89359 SDIO User Manual

Driver Bring-up and Supplicant Commands

Table of Contents

Bringup of Driver

Compilation flags of Wifi Driver

Below are the compilation flags to be added or removed to DHDCFLAGS at bcmdhd/Makefile.

DHDCFLAGS for DT or Device-Tree architecture:

1. Add “-DCONFIG_DTS” flag to DHDCFLAGS in bcmdhd/Makefile.
2. Modify DTS file accordingly for wlreg_on and Out-of band interrupt line.

For Ex:

```
wlreg_on: fixedregulator@2 {
        compatible = "regulator-fixed";
        regulator-name = "wlreg_on";
        gpio = <&gpio6 27 0>;
        startup-delay-us = <600000>;
        enable-active-high;
};

bcmdhd_wlan_0: bcmdhd_wlan@0 {
        compatible = "android,bcmdhd_wlan";
        gpios = <&gpio6 28 0>;
        wlreg_on-supply = <&wlreg_on>;
};
```

DHDCFLAGS for Non-DT or Board file architecture:

1. Remove “-DCONFIG_DTS” flag to DHDCFLAGS in bcmdhd/Makefile.
2. Modify board file as per below reference.

You can follow below url for your reference:

https://android.googlesource.com/kernel/omap.git/+/android-omap-tuna-3.0-ics-mr1/arch/arm/mach-omap2/board-tuna-wifi.c

There are three important code areas in "board-xx.c" files.

1. Telling to the platform that how are we going to control this device and device details.

```
static struct wifi_platform_data manta_wifi_control = {
        .set_power              = manta_wifi_power,
        .set_reset              = manta_wifi_reset,
        .set_carddetect         = manta_wifi_set_carddetect,
        .mem_prealloc           = NULL,
        .get_mac_addr           = manta_wifi_get_mac_addr,
        .get_country_code       = manta_wifi_get_country_code,
};

static struct platform_device manta_wifi_device = {
        .name           = "bcmdhd_wlan",
        .id             = 1,
        .num_resources  = ARRAY_SIZE(manta_wifi_resources),
        .resource       = manta_wifi_resources,
        .dev            = {
                .platform_data = &manta_wifi_control,
        },
};
```

2. Implementation of important functions like; static int manta_wifi_power(int on) static int manta_wifi_set_carddetect(int val) etc..

3. Registering our device with Platform:

```
int __init tuna_wlan_init(void)
{
        pr_debug("%s: start\n", __func__);
        tuna_wlan_gpio();
        tuna_init_wifi_mem();
        platform_device_register(&omap_vwlan_device);
        return platform_device_register(&tuna_wifi_device);
}
```

DHDCFLAGS for Interrupt mechanism:

Broadcom Wifi driver provides two ways to configure Interrupt from chip.

1. **In-band Interrupt**: In this mechanism, sdio interrupt is triggered through DAT1 line (normal SDIO interrupt raising mechanism).
   Platforms based on this interrupt mechanism need to add **SDIO_ISR_THREAD** to DHDCFLAGS and remove **OOB_INTR_ONLY** and **HW_OOB** to DHDCFLAGS.

2. **Out-of band Interrupt**: In this mechanism, chip will trigger an interrupt through GPIO line which is specifically meant for interrupts.
   Platforms based on this interrupt mechanism need to add **OOB_INTR_ONLY** and **HW_OOB** to DHDCFLAGS and remove **SDIO_ISR_THREAD** to DHDCFLAGS.

DHDCFLAGS based on driver build-type:

1. Add below Kernel configurations for Built-in driver:
   a. **CONFIG_BCMDHD=y**
   b. **CONFIG_BCMDHD_SDIO=y**
   c. **CONFIG_BCM4359=y**
2. Add below Kernel configurations for Module type driver:
   a. **CONFIG_BCMDHD=m**
   b. **CONFIG_BCMDHD_SDIO=y**
   c. **CONFIG_BCM4359=y**

DHDCFLAGS based on kernel version:

1. Use **WL_CFG80211_ACL** for kernel version above 3.4.0
2. Use **CUSTOM_FORCE_NODFS_FLAG** if your kernel supports wifi control function framework (CONFIG_WIFI_CONTROL_FUNC) and get_country_code function accepts flags as below:
   *void *(*get_country_code)(char *ccode, u32 flags);*
   This is generally found in android kernel version 3.10.58 and above.
3. Use **CUSTOM_COUNTRY_CODE** if your kernel supports wifi control function framework (CONFIG_WIFI_CONTROL_FUNC) and get_country_code function accepts flags as below:
   *void *(*get_country_code)(char *ccode, u32 flags);*
   This is generally found in android kernel version 3.10.40 and above.

DHDCFLAGS based on requirement:

1. DHDCFLAGS += **ENABLE_INSMOD_NO_FW_LOAD**
   BCMDHD module would be loaded at system boot or on insmod/modprobe, but firmware & nvram will be downloaded only after *'ifconfig wlan0 up'* or when *wpa_supplicant* is run

DHDCFLAGS for IMX6 boards:

1. DHDCFLAGS += **CUSTOMER_IMX**
   Use this flag for IMX boards. This flag is needed when wifi control function framework (CONFIG_WIFI_CONTROL_FUNC) is not supported.
   CONFIG_WIFI_CONTROL_FUNC is by default supported in android and not supported in linux kernel.
   BCMDHD calls wifi_card_detect() hook provided by IMX SDIO host-controller driver to trigger mmc_detect_change().
2. DHDCFLAGS += CONFIG_DTS
3. Even if you have Hardware support for Out-Of-Band Interrupt, for the first time bringup of wlan, please use In-Band Interrupt.
   Refer "*DHDCFLAGS for Interrupt mechanism*" section of this document to enable In-band interrupt in DHD.
4. Once you are able to bring-up wlan with In-Band interrupt mode, if you have hardware support for Out-of-Band Interrupt, then go for bringup with Out-of-Band interrupt.
   Refer "*DHDCFLAGS for Interrupt mechanism*" section of this document to enable Out-of-band interrupt in DHD.

DHDCFLAGS based on sdio_reset_comm() definition in kernel:

1. If kernel does not define sdio_reset_comm() function then add **NO_SDIO_RESET** to DHDCFLAGS.
   This is generally found in linux kernel which are not based on android linux kernel.

Compilation Steps:

1. Copy wifi driver to kernel source tree and cd to *<kernel-dir>*
   cp -fr bcmdhd <kernel-dir>/drivers/net/wireless/
2. For built-in to kernel build type:
   make ARCH=<arch> CROSS_COMPILE=<COMPILER_PATH> -j4
3. For built-in to kernel build type:
   make ARCH=<arch> CROSS_COMPILE=<COMPILER_PATH> modules -j4

Loading driver:

For module-type driver:

1. Copy firmware binary and nvram file to target filesystem.
2. Use nvram file for your wlan hardware module. If you do not have one, please request for it from Module-Vendor or Cypress HW-AE/FAE.
3. Copy built bcmdhd.ko to target filesystem
4. Load bcmdhd.ko and firmware/nvram as below:
   insmod bcmdhd.ko firmware_path=<firmware_path> nvram_path=<nvram_path> clm_path=<path-to-clm_blob>/4359b1.clm_blob
5. After successful loading of driver, you can verify *wlan0* network interface created.
6. Bring-up interface using *ifconfig wlan0 up*
7. If you have wl-utility built for your platform, you can run scan to check the interface is up and running.
   wlutil -i wlan0 up
   wlutil -i wlan0 scan
   wlutil -i wlan0 scanresults

For built-in driver:

1. Copy firmware binary and nvram file to target filesystem
   For ex: *cp fw_bcmdhd_hu.bin /system/vendor/firmware/*
   cp nvram.txt /system/vendor/firmware/
2. Specify firmware and nvram paths in kernel configuration file using below CONFIGs.
   CONFIG_BCMDHD_FW_PATH="*/system/vendor/firmware/fw_bcmdhd_hu.bin*"
   CONFIG_BCMDHD_NVRAM_PATH="*/system/vendor/firmware/nvram.txt*"
3. After successful loading of driver, you can verify *wlan0* network interface created.
   For ex: On Linux platforms: *ifconfig wlan0*
   On Android platforms: *netcfg*
4. Bring-up interface using *ifconfig wlan0 up*
5. If you have wl-utility built for your platform, you can run scan to check the interface is up and running.
   wlutil -i wlan0 up
   wlutil -i wlan0 scan
   wlutil -i wlan0 scanresults

Compat Wireless for kernel version less than 3.5

For kernels less than 3.5, if needed use compat wireless. And below are the steps for how to compile compat wireless:

Howtocompatwireless for android and non x86 platform:

Get compat-wireless from the package:

Copy **compat-wireless-3.5.4-1-brcm.tar.bz2** to **<workspace>**

Build compat-wirless package:

1. tar -xjvf compat-wireless-3.5.4-1-brcm.tar.bz2
2. cd compat-wireless-3.5.4-1-brcm
3. export ARCH=<arm>
4. export CROSS_COMPILE=<cross compiler full path/arm-eabi->./scripts/driver-select brcm80211
5. make ARCH=arm CROSS_COMPILE=<cross compiler full path/arm-eabi-> LIB=<target kernel source full path> KLIB_BUILD=<target kernel source full path>
6. cp ./net/wireless/lib80211.ko <target file system>
7. cp ./net/wireless/cfg80211.ko <target file system>
8. cp ./compat/compat.ko <target file system>

Load compat-wireless:

The steps listed below are made under the assumption that compat-wireless kernel modules are already copied to target and target command shell is up and running.
insmod <target file system copied path/compat.ko>
insmod <target file system copied path/lib80211.ko>
insmod <target file system copied path/cfg80211.ko>

Get DHD Driver:

cp bcmdhd.tgz <workspace>
tar –xzvf bcmdhd.tgz

Build DHD driver:

1. **For kernel below linux-3.8.0**:
2. Disable macro definition - *WL_CFG80211_P2P_DEV_IF* and enable macro definition - *WL_ENABLE_P2P_IF* and WL_COMPAT_WIRELESS in the driver Makefile. Also do not enable *WL_IFACE_COMB_NUM_CHANNELS* in the driver Makefile.
3. cd <driver_directory>
4. export COMPAT_WIRELESS=<compat-wireless source full path>
5. make ARCH=arm CROSS_COMPILE=<cross compile full path/arm-eabi-> -C <target kernel source full path> M=$PWD

Load DHD driver:

insmod <path-to-bcmdhd-binary>/bcmdhd.ko> firmware_path=<path-to-firmware-binary>/<firmware name> nvram_path=<path-to-nvram-file>/nvram.txt clm_path=<path-to-clm_blob>/4359b1.clm_blob

ifconfig wlan0 up
wl scan
wl scanresults

RSDB Concurrency Scenarios - Bringup commands

SoftAP bringup commands

Scenario	Steps
1. 5G Soft-AP	wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0 wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm0 remove_net all wpa_cli> IFNAME=bcm0 add_net wpa_cli> IFNAME=bcm0 set_net 0 ssid '"BRCM_5G"' wpa_cli> IFNAME=bcm0 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm0 set_net 0 frequency 5180 wpa_cli> IFNAME=bcm0 set_net 0 mode 2 wpa_cli> IFNAME=bcm0 select_net 0 ifconfig bcm0 192.168.20.50
2. 2G Soft-AP	wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm1 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 remove_net all wpa_cli> IFNAME=bcm1 add_net wpa_cli> IFNAME=bcm1 set_net 0 ssid ‘”BRCM_2G”’ wpa_cli> IFNAME=bcm1 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm1 set_net 0 frequency 2437 wpa_cli> IFNAME=bcm1 set_net 0 mode 2 wpa_cli> IFNAME=bcm1 select_net 0 ifconfig bcm1 192.168.30.40

Note: When creating AP + AP, 5G AP should be started first since BT (being always on 2G) is also connected to core0. Incase 2G AP is started first it will come up on core0 and might face interference with BT on 2G.

SoftAP bringup commands with Security

Scenario	Steps
1. 2G Soft-AP	**WPA-PSK-TKIP:** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm1 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 remove_net all wpa_cli> IFNAME=bcm1 add_net wpa_cli> IFNAME=bcm1 set_net 0 ssid ‘"BRCM_2G"’ wpa_cli> IFNAME=bcm1 set_net 0 key_mgmt WPA-PSK wpa_cli> IFNAME=bcm1 set_net 0 pairwise TKIP wpa_cli> IFNAME=bcm1 set_net 0 psk ‘"9876543210"’ wpa_cli> IFNAME=bcm1 set_net 0 frequency 2437 wpa_cli> IFNAME=bcm1 set_net 0 mode 2 wpa_cli> IFNAME=bcm1 select_net 0 wpa_cli> IFNAME=bcm1 status **WPA-PSK-AES:** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm1 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 remove_net all wpa_cli> IFNAME=bcm1 add_net wpa_cli> IFNAME=bcm1 set_net 0 ssid ‘"BRCM_2G"’ wpa_cli> IFNAME=bcm1 set_net 0 key_mgmt WPA-PSK wpa_cli> IFNAME=bcm1 set_net 0 pairwise CCMP wpa_cli> IFNAME=bcm1 set_net 0 psk ‘"9876543210"’ wpa_cli> IFNAME=bcm1 set_net 0 frequency 2437 wpa_cli> IFNAME=bcm1 set_net 0 mode 2 wpa_cli> IFNAME=bcm1 select_net 0 wpa_cli> IFNAME=bcm1 status **WPA2-PSK-TKIP:** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm1 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 remove_net all wpa_cli> IFNAME=bcm1 add_net wpa_cli> IFNAME=bcm1 set_net 0 ssid "BRCM_2G" wpa_cli> IFNAME=bcm1 set_net 0 proto WPA2 wpa_cli> IFNAME=bcm1 set_net 0 key_mgmt WPA-PSK wpa_cli> IFNAME=bcm1 set_net 0 pairwise TKIP

	wpa_cli> IFNAME=bcm1 set_net 0 psk ‘"9876543210"’ wpa_cli> IFNAME=bcm1 set_net 0 frequency 2437 wpa_cli> IFNAME=bcm1 set_net 0 mode 2 wpa_cli> IFNAME=bcm1 select_network 0 wpa_cli> IFNAME=bcm1 status **WPA2-PSK-AES:** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm1 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 remove_net all wpa_cli> IFNAME=bcm1 add_net wpa_cli> IFNAME=bcm1 set_net 0 ssid "BRCM_2G" wpa_cli> IFNAME=bcm1 set_net 0 proto WPA2 wpa_cli> IFNAME=bcm1 set_net 0 key_mgmt WPA-PSK wpa_cli> IFNAME=bcm1 set_net 0 pairwise CCMP wpa_cli> IFNAME=bcm1 set_net 0 psk ‘"9876543210"’ wpa_cli> IFNAME=bcm1 set_net 0 frequency 2437 wpa_cli> IFNAME=bcm1 set_net 0 mode 2 wpa_cli> IFNAME=bcm1 select_net 0 wpa_cli> IFNAME=bcm1 status **WAPI-PSK:** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm1 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 remove_net all wpa_cli> IFNAME=bcm1 add_net wpa_cli> IFNAME=bcm1 set_net 0 ssid ‘"wapiap2G_bcm1"’ wpa_cli> IFNAME=bcm1 set_net 0 key_mgmt WAPI-PSK wpa_cli> IFNAME=bcm1 set_net 0 proto WAPI wpa_cli> IFNAME=bcm1 set_net 0 psk ‘”1234506789”’ wpa_cli> IFNAME=bcm1 set_net 0 frequency 2412 wpa_cli> IFNAME=bcm1 set_net 0 mode 2 wpa_cli> IFNAME=bcm1 select_net 0 wpa_cli> IFNAME=bcm1 status
2. 5G Soft-AP	**WPA-PSK-TKIP:** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0 wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm0 remove_network all wpa_cli> IFNAME=bcm0 add_network wpa_cli> IFNAME=bcm0 set_network 0 ssid ‘"BRCM_5G"’

wpa_cli> IFNAME=bcm0 set_network 0 key_mgmt WPA-PSK
wpa_cli> IFNAME=bcm0 set_network 0 pairwise TKIP
wpa_cli> IFNAME=bcm0 set_network 0 psk ‘"9876543210"’
wpa_cli> IFNAME=bcm0 set_net 0 frequency 5745
wpa_cli> IFNAME=bcm0 set_net 0 mode 2
wpa_cli> IFNAME=bcm0 select_network 0
wpa_cli> IFNAME=bcm0 status

WPA-PSK-AES:

wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0

wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211
wpa_cli> IFNAME=bcm0 remove_network all
wpa_cli> IFNAME=bcm0 add_network
wpa_cli> IFNAME=bcm0 set_network 0 ssid ‘"BRCM_5G"’
wpa_cli> IFNAME=bcm0 set_network 0 key_mgmt WPA-PSK
wpa_cli> IFNAME=bcm0 set_network 0 pairwise CCMP
wpa_cli> IFNAME=bcm0 set_network 0 psk ‘"9876543210"’
wpa_cli> IFNAME=bcm0 set_net 0 frequency 5745
wpa_cli> IFNAME=bcm0 set_net 0 mode 2
wpa_cli> IFNAME=bcm0 select_network 0
wpa_cli> IFNAME=bcm0 status

WPA2-PSK-TKIP:

wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0

wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211
wpa_cli> IFNAME=bcm0 remove_network all
wpa_cli> IFNAME=bcm0 add_network
wpa_cli> IFNAME=bcm0 set_network 0 ssid ‘"BRCM_SOFTAP"’
wpa_cli> IFNAME=bcm0 set_network 0 proto WPA2
wpa_cli> IFNAME=bcm0 set_network 0 key_mgmt WPA-PSK
wpa_cli> IFNAME=bcm0 set_network 0 pairwise TKIP
wpa_cli> IFNAME=bcm0 set_network 0 psk ‘"9876543210"’
wpa_cli> IFNAME=bcm0 set_net 0 frequency 5745
wpa_cli> IFNAME=bcm0 set_net 0 mode 2
wpa_cli> IFNAME=bcm0 select_network 0
wpa_cli> IFNAME=bcm0 status

WPA2-PSK-AES:

wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0

	wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm0 remove_network all wpa_cli> IFNAME=bcm0 add_network wpa_cli> IFNAME=bcm0 set_network 0 ssid ‘"BRCM_5G"’ wpa_cli> IFNAME=bcm0 set_network 0 proto WPA2 wpa_cli> IFNAME=bcm0 set_network 0 key_mgmt WPA-PSK wpa_cli> IFNAME=bcm0 set_network 0 pairwise CCMP wpa_cli> IFNAME=bcm0 set_network 0 psk ‘"9876543210"’ wpa_cli> IFNAME=bcm0 set_net 0 frequency 5745 wpa_cli> IFNAME=bcm0 set_net 0 mode 2 wpa_cli> IFNAME=bcm0 select_network 0 wpa_cli> IFNAME=bcm0 status **WAPI-PSK:** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0 wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm0 remove_net all wpa_cli> IFNAME=bcm0 add_net wpa_cli> IFNAME=bcm0 set_net 0 ssid ‘"wapiap5G_bcm0"’ wpa_cli> IFNAME=bcm0 set_net 0 key_mgmt WAPI-PSK wpa_cli> IFNAME=bcm0 set_net 0 proto WAPI wpa_cli> IFNAME=bcm0 set_net 0 psk ‘”1234506789”’ wpa_cli> IFNAME=bcm0 set_net 0 frequency 5180 wpa_cli> IFNAME=bcm0 set_net 0 mode 2 wpa_cli> IFNAME=bcm0 select_net 0 wpa_cli> IFNAME=bcm0 status

SoftAP + STA bringup commands

Scenario	Steps
1. 5G Soft-AP + 2G STA	**//5G AP** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0 wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm0 remove_net all

	wpa_cli> IFNAME=bcm0 add_net wpa_cli> IFNAME=bcm0 set_net 0 ssid '"BRCM_5G"' wpa_cli> IFNAME=bcm0 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm0 set_net 0 frequency 5180 wpa_cli> IFNAME=bcm0 set_net 0 mode 2 wpa_cli> IFNAME=bcm0 select_net 0 ifconfig bcm0 192.168.20.50 **//2G STA** wpa_cli> IFNAME=wlan0 disconnect wpa_cli> IFNAME=wlan0 list_network wpa_cli> IFNAME=wlan0 remove_network 0 wpa_cli> IFNAME=wlan0 add_network wpa_cli> IFNAME=wlan0 set_network 0 ssid '"BRCM_2G"' wpa_cli> IFNAME=wlan0 set_network 0 key_mgmt NONE wpa_cli> IFNAME=wlan0 save_config wpa_cli> IFNAME=wlan0 enable_network 0 wpa_cli> IFNAME=wlan0 select_network 0 wpa_cli> IFNAME=wlan0 status ifconfig wlan0 192.168.1.6
2. 5G Soft-AP + 5G STA	**//5G AP** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0 wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm0 remove_net all

	wpa_cli> IFNAME=bcm0 add_net wpa_cli> IFNAME=bcm0 set_net 0 ssid '"BRCM_5G"' wpa_cli> IFNAME=bcm0 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm0 set_net 0 frequency 5180 wpa_cli> IFNAME=bcm0 set_net 0 mode 2 wpa_cli> IFNAME=bcm0 select_net 0 ifconfig bcm0 192.168.20.50 **//5G STA** wpa_cli> IFNAME=wlan0 disconnect wpa_cli> IFNAME=wlan0 list_network wpa_cli> IFNAME=wlan0 remove_network 0 wpa_cli> IFNAME=wlan0 add_network wpa_cli> IFNAME=wlan0 set_network 0 ssid '"BRCM_5G"' wpa_cli> IFNAME=wlan0 set_network 0 key_mgmt NONE wpa_cli> IFNAME=wlan0 save_config wpa_cli> IFNAME=wlan0 enable_network 0 wpa_cli> IFNAME=wlan0 select_network 0 wpa_cli> IFNAME=wlan0 status ifconfig wlan0 192.168.1.6

Softap + Softap bringup commands

Scenario	Steps
1. **5G Soft-AP + 2G Soft-AP**	**//5G AP** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0

	wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm0 remove_net all wpa_cli> IFNAME=bcm0 add_net wpa_cli> IFNAME=bcm0 set_net 0 ssid '"BRCM_5G"' wpa_cli> IFNAME=bcm0 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm0 set_net 0 frequency 5180 wpa_cli> IFNAME=bcm0 set_net 0 mode 2 wpa_cli> IFNAME=bcm0 select_net 0 ifconfig bcm0 192.168.20.50 **//2G AP** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm1 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 DRIVER interface_create bcm0 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 remove_net all wpa_cli> IFNAME=bcm1 add_net wpa_cli> IFNAME=bcm1 set_net 0 ssid '"BRCM_2G"' wpa_cli> IFNAME=bcm1 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm1 set_net 0 frequency 2437 wpa_cli> IFNAME=bcm1 set_net 0 mode 2 wpa_cli> IFNAME=bcm1 select_net 0 ifconfig bcm1 192.168.30.40

P2P + STA bringup commands

Scenario	Steps
1. **5G P2P AGO + 2G STA**	**//5G P2P_GO** wpa_cli > p2p_group_add freq=5180 ifconfig p2p-wlan0-0 192.168.49.1 **//2G STA (on primary interface)** wpa_cli> IFNAME=wlan0 disconnect wpa_cli> IFNAME=wlan0 list_network wpa_cli> IFNAME=wlan0 remove_network 0 wpa_cli> IFNAME=wlan0 add_network wpa_cli> IFNAME=wlan0 set_network 0 ssid '"BRCM_2G"' wpa_cli> IFNAME=wlan0 set_network 0 key_mgmt NONE wpa_cli> IFNAME=wlan0 save_config wpa_cli> IFNAME=wlan0 enable_network 0 wpa_cli> IFNAME=wlan0 select_network 0 wpa_cli> IFNAME=wlan0 status ifconfig wlan0 192.168.20.50
2. **5G P2P AGO + 5G STA**	**//5G P2P_GO** wpa_cli > p2p_group_add freq=5180 ifconfig p2p-wlan0-0 192.168.49.1 **//5G STA (on primary interface)** wpa_cli> IFNAME=wlan0 disconnect wpa_cli> IFNAME=wlan0 list_network wpa_cli> IFNAME=wlan0 remove_network 0

	wpa_cli> IFNAME=wlan0 add_network wpa_cli> IFNAME=wlan0 set_network 0 ssid '"BRCM_5G"' wpa_cli> IFNAME=wlan0 set_network 0 key_mgmt NONE wpa_cli> IFNAME=wlan0 save_config wpa_cli> IFNAME=wlan0 enable_network 0 wpa_cli> IFNAME=wlan0 select_network 0 wpa_cli> IFNAME=wlan0 status ifconfig wlan0 192.168.20.50

AP + AP + STA bringup commands

Scenario	Steps
1. 2G AP + 5G AP + 2G STA	**//2G AP** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0 wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm0 remove_net all wpa_cli> IFNAME=bcm0 add_net wpa_cli> IFNAME=bcm0 set_net 0 ssid '"BRCM_2G"' wpa_cli> IFNAME=bcm0 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm0 set_net 0 frequency 2437 wpa_cli> IFNAME=bcm0 set_net 0 mode 2 wpa_cli> IFNAME=bcm0 select_net 0 ifconfig bcm0 192.168.20.50 **//5G AP**

	wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm1 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 remove_net all wpa_cli> IFNAME=bcm1 add_net wpa_cli> IFNAME=bcm1 set_net 0 ssid '"BRCM_5G"' wpa_cli> IFNAME=bcm1 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm1 set_net 0 frequency 5180 wpa_cli> IFNAME=bcm1 set_net 0 mode 2 wpa_cli> IFNAME=bcm1 select_net 0 ifconfig bcm1 192.168.30.40 **//2G STA (on primary interface)** wpa_cli> IFNAME=wlan0 disconnect wpa_cli> IFNAME=wlan0 list_network wpa_cli> IFNAME=wlan0 remove_network 0 wpa_cli> IFNAME=wlan0 add_network wpa_cli> IFNAME=wlan0 set_network 0 ssid '"BRCM_EXT_2G"' wpa_cli> IFNAME=wlan0 set_network 0 key_mgmt NONE wpa_cli> IFNAME=wlan0 save_config wpa_cli> IFNAME=wlan0 enable_network 0 wpa_cli> IFNAME=wlan0 select_network 0 wpa_cli> IFNAME=wlan0 status ifconfig wlan0 192.168.1.6
2. 2G AP + 5G AP + 5G STA	**//2G AP**

	wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0 wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm0 remove_net all wpa_cli> IFNAME=bcm0 add_net wpa_cli> IFNAME=bcm0 set_net 0 ssid '"BRCM_2G"' wpa_cli> IFNAME=bcm0 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm0 set_net 0 frequency 2437 wpa_cli> IFNAME=bcm0 set_net 0 mode 2 wpa_cli> IFNAME=bcm0 select_net 0 ifconfig bcm0 192.168.20.50 **//5G AP** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm1 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 remove_net all wpa_cli> IFNAME=bcm1 add_net wpa_cli> IFNAME=bcm1 set_net 0 ssid '"BRCM_5G"' wpa_cli> IFNAME=bcm1 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm1 set_net 0 frequency 5180 wpa_cli> IFNAME=bcm1 set_net 0 mode 2 wpa_cli> IFNAME=bcm1 select_net 0 ifconfig bcm1 192.168.30.40 **//5G STA (primary interface)**

	wpa_cli> IFNAME=wlan0 disconnect wpa_cli> IFNAME=wlan0 list_network wpa_cli> IFNAME=wlan0 remove_network 0 wpa_cli> IFNAME=wlan0 add_network wpa_cli> IFNAME=wlan0 set_network 0 ssid '"BRCM_EXT_5G"' wpa_cli> IFNAME=wlan0 set_network 0 key_mgmt NONE wpa_cli> IFNAME=wlan0 save_config wpa_cli> IFNAME=wlan0 enable_network 0 wpa_cli> IFNAME=wlan0 select_network 0 wpa_cli> IFNAME=wlan0 status ifconfig wlan0 192.168.1.6 <ping test>
3. 5G AP + 2G AP + 2G STA	**//5G AP** wpa_cli> IFNAME=wlan0 DRIVER interface_create bcm0 wpa_cli> interface_add bcm0 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm0 remove_net all wpa_cli> IFNAME=bcm0 add_net wpa_cli> IFNAME=bcm0 set_net 0 ssid '"BRCM_5G"' wpa_cli> IFNAME=bcm0 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm0 set_net 0 frequency 5180 wpa_cli> IFNAME=bcm0 set_net 0 mode 2 wpa_cli> IFNAME=bcm0 select_net 0 ifconfig bcm0 192.168.20.50

	//2G AP wpa_cli> IFNAME=wlan0 interface_create bcm1 wpa_cli> interface_add bcm1 /data/misc/wifi/wpa_supplicant_ap.conf nl80211 wpa_cli> IFNAME=bcm1 remove_net all wpa_cli> IFNAME=bcm1 add_net wpa_cli> IFNAME=bcm1 set_net 0 ssid '"BRCM_2G"' wpa_cli> IFNAME=bcm1 set_net 0 key_mgmt NONE wpa_cli> IFNAME=bcm1 set_net 0 frequency 2437 wpa_cli> IFNAME=bcm1 set_net 0 mode 2 wpa_cli> IFNAME=bcm1 select_net 0 ifconfig bcm1 192.168.30.40 **//2G STA (primary interface)** wpa_cli> IFNAME=wlan0 disconnect wpa_cli> IFNAME=wlan0 list_network wpa_cli> IFNAME=wlan0 remove_network 0 wpa_cli> IFNAME=wlan0 add_network wpa_cli> IFNAME=wlan0 set_network 0 ssid '"BRCM_EXT_2G"' wpa_cli> IFNAME=wlan0 set_network 0 key_mgmt NONE wpa_cli> IFNAME=wlan0 save_config wpa_cli> IFNAME=wlan0 enable_network 0 wpa_cli> IFNAME=wlan0 select_network 0 wpa_cli> IFNAME=wlan0 status ifconfig wlan0 192.168.1.6

AGO + AGO bringup commands

Scenario	Steps
1. 5G AGO + 2G AGO	**//5G AGO** wpa_cli > p2p_group_add freq=5180 ifconfig p2p-wlan0-1 192.168.49.1 **//2G AGO** wpa_cli > p2p_group_add freq=2412 ifconfig p2p-wlan0-0 192.168.59.1

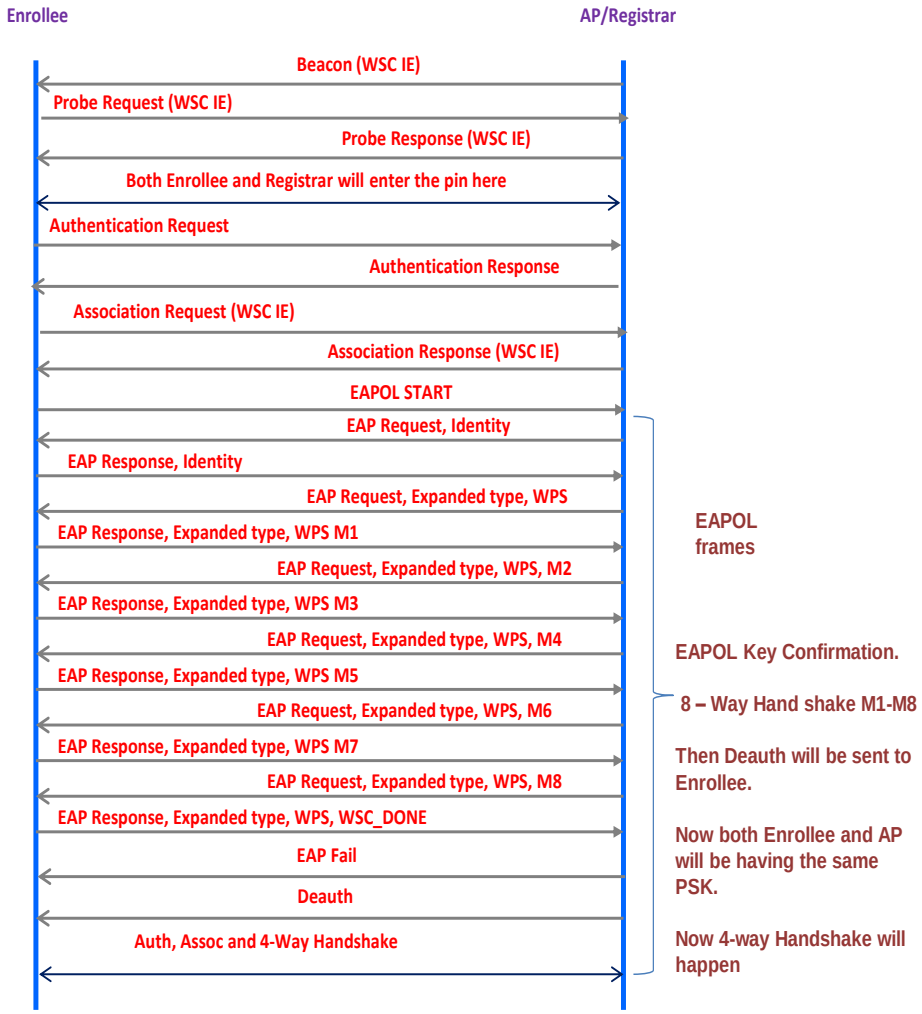
AGO + AGO + STA bringup commands

Scenario	Steps
1. 5G AGO + 2G AGO + 2G STA	**//5G AGO** wpa_cli > p2p_group_add freq=5180 ifconfig p2p-wlan0-1 192.168.49.1 **//2G AGO** wpa_cli > p2p_group_add freq=2412 ifconfig p2p-wlan0-0 192.168.59.1 **//2G STA (primary interface)** wpa_cli> IFNAME=wlan0 disconnect wpa_cli> IFNAME=wlan0 list_network wpa_cli> IFNAME=wlan0 remove_network 0 wpa_cli> IFNAME=wlan0 add_network wpa_cli> IFNAME=wlan0 set_network 0 ssid '"BRCM_EXT_2G"' wpa_cli> IFNAME=wlan0 set_network 0 key_mgmt NONE wpa_cli> IFNAME=wlan0 save_config wpa_cli> IFNAME=wlan0 enable_network 0

	wpa_cli> IFNAME=wlan0 select_network 0 wpa_cli> IFNAME=wlan0 status ifconfig wlan0 192.168.1.6
2. 5G AGO + 2G AGO + 5G STA	**//5G AGO** wpa_cli > p2p_group_add freq=5180 ifconfig p2p-wlan0-1 192.168.49.1 **//2G AGO** wpa_cli > p2p_group_add freq=2412 ifconfig p2p-wlan0-0 192.168.59.1 **//5G STA (primary interface)** wpa_cli> IFNAME=wlan0 disconnect wpa_cli> IFNAME=wlan0 list_network wpa_cli> IFNAME=wlan0 remove_network 0 wpa_cli> IFNAME=wlan0 add_network wpa_cli> IFNAME=wlan0 set_network 0 ssid '"BRCM_EXT_5G"' wpa_cli> IFNAME=wlan0 set_network 0 key_mgmt NONE wpa_cli> IFNAME=wlan0 save_config wpa_cli> IFNAME=wlan0 enable_network 0 wpa_cli> IFNAME=wlan0 select_network 0 wpa_cli> IFNAME=wlan0 status ifconfig wlan0 192.168.1.6

WPS bringup commands:

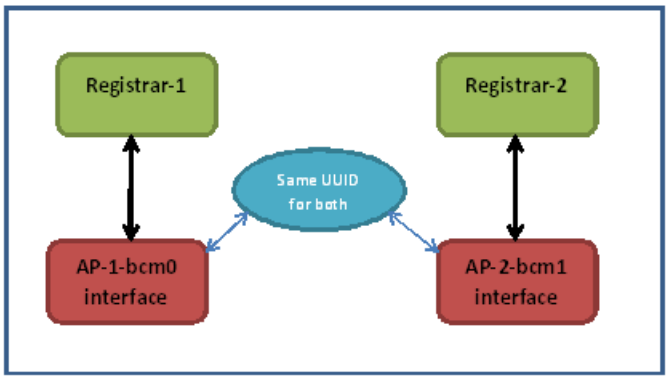
Sample WPS Handshake procedure between STA/Enrollee and AP.



From the above picture (ref:WPS-Spec) handshake procedure you can undertsnd that after 8-way handshake is complete, EAP-FAIL and Deauth will happen.

After the 8-way handshake both Enrollee and AP will be having the same Passphrase then they will use the 4-way handshake to connect.

Wpa_supplicant architecture with reference to WPS-Registrars and UUID for Dual AP/ Dual GO:



Here in the above pic we can see that both AP1 (bcm0 interface) and AP2 (bcm1 interface) have separate Registrars in wpa_supplicant.

Both will use the same UUID.

We can start PBC/PIN WPS sessions on both the interfaces at the same time.

As both AP's use the same UUID, there will not be any WPS session overlap issue happens.

So, if you start WPS sessions on both the AP's, The external STA on which you started WPS session, may connect to either of the AP's (depending on its scan sequence). External STA can connect to particular AP using its BSSID.

P2P Connection using PBC/PIN between a peer and a P2P-GO:

Make sure your p2p_supplicant.conf file is populated with all supported config methods.

Here you can see sample p2p_supplicant.conf

Sample P2P_supplicant.conf	ctrl_interface=/data/misc/wifi/sockets disable_scan_offload=1 driver_param=use_p2p_group_interface=1 update_config=1 device_name=Android_4494

	device_type=10-0050F204-5 **config_methods=virtual_push_button physical_display keypad** p2p_ssid_postfix=-Android_4494 persistent_reconnect=1

1. WPS-PBC method	P1:wpa_cli -p/data/misc/wifi/sockets -ip2p0 p2p_group_add freq=<2g/5g freq> P2:wpa_cli -p/data/misc/wifi/sockets -ip2p0 p2p_find P2:wpa_cli -p/data/misc/wifi/sockets -ip2p0 p2p_connect <device address of GO> pbc join P1:wpa_cli -p/data/misc/wifi/sockets –i<group interface> wps_pbc /*run wps_pbc command in group interface*/
2. WPS-PIN Display/Keypad/ Label Generate	P1:wpa_cli -p/data/misc/wifi/sockets -ip2p0 p2p_group_add P2:wpa_cli -p/data/misc/wifi/sockets -ip2p0 p2p_find P2:wpa_cli -p/data/misc/wifi/sockets -ip2p0 p2p_connect <device address of GO> pin <display | keypad | label> join /*Generate PIN at P1*/ P1:wpa_cli -p/data/misc/wifi/sockets –i<group interface> wps_pin any <use generated pin at P2> /*Use the same generate PIN at P2*/

Generating random pin:

We can use the below command either on GO/STA to generate a random pin for use:

If you want to generate on STA, **IFNAME=wlan0 wps_pin get**

If you want to generate on GO, **IFNAME=<p2p-group-interface/AP interface> wps_pin get**

Generated pin we can use for starting wps_pin session.

WPS based connection commands between Legacy STA and P2P-GO:

WPS-PBC:

WPS Connection between GO and STA (without using BSSID of GO):

GO	STA
P2P_GROUP_ADD	IFNAME=wlan0 wps_pbc any
IFNAME=<group interface> wps_pbc	

WPS Connection between GO and STA (using BSSID of GO):

GO	STA
P2P_GROUP_ADD	IFNAME=wlan0 wps_pbc <BSSID of GO>
IFNAME=<group interface> wps_pbc	

WPS-PIN:

We can not specify whether we need to use display/keypad/label method while connecting to P2P-GO/SoftAP from Legacy STA. Please see below commands.

WPS Connection between GO and STA (without knowing BSSID of GO):

GO	STA
P2P_GROUP_ADD	IFNAME=wlan0 wps_pin any <pin>
IFNAME=<group interface> wps_pin any <pin>	

WPS Connection between GO and STA (using BSSID of GO):

GO	STA
P2P_GROUP_ADD	IFNAME=wlan0 wps_pin <BSSID of GO> <pin>
IFNAME=<group interface> wps_pin any <pin>	

WPS based connection commands between Legacy STA and SoftAP:

WPS-PBC:

WPS Connection between AP and STA (without using BSSID of AP):

AP	STA
Start AP on bcm0 interface.	IFNAME=wlan0 wps_pbc any
IFNAME=<AP interface> wps_pbc	

WPS Connection between AP and STA (using BSSID of AP):

AP	STA
Start AP on bcm0 interface.	IFNAME=wlan0 wps_pbc <BSSID of AP>
IFNAME=<AP interface> wps_pbc	

WPS-PIN:

WPS Connection between AP and STA (without knowing BSSID of AP):

AP	STA
Start AP on bcm0 interface.	IFNAME=wlan0 wps_pin any <pin>
IFNAME=<AP interface> wps_pin any <pin>	

WPS Connection between AP and STA (using BSSID of AP):

AP	STA
Start AP on bcm0 interface.	IFNAME=wlan0 wps_pin <BSSID of AP> <pin>
IFNAME=<AP interface> wps_pin any <pin>	

How to disable WPS:

WPS can be disabled only in case of SoftAP. In case of P2P-GO we shouldn't disable WPS as its support is mandatory according to p2p-spec.

In SoftAP, WPS can be disabled by adding "wps_disabled=1" to the "wpa_supplicant_ap.conf" that we use to bringup particular SoftAP interface.

Sample wpa_supplicant_ap.conf	ctrl_interface=/data/misc/wifi/sockets update_config=1 p2p_disabled=1 **wpa_disabled=1**

WPS will be disabled in case of WAPI. User doesn't need to do any thing. Wpa_supplicant will autodetect WAPI-PSK and disable WPS.