19647.577100] wlan: SCAN COMPLETED: scanned AP count=0
uap0: ACS-COMPLETED freq=2412 channel=1
    Using interface uap0 with hwaddr 00:50:43:24:84:c4 and ssid
 "NXP_Demo"
    random: Cannot read from /dev/random: Resource temporarily
 unavai[19647.619978] wlan: Starting AP
    lable
    random: Only 0/20 bytes of strong random data available
    r[19647.628422] Get ht_cap from beacon ies: 0xc
    andom: Not enough entropy pool available for secure operations
    WPA: Not enough entropy in random pool for secure operations -
 up[19647.647054] wlan: AP started
    date keys later when the first station connects
    [19647.654865] Set AC=3, txop=47 cwmin=3, cwmax=7 aifs=1
    [19647.663319] Set AC=2, txop=94 cwmin=7, cwmax=15 aifs=1
    [19647.670708] Set AC=0, txop=0 cwmin=15, cwmax=63 aifs=3
    [19647.678105] Set AC=1, txop=0 cwmin=15, cwmax=1023 aifs=7
uap0: interface state ACS->ENABLED
uap0: AP-ENABLED
```

2.4.2 Configure WPA2 for the AP using open source supplicant

This section shows the hostapd configuration used to configure WPA2 security and start the Access Point using open source supplicant.

Create the configuration file

Configure WPA2 for open source supplicant:

```
root@imx8mqevk:~# nano /etc/hostapd-wpa2.conf
```

Configuration file content:

```
interface=uap0
hw_mode=g
channel=0
country_code=US
ssid=NXP_Demo
auth_algs=1
ieee80211n=1
wpa=2
wpa_key_mgmt=WPA-PSK
rsn_pairwise=CCMP
wpa_passphrase=123456789
```

Start the Access Point

Run the following command to start the Access Point:

```
root@imx8mqevk:~# killall wpa_supplicant
root@imx8mqevk:~# killall hostapd
root@imx8mqevk:~# ifconfig uap0 192.168.1.2 netmask 255.255.255.0 up
root@imx8mqevk:~# route add default gw 192.168.1.1
root@imx8mqevk:~# hostapd /etc/hostapd-wpa2.conf
```

Command output example:

```
Configuration file: /etc/hostapd-wpa2.conf
rfkill: Cannot open RFKILL control device
uap0: interface state UNINITIALIZED->COUNTRY_UPDATE
ACS: Automatic channel selection started, this may take a bit
[20039.122775] wlan: SCAN COMPLETED: scanned AP count=0
uap0: interface state COUNTRY_UPDATE->ACS
uap0: ACS-STARTED
[20039.603743] wlan: SCAN COMPLETED: scanned AP count=0
[20040.087711] wlan: SCAN COMPLETED: scanned AP count=0
[20040.571527] wlan: SCAN COMPLETED: scanned AP count=0
[20041.051179] wlan: SCAN COMPLETED: scanned AP count=0
uap0: ACS-COMPLETED freq=2412 channel=1
Using interface uap0 with hwaddr 00:50:43:24:84:c4 and ssid "NXP_Demo"
random: Cannot read from /dev/random: Resource temporarily
 unavai[20041.091736] wlan: Starting AP
lable
random: Only 0/20 bytes of strong random data available
r[20041.100179] Get ht_cap from beacon ies: 0xc
andom: Not enough entropy pool available for secure operations
WPA: Not enough entropy in random pool for secure operations - update
 keys later when the first station connects
[20041.120927] wlan: AP started
[20041.130905] Set AC=3, txop=47 cwmin=3, cwmax=7 aifs=1
[20041.138053] Set AC=2, txop=94 cwmin=7, cwmax=15 aifs=1
[20041.145411] Set AC=0, txop=0 cwmin=15, cwmax=63 aifs=3
[20041.152528] Set AC=1, txop=0 cwmin=15, cwmax=1023 aifs=7
uap0: interface state ACS->ENABLED
    uap0: AP-ENABLED
```

2.4.3 WPA3 security

WPA3 is the next generation of Wi-Fi security with new capabilities that enhance the Wi-Fi protection in personal and enterprise networks. Built on the widely adopted WPA2™, WPA3 adds new features to simplify Wi-Fi security, enable more robust authentication, and deliver increased cryptographic strength for highly sensitive data markets.

All WPA3 networks use the latest security methods, disallow outdated legacy protocols, and require the use of Protected Management Frames (PMF) to maintain resiliency of mission critical networks.

This section describes the configurations for WPA3 SAE and SAE transition mode (backward compatibility with WPA2-PSK) using hostapd and wpa_supplicant for Access Point and Station. To test WPA3 security, load the drivers with the module parameter `host_mlme` set to `1` (`host_mlme=1`) to enable the host mlme support.

Configure WPA3 SAE mode for the Access Point

Create the configuration file:

```
root@imx8mqevk:~# nano /etc/hostapd-wpa3-sae.conf
```

Configuration file content:

```
interface=uap0
driver=nl80211
ssid=NXP_Demo
hw_mode=g
channel=6
wmm_enabled=1
ieee80211n=1
auth_algs=1
wpa=2
wpa_pairwise=CCMP
wpa_passphrase=1234567890
wpa_key_mgmt=SAE
wpa_group_rekey=1800
rsn_pairwise=CCMP
ieee80211w=2
sae_groups=19 20 21 25 26
sae_require_mfp=1
sae_anti_clogging_threshold=10
```

Configure WPA3 SAE transition mode for the Access Point

Create the configuration file:

```
root@imx8mqevk:~# nano /etc/hostapd-wpa3-sae.conf
```

Configuration file content:

```
interface=uap0
driver=nl80211
ssid=NXP_Demo
hw_mode=g
channel=6
wmm_enabled=1
ieee80211n=1
auth_algs=1
wpa=2
wpa_pairwise=CCMP
wpa_passphrase=1234567890
wpa_key_mgmt=WPA-PSK SAE
wpa_group_rekey=1800
rsn_pairwise=CCMP
ieee80211w=1
sae_groups=19 20 21 25 26
sae_require_mfp=1
sae_anti_clogging_threshold=10
```

Start the Access Point

Run the following command to start the Access Point:

```
root@imx8mqevk:~# killall wpa_supplicant
root@imx8mqevk:~# killall hostapd
root@imx8mqevk:~# ifconfig uap0 192.168.1.2 netmask 255.255.255.0 up
root@imx8mqevk:~# route add default gw 192.168.1.1
root@imx8mqevk:~# hostapd /etc/hostapd-wpa3-sae.conf
```

Command output example:

```
Configuration file: /etc/hostapd-wpa3-sae.conf
rfkill: Cannot open RFKILL control device
[  758.651665] wlan: HostMlme uap0 send deauth/disassoc
Using interface uap0 with hwaddr 70:66:55:26:8b:6b and ssid "NXP_Demo"
random: Only 19/20 bytes of stron[  758.671118] wlan: Starting AP
g random data available
random: [  758.677273] Get ht_cap from beacon ies: 0xc
Not enough entropy pool available[  758.683524] fw doesn't support
 11ax
for secure operations
WPA: Not enough entropy in random pool for secure operations - update
 key[  758.698627] wlan: AP started
s later when the first station co[  758.703624] Set AC=3, txop=47
 cwmin=3, cwmax=7 aifs=1
nnects
[  758.711651] Set AC=2, txop=94 cwmin=7, cwmax=15 aifs=1
[  758.718678] Set AC=0, txop=0 cwmin=15, cwmax=63 aifs=3
[  758.725701] Set AC=1, txop=0 cwmin=15, cwmax=1023 aifs=7
uap0: interface state UNINITIALIZED->ENABLED
uap0: AP-ENABLED
```

Configure WPA3 SAE mode to connect with the AP configured with WPA3 SAE

Create the configuration file

```
root@imx8mqevk:~# nano /etc/wpa_supplicant.conf
```

Configuration file content:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1
network={
        ssid="NXP_Demo"
        scan_ssid=1
        key_mgmt=SAE
        proto=RSN
        pairwise=CCMP
        group=CCMP
        psk="1234567890"
        ieee80211w=2
}
```

Connect with the AP

Commands to connect the device with the AP

```
root@imx8mqevk:~# killall wpa_supplicant
root@imx8mqevk:~# killall hostapd
root@imx8mqevk:~# wpa_supplicant -i mlan0 -Dnl80211 -c /etc/
wpa_supplicant.conf
```

Command output example

```
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL control device
[  145.646509] wlan: mlan0 START SCAN
[  150.026650] wlan: SCAN COMPLETED: scanned AP count=2
mlan0: SME: Trying to authenticate with 00:50:43:23:3c:0e
 (SSID='NXP_Demo' freq=2437 MHz)
[  150.056535] mlan0:
[  150.056539] wlan: HostMlme Auth received from 00:XX:XX:XX:3c:0e
mlan0: SME: Trying to authenticate with 00:50:43:23:3c:0e
 (SSID='NXP_Demo' freq=2437 MHz)
[  150.073767] mlan0:
[  150.073771] wlan: HostMlme Auth received from 00:XX:XX:XX:3c:0e
mlan0: PMKSA-CACHE-ADDED 00:50:43:23:3c:0e 0
mlan0: Trying to associate with 00:50:43:23:3c:0e (SSID='NXP_Demo'
 freq=2437 MHz)
[  150.097756] wlan: HostMlme mlan0 Connected to bssid
 00:XX:XX:XX:3c:0e successfully
mlan0: Associated with 00:50:43:23:3c:0e[  150.107410] mlan0:
mlan0: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
[  150.107416] wlan: Send EAPOL pkt to 00:XX:XX:XX:3c:0e
[  150.126490] mlan0:
[  150.126494] wlan: Send EAPOL pkt to 00:XX:XX:XX:3c:0e
mlan0: WPA: Key negotiation completed with 00:50:43:23:3c:0e [PTK[
  150.135343] IPv6: ADDRCONF(NETDEV_CHANGE): mlan0: link becomes ready
=CCMP GTK=CCMP]
mlan0: CTRL-EVENT-CONNECTED - Connection to 00:5[  150.148453]
 woal_cfg80211_set_rekey_data return: gtk_rekey_offload is DISABLE
0:43:23:3c:0e completed [id=0 id_str=]
```

2.4.4 Configure WPA to connect with AP using the open source supplicant

This section describes the configuration for wpa_supplicant to configure WPA security and connect with the Access Point.

Edit the configuration

Command to configure WPA to connect with AP for an open source supplicant:

```
root@imx8mqevk:~# nano /etc/wpa_supplicant.conf
```

Configuration file content:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1
network={
        ssid="NXP_Demo"
        key_mgmt=WPA-PSK
        psk="123456789"
}
```

Connect with the AP

Commands to connect the device with the WPA-based AP:

```
root@imx8mqevk:~# killall wpa_supplicant
root@imx8mqevk:~# killall hostapd
root@imx8mqevk:~# wpa_supplicant -i mlan0 -Dnl80211 -c /etc/
wpa_supplicant.conf
```

Command output example:

```
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL contro[29837.078281] IPv6:
 ADDRCONF(NETDEV_UP): mlan0: link is not read
[29837.210057] wlan: mlan0 START SCAN
[29841.854417] wlan: SCAN COMPLETED: scanned AP count=20
mlan0: Trying to associate with 40:49:0f:9e:66:fd (SSID='NXP_Demo'
 freq=2437 MHz)
[29841.873434] wlan: Connected to bssid 40:XX:XX:XX:66:fd successfully
mlan0: Associated with 40:49:0f:9e:66:fd
mlan0: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
mlan0: WPA: Key negotiation compl[29841.891468] IPv6:
 ADDRCONF(NETDEV_CHANGE): mlan0: link becomes ready
eted with 40:49:0f:9e:66:fd [PTK=[29841.901022]
 woal_cfg80211_set_rekey_data return: gtk_rekey_offload is DISABLE
TKIP GTK=TKIP]
mlan0: CTRL-EVENT-CONNECTED - Connection to 40:49:0f:9e:66:fd
 completed [id=0 id_str=]
```

2.4.5 Configure WPA2 to connect with the AP using an open source supplicant

This section describes the configuration for wpa_supplicant to configure WPA2 security and connect with the Access Point.

Edit the configuration

Command to configure WPA2 to connect with the AP for an open source supplicant:

```
root@imx8mqevk:~# nano /etc/wpa_supplicant.conf
```

Configuration file content:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1
network={
        ssid="NXP_Demo"
        key_mgmt=WPA-PSK
        psk="123456789"
}
```

Connect with the AP

Run the following commands to connect the device with the WPA2-based AP:

```
root@imx8mqevk:~# killall wpa_supplicant
root@imx8mqevk:~# killall hostapd
root@imx8mqevk:~# wpa_supplicant -i mlan0 -Dnl80211 -c /etc/
wpa_supplicant.conf
```

Command output example:

```
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL control device
[33831.070886] wlan: mlan0 START SCAN
[33835.714782] wlan: SCAN COMPLETED: scanned AP count=19
mlan0: Trying to associate with 44:82:e5:10:2c:38 (SSID='NXP_Demo'
 freq=2427 MHz)
[33835.735938] wlan: Connected to bssid 44:XX:XX:XX:2c:38 successfully
mlan0: Associated with 44:82:e5:10:2c:38
mlan0: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
mlan0: CTRL-EVENT-REGDOM-CHANGE init=COUNTRY_IE type=COUNTRY alpha2=IN
mlan0: WPA: Key negotiation compl[33835.760667] IPv6:
 ADDRCONF(NETDEV_CHANGE): mlan0: link becomes ready
eted with 44:82:e5:10:2c:38 [PTK=[33835.770113]
 woal_cfg80211_set_rekey_data return: gtk_rekey_offload is DISABLE
CCMP GTK=TKIP]
mlan0: CTRL-EVENT-CONNECTED - Connection to 44:82:e5:10:2c:38
 completed [id=0 id_str=]
```

2.5 Configure IEEE 802.11a/b/g/n/ac standards

This section describes the different IEEE 802.11 standard configurations for the Access Point. The standards use different bandwidths and network speed. The IEEE 802.11a and IEEE 802.11ac standards operate only at 5 GHz, IEEE 802.11b/g operate at 2.4 GHz and IEEE 802.11n can operate at both 2.4 GHz and 5 GHz bands.

2.5.1 Configure IEEE 802.11b standard

Configure the Access Point for IEEE 802.11b standard.

```
root@imx8mqevk:~# nano /etc/hostapd.conf
```

Configuration file content:

```
interface=uap0
hw_mode=b
channel=0
country_code=US
ssid=NXP_Demo
```

2.5.2 Configure IEEE 802.11a standard

Configure the Access Point for IEEE 802.11a standard.

```
root@imx8mqevk:~# nano /etc/hostapd.conf
```

Configuration file content:

```
interface=uap0
hw_mode=a
channel=0
country_code=US
ssid=NXP_Demo
```

2.5.3 Configure IEEE 802.11g standard

Configure the Access Point for IEEE 802.11g standard.

```
root@imx8mqevk:~# nano /etc/hostapd.conf
```

Configuration file content:

```
interface=uap0
hw_mode=g
channel=0
country_code=US
ssid=NXP_Demo
```

2.5.4 Configure IEEE 802.11n standard

Use the following to configure the Access Point configuration for IEEE 802.11n standard for 2.4 GHz and 5 GHz.

Edit the configuration for 2.4 GHz

```
root@imx8mqevk:~# nano /etc/hostapd.conf
```

Configuration file content:

```
interface=uap0
hw_mode=g
ieee80211n=1
channel=0
country_code=US
ssid=NXP_Demo
```

Edit the configuration for 5 GHz

```
root@imx8mqevk:~# nano /etc/hostapd.conf
```

Configuration file content:

```
interface=uap0
hw_mode=a
ieee80211n=1
channel=0
country_code=US
ssid=NXP_Demo
```

2.5.5 Configure IEEE 802.11ac standard

Configure the Access Point configuration for IEEE 802.11ac standard.

```
root@imx8mqevk:~# nano /etc/hostapd.conf
```

Configuration file content:

```
interface=uap0
hw_mode=a
ieee80211ac=1
channel=0
country_code=US
ssid=NXP_Demo
```

2.6 Configure and test Wi-Fi direct

This section describes the Wi-Fi direct feature which is used to establish a connection that allows for device-to-device or peer-to-peer communication, linking devices together without a nearby centralized network.

Use the following steps to test the Wi-Fi direct feature:

- Edit *wifi_mod_para_sd8987.conf* configuration file to add `p2p_enh=1` and `host_mlme=1` parameter-value pairs

```
root@imx8mqevk:~# nano /lib/firmware/nxp/wifi_mod_para_sd8987.conf
```

- Updates in the configuration file:

```
SD8987 = {
         cfg80211_wext=0xf
         wfd_name=p2p
         max_vir_bss=1
         cal_data_cfg=none
         drv_mode=7
         ps_mode=2
         auto_ds=2
         fw_name=nxp/sdiouart8987_combo_v0.bin
         p2p_enh=1
         host_mlme=1
}
```

- Load the modules in the kernel:

```
root@imx8mqevk:~# modprobe moal mod_para=nxp/wifi_mod_para_sd8987.conf
```

Command output example:

```
[ 2896.073550] wlan: Loading MWLAN driver
[ 2896.178399] vendor=0x02DF device=0x9149 class=0 function=1
[ 2896.183972] Attach moal handle ops, card interface type:0x105
[ 2896.189935] SD8987: init module param from usr cfg
[ 2896.194807] card_type: SD8987, config block: 0
[ 2896.199291] cfg80211_wext=0xf
[ 2896.202274] wfd_name=p2p
[ 2896.204855] max_vir_bss=1
[ 2896.207509] cal_data_cfg=none
[ 2896.210494] drv_mode = 7
[ 2896.213054] ps_mode = 2
[ 2896.215532] auto_ds = 2
[ 2896.218007] fw_name=nxp/sdiouart8987_combo_v0.bin
[ 2896.222747] host_mlme=disable
[ 2896.225747] SDIO: max_segs=128 max_seg_size=65535
[ 2896.230505] rx_work=1 cpu_num=4
[ 2896.233724] Attach mlan adapter operations.card_type is 0x105.
[ 2896.240142] wlan: Enable TX SG mode
[ 2896.243666] wlan: Enable RX SG mode
[ 2896.251307] Request firmware: nxp/sdiouart8987_combo_v0.bin
[ 2896.667066] Wlan: FW download over, firmwarelen=526996 downloaded 526996
[ 2897.601246] WLAN FW is active
[ 2897.604233] on_time is 2897601629500
[ 2897.638511] fw_cap_info=0x181c3f03, dev_cap_mask=0xffffffff
[ 2897.644138] max_p2p_conn = 8, max_sta_conn = 8
[ 2897.671116] wlan: version = SD8987---16.92.10.p207-MXM4X16186.p6-GPL-(FP92)
[ 2897.682184] wlan: Driver loaded successfully
```

- Start the wpa_supplicant on i.MX 8M Quad:

```
root@imx8mqevk:~# killall wpa_supplicant
root@imx8mqevk:~# killall hostapd
root@imx8mqevk:~# wpa_supplicant -g/var/run/wpa_supplicant-global -D
 nl80211 -B &
```

- Start the wpa_cli on i.MX 8M Quad:

```
root@imx8mqevk:~#  wpa_cli -g/var/run/wpa_supplicant-global
 interface_add p2p0 "" nl80211 /var/run/wpa_supplicant
root@imx8mqevk:~# wpa_cli
> set p2p_no_group_iface 1
```

- Start Wi-Fi Direct on the User Interface for the peer or mobile device.
- Enable the p2p device discovery on i.MX 8M Quad:

```
> p2p_find
```

Command output example:

```
OK
[   59.511943] wlan: p2p0 START SCAN
<3>CTRL-EVENT-SCAN-STARTED
> [   59.643151] wlan: SCAN COMPLETED: scanned AP count=0
[   59.979884] wlan: p2p0 START SCAN
<3>CTRL-EVENT-SCAN-STARTED
> [   60.111610] wlan: SCAN COMPLETED: scanned AP count=1
<3>P2P-DEVICE-FOUND ba:c7:4a:41:e3:3f p2p_dev_addr=ba:c7:4a:41:e3:3f
 pri_dev_type=10-0050F204-5 name='Peer_Device' config_methods=0x188
 dev_capab=0x25
        group_capab=0x0 new=1
```

- Stop the ongoing p2p discovery

```
> p2p_stop_find
```

- Connect with the p2p device

```
> p2p_connect <peer_device_addr> pbc go_intent=1 freq=2412
        [   89.733043] Add virtual interface p2p-p2p0-0
        [   89.747139] Can not set data rate in disconnected state
        OK[   89.752877] wlan: TX P2P Group Owner Negotiation Req
 Frame, channel=6
        > [   90.093921] wlan: TX P2P Group Owner Negotiation Req
 Frame, channel=6
        [   90.110123] wlan: RX P2P Group Owner Negotiation Rsp Frame,
 channel=6
```

- Accept the Wi-Fi Direct connection request on the peer or on the mobile device

```
[   91.756563] wlan: RX P2P Group Owner Negotiation Req Frame, channel=11
       [   91.763402] wlan: TX P2P Group Owner Negotiation Rsp Frame,
 channel=11
       [   91.774831] wlan: RX P2P Group Owner Negotiation Confirm Frame,
 channel=11
       <3>P2P-GO-NEG-SUCCESS role=client freq=2412 ht40=0 peer_dev=ba:c[
  91.807376] Can not set data rate in disconnected state
       7:4a:41:e3:3f peer_iface=ba:c7:4a:41:e3:3f wps_method=PBC
       > [   91.819478] Can not set data rate in disconnected state
       [   91.867977] wlan: p2p-p2p0-0 START SCAN
       [   91.915383] wlan: SCAN COMPLETED: scanned AP count=0
       <3>CTRL-EVENT-SCAN-RESULTS
       > [   92.171472] wlan: p2p-p2p0-0 START SCAN
       [   92.219913] wlan: SCAN COMPLETED: scanned AP count=1
       [   92.437774] p2p-p2p0-0:
       [   92.437781] wlan: HostMlme Auth received from ba:XX:XX:XX:e3:3f
       [   92.466671] wlan: HostMlme p2p-p2p0-0 Connected to bssid
 ba:XX:XX:XX:e3:3f successfully
       [   92.475454] p2p-p2p0-0:
       [   92.475460] wlan: Send EAPOL pkt to ba:XX:XX:XX:e3:3f
       [   92.556358] p2p-p2p0-0:
       [   92.556365] wlan: Send EAPOL pkt to ba:XX:XX:XX:e3:3f
       [   92.683359] p2p-p2p0-0:
       [   92.683364] wlan: Send EAPOL pkt to ba:XX:XX:XX:e3:3f
       [   92.757997] p2p-p2p0-0:
       [   92.758004] wlan: Send EAPOL pkt to ba:XX:XX:XX:e3:3f
       [   92.770529] p2p-p2p0-0:
       [   92.770532] wlan: Send EAPOL pkt to ba:XX:XX:XX:e3:3f
       [   92.783601] p2p-p2p0-0:
       [   92.783608] wlan: Send EAPOL pkt to ba:XX:XX:XX:e3:3f
       <3>P2P-GROUP-FORMATION-SUCCESS
       > [   92.847516] p2p-p2p0-0:
       [   92.847521] wlan: HostMlme Disconnected: sub_type=10
       [   92.855281] wlan: Received disassociation request on p2p-p2p0-0,
 reason: 3
       [   92.862198] wlan: REASON: (Deauth) Sending STA is leaving (or has
 left) IBSS or ESS
       [   92.869898]  Already disconnected
       <3>CTRL-EVENT-REGDOM-CHANGE init=CORE type=WORLD
       > [   92.974484] wlan: p2p-p2p0-0 START SCAN
       [   93.017544] wlan: SCAN COMPLETED: scanned AP count=1
       [   93.257289] p2p-p2p0-0:
       [   93.257296] wlan: HostMlme Auth received from ba:XX:XX:XX:e3:3f
       [   93.280678] wlan: HostMlme p2p-p2p0-0 Connected to bssid
 ba:XX:XX:XX:e3:3f successfully
       [   93.360455] p2p-p2p0-0:
       [   93.360459] wlan: Send EAPOL pkt to ba:XX:XX:XX:e3:3f
       [   93.372269] p2p-p2p0-0:
       [   93.372275] wlan: Send EAPOL pkt to ba:XX:XX:XX:e3:3f
       [   93.381399] IPv6: ADDRCONF(NETDEV_CHANGE): p2p-p2p0-0: link becomes
 ready
       <3>P2P-GROUP-STARTED p2p-p2p0-0 client ssid="DIRECT-fi-
 Peer_Device" freq=2412 [   93.389544] woal_cfg80211_set_rekey_data return:
 gtk_rekey_offload is DISABLE
       psk=cb765fd7d25a10e1c8acaef5e773658663ff03931615bc3ac6d8d0b1548bd673
 go_dev_addr=ba:c7:4a:41:e3:3f [PERSISTENT]
```

- Close the wpa_cli prompt

```
> quit
```

2.7 Measure the throughput with iPerf

This section describes the throughput test using the iPerf tool. iPerf is an open source tool used for network throughput measurements. The tool can test either TCP or UDP throughput. To perform an iPerf test the user must configure an iPerf server to act as a traffic sink, and an iPerf client to generate network traffic.

Figure 1 shows the test setup with iPerf.

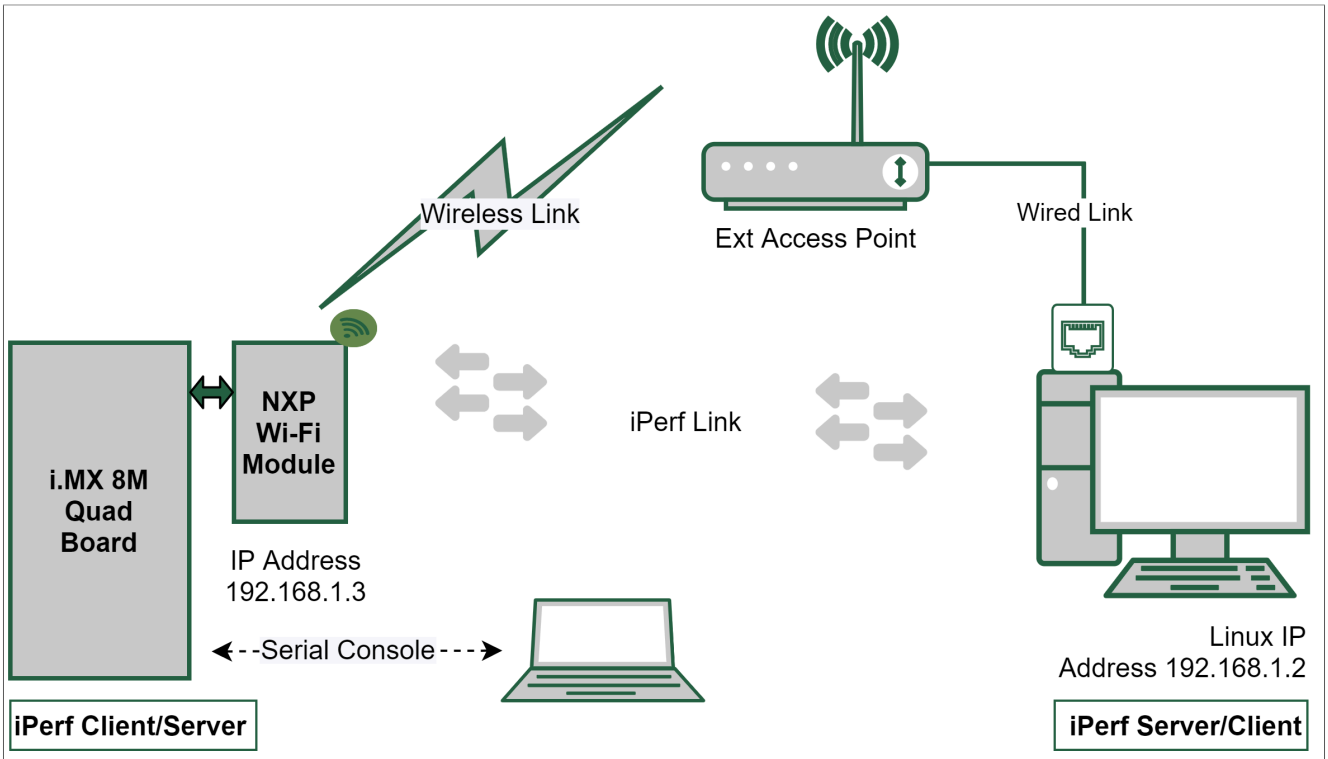

Figure 1. iPerf setup diagram

Step 1 - Connect the device with the Access point

Follow the instructions given in Section 2.3 "Connect with the Access Point" to connect the device i.MX 8M Quad with the Access Point (External Wi-Fi router).

Use the following steps to test the throughput of the network established between the AP and the station or client.

Step 2 - Get the IP address of the Access Point

Run the command:

```
root@imx8mqevk:~# udhcpc -i mlan0
```

Command output example:

```
udhcpc -i mlan0
udhcpc: started, v1.31.0
udhcpc: sending discover
udhcpc: sending select for 192.168.1.3
udhcpc: lease of 192.168.1.3 obtained, lease time 600
/etc/udhcpc.d/50default: Adding DNS 192.168.1.1
```

Step 3 - Start iPerf on the server

Use the following command to start iPerf on the Ubuntu station connected with the external Wi-Fi Router as a server.

```
root@ubuntu-desktop:/# iperf -s -i 1
```

Command output example:

```
------------------------------------------------------------
Server listening on 5201
------------------------------------------------------------
Accepted connection from 192.168.1.3, port 47590
[  5] local 192.168.1.2 port 5201 connected to 192.168.1.3 port 55286
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total
 Datagrams
[  5]   0.00-1.00   sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
[  5]   1.00-2.00   sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
[  5]   2.00-3.00   sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
[  5]   3.00-4.00   sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
[  5]   4.00-5.00   sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
[  5]   5.00-6.00   sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
[  5]   6.00-7.00   sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
[  5]   7.00-8.00   sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
[  5]   8.00-9.00   sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
[  5]   9.00-10.00  sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
[  5]  10.00-10.52  sec  XX.XX MBytes  XXX.X Mbits/sec  X.XXX ms  XX/XXXX  (XX
%)
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Jitter    Lost/Total
 Datagrams
[  5]   0.00-10.52  sec  XXX.X MBytes  XXX.X Mbits/sec  X.XXX ms  XXXX/XXXXX
 (XX%)
------------------------------------------------------------
```

Step 4 - Start iPerf on the client

Iperf server is running and the system is connected to the external AP.

Run the following command to start iPerf on i.MX 8M Quad station as a client. The command takes iPerf server IP address as an argument.

```
root@imx8mqevk:~# iperf -c 192.168.1.2 -u -b 50M -i 1
```

Command output example:

```
Connecting to host 192.168.1.2, port 5201
[  5] local 192.168.1.3 port 55286 connected to 192.168.1.2 port 5201
[ ID] Interval           Transfer     Bitrate           Total Datagrams
[  5]   0.00-1.00   sec  XX.XX MBytes  XXX.X Mbits/sec  XXXX
[  5]   1.00-2.00   sec  XX.XX MBytes  XXX.X Mbits/sec  XXXX
[  5]   2.00-3.00   sec  XX.XX MBytes  XXX.X Mbits/sec  XXXX
[  5]   3.00-4.00   sec  XX.XX MBytes  XXX.X Mbits/sec  XXXX
[  5]   4.00-5.00   sec  XX.XX MBytes  XXX.X Mbits/sec  XXXX
[  5]   5.00-6.00   sec  XX.XX MBytes  XXX.X Mbits/sec  XXXX
[  5]   6.00-7.00   sec  XX.XX MBytes  XXX.X Mbits/sec  XXXX
[  5]   7.00-8.00   sec  XX.XX MBytes  XXX.X Mbits/sec  XXXX
[  5]   8.00-9.00   sec  XX.XX MBytes  XXX.X Mbits/sec  XXXX
[  5]   9.00-10.00  sec  XX.XX MBytes  XXX.X Mbits/sec  XXXX
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Jitter    Lost/Total
 Datagrams
[  5]   0.00-10.00  sec  XXX.X MBytes  XXX.X Mbits/sec  X.XXX ms   XXXX/XXXXX
 (XX%)  sender
[  5]   0.00-10.00  sec  XXX.X MBytes  XXX.X Mbits/sec  X.XXX ms   XXXX/XXXXX
 (XX%)  receiver
iperf Done.
```

2.8 RF test mode

This section shows how to use RF test mode to set the Tx/Rx antenna configuration and the RF band.

Read more in the *README_MLAN* located in */usr/share/nxp_wireless/bin_mxm_wifiex/* directory.

2.8.1 Set Tx/Rx antenna configuration

This section describes how to test either Tx or Rx antenna mode. The test requires to first load the driver modules to bring-up the wireless module and to set the antenna to either transmit or receive mode.

Step 1 - Enable RF test mode

```
root@imx8mqevk:~# echo "rf_test_mode=1" >> /proc/mwlan/
adapterX/config
```

Verify that the value of `rf_test_mode` is `1`.

```
root@imx8mqevk:~# cat /proc/mwlan/adapterX/config
```

Step 2 - Set the antenna configuration

- Command to set the antenna configuration for **transmission mode** which will be used later when the transmission happens.

```
root@imx8mqevk:~# echo "tx_antenna=1" >> /proc/mwlan/adapterX/
config
```

Verify that the value of `tx_antenna` is `1`.

```
root@imx8mqevk:~# cat /proc/mwlan/adapterX/config
```

- Command to set the antenna configuration for **receive mode**:

```
root@imx8mqevk:~# echo "rx_antenna=1" >> /proc/mwlan/adapterX/
config
```

Verify that the value of `rx_antenna` is `1`.

```
root@imx8mqevk:~# cat /proc/mwlan/adapterX/config
```

2.8.2 Set the RF frequency band

This section shows how to set the RF frequency band to 2.4 GHz or 5 GHz. The test requires first to load the driver modules to bring-up the wireless module.

Step 1 - Enable the RF test mode

```
root@imx8mqevk:~# echo "rf_test_mode=1" >> /proc/mwlan/
adapterX/config
```

Verify that the output value of `rf_test_mode` is `1`.

```
root@imx8mqevk:~# cat /proc/mwlan/adapterX/config
```

Step 2 - Set the RF frequency band to 2.4 GHz

For example:

```
root@imx8mqevk:~# echo "band=0" >> /proc/mwlan/adapterX/config
```

Verify the output value of `band` parameter.

```
root@imx8mqevk:~# cat /proc/mwlan/adapterX/config
```

Parameter	Definition
`band`	RF band • 0 = 2.4 GHz • 1 = 5 GHz

Step 3 - Run a negative test

Use an invalid input for `band` parameter, for example band=2.

```
root@imx8mqevk:~# echo "band=2" > /proc/mwlan/adapterX/config
```

Verify that the command fails and returns `command error` which shows that the wrong input was provided.

2.8.3 Set the RF bandwidth

This section shows how to set the RF bandwidth to 20/40/80 MHz. The test requires first to load the driver modules to bring-up the NXP-based wireless module.

Step 1 - Enable the RF test mode

```
root@imx8mqevk:~# echo "rf_test_mode=1" >> /proc/mwlan/adapterX/config
```

Verify that the output value of `rf_test_mode` is `1`.

```
root@imx8mqevk:~# cat /proc/mwlan/adapterX/config
```

Step 2 - Set the RF bandwidth

For example:

```
root@imx8mqevk:~# echo "bw=0" >> /proc/mwlan/adapterX/config
```

Where:

Parameter	Definition
`bw`	RF bandwidth • 0 = 20 MHz • 1 = 40 MH • 4 = 80 MHz

2.8.4 Set the RF channel

This section shows how to set the RF channel. The test requires first to load the driver modules to bring-up the NXP-based wireless module.

Step 1 - Enable the RF test mode

```
root@imx8mqevk:~# echo "rf_test_mode=1" >> /proc/mwlan/adapterX/config
```

Verify that the output value of `rf_test_mode` is `1`.

```
root@imx8mqevk:~# cat /proc/mwlan/adapterX/config
```

Step 2 - Set the RF channel

For example, to set channel 6:

```
root@imx8mqevk:~#  echo "channel=6" >> /proc/mwlan/adapterX/config
```

2.8.5 Set Tx power

This section shows how to set the Tx power. The commands only set power if caldata is already loaded in the firmware. The test also requires first to load the driver modules to bring-up NXP-based wireless module.

Step 1 - Enable the RF test mode

```
root@imx8mqevk:~# echo "rf_test_mode=1" >> /proc/mwlan/adapterX/config
```

Verify that the output value of `rf_test_mode` is `1`.

```
root@imx8mqevk:~# cat /proc/mwlan/adapterX/config
```

Step 2 - Set the Tx power

For example:

```
root@imx8mqevk:~#  echo "tx_power=16 2 0" >> /proc/mwlan/adapterX/
config
```

<tx_power> combines three parameters:

```
<tx_power> = <power> <modulation> <path id>
```

Where:

Parameter	Definition
`power`	Power value in the range of 0 to 24 dBm
`modulation`	Signal modulation • 0 = CCK • 1 = OFDM • 2 = MCS
`path id`	Tx signal path name • 0 = path A • 1 = path B • 2 = path A + path B

2.8.6 Set Tx continuous mode

This section shows how to set Tx continuous mode. The test requires first to load the driver modules to bring-up NXP-based wireless module.

Step 1 - Enable the RF test mode

```
root@imx8mqevk:~# echo "rf_test_mode=1" >> /proc/mwlan/adapterX/config
```

Verify that the output value of `rf_test_mode` is `1`.

```
root@imx8mqevk:~# cat /proc/mwlan/adapterX/config
```

Step 2 - Start continuous mode

- Packet mode

First stop any ongoing transmission:

```
root@imx8mqevk:~# echo "tx_continuous=0" >> /proc/mwlan/adapterX/
config
```

Start Tx continuous mode:

```
root@imx8mqevk:~# echo "tx_continuous=1 0 0xAAA 0 3 7" >> /proc/mwlan/
adapterX/config
```

- Wave mode

First stop any ongoing transmission:

```
root@imx8mqevk:~# echo "tx_continuous=0" >> /proc/mwlan/adapterX/
config
```

Start Tx continuous wave:

```
root@imx8mqevk:~#  echo "tx_continuous=1 1 0xAAA 0 3 7" >> /proc/
mwlan/adapterX/config
```

Where `tx_continuous` combines six parameter:

```
tx_continuous= <start/stop> <continuous wave mode> <payload pattern>
 <cs mode> <active subchannel> <tx data rate>
```

Where:

Parameter	Definition
`start/stop`	Start/stop enable • 0 = disable • 1 = enable
`continuous wave mode`	Continuous wave mode enable • 0 = disable • 1 = enable
`payload pattern`	Payload pattern value in the range of 0 to 0xFFFFFFFF.
`cs mode`	CS mode - Applicable only when continuous wave is disabled • 0 = disable • 1 = enable
`active subchannel`	Active sub-channel • 0 = low • 1 = upper • 3 = both
`tx data rate`	Transmit data rate index corresponding to legacy/HT/VHT rates

2.8.7 Set Tx frame

This section shows how to set Tx frame. The test requires first to load the driver modules to bring-up NXP-based wireless module.

Step 1 - Enable the RF test mode

```
root@imx8mqevk:~# echo "rf_test_mode=1" >> /proc/mwlan/adapterX/config
```

Verify that the output value of `rf_test_mode` is `1`.

```
root@imx8mqevk:~# cat /proc/mwlan/adapterX/config
```

Step 2 - Start Tx frame

First stop any ongoing Tx frame:

```
root@imx8mqevk:~# echo "tx_frame=0" >> /proc/mwlan/adapterX/config
```

Command to start Tx frame with duty cycle:

```
root@imx8mqevk:~#  echo "tx_frame=1 7 0xAAA 0x100 1 20 0 0 0 0 0 0 0
 05:43:3f:c4:51" >> /proc/mwlan/adapterX/config
```

`tx_frame` combines the following parameters:

```
tx_continuous= <start/stop> <tx data rate> <payload pattern> <payload
 length> <adjust burst SIFS gap> <adjust SIFS> <short preamble>
 <active subchannel> <short GI> <adv coding> <beamforming> <greenfield
 mode> <STBC> <BSSID>
```

Where:

Parameter	Definition
`start/stop`	Start/stop enable • 0 = disable • 1 = enable
`tx data rate`	Transmit the data rate index corresponding to the legacy/HT/VHT rates. See Table 2 "Data rate parameter"
`payload pattern`	Payload pattern value in the range of 0 to 0xFFFFFFFF.
`payload length`	Payload length value in the range of 1 to 0x400.
`adjust burst SIFS gap`	• 0 = disable • 1 = enable
`adjust SIFS`	Burst SIFS duration in us, in the range of 0 to 255 us
`short preamble`	• 0 = disable • 1 = enable
`active subchannel`	• 0 = low • 1 = upper • 3 = both
`short GI`	Short guard interval • 0 = disable • 1 = enable

Parameter	Definition
`adv coding`	Advanced coding • 0 = disable • 1 = enable
`beamforming`	Beamforming enable • 0 = disable • 1 = enable
`greenfield mode`	Greenfield mode enable • 0 = disable • 1 = enable
`STBC`	Space time block coding enable • 0 = disable • 1 = enable
`BSSID`	Basic service set identifiers - format = xx:xx:xx:xx:xx:xx

Table 2. Data rate parameter

ID	Data rate
1	1Mbit/s
2	2Mbit/s
3	5.5Mbit/s
4	11Mbit/s
5	Reserved
6	6Mbit/s
7	9Mbit/s
8	12Mbit/s
9	18Mbit/s
10	24Mbit/s
11	36Mbit/s
12	48Mbit/s
13	54Mbit/s
14	Reserved
15	HT_MCS0
16	HT_MCS1
17	HT_MCS2
18	HT_MCS3
19	HT_MCS4
20	HT_MCS5
21	HT_MCS6
22	HT_MCS7
23	HT_MCS8
24	HT_MCS9
25	HT_MCS10

Table 2. Data rate parameter*...continued*

ID	Data rate
26	HT_MCS11
27	HT_MCS12
28	HT_MCS13
29	HT_MCS14
30	HT_MCS15
101	VHT_SS1_MCS0
102	VHT_SS1_MCS1
103	VHT_SS1_MCS2
104	VHT_SS1_MCS3
105	VHT_SS1_MCS4
106	VHT_SS1_MCS5
107	VHT_SS1_MCS6
108	VHT_SS1_MCS7
109	VHT_SS1_MCS8
110	VHT_SS1_MCS9
111	VHT_SS2_MCS0
112	VHT_SS2_MCS1
113	VHT_SS2_MCS2
114	VHT_SS2_MCS3
115	VHT_SS2_MCS4
116	VHT_SS2_MCS5
117	VHT_SS2_MCS6
118	VHT_SS2_MCS7
119	VHT_SS2_MCS8
120	VHT_SS2_MCS9

Note: *Some data rates are only available in some channel bandwidth modes. For example, VHT_MCS9 is only available in 40 MHz, 80 MHz, and 160 MHz modes.*

2.9 ED-MAC and Tx power enable

This section provides the guidelines to enable energy detection (ED) for the adaptivity test and to load the converted Tx power table.

2.9.1 ED-MAC enable

This section provides the instructions to edit the configuration file and enable ED-MAC configuration.

Step 1 - Edit the configuration file

The path to *ed_mac_ctrl_V3_8987.conf* file is as follows:

/usr/share/nxp_wireless/bin_mxm_wifiex/config/

Edit the *.conf* file to set `ed_ctrl_2g.enable:2 = 0x01` to enable energy detection for the adaptivity test. In the configuration file, the offset values of `ed_ctrl_2g.offset:2` and `ed_ctrl_5g.offset:2` parameters are used to adjust the ED threshold during the EU adaptivity test.

```
ed_mac_ctrl_v3={
   CmdCode=0x0130                #Command code, DO NOT change this line
   ed_ctrl_2g.enable:2=0x1       # 0 - disable EU adaptivity for 2.4GHz band
                                 # 1 - enable EU adaptivity for 2.4GHz band
   ed_ctrl_2g.offset:2=0x6       # 0 - Default Energy Detect threshold
                                 #offset value range: 0x80 to 0x7F
   ed_ctrl_5g.enable:2=0x1       # 0 - disable EU adaptivity for 5GHz band
                                 # 1 - enable EU adaptivity for 5GHz band
   ed_ctrl_5g.offset:2=0x6       # 0 - Default Energy Detect threshold
                                 #offset value range: 0x80 to 0x7F
   ed_ctrl_txq_lock:4=0xFF       #DO NOT Change this line
}
```

Step 2 - Convert the configuration file to a binary format

The ED-MAC configuration file *ed_mac_ctrl_V3_8987.conf* file must be converted to a binary format (*.bin*) so the wireless driver can use it. The following command uses *mlanutl* utility to convert the configuration file to a binary format. *mlanutl* utility is stored at the following location: */usr/share/nxp_wireless/bin_mxm_wifiex/*

```
mlanutl <wireless interface> hostcmd <conf_file_name> generate_raw
 <bin_file_name>
```

Where:

Command parameter	Description
`wireless interface`	Name of the wireless interface used in the system, for example mlan0
`conf_file_name`	Name of ED-MAC configuration file
`bin_file_name`	Name of the output binary file to be generated

Example

```
root@imx8mqevk:~# ./mlanutl mlan0 hostcmd config/
ed_mac_ctrl_V3_8987.conf generate_raw ed_mac.bin
```

Copy the newly generated binary file *ed_mac.bin* and paste into the wireless firmware directory.

For Linux operating systems, the directory is: */lib/firmware/nxp*.

Step 3 - Load ED-MAC configuration

The ED-MAC configuration binary file is activated during the wireless driver load time. In the following example, the driver uses `init_hostcmd_cfg` parameter when loading *ed_mac.bin*.

```
root@imx8mqevk:~# nano /lib/firmware/nxp/wifi_mod_para_sd8987.conf
SD8987 = {
      cfg80211_wext=0xf
      wfd_name=p2p
      cal_data_cfg=none
      max_vir_bss=1
      fw_name=nxp/sdiouart8987_combo_v0.bin
      init_hostcmd_cfg=nxp/ed_mac.bin
}
```

Run the following command to load the modules:

```
root@imx8mqevk:~# modprobe moal mod_para=nxp/
wifi_mod_para_sd8987.conf
```

2.9.2 Transmit power enable

This section provides the steps to enable transmit (Tx) power.

Step 1 - Edit the configuration file

The Tx power levels are set with `txpwrlimit_2g_cfg_set` and `txpwrlimit_5g_cfg _set` data structures defined in *txpwrlimit_cfg.conf* configuration file.

Use the configuration file to specify the transmit power levels for the frequency, channels, power and channel-width.

Step 2 - Convert the configuration file

The TX power configuration file *txpwrlimit_cfg.conf* must be converted to a binary (*.bin*) format so the wireless driver can use it.

The following command uses *mlanutl* utility stored at the following location: */usr/share/nxp_wireless/bin_mxm_wifiex/* to convert the configuration file to a bin format.

```
mlanutl <wireless interface> hostcmd <conf_file_name> generate_raw
 <bin_file_name>
```

- `<wireless interface>` is the wireless interface used in the system, for example mlan0
- `<conf_file_name>` is the name of Tx power configuration file
- `<bin_file_name>` is the name of the output bin file to be generated

Example

```
root@imx8mqevk:~# ./mlanutl mlan0 hostcmd txpwrlimit_cfg_8987.conf
 generate_raw txpower_US.bin
```

returns:

```
root@imx8mqevk:~# ./mlanutl mlan0 hostcmd  ./config/
txpwrlimit_cfg.conf generate_raw txpower_US.bin
```

Copy the newly generated binary files into the wireless firmware directory. For Linux systems, the directory is: */lib/firmware/nxp*.

Step 3 - Load Tx power table configuration

The Tx Power table binary file is activated during the wireless driver load time. In the following example, the driver loads *txpower_US.bin* using `txpwrlimit_cfg` parameter:

```
root@imx8mqevk:~# nano /lib/firmware/nxp/wifi_mod_para_sd8987.conf
SD8987 = {
      cfg80211_wext=0xf
      wfd_name=p2p
      cal_data_cfg=none
      max_vir_bss=1
      fw_name=nxp/sdiouart8987_combo_v0.bin
      txpwrlimit_cfg=nxp/txpower_US.bin
}
```

Run the following command to load the module:

```
root@imx8mqevk:~# modprobe moal mod_para=nxp/wifi_mod_para_sd8987.conf
```

Step 4 - Get Tx power configurations

The command get the Tx power configurations uses *mlanutl* utility stored at the following location:

/usr/share/nxp_wireless/bin_mxm_wifiex/

Syntax: `mlanutl <ifname> <cmd> [...]`

Where:

`ifname`: wireless network interface name, such as mlanX or uapX

Usage:

```
mlanutl mlanX get_txpwrlimit <n> [raw_data_file]
```

Where:

Command Parameter	Description
`n`	Hexadecimal value set for a specific tx power table 0 = Get 2.4G txpwrlimit table 0x10 = Get 5G sub0 txpwrlimit table 0x11 = Get 5G sub1 txpwrlimit table 0x12 = Get 5G sub2 txpwrlimit table 0x1f = Get all 5G txpwrlimit table 0xff = Get both 2G and 5G txpwrlimit table
`raw_data_file`	Name of the file in which the driver will save the firmware raw data

Example(s)

`mlanutl mlan0 get_txpwrlimit 0` // Get 2G txpwrlimit table

`mlanutl mlan0 get_txpwrlimit 0x10` // Get 5G sub band0 txpwrlimit table

`mlanutl mlan0 get_txpwrlimit 0xff txpwrlimit.bin` // Get both 2G/5G txpwrlimit table and save to *txpwrlimit.bin*

2.10 Set the Wi-Fi device in suspend state

This section explains how to capture the Wi-Fi driver logs even when the i.MX host processor is in suspend state. It also covers the steps to put the i.MX host processor in suspend state.

Step 1 - Update the configuration file

Edit *wifi_mod_para_sd8987.conf* file:

```
root@imx8mqevk:~# nano /lib/firmware/nxp/wifi_mod_para_sd8987.conf
```

Update `ps_mode` in the configuration file:

```
SD8987 = {
    cfg80211_wext=0xf
    wfd_name=p2p
    max_vir_bss=2
    cal_data_cfg=none
    drv_mode=7
    ps_mode=1
    auto_ds=2
    fw_name=nxp/sdiouart8987_combo_v0.bin
}
```

Step 2 - Load the modules in the kernel

```
root@imx8mqevk:~# modprobe moal mod_para=nxp/wifi_mod_para_sd8987.conf
```

Step 3 - Connect the client (STA) device to the external access point using wpa_supplicant

Refer to Section 2.3 "Connect with the Access Point"

Step 4 - Enable WoWLAN

```
root@imx8mqevk:~# iw phy#0 wowlan enable any
```

Step 5 - Set the device in suspend state

```
root@imx8mqevk:~# echo mem >> /sys/power/state
```

2.10.1 Enable dmesg logs for device in supended state

Step 1 - Boot the i.MX 8M Quad EVK to the U-boot console

```
U-Boot >
```

Step 2 - Enable dmesg logs while the device is in suspended state

```
U-Boot > setenv mmcargs $mmcargs no_console_suspend
```

Step 3 - Boot the i.MX 8M Quad EVK

```
U-Boot > boot
```

2.10.2 High throughput, DDR frequency, and AXIHP bus frequency

This section shows how to configure PCIe bus frequency to high power mode and achieving a higher throughput. It also provides the step to reverse to low power mode.

- Enable high throughput using high power mode

```
root@imx8mqevk:# echo 1 >> /sys/devices/platform/soc@0/33c00000.pcie/
bus_freq
[ 204.287147] imx6q-pcie 33c00000.pcie: pcie request bus freq high
```

- Set low power mode

```
root@imx8mqevk:# echo 0 >> /sys/devices/platform/soc@0/33c00000.pcie/
bus_freq
[ 252.812327] imx6q-pcie 33c00000.pcie: pcie release bus freq high
```

3 Wi-Fi driver debugging

This section explains how to enable driver debug logs and access debug information. Some debug logs can be enabled at runtime and driver load time. Other logs require to first rebuild the driver with a specific configuration. The firmware dump methods to generates dump file to debug on the firmware front are also detailed in this section.

3.1 Enable the driver debug logs

This section provides the guidelines on how to enable the driver debug logs using either the module parameter or `/proc` at runtime.

3.1.1 Use the module parameter

Use the `drvdbg` module parameter to enable driver debug logs while the driver is loading. Refer to Section 3.2 "Driver debug log types" for the module parameter values.

Edit the configuration file:

```
root@imx8mqevk:~# nano /lib/firmware/nxp/wifi_mod_para_sd8987.conf
```

Configuration file content:

```
SD8987 = {
    cfg80211_wext=0xf
    wfd_name=p2p
    cal_data_cfg=none
    fw_name=nxp/sdiouart8987_combo_v0.bin
    drvdbg=< bit masks of driver debug message control>
}
```

3.1.2 Use /proc at runtime

Use the following command to enable driver debug logs when the driver is already loaded:

```
root@imx8mqevk:~# echo "drvdbg=<bit masks of driver debug
 message control>" > /proc/mwlan/adapter0/<interface>/debug
```

3.2 Driver debug log types

Table 3 lists the debug logs types exposed by the NXP driver for `drvdbg` parameter. The debug information can be enabled or disabled using either the module parameter (Section 3.1.1) or /proc at runtime (Section 3.1.2).

Table 3. NXP driver debug log types

Bit	Message type	Log format	Description
Bit 0	MMSG	PRINTM(MMSG,...)	Set bit 0 to enable all driver logs with log level MMSG.
Bit 1	MFATAL	PRINTM(MFATAL,...)	Set bit 1 to enable all driver logs with log level MFATAL.
Bit 2	MERROR	PRINTM(MERROR,...)	Set bit 2 to enable all driver logs with log level MERROR.
Bit 3	MDATA	PRINTM(MDATA,...)	Set bit 3 to enable all driver logs with log level MDATA.
Bit 4	MCMND	PRINTM(MCMND,...)	Set bit 4 to enable all driver logs with log level MCMND.
Bit 5	MEVENT	PRINTM(MEVENT,...)	Set bit 5 to enable all driver logs with log level MEVENT.
Bit 6	MINTR	PRINTM(MINTR,...)	Set bit 6 to enable all driver logs with log level MINTR.
Bit 7	MIOCTL	PRINTM(MIOCTL,...)	Set bit 7 to enable all driver logs with log level MIOCTL.
Bit 16	MDAT_D	PRINTM(MDAT_D,...), DBG_HEXDUMP(MDAT_D,...)	Set bit 16 to enable all driver logs with log level MDAT_D and provide the corresponding hexdump in dmesg logs.
Bit 17	MCMD_D	PRINTM(MCMD_D,...), DBG_HEXDUMP(MCMD_D,...)	Set bit 17 to enable all driver logs with log level MCMD_D and provide the corresponding hexdump in dmesg logs.
Bit 18	MEVT_D	PRINTM(MEVT_D,...), DBG_HEXDUMP(MEVT_D,...)	Set bit 18 to enable all driver logs with log level MEVT_D and provide the corresponding hexdump in dmesg logs.
Bit 19	MFW_D	PRINTM(MFW_D,...), DBG_HEXDUMP(MFW_D,...)	Set bit 19 to enable all driver logs with log level MFW_D and provide the corresponding hexdump in dmesg logs.
Bit 20	MIF_D	PRINTM(MIF_D,...), DBG_HEXDUMP(MIF_D,...)	Set bit 20 to enable all driver logs with log level MIF_D and provide the corresponding hexdump in dmesg logs.
Bit 28	MENTRY	PRINTM(MENTRY,...), ENTER(), LEAVE()	Set bit 28 to enable all driver logs with API entry and exit.
Bit 29	MWARN	PRINTM(MWARN,...)	Set bit 29 to enable all driver logs with log level MWARN.
Bit 30	MINFO	PRINTM(MINFO,...)	Set bit 30 to enable all driver logs with log level MINFO.

3.3 Firmware dump

This section shows how to retrieve the firmware memory from the device and dump it into a file for debugging purposes. A firmware dump can be triggered from the /proc. The driver also dumps the firmware memory in the dmesg logs when a fatal error occurs for example due to command timeout or Tx timeout.

Command to trigger the firmware dump:

```
root@imx8mqevk:~# echo "debug_dump" >> /proc/mwlan/adapter0/config
```

Command output example:

```
[ 4647.885895] Receive debug_dump command
[ 4647.889735] ------------mlan_debug_info-------------
[ 4647.894749] mlan_processing =0
[ 4647.897838] main_lock_flag =0
[ 4647.900808] main_process_cnt =50
[ 4647.904070] delay_task_flag =0
[ 4647.907151] mlan_rx_processing =0
[ 4647.910498] rx_pkts_queued=0
[ 4647.913381] tx_pkts_queued=0
[ 4648.314446] Create directory /var/dump_4648 successfully
[ 4648.319463] SDIO Func1 (0x10-0x17): 00 00 00 00 ff ff ff ff
[ 4648.325351] Directory name is /var/dump_4648
[ 4648.325353] === START DRIVER INFO DUMP===
[ 4648.325407] DRV dump data in /var/dump_4648/file_drv_info
[ 4648.330895] SDIO Func1: (0x8) c3 (0x58) 00 (0x5c) 08 (0x5d) 00
 (0x60) 07 (0x61) 0c (0x62) 00 (0x64) 10 (0x65) 00 (0x66) 00 (0x68) 00
 (0x69) 00 (0x6a
[ 4648.363988] SDIO Func1 (0xe8-0xf2): dc fe 04 00 d9 00 00 20 20 01
 70
[ 4648.448491] Drv info total bytes = 349618 (0x555b2)
[ 4648.453388] === DRIVER INFO DUMP END===
[ 4648.453448] ==== DEBUG MODE OUTPUT START: 4648.303870 ====
[ 4648.464517] DUMP_SIZE=0xf0000
[ 4648.467648] Start DUMP output 4648.318072, please wait...
[ 4691.532749] DUMP done:size = 0x129780
[ 4691.536605] Dump data file_sdio_DUMP saved in /var/dump_4648/
file_sdio_DUMP
[ 4691.552935] Dump data file_sdio_DUMP saved in /var/dump_4648/
file_sdio_DUMP successfully
[ 4691.561469] ==== DEBUG MODE OUTPUT END: 4691.411892 ====
[ 4691.566806] ==== DEBUG MODE END ====
```

The firmware dump files are created in `dump_XXXX` directory.

Command to list the content of `dump_XXXX` directory:

```
root@imx8mqevk:~# ls /var/dump_XXXX/
```

Command output example:

```
file_drv_info   file_sdio_DUMP
```

4 Bluetooth classic/Bluetooth LE features and configurations

4.1 Scan, pair and connect to Bluetooth classic/Bluetooth LE

This section describes the configuration steps to scan, pair and connect with a Remote Bluetooth device from NXP-based wireless module. BlueZ Stack provides the *bluetoothctl* command line utility to connect with a Remote Bluetooth device.

Use the following steps to scan, pair and connect the remote Bluetooth Classic/Bluetooth Low Energy device using *bluetoothctl* utility.

Start *bluetoothctl*

Start *bluetoothctl* to interact with the Bluetooth daemon from the command line:

```
root@imx8mqevk:~# bluetoothctl
Agent registered
[bluetooth]#
```

Authenticate

Since the pairing procedure will involve authentication by PIN, it is required to register with an authentication agent. The agent handles the PIN prompt:

```
[bluetooth]# agent on
Agent is already registered
[bluetooth]# default-agent
Default agent request successful
```

Start scanning

Run the following command to start scanning for remote Bluetooth Classic/Bluetooth LE device(s).

```
[bluetooth]# scan on
Discovery started
[CHG] Controller 00:50:43:24:34:F7 Discovering: yes
[NEW] Device B4:F5:00:31:CB:4E Moto E
```

Stop scanning

Stop the scanning and check for available remote Bluetooth Classic/Bluetooth LE device(s) for pairing.

```
[bluetooth]# scan off
Discovery stopped
[CHG] Controller 00:50:43:24:34:F7 Discovering: no
[bluetooth]# devices
Device B4:F5:00:31:CB:4E Moto E
[bluetooth]#
```

Start pairing

Start pairing with the remote Bluetooth Classic/Bluetooth LE device.

```
[bluetooth]# pair B4:F5:00:31:CB:4E
Attempting to pair with B4:F5:00:31:CB:4E
[CHG] Device B4:F5:00:31:CB:4E Connected: yes
Request confirmation
[agent] Confirm passkey 666330 (yes/no):
```

Confirm pairing

If the passkey matches, type yes to complete the pairing procedure and establish a connection:

```
[agent] Confirm passkey 666330 (yes/no): yes
[CHG] Device B4:F5:00:31:CB:4E Modalias: bluetooth:v000Fp1200d1436
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 00001105-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 0000110a-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 0000110c-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 0000110e-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 00001112-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 00001115-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 00001116-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 0000111f-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 0000112f-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 00001132-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 00001200-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 00001800-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E UUIDs:
 00001801-0000-1000-8000-00805f9b34fb
[CHG] Device B4:F5:00:31:CB:4E ServicesResolved: yes
[CHG] Device B4:F5:00:31:CB:4E Paired: yes
Pairing successful
[CHG] Device B4:F5:00:31:CB:4E ServicesResolved: no
[CHG] Device B4:F5:00:31:CB:4E Connected: no
[bluetooth]#
```

Connect with the remote Bluetooth Classic/Bluetooth LE device

The remote Bluetooth Classic/Bluetooth LE device is not yet in the connected state. Run the following command to set the connected state.

```
[bluetooth]# trust B4:F5:00:31:CB:4E
[CHG] Device B4:F5:00:31:CB:4E Trusted: yes
Changing B4:F5:00:31:CB:4E trust succeeded
[bluetooth]# connect B4:F5:00:31:CB:4E
Attempting to connect to B4:F5:00:31:CB:4E
[CHG] Device B4:F5:00:31:CB:4E Connected: yes
Connection successful
[CHG] Device B4:F5:00:31:CB:4E ServicesResolved: yes
[Moto E]#
```

Disconnect the remote Bluetooth Classic/Bluetooth LE device

```
[Moto E]# disconnect B4:F5:00:31:CB:4E
Attempting to disconnect from B4:F5:00:31:CB:4E
[CHG] Device B4:F5:00:31:CB:4E ServicesResolved: no
Successful disconnected
[CHG] Device B4:F5:00:31:CB:4E Connected: no
[bluetooth]#
```

Remove the remote Bluetooth Classic/Bluetooth LE device

```
[bluetooth]# remove B4:F5:00:31:CB:4E
[DEL] Device B4:F5:00:31:CB:4E Moto E
Device has been removed
[bluetooth]#
```

Exit from *bluetoothctl* command line interface

```
[Moto E]# quit
root@imx8mqevk:~#
```

4.2 Advanced audio distribution profile

This section describes the configuration steps for i.MX 8M Quad board to act as an A2DP sink or source. The Pulseaudio and Bluetooth-player packages are used while configuring these profiles.

4.2.1 Configure A2DP sink

Make sure the remote Bluetooth device supports the A2DP source profile feature.

Use the following steps to set up the i.MX 8M Quad as an A2DP sink:

Start pulseaudio on i.MX 8M Quad

```
root@imx8mqevk:~# pulseaudio &
[1] 507
root@imx8mqevk:~#
```

Connect with a remote Bluetooth device that supports Audio Source Profile

Refer to Section 4.1 "Scan, pair and connect to Bluetooth classic/Bluetooth LE"

Run the following command to verify the Audio Source Profile capability of the connected Bluetooth device:

```
root@imx8mqevk:~# bluetoothctl
[c8ca22c1]# info 20:39:56:C6:6C:6C
Device 20:39:56:C6:6C:6C
Name: c8ca22c1
Alias: c8ca22c1
Class: 0x005a020c
Icon: phone
Paired: yes
Trusted: yes
Blocked: no
Connected: yes
LegacyPairing: no
UUID: Dialup Networking (00001103-0000-1000-8000-00805f9b34fb)
UUID: OBEX Object Push (00001105-0000-1000-8000-00805f9b34fb)
UUID: Audio Source (0000110a-0000-1000-8000-00805f9b34fb)
UUID: A/V Remote Control Target
      (0000110c-0000-1000-8000-00805f9b34fb)
UUID: A/V Remote Control (0000110e-0000-1000-8000-00805f9b34fb)
UUID: Headset AG (00001112-0000-1000-8000-00805f9b34fb)
UUID: PANU (00001115-0000-1000-8000-00805f9b34fb)
UUID: NAP (00001116-0000-1000-8000-00805f9b34fb)
UUID: Handsfree Audio Gateway (0000111f-0000-1000-8000-00805f9b34fb)
UUID: SIM Access (0000112d-0000-1000-8000-00805f9b34fb)
UUID: Phonebook Access Server (0000112f-0000-1000-8000-00805f9b34fb)
UUID: Message Access Server (00001132-0000-1000-8000-00805f9b34fb)
UUID: PnP Information (00001200-0000-1000-8000-00805f9b34fb)
UUID: Generic Access Profile (00001800-0000-1000-8000-00805f9b34fb)
UUID: Generic Attribute Profile
       (00001801-0000-1000-8000-00805f9b34fb)
Modalias: bluetooth:v001Dp1200d1436
[c8ca22c1]#
```

Exit from *bluetoothctl* command line interface

```
[c8ca22c1]# quit
root@imx8mqevk:~#
```

Check the available sink cards

```
root@imx8mqevk:~# pacmd list-sinks | egrep '(index|name):'
  * index: 0
        name: <alsa_output.platform-sound-bt-sco.mono-fallback>
    index: 1
        name: <alsa_output.platform-sound-spdif.stereo-fallback>
    index: 2
        name: <alsa_output.platform-sound-wm8524.stereo-fallback>
root@imx8mqevk:~#
```

Select `wm8524` as default sink

Select `wm8524` as default sink to listen to music on the audio jack of the i.MX 8M Quad board.

```
root@imx8mqevk:~# pacmd set-default-sink 2
root@imx8mqevk:~# pacmd list-sinks | egrep '(index|name):'
    index: 0
        name: <alsa_output.platform-sound-bt-sco.mono-fallback>
    index: 1
        name: <alsa_output.platform-sound-spdif.stereo-fallback>
  * index: 2
        name: <alsa_output.platform-sound-wm8524.stereo-fallback>
root@imx8mqevk:~#
```

Plug the speaker

Plug the audio jack of the board to the speaker and play music from the Bluetooth device connected to the board.

Play and pause the audio

Use bluetooth-player to play and pause the audio.

```
root@imx8mqevk:~# bluetooth-player
        [NEW] Player /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0
 [default]
        [NEW] Folder /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0
        [NEW] Item /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0/
NowPlaying
                /NowPlaying
        [NEW] Item /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0/
NowPlaying/item3
                <unknown>
        [NEW] Item
                /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0/
NowPlaying/item18446744073709551615
                <unknown>
        [NEW] Item /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0/
Filesystem
                /Filesystem
        [NEW] Item /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0/
Filesystem/item4
                /Filesystem/All Songs
        [NEW] Item /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0/
Filesystem/item3
                /Filesystem/Playlists
        [NEW] Item /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0/
Filesystem/item2
                /Filesystem/Artists
        [NEW] Item /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0/
Filesystem/item1
                /Filesystem/Albums
        [bluetooth]#
        [bluetooth]# pause
        Attempting to pause
        Pause successful
        [CHG] Player /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0
 Status:
                paused
        [CHG] Player /org/bluez/hci0/dev_B4_F5_00_31_CB_4E/player0
 Position:
                0x1e9e84
        [bluetooth]# play
        Attempting to play
        Play successful
        [bluetooth]#
```

Disconnect the device using *bluetoothctl* command line interface

```
root@imx8mqevk:~# bluetoothctl
        [Moto E]# disconnect B4:F5:00:31:CB:4E
        Attempting to disconnect from B4:F5:00:31:CB:4E
        [CHG] Device B4:F5:00:31:CB:4E ServicesResolved: no
        Successful disconnected
        [CHG] Device B4:F5:00:31:CB:4E Connected: no
        [bluetooth]#
```

4.2.2 Configure A2DP source

Make sure the remote Bluetooth device supports the A2DP sink feature.

Use the following steps to configure the i.MX 8M Quad board as an A2DP Source.

Start pulseaudio on i.MX 8M Quad

```
root@imx8mqevk:~# pulseaudio &
[1] 507
root@imx8mqevk:~#
```

Connect with a remote Bluetooth device that supports Audio Sink Profile

Refer to Section 4.1 "Scan, pair and connect to Bluetooth classic/Bluetooth LE"

Run the following command to verify the Audio Sink Profile capability of the connected Bluetooth device:

```
root@imx8mqevk:~# bluetoothctl
[Office speaker 1]# info 7C:2E:BD:48:15:FC
Device 7C:2E:BD:48:15:FC (public)
Name: Office speaker 1
        Alias: Office speaker 1
        Class: 0x00240400
        Icon: audio-card
        Paired: yes
        Trusted: yes
        Blocked: no
        Connected: yes
LegacyPairing: no
        UUID: Audio Sink                (0000110b-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control Target (0000110c-0000-1000-8000-00805f9b34fb)
        UUID: A/V Remote Control        (0000110e-0000-1000-8000-00805f9b34fb)
        UUID: PnP Information           (00001200-0000-1000-8000-00805f9b34fb)
        UUID: Generic Access Profile    (00001800-0000-1000-8000-00805f9b34fb)
        UUID: Generic Attribute Profile (00001801-0000-1000-8000-00805f9b34fb)
        UUID: Google                    (0000fea0-0000-1000-8000-00805f9b34fb)
Modalias: bluetooth:v000Fp1200d1436
[Office speaker 1]#
```

Copy the sample audio file to i.MX 8M Quad

Use the following command to check the available sink cards:

```
root@imx8mqevk:~# pacmd list-sinks | egrep '(index|name):'
    index: 0
        name: <alsa_output.platform-sound-bt-sco.mono-fallback>
    index: 1
        name: <alsa_output.platform-sound-spdif.stereo-fallback>
  * index: 2
        name: <alsa_output.platform-sound-wm8524.stereo-fallback>
    index: 3
        name: <bluez_sink.7C_2E_BD_48_15_FC.a2dp_sink>
root@imx8mqevk:~#
```

Select the Bluetooth speaker as an audio sink device

```
root@imx8mqevk:~# pacmd set-default-sink 3
root@imx8mqevk:~# pacmd list-sinks | egrep '(index|name):'
    index: 0
        name: <alsa_output.platform-sound-bt-sco.mono-fallback>
    index: 1
        name: <alsa_output.platform-sound-spdif.stereo-fallback>
    index: 2
        name: <alsa_output.platform-sound-wm8524.stereo-fallback>
  * index: 3
        name: <bluez_sink.7C_2E_BD_48_15_FC.a2dp_sink>
root@imx8mqevk:~#
```

Play the audio file using pulseaudio play utility

```
root@imx8mqevk:~# paplay <path to audio file>/sample_audio.wav &
[3] 562
[2]   Done(1)                 paplay /usr/local/sample_audio.wav
root@imx8mqevk:~#
```

Set the volume of sink device using pulseaudio commands

Here, 0 is muted and 0x10000 is normal volume.

```
root@imx8mqevk:~# pacmd set-sink-volume <sink_index> <volume_level>
root@imx8mqevk:~# pacmd set-sink-volume 3 0x9000
```

Disconnect the device using *Bluetoothctl* command line interface

```
root@imx8mqevk:~# bluetoothctl
Agent registered
[Office speaker 1]# disconnect 7C:2E:BD:48:15:FC
Attempting to disconnect from 7C:2E:BD:48:15:FC
[CHG] Device 7C:2E:BD:48:15:FC ServicesResolved: no
Successful disconnected
[CHG] Device 7C:2E:BD:48:15:FC Connected: no
[bluetooth]#
```

4.3 Hands-free profile (HFP)

This section introduces the Ofono package used to dial, hang-up and answer telephone calls, and details the configuration for the hands-free profile.

4.3.1 Ofono package

Ofono package

Ofono provides a mobile telephony application development framework with minimal and easy-to-use APIs. Ofono is controlled through D-Bus.

ofonod is the daemon that provides the Ofono stack for interfacing mobile telephony devices. For example, one can tell ofonod to send AT commands over */dev/rfcomm0* by calling the D-Bus method `org.ofono.at.Manager.Create`.

Table 4 describes the AT commands used to dial, hang-up and answer the calls:

Table 4. Hands-free profile AT commands

command	Description
ATA	Command used to answer a call
ATD	Command used to place a call to a phone number
AT+CHUP	Command used to hang up. This command causes the AG to end an active call

For more information on AT commands refer to the Hands-Free Profile Specification.

4.3.2 Hands-free profile configuration

Step 1 - Enable the host interface

- **88W8987**: Use *imx8mq-evk-usdhc2-m2.dtb* file to enable SDIO on M.2 connector
- **88W8997**: Use *imx8mq-evk-pcie1-m2.dtb* file to enable PCIe on M.2 connector

Note: *Refer to UM11483 for details on interface enable.*

Step 2 - Stop pulseaudio

If pulseaudio is running, run the command to stop pulseaudio:

```
root@imx8mqevk:~# killall pulseaudio
```

Command output example:

```
pulseaudio: no process found
root@imx8mqevk:~#
```

Step 3 - Update the sample rate in the configuration file

Edit the configuration file to change the default sample rate to 8000 samples per second:

```
root@imx8mqevk:~# nano /etc/pulse/daemon.conf
; default-sample-rate = 44100
default-sample-rate = 8000
root@imx8mqevk:~#
```

Step 4 - Start pulseaudio on i.MX 8M Quad EVK

```
root@imx8mqevk:~# pulseaudio &
```

Command output:

```
[1] 507
root@imx8mqevk:~#
```

Step 5 - List the available sink devices

```
root@imx8mqevk:~# pacmd list-sinks | egrep '(index|name):'
```

Command output example:

```
  index: 0
      name: <alsa_output.platform-sound-bt-sco.mono-fallback>
  index: 1
      name: <alsa_output.platform-sound-spdif.stereo-fallback>
* index: 2
      name: <alsa_output.platform-sound-wm8524.stereo-fallback>
root@imx8mqevk:~#
```

Step 6 - Set external sink device as default sink

```
root@imx8mqevk:~# pacmd set-default-sink <index number of external
 sink device>
```

Note: *As the i.MX 8M Quad EVK does not have a built-in mic, an external USB audio mic adapter is required to enable the two-way audio.*

Step 7 - List the available source devices

```
root@imx8mqevk:~# pacmd list-sources | egrep '(index|name):'
```

Command output example:

```
  index: 0
      name: <alsa_output.platform-sound-bt-sco.mono-fallback.monitor>
* index: 1
      name: <alsa_input.platform-sound-bt-sco.mono-fallback>
  index: 2
      name: <alsa_input.platform-sound-micfil.stereo-fallback>
  index: 3
      name: <alsa_output.platform-sound-spdif.stereo-fallback.monitor>
  index: 4
      name: <alsa_input.platform-sound-spdif.stereo-fallback>
  index: 5
      name: <alsa_output.platform-sound-wm8524.stereo-
fallback.monitor>
root@imx8mqevk:~#
```

Step 8 - Set external source device as default source

```
root@imx8mqevk:~# pacmd set-default-source <index number of external
 source device>
```

Step 9 - Enable the audio route

```
root@imx8mqevk:~# pactl load-module module-loopback latency_msec=1
 source=alsa_input.platform-sound-bt-sco.mono-fallback sink=<External sink
 device>

root@imx8mqevk:~# pactl load-module module-loopback latency_msec=1
 source=<External source device> sink=alsa_output.platform-sound-bt-sco.mono-
fallback
```

Step 10 - Enable PCM line management by the host software

The following command controls the PCM lines. This command should be sent when the host wants to control the PCM lines.

The host needs to issue this command to have multiple voices at the same time, to let the controller know that the host software provides the PCM configuration.

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x3f 0x0070 0x01
```

Command output example:

```
< HCI Command: ogf 0x3f, ocf 0x0070, plen 1
  01
> HCI Event: 0x0e plen 4
  01 70 FC 00
```

where:

Parameter	Length	Definition
OGF	1	0x3F
OCF	2	0x0070
Action	1	0x00 = the controller manages the PCM lines(default) 0x01 = the host software manages the PCM lines (enables new use cases)

Step 11 - Connect with a remote Bluetooth device

Refer to Section 4.1 "Scan, pair and connect to Bluetooth classic/Bluetooth LE" and connect with a remote Bluetooth device that supports HFP Audio Gateway.

Step 12 - Verify the Hands-free Audio Gateway Profile capability of the connected Bluetooth device

```
root@imx8mqevk:~# bluetoothctl
[GT-S7560M]# info 04:1B:BA:C7:92:36
```

Output command example:

```
Device 04:1B:BA:C7:92:36 (public)

    Name: GT-S7560M
    Alias: GT-S7560M
    Class: 0x005a020c
    Icon: phone
    Paired: yes
    Trusted: yes
    Blocked: no
    Connected: yes
    LegacyPairing: no
    UUID: OBEX Object Push          (00001105-0000-1000-8000-00805f9b34fb)
    UUID: Audio Source              (0000110a-0000-1000-8000-00805f9b34fb)
    UUID: A/V Remote Control Target (0000110c-0000-1000-8000-00805f9b34fb)
    UUID: Headset AG                (00001112-0000-1000-8000-00805f9b34fb)
    UUID: NAP                       (00001116-0000-1000-8000-00805f9b34fb)
    UUID: Handsfree Audio Gateway   (0000111f-0000-1000-8000-00805f9b34fb)
    UUID: Phonebook Access Server   (0000112f-0000-1000-8000-00805f9b34fb)
    UUID: Message Access Server     (00001132-0000-1000-8000-00805f9b34fb)
    UUID: PnP Information           (00001200-0000-1000-8000-00805f9b34fb)
    Modalias: bluetooth:v0075p0100d0100
[GT-S7560M]#
```

Step 13 - Set SCO voice data path through PCM interface

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x3F 0x001D 0x01
```

Command output example:

```
< HCI Command: ogf 0x3f, ocf 0x001d, plen 1
  01
> HCI Event: 0x0e plen 4
  01 1D FC 00
root@imx8mqevk:~#
```

where:

Parameter	Length	Definition
OGF	1	0x3F
OCF	2	0x001D
Voice path	1	0x00 = Host 0x01 = PCM

Step 14 - Write PCM settings

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x3F 0x0007 0x00
```

Command output example:

```
< HCI Command: ogf 0x3f, ocf 0x0007, plen 1
  00
> HCI Event: 0x0e plen 4
  01 07 FC 00
root@imx8mqevk:~#
```

where:

Parameter	Length	Definition
OGF	1	0x3F
OCF	2	0x0007
PCM_Setting	1	Bit[4]: PCM Clock On • 0 = PCM clock is terminated after last data bit has been transmitted • 1 = make PCM clock available continuously Bit[3]: Reserved Bit[2]: PCM Sync Source • 0 = PCM sync page generated from system clock • 1 = PCM sync page generated from frame clock Bit[1]: Master/Slave • 0 = PCM I/F slave, external PCM clock and synchronization • 1 = PCM I/F master, internal PCM clock and synchronization Bit[0]: PCM Direction • 0 = port A receive, port B transmit • 1 = port A transmit, port B receive

Step 15 - Write PCM sync settings

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x3F 0x0028 0x03 0x00 0x03
```

Command output example:

```
< HCI Command: ogf 0x3f, ocf 0x0028, plen 3
   03 00 03
> HCI Event: 0x0e plen 4
   01 28 FC 00
root@imx8mqevk:~#
```

where:

Parameter	Length	Definition
OGF	1	0x3F
OCF	2	0x0028
PCM Sync Settings 1	1	Default = 0x03 ISR (IramSyncRate) only valid if IF = Host: PCM Sync Settings 1 Bit[0]: • 0 = bursts controlled by Tx or Rx of voice packets • 1 = fixed rate of 8 ksample/s ISS (IramSyncSource) only valid if IF = Host and ISR = Fixed Rate: PCM Sync Settings 1 Bit[1]: • 0 = 0 = ISR not aligned to frame tick • 1 = ISR aligned to frame tick (this field should be set to 1)
PCM Sync Settings 2	2	Default = 0x0000 pcmIfMode in PCM Descriptor PCM Sync Settings 2 Bits[1:0]: • 00 = PCM short sync • 01 = PCM long sync • 10 = I2S audio mode pcmLRCPol in PCM Descriptor PCM Sync Settings 2 Bit[4]: • 0 = LRC is same polarity as PCM sync • 1 = LRC is inverted pcmMClkEn in PCM Descriptor PCM Sync Settings 2 Bit[8]: • 0 = disable generation of PCM main clock • 1 = enable pcm2048MClkSel in PCM Descriptor PCM Sync Settings 2 Bit[9]: • 0 = default • 1 = select 2.048 MHz clock for PCM clock 16k Sync in PCM PCM Sync Settings 2 Bit[10]: • 0 = 8k Sync • 1 = 16k Sync

Step 16 - Write PCM Link settings

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x3F 0x0029 0x04 0x00
```

Command output example:

```
< HCI Command: ogf 0x3f, ocf 0x0029, plen 2
   04 00
> HCI Event: 0x0e plen 4
   01 29 FC 00
root@imx8mqevk:~#
```

where:

Parameter	Length	Definition
OGF	1	0x3F
OCF	2	0x0029
PCM_Link_Setting	2	Bits [13:10]: Each bit corresponds to 1 of 4 PCM timeslots (if 0, the slot is used by the BTU) Bits[9:2]: Defines start of PCM slot relative to start of PCM synchronization (must be greater than the size of the PCM slot) Bits[1:0]: Indicates which PCM slots should be used. Default: • 0x0004 = first SCO link • 0x0045 = second SCO link • 0x0086 = third SCO link

Note: *The PCM Link settings command should be given after HCI reset and before setting up the voice link. Also, if multiple voice links are supported, this command should be given before setting up the voice link with the respective parameters of each voice link.*

Step 17 - Write the voice settings

Use `HCI_Write_Voice_Setting` command defined in Bluetooth Core Specifications v5.2 to write the required voice settings.

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x03 0x0026 0x60 0x00
```

Command output example:

```
< HCI Command: ogf 0x03, ocf 0x0026, plen 2
   60 00
> HCI Event: 0x0e plen 4
   01 26 0C 00
root@imx8mqevk:~#
```

Step 18 - Run Ofono test scripts

Go to the repository where Ofono test scripts are available:

```
root@imx8mqevk:~# cd /usr/lib/ofono/test/
```

Run the dial number test script

```
root@imx8mqevk:/usr/lib/ofono/test# python3 ./dial-number 1234567890
```

Command output example:

```
Using modem /hfp/org/bluez/hci0/dev_04_1B_BA_C7_92_36
/hfp/org/bluez/hci0/dev_04_1B_BA_C7_92_36/voicecall01
root@imx8mqevk:~/test#
```

Step 19 - Initialize the PCM interface

The command below initializes and configures PCM. This command should be sent in any of the following situations:

- To initialize PCM after starting voice call on a particular SCO connection
- To switch call from SCO connection 1 to SCO connection 2
- To route SCO connection 1 voice data to SCO connection 2
- To de-initialize PCM once the voice call is over on a particular SCO connection

To initialize the PCM interface:

```
root@imx8mqevk:/usr/lib/ofono/test# hcitool -i hci0 cmd 0x3f 0x006f
 0x00 0x00 0x08 0x00 0x00 0x00
```

Command output example:

```
< HCI Command: ogf 0x3f, ocf 0x006f, plen 6
   00 00 08 00 00 00
> HCI Event: 0x0e plen 4
   01 6F FC 00
root@imx8mqevk:/usr/lib/ofono/test#
```

Command to de-initialize the PCM interface

```
root@imx8mqevk:/usr/lib/ofono/test# hcitool -i hci0 cmd 0x3f 0x006f
 0x01 0x00 0x08 0x00 0x00 0x00
```

Command output example:

```
< HCI Command: ogf 0x3f, ocf 0x006f, plen 6
   01 00 08 00 00 00
> HCI Event: 0x0e plen 4
   01 6F FC 00
root@imx8mqevk:/usr/lib/ofono/test#
```

where:

Parameter	Length	Definition
OGF	1	0x3F
OCF	2	0x006F
Action	1	0x00 = PCM will be initialized 0x01 = PCM will be de-initialized
Operation mode	1	0x00 = normal mode This mode is used when only 1 voice call needs to be active at a time. Depending on the Action parameter, either the PCM will be initialized or de-initialized. 0x01 = internal loopback This mode is used when there are 2 HF connections and audio data on first connection needs to be routed to second connection. This can be used only when both SCO links are of the same type (that is, NB to NB, or WB to WB only). 0x02 = remote loopback on same link 0x03 = local loopback on same link 0x04 to 0xFF = reserved
SCO Handle1	2	Synchronous connection handle for which PCM configuration is to be done Range from 8 to 10
SCO Handle2	2	Synchronous connection handle for which PCM configuration is to be done Parameter valid only when Operation Mode = 0x01.

Step 20 - Use Ofono test script to hang up a call

Ofono hangup-all script uses the AT command `AT+CHUP` to hangup a call.

```
root@imx8mqevk:/usr/lib/ofono/test# python3 ./hangup-all
root@imx8mqevk:/usr/lib/ofono/test#
```

Step 21 - Use Ofono test script to answer a call

Ofono answer-calls script uses the AT command `ATA` to answer a call.

```
root@imx8mqevk:/usr/lib/ofono/test#  python3 ./answer-calls
```

Command output example:

```
[ /hfp/org/bluez/hci0/dev_04_1B_BA_C7_92_36 ]
[ /hfp/org/bluez/hci0/dev_04_1B_BA_C7_92_36/voicecall01 ] incoming
root@imx8mqevk:/usr/lib/ofono/test#
```

Step 22 - Initialize PCM

Refer to Step 19.

Step 23 - Use Ofono test script to hang up an active call

Ofono hangup-active script uses the AT command `AT+CHUP` to hang up an active call

```
root@imx8mqevk:/usr/lib/ofono/test# python3 ./hangup-active
[ /hfp/org/bluez/hci0/dev_04_1B_BA_C7_92_36/voicecall01 ] active
root@imx8mqevk:/usr/lib/ofono/test#
```

4.4 Object Push Profile

This section explains how to send the file to a remote device over Bluetooth. BlueZ Stack provides the obexctl user space utility to send the files to a Bluetooth device.

Use the following steps to configure Object Push Profile using Obex Control utility (obexctl)

Start the obex daemon on i.MX 8M Quad

```
root@imx8mqevk:~# /usr/libexec/bluetooth/obexd &
    [1] 768
root@imx8mqevk:~#
```

Connect with a remote Bluetooth device that supports Object Push Profile

Refer to Section 4.1 "Scan, pair and connect to Bluetooth classic/Bluetooth LE"

Run the following commands to verify the Object Push Profile capability of the connected Bluetooth device.

```
root@imx8mqevk:~# bluetoothctl
    Agent registered
    [Moto E]# info B4:F5:00:31:CB:4E
    Device B4:F5:00:31:CB:4E (public)
            Name: Moto E
            Alias: Moto E
            Class: 0x005a020c
            Icon: phone
            Paired: yes
            Trusted: yes
            Blocked: no
            Connected: yes
    LegacyPairing: no
            UUID: OBEX Object Push
 (00001105-0000-1000-8000-00805f9b34fb)
            UUID: Audio Source
 (0000110a-0000-1000-8000-00805f9b34fb)
            UUID: A/V Remote Control Target
 (0000110c-0000-1000-8000-00805f9b34fb)
            UUID: A/V Remote Control
 (0000110e-0000-1000-8000-00805f9b34fb)
            UUID: Headset AG
 (00001112-0000-1000-8000-00805f9b34fb)
            UUID: PANU
 (00001115-0000-1000-8000-00805f9b34fb)
            UUID: NAP
 (00001116-0000-1000-8000-00805f9b34fb)
            UUID: Handsfree Audio Gateway
 (0000111f-0000-1000-8000-00805f9b34fb)
            UUID: Phonebook Access Server
 (0000112f-0000-1000-8000-00805f9b34fb)
            UUID: Message Access Server
 (00001132-0000-1000-8000-00805f9b34fb)
            UUID: PnP Information
 (00001200-0000-1000-8000-00805f9b34fb)
            UUID: Generic Access Profile
 (00001800-0000-1000-8000-00805f9b34fb)
            UUID: Generic Attribute Profile
 (00001801-0000-1000-8000-00805f9b34fb)
    Modalias: bluetooth:v000Fp1200d1436
    [Moto E]#
```

Start Obex Control

Use the following commands to start the Obex Control and connect to the remote Bluetooth device paired using Bluetooth Control.

```
root@imx8mqevk:~# obexctl
    [NEW] Client /org/bluez/obex
    [obex]# connect B4:F5:00:31:CB:4E
    Attempting to connect to B4:F5:00:31:CB:4E
    [NEW] Session /org/bluez/obex/client/session0 [default]
    [NEW] ObjectPush /org/bluez/obex/client/session0
    Connection successful
    [B4:F5:00:31:CB:4E]#
```

Send a file from i.MX 8M Quad to the connected remote Bluetooth device

```
[B4:F5:00:31:CB:4E]# send /home/root/sample_audio.wav
    Attempting to send /home/root/sample_audio.wav to /org/bluez/obex/
client/session0
    [NEW] Transfer /org/bluez/obex/client/session0/transfer0
    Transfer /org/bluez/obex/client/session0/transfer0
            Status: queued
            Name: sample_audio.wav
            Size: 1073218
            Filename: /home/root/sample_audio.wav
    Session: /org/bluez/obex/client/session0
    [CHG] Transfer /org/bluez/obex/client/session0/transfer0 Status:
 active
    [CHG] Transfer /org/bluez/obex/client/session0/transfer0
 Transferred: 8030 (@8KB/s 02:12)
    [B4:F5:00:31:CB:4E]#
```

Select ACCEPT or DECLINE on the remote Bluetooth device

Select ACCEPT or DECLINE on the remote Bluetooth device depending on the file to be received. The following logs appear once the file transfer is completed.

```
[CHG] Transfer /org/bluez/obex/client/session1/transfer1 Status:
 complete
    [DEL] Transfer /org/bluez/obex/client/session1/transfer1
    [B4:F5:00:31:CB:4E]#
```

Disconnect and exit from the obexctl

```
[B4:F5:00:31:CB:4E]# disconnect
    Attempting to disconnect to /org/bluez/obex/client/session2
    [DEL] Session /org/bluez/obex/client/session2 [default]
    [DEL] ObjectPush /org/bluez/obex/client/session2
    Disconnection successful
    [obex]# quit
root@imx8mqevk:~#
```

Note: *The Obex Control receiving is not working with the current version 5.50 of BlueZ. It may be resolved in a later version.*

4.5 Human Interface Device Profile

This section provides the configuration steps to connect a Bluetooth keyboard and/or mouse. The Human Interface Device Profile is used to establish the connection.

Use the following steps to configure Human Interface Device (HID) Profile to connect with the Bluetooth device.

Connect with a remote Bluetooth device that supports Human Interface Device profile

Refer to Section 4.1 "Scan, pair and connect to Bluetooth classic/Bluetooth LE"

Verify the Human Interface Device Profile capability of the connected Bluetooth device

```
[Rapoo E6700]# info 6C:5D:63:20:40:AA
        Device 6C:5D:63:20:40:AA
        Name: Rapoo E6700
        Alias: Rapoo E6700
        Class: 0x002540
        Icon: input-keyboard
        Paired: yes
        Trusted: yes
        Blocked: no
        Connected: yes
        LegacyPairing: no
        UUID: Service Discovery Server (00001000-0000-1000-8000-00805f9b34fb)
        UUID: Human Interface Device    (00001124-0000-1000-8000-00805f9b34fb)
        UUID: PnP Information           (00001200-0000-1000-8000-00805f9b34fb)
        Modalias: usb:v0A5Cp8502d011B
        [Rapoo E6700]#
```

The connected keyboard can be used as a normal keyboard now.

4.6 Bluetooth LE device as GATT server

This section shows how to configure the i.MX 8M Quad board as a Bluetooth LE GATT Server using the example of Battery Service registration.

Use the following steps to configure the Battery Service for LE GATT server. To configure more services please refer the specification/gatt.

Start *bluetoothctl* to interact with the Bluetooth daemon from the command line

```
root@imx8mqevk:~# bluetoothctl
    Agent registered
    [bluetooth]#
```

Register the Battery Service

```
[bluetooth]# menu gatt
    Menu gatt:
    [bluetooth]# register-service 0x180F
    [NEW] Primary Service
            /org/bluez/app/service0xbf29850
            0x180F
            Battery Service
    [/org/bluez/app/service0xbf29850] Primary (yes/no): yes
[bluetooth]#
```

Configure the characteristics for Battery Service

```
[bluetooth]# register-characteristic 0x2A19 notify,read
    [NEW] Characteristic
            /org/bluez/app/service0xbf29850/chrc0xbf32890
            0x2A19
            Battery Level
    [/org/bluez/app/service0xbf29850/chrc0xbf32890] Enter value: 0x64
[bluetooth]#
```

Register the Battery Service

```
[bluetooth]# register-application
 0000180F-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 00001112-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000110a-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 00001200-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000110e-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000110b-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000110c-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 00001800-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 00001801-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000180f-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000111e-0000-1000-8000-00805f9b34fb
    Application registered
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 00001112-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000110a-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 00001200-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000110e-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000110b-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000110c-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 00001800-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 00001801-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000180f-0000-1000-8000-00805f9b34fb
    [CHG] Controller 00:50:43:24:34:F7 UUIDs:
 0000111e-0000-1000-8000-00805f9b34fb
    [bluetooth]#
```

Check that the Battery Service was added successfully

```
[bluetooth]# back
    Menu main:
[bluetooth]# show
    Controller 00:50:43:24:34:F7 (public)
            Name: imx8mqevk
            Alias: imx8mqevk
            Class: 0x002c0000
            Powered: yes
            Discoverable: no
    DiscoverableTimeout: 0x000000b4
            Pairable: yes
            UUID: Headset AG
 (00001112-0000-1000-8000-00805f9b34fb)
            UUID: Audio Source
 (0000110a-0000-1000-8000-00805f9b34fb)
            UUID: PnP Information
 (00001200-0000-1000-8000-00805f9b34fb)
            UUID: A/V Remote Control
 (0000110e-0000-1000-8000-00805f9b34fb)
            UUID: Audio Sink
 (0000110b-0000-1000-8000-00805f9b34fb)
            UUID: A/V Remote Control Target
 (0000110c-0000-1000-8000-00805f9b34fb)
            UUID: Generic Access Profile
 (00001800-0000-1000-8000-00805f9b34fb)
            UUID: Generic Attribute Profile
 (00001801-0000-1000-8000-00805f9b34fb)
            UUID: Battery Service
 (0000180f-0000-1000-8000-00805f9b34fb)
            UUID: Handsfree
 (0000111e-0000-1000-8000-00805f9b34fb)
    Modalias: usb:v1D6Bp0246d0532
            Discovering: no
[bluetooth]#
```

Start Bluetooth LE advertising

```
[bluetooth]# advertise on
    [CHG] Controller 00:50:43:24:34:F7 SupportedInstances: 0x04
    [CHG] Controller 00:50:43:24:34:F7 ActiveInstances: 0x01
    Advertising object registered
    Tx Power: off
    Name: off
    Apperance: off
    Discoverable: off
[bluetooth]#
```

Connect to the i.MX 8M Quad board using nRF connect LE Cell Phone application

Check for the available Battery Service in the application.

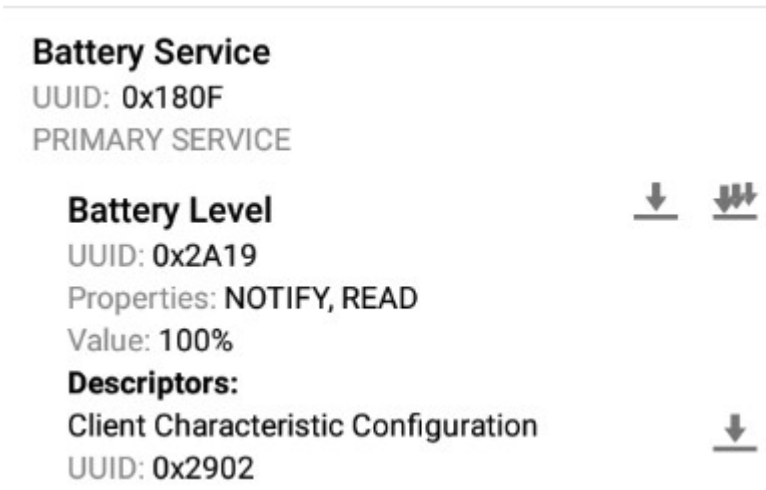

Figure 2. Battery Service

Read operation can be performed using the options from the nRF connect application. The following logs will be displayed on the i.MX 8M Quad console:

```
ReadValue: 41:CB:E2:8F:BA:62 offset 0 link LE
```

4.7 Bluetooth LE device as GATT client

This section describes the procedure to configure the i.MX 8M Quad as a Bluetooth LE client. It also shows the options to list the attributes available in the Bluetooth LE server and access information.

Follow these steps to configure the Bluetooth LE GATT client.

Connect with a remote Bluetooth LE device

Refer to Section 4.1 "Scan, pair and connect to Bluetooth classic/Bluetooth LE".

List the attributes

```
[LE Device]# menu gatt
    Menu gatt
    [LE Device]# list-attributes
Primary Service
    /org/bluez/hci0/dev_5F_82_1E_F1_DE_CC/service0001
    00001801-0000-1000-8000-00805f9b34fb
    Generic Attribute Profile
Characteristic
/org/bluez/hci0/dev_5F_82_1E_F1_DE_CC/service0001/char0002
    00002a05-0000-1000-8000-00805f9b34fb
    Service Changed
[LE Device]#
```

Retrieve the attribute information

```
[LE Device]# attribute-info
    /org/bluez/hci0/dev_5F_82_1E_F1_DE_CC/service0001/char0002
    Characteristic - Service Changed
    UUID: 00002a05-0000-1000-8000-00805f9b34fb
    Service: /org/bluez/hci0/dev_5F_82_1E_F1_DE_CC/service0001
    Notifying: no
    Flags: indicate
[LE Device]#
```

4.8 Human Interface Device Service

This section describes the configuration steps to connect the NXP-based Bluetooth LE module with a Bluetooth Low Energy device that supports Human Interface Device Profile.

Follow these steps to configure Human Interface Device Service.

Connect with the Bluetooth LE device

Refer to Section 4.1 "Scan, pair and connect to Bluetooth classic/Bluetooth LE".

Verify the Human Interface Device Service capability of the connected Bluetooth LE device

```
root@imx8mqevk:~# bluetoothctl
Agent registered
[LE Device]# info D4:18:20:A0:48:5F
Device D4:18:20:A0:48:5F (public)
    Name: LE Device
    Alias: LE Device
    Paired: yes
    Trusted: yes
    Blocked: no
    Connected: yes
    LegacyPairing: no
    UUID: Generic Access Profile
 (00001800-0000-1000-8000-00805f9b34fb)
    UUID: Generic Attribute Profile
 (00001801-0000-1000-8000-00805f9b34fb)
    UUID: Device Information
 (0000180a-0000-1000-8000-00805f9b34fb)
    UUID: Battery Service
 (0000180f-0000-1000-8000-00805f9b34fb)
    UUID: Human Interface Device
 (00001812-0000-1000-8000-00805f9b34fb)
    UUID: Vendor specific
 (3dda0001-957f-7d4a-34a6-74696673696d)
    ManufacturerData Key: 0x00df
    ManufacturerData Value:
57 30 46 39 30 32 31 39 59 32                    W0F90219Y2
[LE Device]#
```

The connected keyboard can be used as a normal keyboard now.

4.9 RF test mode

This section describes the commands to perform the RF Test for Bluetooth Classic and Bluetooth Low Energy on the i.MX 8M Quad EVK.

4.9.1 Bluetooth Classic

4.9.1.1 Enable test mode for qualification

The `HCI_Enable_Device_Under_Test_Mode` command allows the local BR/EDR controller to enter test mode via LMP test commands for BR/EDR controllers. Read more about the command in Bluetooth Core Specification v5.2.

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x06 0x0003
```

Command output example:

```
< HCI Command: ogf 0x06, ocf 0x0003, plen 0
< HCI Command: Enable Device Under Test Mode (0x06|0x0003) plen 0
> HCI Event: 0x0e plen 4
   01 03 18 00
> HCI Event: Command Complete (0x0e) plen 4
   Enable Device Under Test Mode (0x06|0x0003) ncmd 1
   Status: Success (0x00)
root@imx8mqevk:~#
```

4.9.1.2 Set the receive test parameters

This command sets the receive test parameters. An HCI reset command is required after this test to resume normal Bluetooth operation.

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x3F 0x0018 0x01 0x01 0x02 0x0F 0x02 0x00
 0x00 0x00 0x01 0x00 0x01 0x6C 0x6C 0xC6 0x56 0x39 0x20 0x00
```

Command output example:

```
< HCI Command: ogf 0x3f, ocf 0x0018, plen 18
   01 01 02 0F 02 00 00 00 01 00 01 6C 6C C6 56 39 20 00
< HCI Command: Vendor (0x3f|0x0018) plen 18
   ->01 01 02 0f 02 00 00 00 01 00 01 6c 6c c6 56 39
   ->20 00
> HCI Event: 0x0e plen 4
   ->01 18 FC 00
> HCI Event: Command Complete (0x0e) plen 4
     Vendor (0x3f|0x0018) ncmd 1
     Status: Success (0x00)
root@imx8mqevk:~#
```

where:

Parameter	Length	Definition
OGF	1	0x3F
OCF	2	0x0018

Parameter	Length	Definition
TestScenario	1	Test scenario • 0x01 = receiver test, 0–pattern • 0x02 = receiver test, 1–pattern • 0x03 = receiver test, 1010–pattern • 0x04 = receiver test, PRBS–pattern • 0x09 = receiver test, 1111 0000–pattern • 0xFF = abort test mode
TxFrequency	1	Transmit frequency f = (2402+k) MHz
RxFrequency	1	Receive frequency f = (2402+k) MHz
TestPacketType	1	Test Packet Type • 0x03 = DM1 • 0x04 = DH1 • 0x0A = DM3 • 0x0B = DH3 • 0x0E = DM5 • 0x0F = DH5 • 0x14 = 2-DH1 • 0x18 = 3-DH1 • 0x1A = 2-DH3 • 0x1B = 3-DH3 • 0x1E = 2-DH5 • 0x1F = 3-DH5
Expected Number of Packets	4	--
Length of Test Data	2	Should not be bigger than the maximum size of the specified test packet type
Tx AM Address	1	Default = 0x01
Transmitter BD Address	6	This is used to derive the access code
Report error packets	1	Report Error Packets • 0x00 = none (default) • 0x01 to 0xFE = number of packets to report

4.9.1.3 Set the transmit test parameters

This command sets the transmit test parameters. An HCI Reset command is required after this test to resume normal Bluetooth operations.

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x3F 0x0019 0x80 0x80 0x80 0x80 0x01 0x00
 0x01 0x02 0x01 0x0F 0x01 0x00 0x00 0x01 0x00 0x00 0x00 0x04
```

Command output example:

```
< HCI Command: ogf 0x3f, ocf 0x0019, plen 18
     80 80 80 80 01 00 01 02 01 0F 01 00 00 01 00 00 00 04
< HCI Command: Vendor (0x3f|0x0019) plen 18

        80 80 80 80 01 00 01 02 01 0f 01 00 00 01 00 00
        00 04

< HCI Event: 0x0e plen 4
     01 19 FC 00
> HCI Event: Command Complete (0x0e) plen 4

       Vendor (0x3f|0x0019) ncmd 1
       Status: Success (0x00)
> HCI Event: Vendor (0xff) plen 6

       19 00 01 00 00 00
```

where:

Parameter	Length	Definition
OGF	1	0x3F
OCF	2	0x0019
RxOnStart	1	These 4 parameters should be set to 0x80.
SyntOnStart 1	1	
TxOnStart	1	
PhdOffStart	1	
TestScenario	1	Test scenario • 0x01 = PATTERN_00 (data pattern: 0x00) • 0x02 = PATTERN_FF (data pattern: 0xFF) • 0x03 = PATTERN_55 (data pattern: 0x55) • 0x04 = PATTERN_PRBS (data pattern: 0xFE) • 0x09 = PATTERN_0F (data pattern: 0x0F) • 0xFF = exit test
HoppingMode	1	• 0x00 = fix frequency • 0x01 = hopping set
TxChannel	1	Transmit Frequency = (2402+k) MHz, where k is the value of TxChannel
RxChannel	1	Receive Frequency = (2402+k) MHz, where k is the value of RxChannel
TxTestInterval	1	Poll interval in frames for the link (units, 1.25 ms)

Parameter	Length	Definition
PacketType	1	Transmit Packet Type • 0x03 = DM1 • 0x04 = DH1 • 0x0A = DM3 • 0x0B = DH3 • 0x0E = DM5 • 0x0F = DH5 • 0x14 = 2-DH1 • 0x18 = 3-DH1 • 0x1A = 2-DH3 • 0x1B = 3-DH3 • 0x1E = 2-DH5 • 0x1F = 3-DH5
Length	2	Length of test data
Whitening	1	• 0x00 = disabled • 0x01 = enabled
Number of Test Packets	4	0 = infinite (default)
Tx Power	1	Signed value of Tx power (dBm) Range = -20 dBm to 12 dBm (default = 4 dBm)

4.9.2 Bluetooth Low Energy (LE)

Two commands are used to test the RF for Bluetooth Low Energy (LE): LE Receiver Test and LE Transmitter Test. And one command is used to end the test.

4.9.2.1 Bluetooth LE receiver test

To start a test where the DUT receives test reference packets at a fixed interval, use `LE Receiver Test[V2]` command. For more details on the command please refer to Bluetooth Core Specification v5.2.

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x08 0x0033 0x01 0x02 0x00
```

Command output example:

```
< HCI Command: ogf 0x08, ocf 0x0033, plen 3
  01 02 00
< HCI Command: LE Enhanced Receiver Test (0x08|0x0033) plen 3

      RX channel frequency: 2404 MHz (0x01)
      PHY: LE 2M (0x02)
      Modulation index: Standard (0x00)
> HCI Event: 0x0e plen 4
> HCI Event: Command Complete (0x0e) plen 4

01 33 20 00
LE Enhanced Receiver Test (0x08|0x0033) ncmd 1
      Status: Success (0x00)
root@imx8mqevk:~#
```

4.9.2.2 Bluetooth LE transmitter test

To start a test where the DUT generates test reference packets at a fixed interval, use `LE Transmitter Test[V2]` command. For more details on the command please refer to Bluetooth Core Specification v5.2.

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x08 0x0034 0x01 0x01 0x00 0x02
```

Command output example:

```
< HCI Command: ogf 0x08, ocf 0x0034, plen 4
  01 01 00 02
< HCI Command: LE Enhanced Transmitter Test (0x08|0x0034) plen 4

        TX channel frequency: 2404 MHz (0x01)
        Test data length: 1 bytes
        Packet payload: 0x00
        PHY: LE 2M (0x02)
> HCI Event: Command Complete (0x0e) plen 4

> HCI Event: 0x0e plen 4
        LE Enhanced Transmitter Test (0x08|0x0034) ncmd 1
        01 34 20 00
        Status: Success (0x00)
root@imx8mqevk:~#
```

4.9.2.3 End Bluetooth LE test

To stop any test which is in progress, use `LE Test End` command. For more details on the command please refer to Bluetooth Core Specification v5.2.

```
root@imx8mqevk:~# hcitool -i hci0 cmd 0x08 0x001F
```

Command output example:

```
< HCI Command: ogf 0x08, ocf 0x001f, plen 0
< HCI Command: LE Test End (0x08|0x001f) plen 0

> HCI Event: 0x0e plen 6
   01 1F 20 00 00 00
> HCI Event: Command Complete (0x0e) plen 6

   LE Test End (0x08|0x001f) ncmd 1
   Status: Success (0x00)
   Number of packets: 0
root@imx8mqevk:~#
```

5 Bluetooth debugging

This section details the Bluetooth Protocol and Driver Debugging using *hcidump*, *btmon* and *dmesg* logs.

5.1 Bluetooth protocol debugging

5.1.1 Capture the HCI logs using *hcidump*

Follow these steps to capture the HCI logs between the i.MX 8M Quad EVK and NXP-based module using *hcidump* utility.

Start capturing HCI logs

Use the following command to start the HCI logs and store these in the file. Use Frontline Bluetooth Software tool to open the log file:

```
root@imx8mqevk:~# hcidump -i hci0 -w sample_hci.log &
[1] 770
HCI sniffer - Bluetooth packet analyzer ver 5.50
btsnoop version: 1 datalink type: 1002
device: hci0 snap_len: 1500 filter: 0x0
root@imx8mqevk:~#
```

Start capturing HCI logs and store in text format

```
root@imx8mqevk:~# hcidump -i hci0 -Xt | tee sample_hci.txt &
[3] 825
root@imx8mqevk:~#
```

Connect with a Bluetooth Classic/LE device

Refer to Section 4.1 "Scan, pair and connect to Bluetooth classic/Bluetooth LE"

Stop capturing HCI logs

```
root@imx8mqevk:~# killall hcidump
[1]+  Terminated              hcidump -i hci0 -w sample_hci.log
root@imx8mqevk:~#
```

Copy the *sample_hci.log* file to the host PC

Note: *hcidump utility may crash during testing. If this occurs, use btmon utility.*

5.1.2 Capture the HCI Logs using btmon

Follow these steps to capture the HCI logs between the i.MX 8M Quad EVK and NXP-based wireless module using *btmon* utility.

Use the following command to start the HCI logs and store these in the file. You can use Wireshark tool and Wireshark Software to open the log file:

```
root@imx8mqevk:~# btmon -w sample_hci.log &
[1] 780
Bluetooth monitor ver 5.50
= Note: Linux version 5.4.24-2.2.0+gd60d1df535b1 (aarch64)
 0.926150
= Note: Bluetooth subsystem version 2.22
 0.926156
= New Index: 00:50:43:24:34:F7 (Primary,UART,hci0)
 [hci0] 0.926159
= Open Index: 00:50:43:24:34:F7
 [hci0] 0.926161
= Index Info: 00:50:43:24:34:F7 (Marvell Technology Group Ltd.)
 [hci0] 0.926162
@ MGMT Open: bluetoothd (privileged) version 1.14
 {0x0001} 0.926164
@ MGMT Open: btmon (privileged) version 1.14
 {0x0002} 0.926262
root@imx8mqevk:~#
```

Start capturing HCI logs and store in text format

```
root@imx8mqevk:~# btmon | tee sample_hci.txt &
[1] 780
Bluetooth monitor ver 5.50
= Note: Linux version 5.4.24-2.2.0+gd60d1df535b1 (aarch64)
 0.926150
= Note: Bluetooth subsystem version 2.22
 0.926156
= New Index: 00:50:43:24:34:F7 (Primary,UART,hci0)
 [hci0] 0.926159
= Open Index: 00:50:43:24:34:F7
 [hci0] 0.926161
= Index Info: 00:50:43:24:34:F7 (Marvell Technology Group Ltd.)
 [hci0] 0.926162
@ MGMT Open: bluetoothd (privileged) version 1.14
 {0x0001} 0.926164
@ MGMT Open: btmon (privileged) version 1.14
 {0x0002} 0.926262
root@imx8mqevk:~#
```

Connect with a Bluetooth Classic/LE device

Refer to Section 4.1 "Scan, pair and connect to Bluetooth classic/Bluetooth LE"

Stop capturing HCI logs

```
root@imx8mqevk:~# killall btmon
root@imx8mqevk:~#
[1]+  Done                    btmon -w sample_hci.log
root@imx8mqevk:~#
```

Copy the *sample_hci.log* file to the host PC

5.1.3 Extract the Link Key for remote Bluetooth Classic/Bluetooth LE devices

Follow these steps to get the link key for paired Bluetooth Classic/Bluetooth LE devices:

Get the information for Bluetooth Classic and/or Bluetooth LE devices

```
root@imx8mqevk:~# cat /var/lib/bluetooth/<DUT BD Address>/<Remote BD
 Address>/info
```

Example - Get the Link Key for a Bluetooth device

```
root@imx8mqevk:~# cat /var/lib/bluetooth/00:50:43:24:34:F7/
B4:F5:00:31:CB:4E/info
        [LinkKey]
        Key=7CC2A6C9AA14F799F9E596B90FC973BC
        Type=8
        PINLength=0
```

Example - Get the Long Term Key for a Bluetooth LE device

```
root@imx8mqevk:~# cat /var/lib/bluetooth/00:50:43:24:34:F7/
D4:18:20:A0:48:5F/info
        [LongTermKey]
        Key=AA3E7062B5B9415F9F27A5010690412A
        Authenticated=0
        EncSize=16
        EDiv=10551
        Rand=730759945871301339
```

5.2 Bluetooth driver debugging

Run the following command to get and analyze the Bluetooth logs using *dmesg* utility.

Use *Dmesg* utility to display the driver logs

```
root@imx8mqevk:~# dmesg | grep Bluetooth
    [    0.225403] Bluetooth: Core ver 2.22
    [    0.225431] Bluetooth: HCI device and connection manager
 initialized
    [    0.225440] Bluetooth: HCI socket layer initialized
    [    0.225446] Bluetooth: L2CAP socket layer initialized
    [    0.225458] Bluetooth: SCO socket layer initialized
    [    1.775437] Bluetooth: HCI UART driver ver 2.3
    [    1.779894] Bluetooth: HCI UART protocol H4 registered
    [    1.785041] Bluetooth: HCI UART protocol BCSP registered
    [    1.790390] Bluetooth: HCI UART protocol LL registered
    [    1.795534] Bluetooth: HCI UART protocol ATH3K registered
    [    1.800958] Bluetooth: HCI UART protocol Three-wire (H5)
 registered
    [    1.807322] Bluetooth: HCI UART protocol Broadcom registered
    [    1.813003] Bluetooth: HCI UART protocol QCA registered
    [    2.284553] Bluetooth: RFCOMM TTY layer initialized
    [    2.289447] Bluetooth: RFCOMM socket layer initialized
    [    2.294614] Bluetooth: RFCOMM ver 1.11
    [    2.298396] Bluetooth: BNEP (Ethernet Emulation) ver 1.3
    [    2.303714] Bluetooth: BNEP filters: protocol multicast
    [    2.308948] Bluetooth: BNEP socket layer initialized
    [    2.313918] Bluetooth: HIDP (Human Interface Emulation) ver 1.2
    [    2.319849] Bluetooth: HIDP socket layer initialized
```

6 Contact information

Use the following links for more details, queries and support.

Website homepage: nxp.com

Web support: nxp.com/support

NXP community: community.nxp.com

iMX community: community.nxp.com/community/imx

7 Acronyms and abbreviations

Table 5. Acronyms and abbreviations

Acronym	Definition
A2DP	Advanced audio distribution profile
AP	Access point
AXIHP	Advanced extensible interface high performance
BD	Bluetooth device
BSSID	Basic service set identifiers
DDR	Double data rate
DUT	Device under test
ED	Energy detection
EVK	Evaluation kit
GATT	Generic attribute profile
HID	Human interface device
HFP	Hands free profile
OCF	Opcode command field
OGF	Opcode group field
OPP	Object push profile
STA	Station
STBC	Space time block coding
TP	Throughput
WLAN	Wireless local area network
WoWLAN	Wake on wireless local area network
WPA	Wi-Fi protected access

8 Legal information

8.1 Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

8.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors. In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer. In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages. Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

8.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

Tables

Figures

Contents

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 29 January 2021
Document identifier: UM11490