NRC7292 Application Note
(Tree-based Mesh)

Ultra-low power & Long-range Wi-Fi Module

Ver1.0
Jul 05, 2019

NEWRACOM, Inc.

NRC7292 Application Note (Tree-based Mesh)
Ultra-low power & Long-range Wi-Fi Module

Office
Newracom, Inc.
25361 Commercentre Drive, Lake Forest, CA 92630 USA
http://www.newracom.com

Contents

List of Figures

1 Overview

This document describes the Tree-based mesh and explains how to expand the coverage of 11ah using NRC7292 EVK and the Tree-based mesh. The Tree-based mesh of NRC7292 is only supported through its host mode.

As shown in Figure 1.1, the Tree-based mesh station supports the concurrent mode that enables the Tree-based mesh station to function as AP and STA at the same time. The concurrent mode operates in the same channel through the time-sharing method. In one Tree-based mesh station, another Tree-based mesh station or 11ah station can be attached.

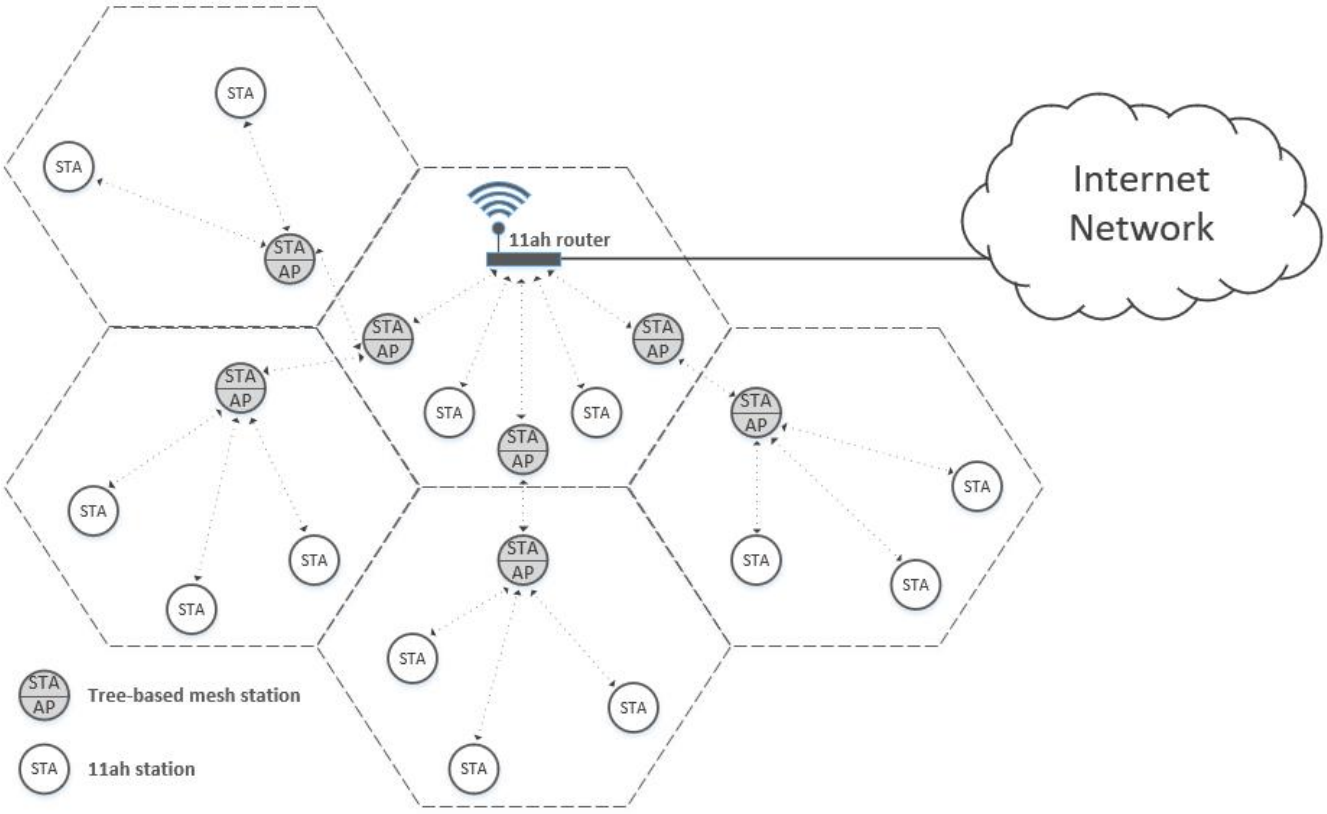

Figure 1.1 Tree-based mesh network

2 NRC7292 EVK Relay operation

The Tree-based mesh station can function as Relay since the Tree-based mesh station supports the concurrent mode. To operate the Tree-based mesh station as a Relay, the concurrent mode must be first enabled. Then, the user must create the extra virtual interface (wlan1) so that the Tree-based mesh station can operate as AP and STA at the same time (these procedures are already completed under start.py script). In this way, the wlan0 interface can be used for AP and wlan1 for STA operation. In addition, Hostapd and DHCH server runs on wlan0 interface, WPA supplicant and DHCP client on wlan1 interface.

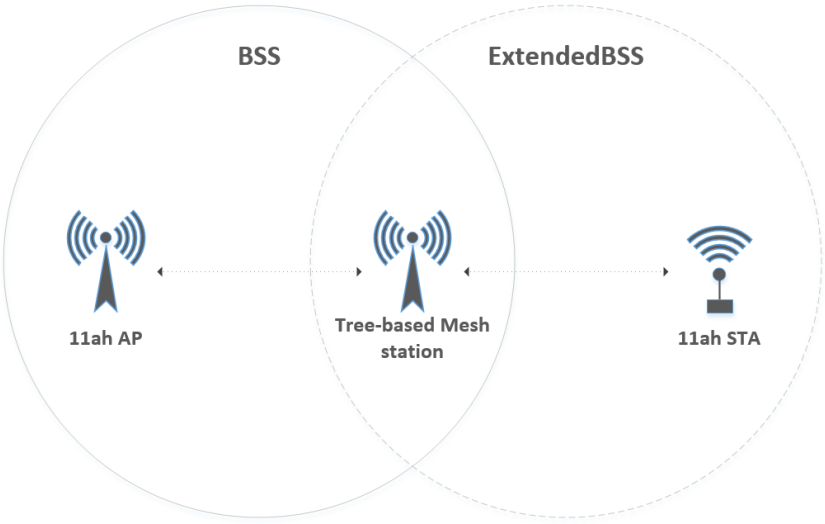

Figure 2.1 Example of the Tree-based mesh used as a 11ah Relay

If there is more than one Tree-based mesh station running as a Relay, the user should use different SSID's and set the different IP address and DHCP regions as shown in Figure 2.2. In this example, two different SSID's ('halow_realy_1' and 'halow_relay_2') are assigned to each Tree-based mesh station. The IP address and DHCP region for each mesh station are set to '192.168.150.xx' and '192.168.140.xx', respectively.

⚠ ***Relay only supports static routing, so the user should set IP & DHCP region and SSID manually.***

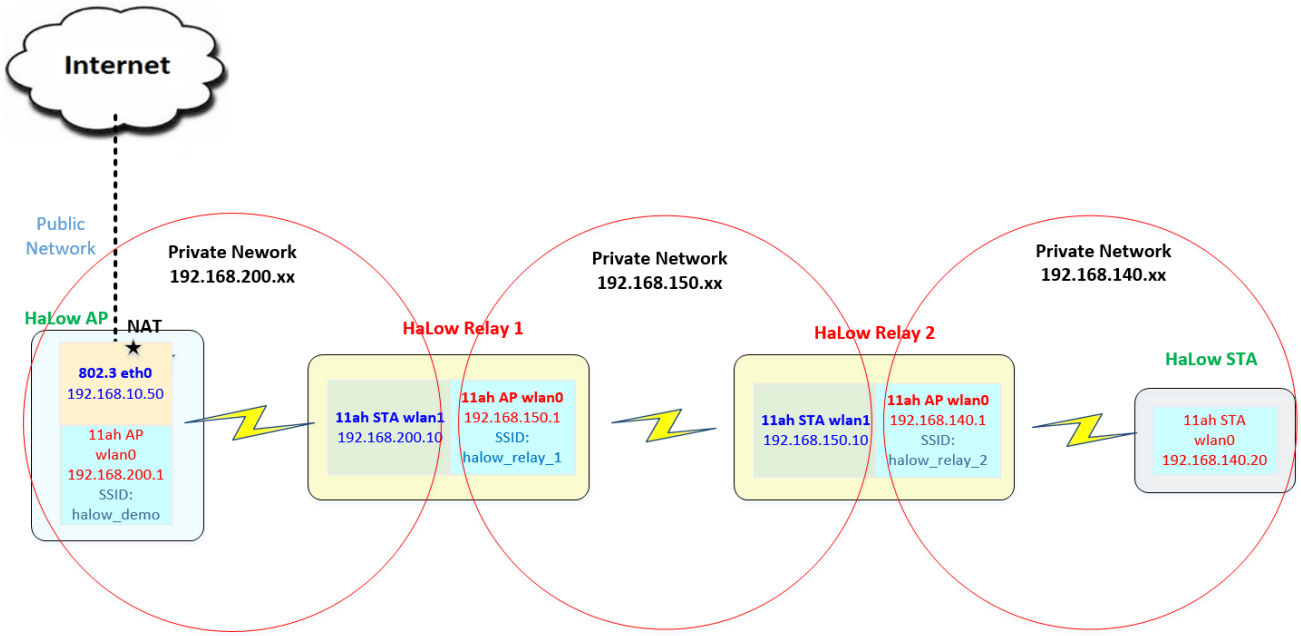

Figure 2.2 Network Topology with 2-hop relay

3 NRC7292 EVK Relay operation

This chapter describes how to run Tree-based mesh station for 11ah Relay on the EVK in detail.

3.1 Set SSID and IP address, DHCP region

Since the Tree-based mesh station functions as AP and STA at the same time, the user must assign a new SSID to the Tree-based mesh station. The user can perform this task by writing the SSID of the Tree-based mesh station on the *ssid* parameter of the configuration file (e.g. ~/nrc_pkg/script/conf/US/ap_halow_open.conf). Figure 3.1 shows an example of this procedure with the SSID, 'halow_relay_1'.

```
ctrl_interface=/var/run/hostapd
country_code=US
interface=wlan0
ssid=halow_relay_1
hw_mode=a
```

Figure 3.1 Parameters for hostapd in the configuration file

Next, the user must set up the IP address and DHCP region to operate 11ah Relay in the 'ip_config.sh' script file. The user can set up these by writing the *RELAY_AP_IP* and *RELAY _DHCP_RANGE* parameters in the script file shown in Figure 3.2.

```
#!/bin/bash

ETH_IP=192.168.100.1
AP_IP=192.168.200.1
RELAY_AP_IP=192.168.150.1

ETH_DHCP_RANGE=192.168.100
AP_DHCP_RANGE=192.168.200
RELAY_DHCP_RANGE=192.168.150
```

Figure 3.2 Parameters for IP & DHCP region in the script

3.2 Start Tree-based mesh station for 11ah Relay

For Tree-based mesh station to operate as 11ah Relay, there must be a 11ah AP that the 11ah Relay's STA can associate to within the 11ah AP's coverage. Once the 11ah Relay's STA successfully associates to the 11ah AP, the 11ah Relay can provide relay service to other 11ah STA around it.

To start Tree-based mesh station as a 11ah Relay, the user can run 'start.py' script by assigning the value '3' to the *sta_type* parameter (refer to "NRC7292_EVK_User_Guide_HM" for other parameters).

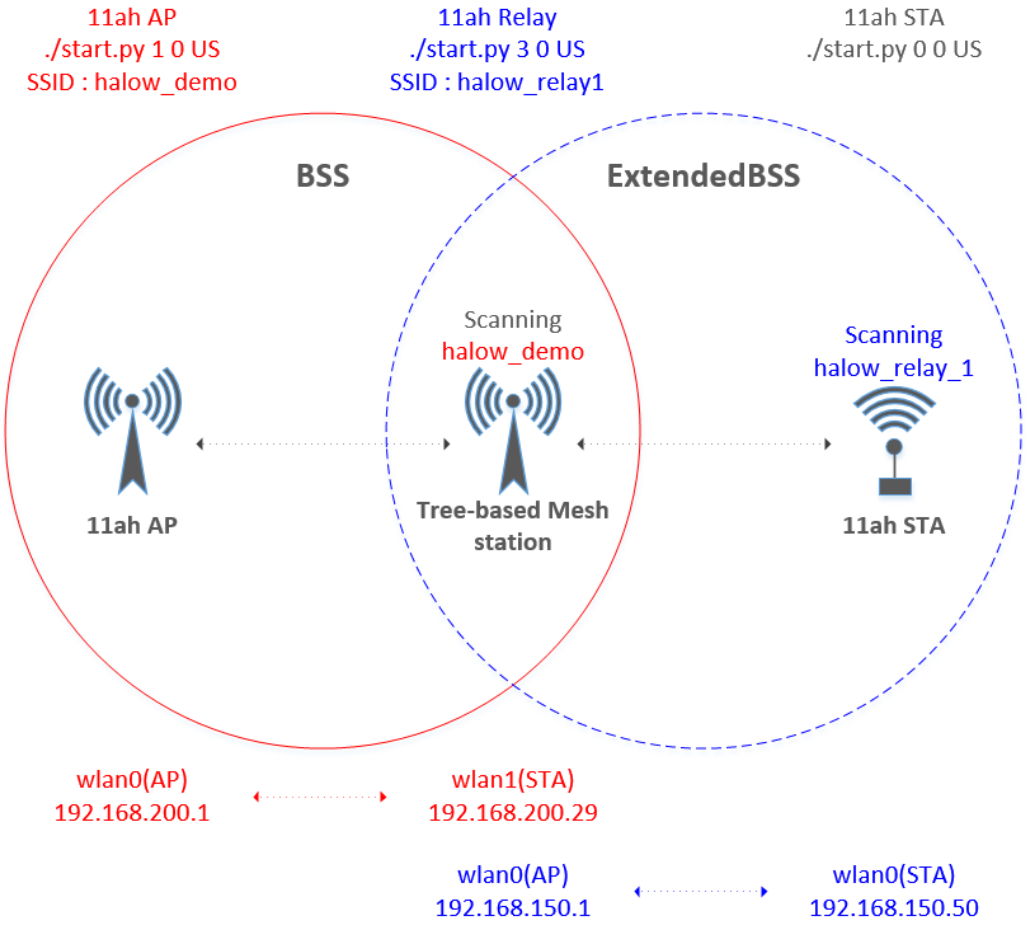

Figure 3.3 Procedure of configuration for one-hop Relay

To configure one-hop Relay network as shown in Figure 3.3, user must start the 11ah AP, 11ah Relay, and 11ah STA in sequence. Figure 3.4 shows the result of starting 11ah AP by running 'start.py' script with giving '1' to *sta_type* parameter.

```
pi@raspberrypi:~/nrc_pkg/script $ ./start.py 1 0 US
------------------------------
Model            : 7292
STA Type         : AP
Security Mode    : OPEN
Country Selected : US
Download FW      : uni_s1g.bin
TX Power         : 17
------------------------------
NRC AP setting for HaLow...
[*] Set Max CPU Clock on RPi
1200000
```

•
•
•

```
-rw-r--r-- 1 root root 228664 Jul  1 21:30 /lib/firmware/uni_s1g.bin
Config for AP is done!
done
[2] Loading module
sudo insmod ~/nrc_pkg/sw/driver/nrc.ko fw_name=uni_s1g.bin disable_cqm=0 hifspeed=16000000
[3] Set tx power
nrf txpwr 17
success
```

•
•
•

```
wlan0: interface state COUNTRY_UPDATE->ENABLED
wlan0: AP-ENABLED
[7] Start NAT
[8] Start DNSMASQ
[9] ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
```

•
•
•

```
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 227  bytes 22946 (22.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 227  bytes 22946 (22.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.200.1  netmask 255.255.255.0  broadcast 192.168.200.255
        inet6 fe80::6154:4ff9:1632:e2a0  prefixlen 64  scopeid 0x20<link>
        ether 00:10:40:39:78:62  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 25  bytes 3780 (3.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

HaLow AP ready
Done.
```

Figure 3.4 Results of running 11ah AP

In the 2nd step, user must start the Tree-based mesh station as a 11ah Relay with the 'start.py' script by giving '3' to *sta_type* parameter as shown in Figure 3.5. User also can find the 11ah Relay initiates the WPA supplicant, associates to 11ah AP, and get an IP address for wlan1 interface through the DHCP client.

```
pi@raspberrypi:~/nrc_pkg/script $ ./start.py 3 0 US
------------------------------
Model            : 7292
STA Type         : RELAY
Security Mode    : OPEN
Country Selected : US
Download FW      : uni_s1g.bin
TX Power         : 17
------------------------------
NRC RELAY setting for HaLow...
[*] Set Max CPU Clock on RPi
1200000
1200000
1200000
1200000
Done
[0] Clear
wireshark-gtk: no process found
[1] Copy
total 460
drwxr-xr-x 2 pi pi   4096 Jul  3 20:14 .
drwxr-xr-x 4 pi pi   4096 Jul  3 20:14 ..
-rwxr-xr-x 1 pi pi    218 Jul  3 20:14 copy
-rwxr-xr-x 1 pi pi 228808 Jul  3 20:14 nrc7292_cspi.bin
-rwxr-xr-x 1 pi pi 228808 Jul  4 13:44 uni_s1g.bin
-rw-r--r-- 1 root root 228808 Jul  4 13:44 /lib/firmware/uni_s1g.bin
Config for RELAY is done!
done
[2] Loading module
sudo insmod ~/nrc_pkg/sw/driver/nrc.ko fw_name=uni_s1g.bin disable_cqm=0 hifspeed=16000000
[2-1] Create wlan1 for concurrent mode
[3] Set tx power
nrf txpwr 17
success
[4] Set aggregation number
Input TID is (0)
set maxagg 1 8
success
[5] Set guard interval
set gi long
success
[6] Start wpa_supplicant
Successfully initialized wpa_supplicant
wlan1: SME: Trying to authenticate with 00:10:40:39:78:62 (SSID='halow_demo' freq=5765 MHz)
wlan1: Trying to associate with 00:10:40:39:78:62 (SSID='halow_demo' freq=5765 MHz)
wlan1: Associated with 00:10:40:39:78:62
wlan1: CTRL-EVENT-CONNECTED - Connection to 00:10:40:39:78:62 completed [id=0 id_str=]
wlan1: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
[7] Connect and DHCP
Waiting for IP
ip_address=192.168.200.29
IP assigned. HaLow STA ready
```

Figure 3.5 Results of running 11ah Relay (STA operation)

After STA operation, the 11ah Relay starts AP operation including staring hostapd on wlan0 interface with “halow_relay_1” SSID, assigning IP address on wlan0 (pre-allocated by the user), and running the DHCP server.

```
[6] Start hostapd
Configuration file: /home/pi/nrc_pkg/script/conf/US/ap_halow_open.conf
wlan0: interface state UNINITIALIZED->COUNTRY_UPDATE
Using interface wlan0 with hwaddr 02:00:eb:cc:66:42 and ssid "halow_relay_1"
wlan0: interface state COUNTRY_UPDATE->ENABLED
wlan0: AP-ENABLED
[7] Start NAT
[8] Start DNSMASQ
[9] ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.3.37  netmask 255.255.255.0  broadcast 192.168.3.255
        inet6 fe80::3d5a:f38b:929d:d2ee  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:cc:66:41  txqueuelen 1000  (Ethernet)
        RX packets 42679  bytes 29931541 (28.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 34762  bytes 17047011 (16.2 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 63  bytes 9628 (9.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 63  bytes 9628 (9.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.150.1  netmask 255.255.255.0  broadcast 192.168.150.255
        inet6 fe80::59d2:d113:f55e:2fa1  prefixlen 64  scopeid 0x20<link>
        ether 02:00:eb:cc:66:42  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 25  bytes 3800 (3.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.200.29  netmask 255.255.255.0  broadcast 192.168.200.255
        inet6 fe80::24e7:b08b:17f9:8c01  prefixlen 64  scopeid 0x20<link>
        ether 02:01:eb:cc:66:42  txqueuelen 1000  (Ethernet)
        RX packets 3  bytes 431 (431.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 31  bytes 4994 (4.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 3.6 Results of running 11ah Relay (AP operation)

Once the setup for 11ah Relay is finished, user can start running a 11ah STA as shown in Figure 3.7. For the 11ah STA, user must give '0' to *sta_type*.

```
pi@raspberrypi:~/nrc_pkg/script $ ./start.py 0 0 US
------------------------------
Model            : 7292
STA Type         : STA
Security Mode    : OPEN
Country Selected : US
Download FW      : uni_s1g.bin
TX Power         : 17
------------------------------
NRC STA setting for HaLow...
[*] Set Max CPU Clock on RPi
1400000
1400000
1400000
1400000
Done
[0] Clear
hostapd: no process found
wireshark-gtk: no process found
[1] Copy
total 460
drwxr-xr-x 2 pi pi   4096 Jul  3 20:14 .
drwxr-xr-x 4 pi pi   4096 Jul  3 20:14 ..
-rwxr-xr-x 1 pi pi    218 Jul  3 20:14 copy
-rwxr-xr-x 1 pi pi 228808 Jul  3 20:14 nrc7292_cspi.bin
-rwxr-xr-x 1 pi pi 228808 Jul  3 20:54 uni_s1g.bin
-rw-r--r-- 1 root root 228808 Jul  3 20:54 /lib/firmware/uni_s1g.bin
Config for STA is done!
done
[2] Loading module
sudo insmod ~/nrc_pkg/sw/driver/nrc.ko fw_name=uni_s1g.bin disable_cqm=0 hifspeed=16000000
[3] Set tx power
nrf txpwr 17
success
[4] Set aggregation number
Input TID is (0)
set maxagg 1 8
success
[5] Set guard interval
set gi long
success
[6] Start wpa_supplicant
Successfully initialized wpa_supplicant
[7] Connect and DHCP
wlan0: SME: Trying to authenticate with 02:01:eb:cc:66:42 (SSID='halow_relay_1' freq=5765 MHz)
wlan0: Trying to associate with 02:01:eb:cc:66:42 (SSID='halow_relay_1' freq=5765 MHz)
wlan0: Associated with 02:01:eb:cc:66:42
wlan0: CTRL-EVENT-CONNECTED - Connection to 02:01:eb:cc:66:42 completed [id=0 id_str=]
wlan0: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
Waiting for IP
ip_address=192.168.150.50
IP assigned. HaLow STA ready
Done.
```

Figure 3.7 Results of running 11ah STA

If user successfully completes step 1 to 3, there is only one task left for communication between 11ah AP and STA. That is to add routing table on 11ah AP. This task should be done manually or by using ‘add_route.sh’ script as below. The parameter for this script is the last number of IP address of 11ah Relay node.

Figure 3.8 shows a way to add routing table with ‘add_route.sh’ and the success of the ping test after this task.

```
pi@raspberrypi:~/nrc_pkg/script $ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.3.1     0.0.0.0         UG    202    0        0 eth0
192.168.3.0     0.0.0.0         255.255.255.0   U     202    0        0 eth0
192.168.200.0   0.0.0.0         255.255.255.0   U     312    0        0 wlan0
pi@raspberrypi:~/nrc_pkg/script $ ./conf/etc/add_route.sh 29
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.3.1     0.0.0.0         UG    202    0        0 eth0
192.168.3.0     0.0.0.0         255.255.255.0   U     202    0        0 eth0
192.168.150.0   192.168.200.29  255.255.255.0   UG    0      0        0 wlan0
192.168.200.0   0.0.0.0         255.255.255.0   U     312    0        0 wlan0
Done
pi@raspberrypi:~/nrc_pkg/script $ ping 192.168.150.50 -c 10
PING 192.168.150.50 (192.168.150.50) 56(84) bytes of data.
64 bytes from 192.168.150.50: icmp_seq=1 ttl=63 time=18.5 ms
64 bytes from 192.168.150.50: icmp_seq=2 ttl=63 time=12.7 ms
64 bytes from 192.168.150.50: icmp_seq=3 ttl=63 time=14.1 ms
64 bytes from 192.168.150.50: icmp_seq=4 ttl=63 time=25.4 ms
64 bytes from 192.168.150.50: icmp_seq=5 ttl=63 time=23.1 ms
64 bytes from 192.168.150.50: icmp_seq=6 ttl=63 time=28.1 ms
64 bytes from 192.168.150.50: icmp_seq=7 ttl=63 time=15.0 ms
64 bytes from 192.168.150.50: icmp_seq=8 ttl=63 time=15.9 ms
64 bytes from 192.168.150.50: icmp_seq=9 ttl=63 time=18.8 ms
64 bytes from 192.168.150.50: icmp_seq=10 ttl=63 time=14.0 ms

--- 192.168.150.50 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9013ms
rtt min/avg/max/mdev = 12.749/18.605/28.105/5.031 ms
```

Figure 3.8 Adding route on 11ah AP and performing ping test between 11ah AP and STA

4 Usage of Tree-based Mesh (WDS)

One of the usable application for Tree-based mesh is wireless distribution system (WDS). As shown in Figure 4.1, the 11ah APs can communicate each other in WDS in a standardized manner. With a WDS, user can extend network using 11ah AP without wired link to internet network.

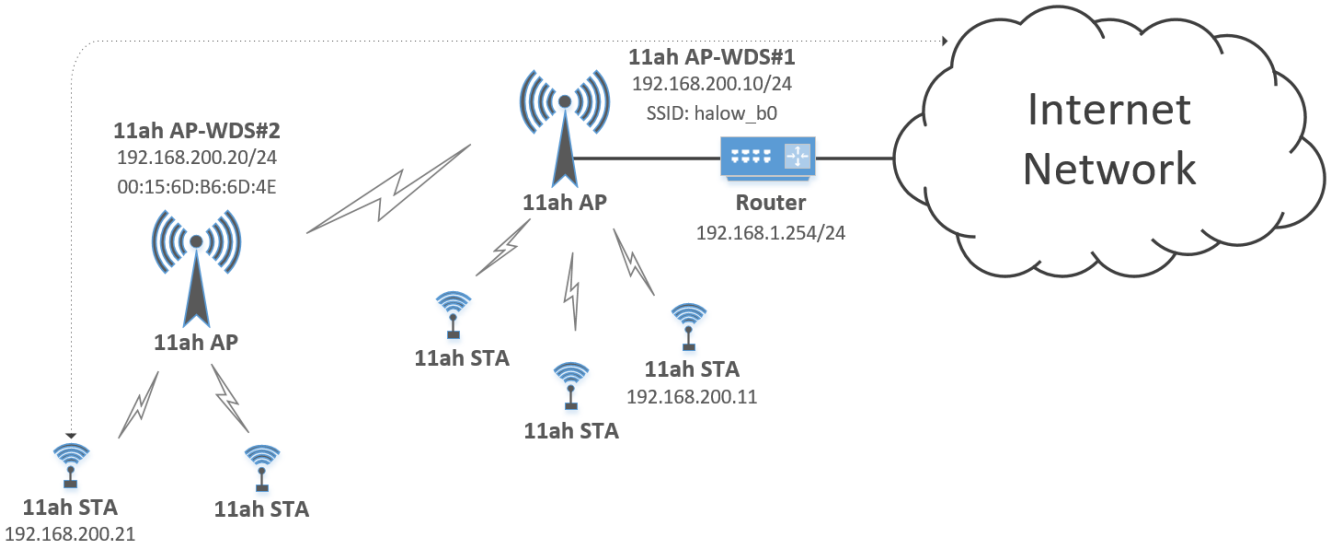

Figure 4.1 WDS with Tree-based mesh

4.1 Kernel build for enabling WDS

To use WDS on the NRC7292 EVK, user should re-build Linux kernel to enable legacy WDS support by following the instructions below.

- **Configure & build kernel reference**
  - https://www.raspberrypi.org/documentation/linux/kernel/configuring.md
  - https://www.raspberrypi.org/documentation/linux/kernel/building.md

- **Instructions**

  $ sudo apt-get update

  $ sudo apt-get install git bc bison flex libssl-dev

  $ sudo apt-get install libncurses5-dev

  $ sudo apt-get install bridge-utils

  $ git clone --depth=1 --branch rpi-4.14.y https://github.com/raspberrypi/linux

```
$ cd linux
$ KERNEL=kernel7
$ make bcm2709_defconfig
$ make menuconfig
```

- Enable “Device Driver → Network Device support → Wireless Lan → mac80211-based Legacy WDS support”

```
$ make -j4 zImage modules dtbs
$ sudo make modules_install
$ sudo cp arch/arm/boot/dts/*.dtb /boot/
$ sudo cp arch/arm/boot/dts/overlays/*.dtb* /boot/overlays/
$ sudo cp arch/arm/boot/dts/overlays/README /boot/overlays/
$ sudo cp arch/arm/boot/zImage /boot/$KERNEL.img
```

4.2 Configure WDS network

This chapter describes how to configure WDS network as shown in Figure 4.2 with NRC7292 EVKs. A user can confirm its operation by sending a ping packet between 11ah STAs connected to the different 11ah network.

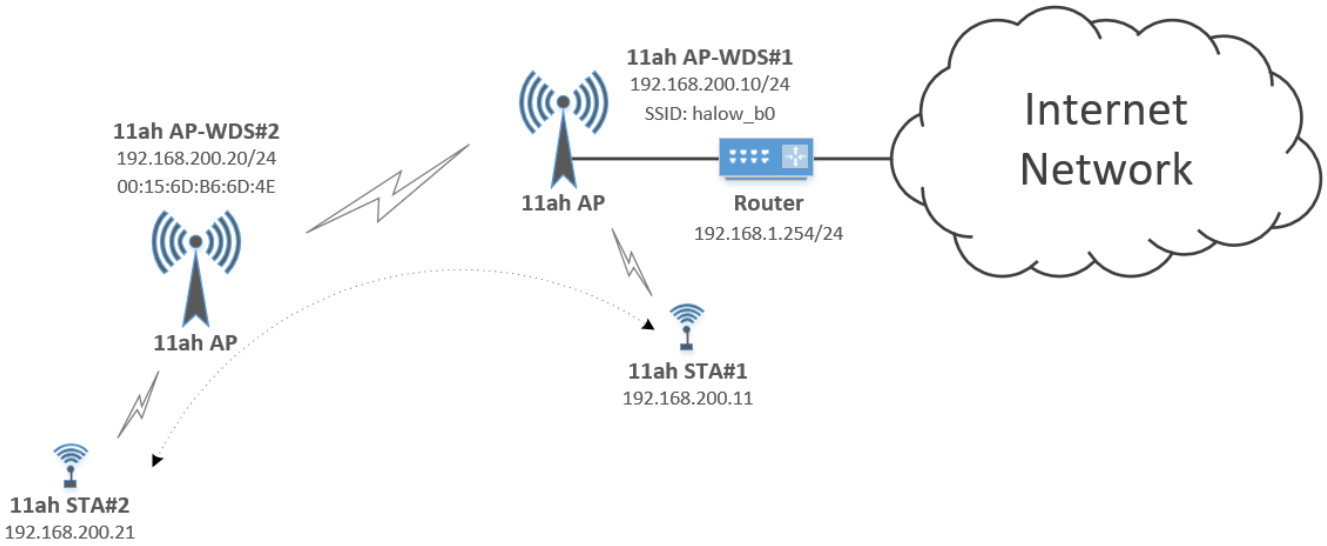

Figure 4.2 WDS network configuration

- **Start 11ah AP-WDS#1 as WDS master**

```
$ sudo insmod ~/NRC_MACSW/host/linux/driver/nrc/nrc.ko fw_name=uni_s1g.bin
$ sudo iw dev wlan0 interface add wlan1 type managed
$ sudo brctl addbr br0
$ cat wds_ap.conf
        ctrl_interface=/var/run/hostapd
        country_code=US
        wds_sta=1
        wds_bridge=br0
        ssid=halow_b0
        hw_mode=a
        beacon_int=100
        channel=159
        ieee80211n=1
        interface=wlan0
        ap_max_inactivity=16779
        wmm_enabled=1

$ sudo hostapd wds_ap.conf
$ sudo ifconfig wlan0 0.0.0.0
$ sudo ifconfig wlan1 0.0.0.0
$ sudo iw dev wlan1 set 4addr on
$ sudo brctl addif br0 wlan0
$ sudo brctl addif br0 wlan1
$ sudo ifconfig br0 192.168.200.10
```

- **Start 11ah AP-WDS#1 as WDS slave**

```
$ sudo insmod ~/NRC_MACSW/host/linux/driver/nrc/nrc.ko fw_name=uni_s1g.bin
$ sudo iw dev wlan0 interface add wlan1 type managed
$ sudo brctl addbr br0
$ cat wds_ap.conf
        ctrl_interface=/var/run/hostapd
        country_code=US
        wds_sta=1
        wds_bridge=br0
        ssid=halow_b1
        hw_mode=a
        beacon_int=100
```

```
        channel=159
        ieee80211n=1
        interface=wlan0
        ap_max_inactivity=16779
        wmm_enabled=1

$ sudo hostapd wds_ap.conf
$ cat wds_sta.conf
        ctrl_interface=/var/run/wpa_supplicant
        country=US
        network={
        ssid="halow_b0"
        key_mgmt=NONE
        }
        p2p_disabled=1

$ sudo wpa_supplicant -iwlan1 -cwds_sta.conf
$ sudo ifconfig wlan0 0.0.0.0
$ sudo ifconfig wlan1 0.0.0.0
$ sudo iw dev wlan1 set 4addr on
$ sudo brctl addif br0 wlan0
$ sudo brctl addif br0 wlan1
$ sudo ifconfig br0 192.168.200.20
```

- **Start 11ah STA#1 to connect to 11ah AP-WDS#1**

```
$ sudo insmod ~/NRC_MACSW/host/linux/driver/nrc/nrc.ko fw_name=uni_s1g.bin
$ cat wds_sta.conf
        ctrl_interface=/var/run/wpa_supplicant
        country=US
        network={
        ssid="halow_b0"
        key_mgmt=NONE
        }
        p2p_disabled=1

$ sudo wpa_supplicant -iwlan0 -cwds_sta.conf
$ sudo ifconfig wlan0 192.168.200.11
```

- **Start 11ah STA#2 to connect to 11ah AP-WDS#2**

  $ sudo insmod ~/NRC_MACSW/host/linux/driver/nrc/nrc.ko fw_name=uni_s1g.bin

  $ cat wds_sta.conf

  ```
  ctrl_interface=/var/run/wpa_supplicant
  country=US
  network={
  ssid="halow_b1"
  key_mgmt=NONE
  }
  p2p_disabled=1
  ```

  $ sudo wpa_supplicant -iwlan0 -cwds_sta.conf

  $ sudo ifconfig wlan0 192.168.200.21

- **Verify connection between 11ah STA#1 and #2 by ping packet**

  $ ping 192.168.200.21 (@11ah STA#1)

  $ ping 192.168.200.11 (@11ah STA#2)

5 Revision History

Revision No	Date	Comments
Ver 1.0	07/05/2019	Internal version for customer release created
Ver 1.1	11/14/2019	Updated script name on Figure 3.3 Added a brief description for add_route.sh Add conf file and update related descriptions for WDS