HOPERF Wi-SUN Border Router Using Guide

2024.6

HOPERF Wi-SUN Border Router Using Guide

Table of Contents

1. HOPERF Wi-SUN Solution Introduction

Wi-SUN FAN (Wireless Smart Network) is a IPv6 Sub-GHz Mesh technology-based solution for smart city and public applications, which can be widely used in large-scale outdoor IoT wireless networking. Our complete Wi-SUN protocol stack is implemented on the HOPERF module, providing a complete Wi-SUN solution including Border Router and Node Router.

This article emphasizes the implementation and application of HOPERF WI-SUN border router.

1.1. HOPERF Wi-SUN FAN1. Network CXonstruction

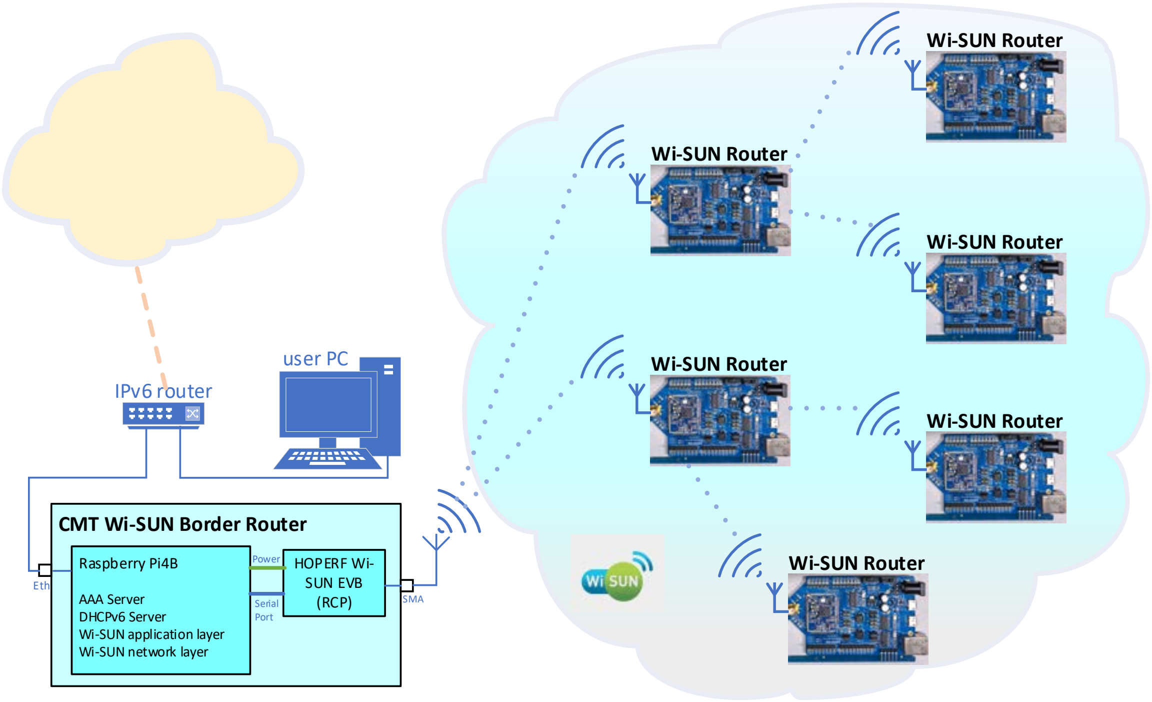


Figure 1-1. HOPERF Wi-SUN FAN 1.0 Network Construction

HOPERF Wi-SUN network supports Sub-GHz mesh networking, which consists of several border router nodes and a larger number of Wi-SUN router nodes. The border routers serve as interfaces between the main network and the wireless Wi-SUN network, enabling connection between the nodes and the internet. The router nodes can be accessed through their own IPv6 addresses or act as relay points in the mesh network to connect non-directly network nodes.

After performing consistent Wi-SUN network parameter configuration and starting each node of the devices, the border router will first establish a Wi-SUN network and broadcast PA to wait for other nodes to join. Then, the joining nodes send PAS broadcasts and select to join the Wi-SUN network of the border router that initiated PA. The nodes will finally join the Wi-SUN network through the process of EAP handshaking, key exchanging, identity authentication, DHCP, RPL establishment. Nodes that are far from the border router can join this network through relay routing by closer nodes that have already joined the Wi-SUN network; however, an additional node route is added as a relay in between during this interaction. The parameter configurations mentioned above include: Network name, Domain, Mode, Class level (Class), Network size (Size), Broadcast Dwell Interval (BDI), Unicast Dwell Interval (UDI), Broadcast Interval (BI), etc.

1.2. HOPERF Wi-SUN Border Router

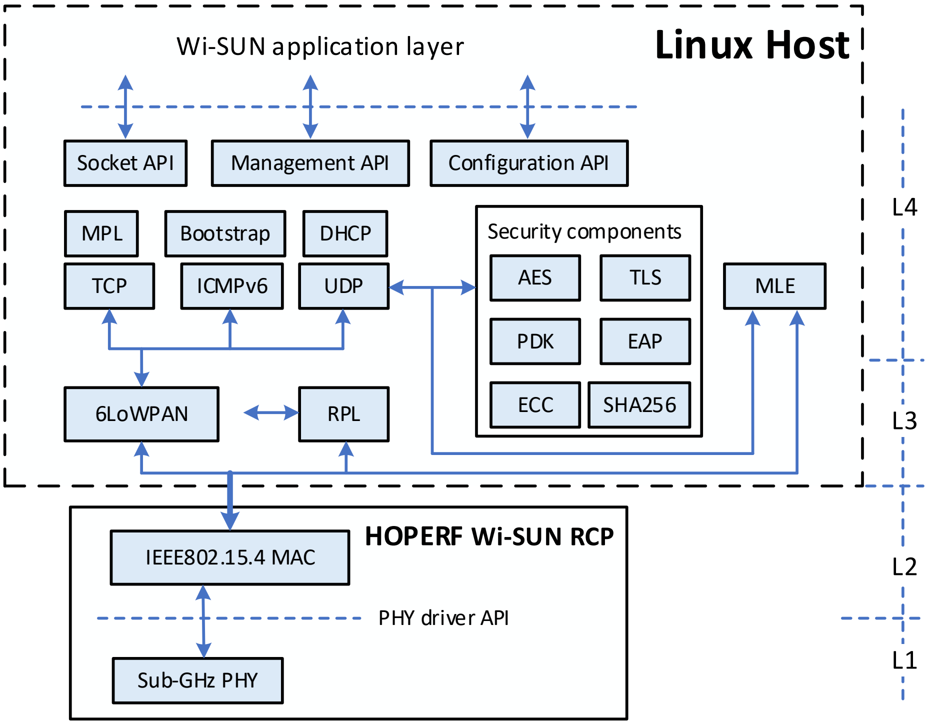


Figure 1-2. HOPERF Border Router Layer

As shown in Figure 1-2, the border router consists of a Linux Host and Radio Co-processor (RCP). The RCP implements the processing of the Wi-SUN physical layer (PHY) and the media access MAC layer, while the Host implements the processing of the network layer, transmission layer, and application layer above the Wi-SUN MAC layer. This construction can improve the access and forwarding capability of Wi-SUN border router by utilizing the relatively sufficient computing and processing power on Host, and can realize more complex processing at the application layer.

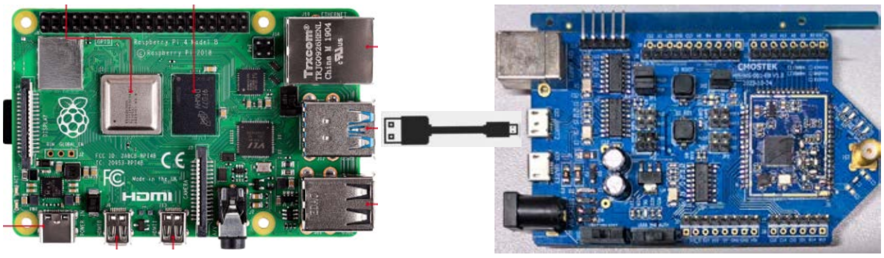


Figure 1-3. Border Router Hardware Component

At present, Raspberry Pi4B is used as Linux Host. RCP is implemented by a Wi-SUN module that integrates a 32bit MCU and a RF front end. The RCP is connected to the Host via USB (serial port @921.600kbps) to transmit Host-RCP control information and inter-layer messages between the network layer (6LoWPAN) and the MAC. Figure 1-3 shows the HOPERF Wi-SUN border router component.

2. Border Router Component

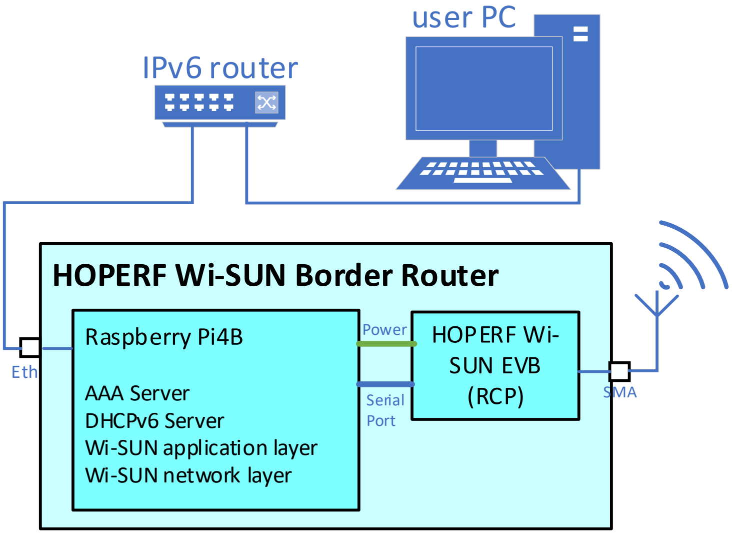


Figure 2-1. Border Router Function Component

As shown in Figure 2-1, the HOPERF Wi-SUN Border Router consists of the Raspberry Pi4B and the HOPERF Wi-SUN hardware module (MCU+HOPERF2310A) EVB board. Wi-SUN network layer and application layer software protocol stack is operated on Raspberry Linux Host: wsbrd implementation, MAC layer and PHY layer processing by HOPERF Wi-SUN module implementation. The above two parts are connected through serial port /USB port to exchange data between the MAC layer and the network layer. The application of wsbrd on Host is qualified for AAA, DHCP service function, which enables border router to perform local authentication services and IPv6 address assignment for access Wi-SUN node routes.

2.1. Border Router RCP

RCP is a wireless coprocessor. HOPERF 2310A RF transceiver is integrated in the module to tranceive Wi-SUN RF signals in the Sub-GHz band, and then transform them into PHY frames to the MCU in the module for CSMA of MAC layer, Channel Hopping control and other processing, and interact with the upper layer through MAC layer control plane interface, data palne interface.

The RCP interacts with the upper layer through the serial port between the HOPERF module and the Raspberry PI of the running Wi-SUN upper layer protocol stack. The data command transfer between the two parties on the serial port follows the Spinel protocol with a data rate of 921.6kbps. In addition, the power supply for the RCP part can be supplied via USB from the Raspberry PI.

HOPERF will provide RCP modules with pre-burned firmware.

2.2. Border Router Linux Host

The upper layer protocol stack and application layer of Wi-SUN border router are accomplished by Raspberry Pi4B. Each Wi-SUN network node needs border router to increase the corresponding data buffer at the network layer and application layer. Thus making the access scale and the additional processing requirements are relatively free from the storage and computing capabilities of the hardware platform.

3. Software Compile and Setup

HOPERF provides application code for Wi-SUN border router on Linux hosts. The Wi-SUN border router application needs to be compiled and installed on the Raspberry PI Linux Host. The following section describes how to configure and install the Raspberry PI.

3.1. Respberry Preparation

Download and install the Raspberry PI burning tool, download at URL: https://www.raspberrypi.com/software/。

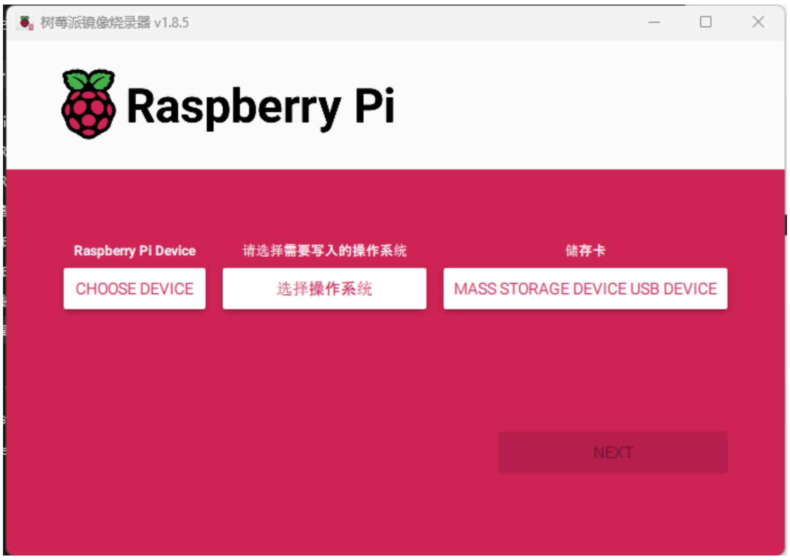
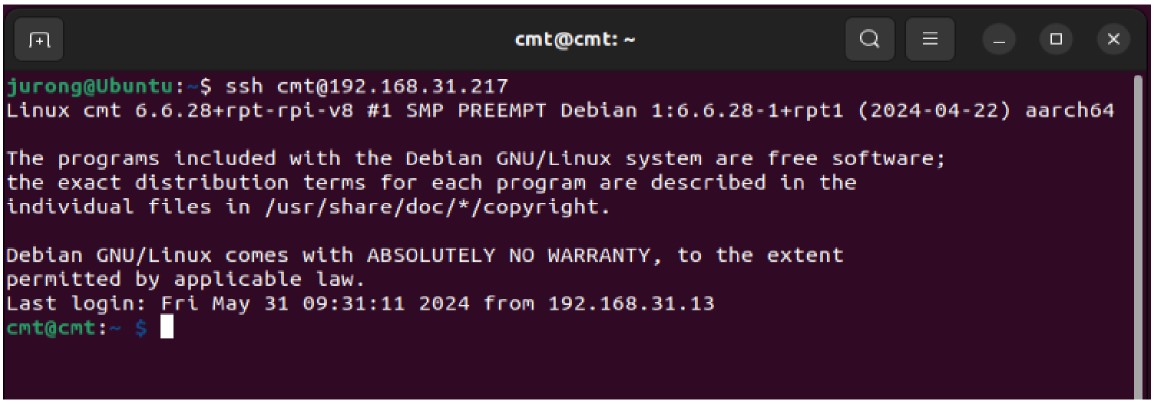


Figure 3-1. Raspberry PI TF Card Burning

Write a clean Raspberry Pi system to a TF card (recommended high-speed card of 32GB or more). CHOOSE DEVICE -> "Raspberry Pi 4"; select operating system -> "Raspberry Pi OS (other)" -> "Raspberry Pi OS Lite (64-bit)"; MASS STORAGE DEVICE USB DEVICE-> choose the target TF storage card.

After successful burning, insert the TF card into the slot below the Raspberry Pi and power it on for the first boot. The non-desktop version of Raspberry Pi OS that we used can not only interact among human and machine by connecting the mini HDMI interface of Raspberry Pi to the monitor, but also achieve remote control through SSH without requiring an additional monitor and keyboard. However, this requires enabling remote control on Raspberry Pi and knowing its IP address. Here we use SSH tool on PC to connect and control Raspberry Pi (MobaXterm installed on Windows PC and ssh command can be directly used on Linux/Ubuntu), as shown in Figure 3-1.



```
jurong@Ubuntu:~$ ssh cmt@192.168.31.217
Linux cmt 6.6.28+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.6.28-1+rpt1 (2024-04-22) aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri May 31 09:31:11 2024 from 192.168.31.13
cmt@cmt:~ $
```

Figure 3-2. Raspberry Pi Remote Control

3.2. Rasperry Pi Tool Pre-installation

```
sudo apt update -y
sudo apt upgrade -y
sudo apt install git net-tools ssh traceroute g++ lm-sensors psensor tcpdump
sudo apt install libnl-3-dev libnl-route-3-dev libcap-dev libpcap-dev python3
sudo apt install libsystemd-dev libdbus-1-dev cargo cmake ninja-build pkg-config lrzsz
```

Install the toolchain necessary to compile the source code on your Raspberry PI system as shown above.

```
# edit /etc/sysctl.conf as the following 2 lines
# net.ipv4.ip_forward=1
# net.ipv6.conf.all.forwarding=1
sudo nano /etc/sysctl.conf
# activate the changes immedialtely
sudo sysctl -p
```

Edit file：related options of ip forward in /etc/sysctl.conf, enabling IP forwarding of Rasperry pi.

```
cd ~
mkdir Git_repository
cd Git_repository
# clone, build and install mbedtls
git clone --branch=v3.0.0 https://github.com/ARMmbed/mbedtls
cd mbedtls
cmake -G Ninja .
ninja
sudo ninja install
```

Creat a Git repository catalog, eg：/Git_repository，down，Compile and install mbedtls (the lated update version is 3.0.0).

```
# clone, build and install wisun-br-linux
cd ~/Git_repository
git clone https://github.com/rongjun72/wisun-br-linux.git
cd wisun-br-linux
cmake -G Ninja .
ninja
sudo ninja install
```

Download, compile, and install the upper-layer software of Wi-SUN. After completing the above compilation and installation, connects the border router hardware (including connecting the RCP and the Raspberry PI USB/ serial port power supply), you can start the border router by using the wsbrd command under the /wisun-br-linux directory.

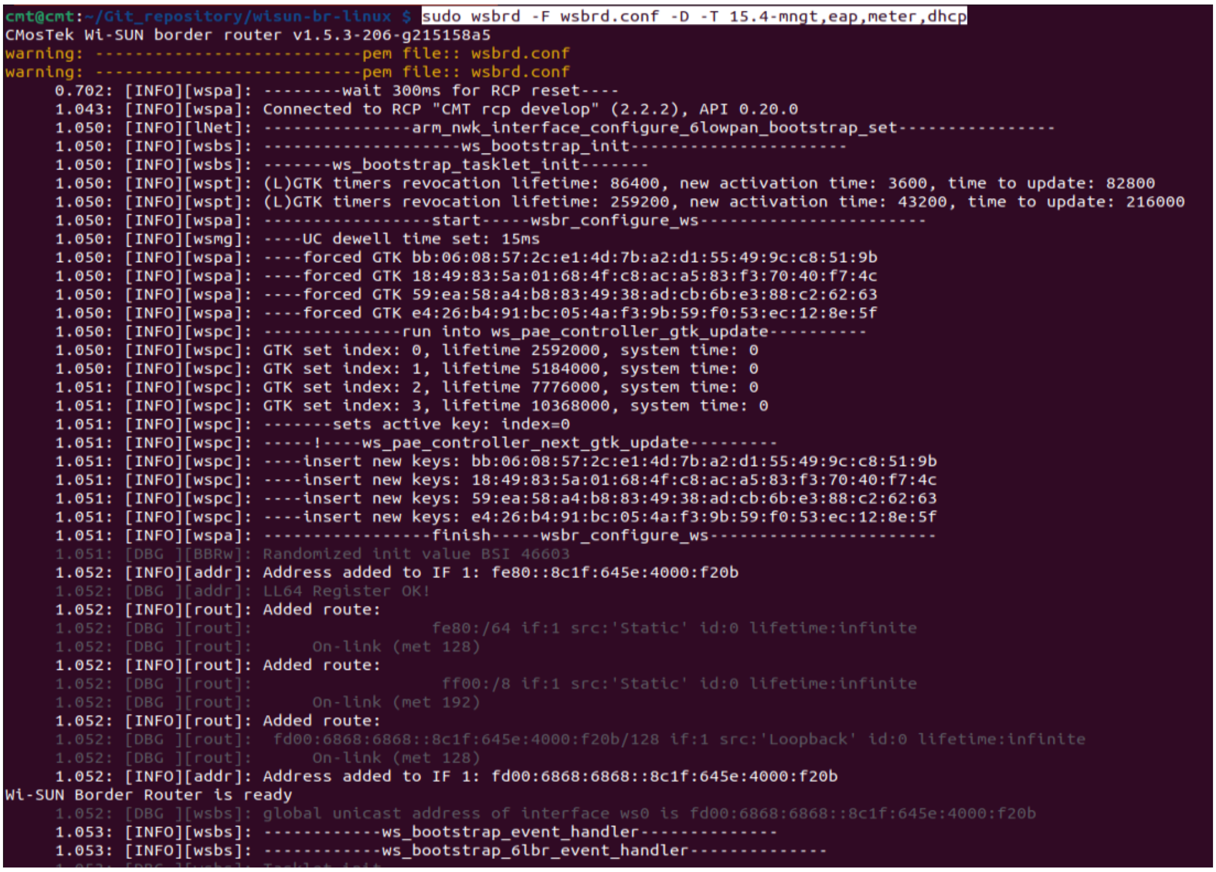
wsbrd is a border router startup command with parameters. You can view the command parameter details by running wsbrd -h. In general, we use sudo wsbrd -F wsbrd.conf -D -T 15.4-mngt,eap,dhcp to start. The *-F* option can specified border router configuration file, If this option is not selected, the configuration file */usr/local/share/doc/wsbrd/examples/wsbrd.conf* will be copied to the directory during the software installation stage by default.After the -D option is enabled, all previously authenticated node routes are cleared. -T is the trace option, which indicates the log related to the function.

Before starting the border router, it is also necessary to modify the Wi-SUN operating mode in the file wsbrd.conf. This file can be the default one: /usr/local/share/doc/wsbrd/examples/wsbrd.conf, or it can be the configuration file specified by the -F option, such as: ../wisun-br-linux/wsbrd.conf.

```
# Connect the Raspberry PI serial port file path of the RCP. If multiple USB devices are
inserted into the Raspberry PI, confirm which one is connected to the RCP in advance.
rcp_uart_device = /dev/ttyUSB0
# Specifies the IPv6 prefix that RPL interactively uses. The prefix does not change during the
lifetime of the network. You can use the GUA prefix of your network directly (for example:
2001:db8::/64).
ipv6_prefix = fd00:6868:6868:0::/64
# The transparent bridge between Wi-SUN and the external network, that is, the name of
the Raspberry PI's Ethernet or wireless card.
neighbor_proxy=eth0
# Wi-SUN network name. \x represents special characters, e.g. \x20 is for spaces.
network_name = Wi-SUN\x20test
# Wi-SUN regulatory domain. Valid values are EU, NA, JP, CN, IN...
domain = NA
# A Wi-SUN Operating Class as in FAN1.0 spec. Accepted values: 1, 2, 3 or 4.
class = 3
# A Wi-SUN Operating Mode as in FAN1.0 spec.
# Accepted values are 1a, 1b, 2a, 2b, 3, 4a, 4b and 5.
mode = 5
```

```
# allowed frequency hopping channel (FHSS). Default 0-255 (all). If only one channel is listed,
FHSS is disabled (i.e., fixed channel mode). Use the connectors "-" and "," to indicate the
channel range, such as: 3-5,10-100
allowed_channels = 0-255
# Unicast Dwell Interval (UDI) is the duration (ms)
unicast_dwell_interval = 15
# Fix the PAN size (number of connected nodes) advertised by the border router
#pan_size = 1000
```

Wsbrd.conf contains the RCP serial port path, Wi-SUN network name, WI-SUN frequency hopping settings, Wi-SUN border router DHCP server and other Wi-SUN border router options. For more detailed configuration and description can be viewed by opening this file.



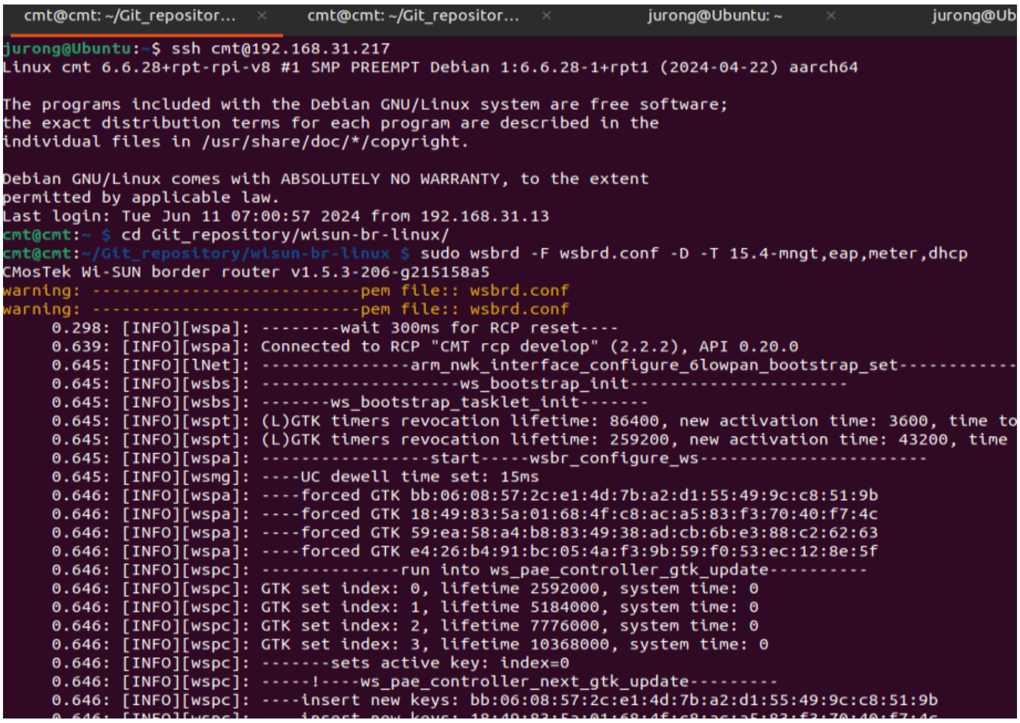
```
cmt@cmt:~/Git_repository/wisun-br-linux $ sudo wsbrd -F wsbrd.conf -D -T 15.4-mngt,eap,meter,dhcp
CMosTek Wi-SUN border router v1.5.3-206-g215158a5
warning: ---------------------------pem file:: wsbrd.conf
warning: ---------------------------pem file:: wsbrd.conf
     0.702: [INFO][wspa]: --------wait 300ms for RCP reset----
     1.043: [INFO][wspa]: Connected to RCP "CMT rcp develop" (2.2.2), API 0.20.0
     1.050: [INFO][lNet]: ---------------arm_nwk_interface_configure_6lowpan_bootstrap_set---------------
     1.050: [INFO][wsbs]: --------------------ws_bootstrap_init----------------------
     1.050: [INFO][wsbs]: -------ws_bootstrap_tasklet_init-------
     1.050: [INFO][wspt]: (L)GTK timers revocation lifetime: 86400, new activation time: 3600, time to update: 82800
     1.050: [INFO][wspt]: (L)GTK timers revocation lifetime: 259200, new activation time: 43200, time to update: 216000
     1.050: [INFO][wspa]: -----------------start-----wsbr_configure_ws----------------------
     1.050: [INFO][wsmg]: ----UC dewell time set: 15ms
     1.050: [INFO][wspa]: ----forced GTK bb:06:08:57:2c:e1:4d:7b:a2:d1:55:49:9c:c8:51:9b
     1.050: [INFO][wspa]: ----forced GTK 18:49:83:5a:01:68:4f:c8:ac:a5:83:f3:70:40:f7:4c
     1.050: [INFO][wspa]: ----forced GTK 59:ea:58:a4:b8:83:49:38:ad:cb:6b:e3:88:c2:62:63
     1.050: [INFO][wspa]: ----forced GTK e4:26:b4:91:bc:05:4a:f3:9b:59:f0:53:ec:12:8e:5f
     1.050: [INFO][wspc]: --------------run into ws_pae_controller_gtk_update----------
     1.050: [INFO][wspc]: GTK set index: 0, lifetime 2592000, system time: 0
     1.050: [INFO][wspc]: GTK set index: 1, lifetime 5184000, system time: 0
     1.051: [INFO][wspc]: GTK set index: 2, lifetime 7776000, system time: 0
     1.051: [INFO][wspc]: GTK set index: 3, lifetime 10368000, system time: 0
     1.051: [INFO][wspc]: -------sets active key: index=0
     1.051: [INFO][wspc]: -----!----ws_pae_controller_next_gtk_update---------
     1.051: [INFO][wspc]: ----insert new keys: bb:06:08:57:2c:e1:4d:7b:a2:d1:55:49:9c:c8:51:9b
     1.051: [INFO][wspc]: ----insert new keys: 18:49:83:5a:01:68:4f:c8:ac:a5:83:f3:70:40:f7:4c
     1.051: [INFO][wspc]: ----insert new keys: 59:ea:58:a4:b8:83:49:38:ad:cb:6b:e3:88:c2:62:63
     1.051: [INFO][wspc]: ----insert new keys: e4:26:b4:91:bc:05:4a:f3:9b:59:f0:53:ec:12:8e:5f
     1.051: [INFO][wspa]: -----------------finish-----wsbr_configure_ws----------------------
     1.051: [DBG ][BBRw]: Randomized init value BSI 46603
     1.052: [INFO][addr]: Address added to IF 1: fe80::8c1f:645e:4000:f20b
     1.052: [DBG ][addr]: LL64 Register OK!
     1.052: [INFO][rout]: Added route:
     1.052: [DBG ][rout]:                   fe80:/64 if:1 src:'Static' id:0 lifetime:infinite
     1.052: [DBG ][rout]:      On-link (met 128)
     1.052: [INFO][rout]: Added route:
     1.052: [DBG ][rout]:                    ff00:/8 if:1 src:'Static' id:0 lifetime:infinite
     1.052: [DBG ][rout]:      On-link (met 192)
     1.052: [INFO][rout]: Added route:
     1.052: [DBG ][rout]:  fd00:6868:6868::8c1f:645e:4000:f20b/128 if:1 src:'Loopback' id:0 lifetime:infinite
     1.052: [DBG ][rout]:      On-link (met 128)
     1.052: [INFO][addr]: Address added to IF 1: fd00:6868:6868::8c1f:645e:4000:f20b
Wi-SUN Border Router is ready
     1.052: [DBG ][wsbs]: global unicast address of interface ws0 is fd00:6868:6868::8c1f:645e:4000:f20b
     1.053: [INFO][wsbs]: ------------ws_bootstrap_event_handler--------------
     1.053: [INFO][wsbs]: ------------ws_bootstrap_6lbr_event_handler--------------
```

Figure 3-3. Border Router Work Log Interface

Finally, after using `sudo wsbrd -F wsbrd.conf -D -T 15.4-mngt,eap,dhcp` to start the border router application, the current console will always display the router's running LOG (LOG), a new tab or ssh interface has to be opened to enter CLI commands. After the wsbrd application is started, the border router then turns on the Wi-SUN FAN1.0 network, sends out PAN broadcasts and receives PAS broadcast signals from other nodes that requiring the PAN network to access.

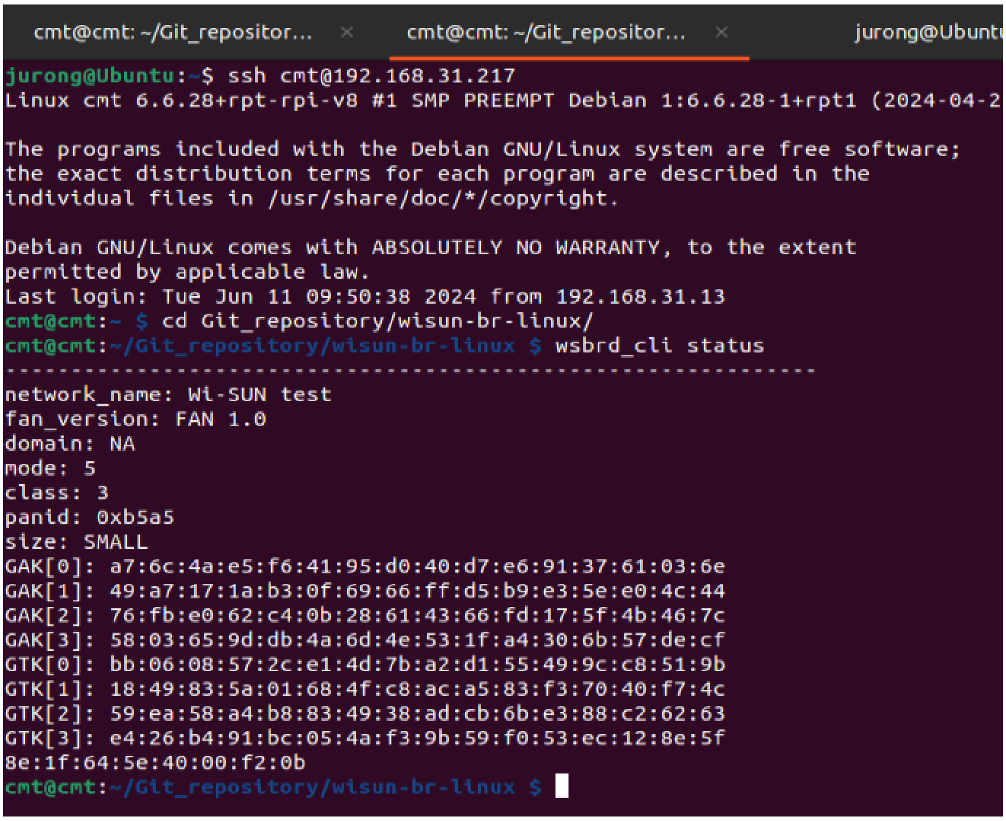
4. CLI Command

The running log and command line input interfaces are separated by the Wi-SUN border router application.



```
jurong@Ubuntu:~$ ssh cmt@192.168.31.217
Linux cmt 6.6.28+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.6.28-1+rpt1 (2024-04-22) aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Jun 11 07:00:57 2024 from 192.168.31.13
cmt@cmt:~ $ cd Git_repository/wisun-br-linux/
cmt@cmt:~/Git_repository/wisun-br-linux $ sudo wsbrd -F wsbrd.conf -D -T 15.4-mngt,eap,meter,dhcp
CMosTek Wi-SUN border router v1.5.3-206-g215158a5
warning: ---------------------------pem file:: wsbrd.conf
warning: ---------------------------pem file:: wsbrd.conf
     0.298: [INFO][wspa]: --------wait 300ms for RCP reset----
     0.639: [INFO][wspa]: Connected to RCP "CMT rcp develop" (2.2.2), API 0.20.0
     0.645: [INFO][lNet]: ---------------arm_nwk_interface_configure_6lowpan_bootstrap_set------------
     0.645: [INFO][wsbs]: --------------------ws_bootstrap_init----------------------
     0.645: [INFO][wsbs]: -------ws_bootstrap_tasklet_init-------
     0.645: [INFO][wspt]: (L)GTK timers revocation lifetime: 86400, new activation time: 3600, time to
     0.645: [INFO][wspt]: (L)GTK timers revocation lifetime: 259200, new activation time: 43200, time
     0.645: [INFO][wspa]: -----------------start-----wsbr_configure_ws-----------------------
     0.645: [INFO][wsmg]: ----UC dewell time set: 15ms
     0.646: [INFO][wspa]: ----forced GTK bb:06:08:57:2c:e1:4d:7b:a2:d1:55:49:9c:c8:51:9b
     0.646: [INFO][wspa]: ----forced GTK 18:49:83:5a:01:68:4f:c8:ac:a5:83:f3:70:40:f7:4c
     0.646: [INFO][wspa]: ----forced GTK 59:ea:58:a4:b8:83:49:38:ad:cb:6b:e3:88:c2:62:63
     0.646: [INFO][wspa]: ----forced GTK e4:26:b4:91:bc:05:4a:f3:9b:59:f0:53:ec:12:8e:5f
     0.646: [INFO][wspc]: --------------run into ws_pae_controller_gtk_update----------
     0.646: [INFO][wspc]: GTK set index: 0, lifetime 2592000, system time: 0
     0.646: [INFO][wspc]: GTK set index: 1, lifetime 5184000, system time: 0
     0.646: [INFO][wspc]: GTK set index: 2, lifetime 7776000, system time: 0
     0.646: [INFO][wspc]: GTK set index: 3, lifetime 10368000, system time: 0
     0.646: [INFO][wspc]: -------sets active key: index=0
     0.646: [INFO][wspc]: -----!----ws_pae_controller_next_gtk_update---------
     0.646: [INFO][wspc]: ----insert new keys: bb:06:08:57:2c:e1:4d:7b:a2:d1:55:49:9c:c8:51:9b
     0.646: [INFO][wspc]: ----insert new keys: 18:49:83:5a:01:68:4f:c8:ac:a5:83:f3:70:40:f7:4c
```

Figure 4-1. Wi-SUN Border Router Work Log Interface

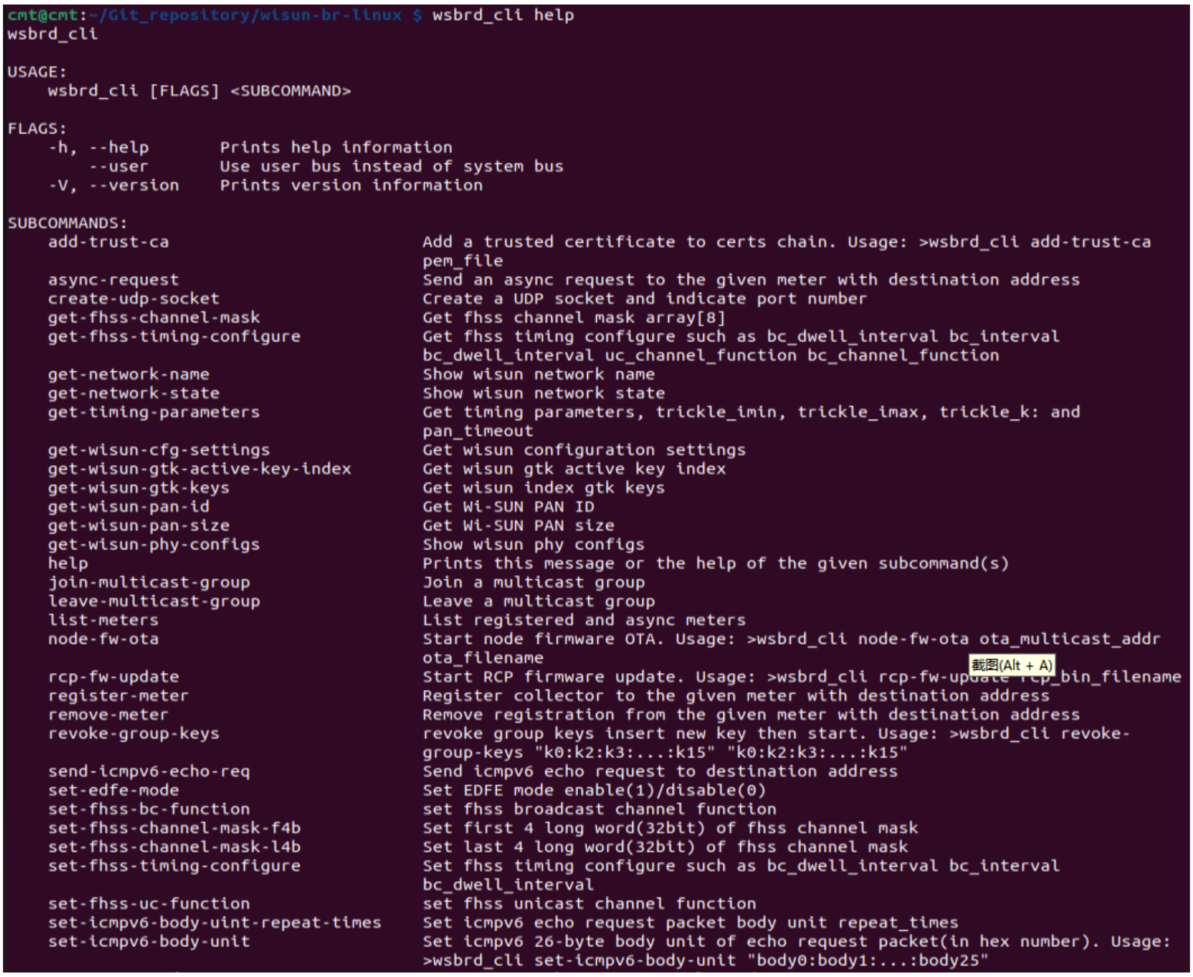


```
jurong@Ubuntu:~$ ssh cmt@192.168.31.217
Linux cmt 6.6.28+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.6.28-1+rpt1 (2024-04-2

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Jun 11 09:50:38 2024 from 192.168.31.13
cmt@cmt:~ $ cd Git_repository/wisun-br-linux/
cmt@cmt:~/Git_repository/wisun-br-linux $ wsbrd_cli status
--------------------------------------------------------------
network_name: Wi-SUN test
fan_version: FAN 1.0
domain: NA
mode: 5
class: 3
panid: 0xb5a5
size: SMALL
GAK[0]: a7:6c:4a:e5:f6:41:95:d0:40:d7:e6:91:37:61:03:6e
GAK[1]: 49:a7:17:1a:b3:0f:69:66:ff:d5:b9:e3:5e:e0:4c:44
GAK[2]: 76:fb:e0:62:c4:0b:28:61:43:66:fd:17:5f:4b:46:7c
GAK[3]: 58:03:65:9d:db:4a:6d:4e:53:1f:a4:30:6b:57:de:cf
GTK[0]: bb:06:08:57:2c:e1:4d:7b:a2:d1:55:49:9c:c8:51:9b
GTK[1]: 18:49:83:5a:01:68:4f:c8:ac:a5:83:f3:70:40:f7:4c
GTK[2]: 59:ea:58:a4:b8:83:49:38:ad:cb:6b:e3:88:c2:62:63
GTK[3]: e4:26:b4:91:bc:05:4a:f3:9b:59:f0:53:ec:12:8e:5f
8e:1f:64:5e:40:00:f2:0b
cmt@cmt:~/Git_repository/wisun-br-linux $ 
```

Figure 4-2. Wi-SUN Border Router Command Line Input Interface

4.1.wsbrd_cli Command Instruction

Wi-SUN command line input: wsbrd_cli help can list all the commands and description.



```
cmt@cmt:~/Git_repository/wisun-br-linux $ wsbrd_cli help
wsbrd_cli

USAGE:
    wsbrd_cli [FLAGS] <SUBCOMMAND>

FLAGS:
    -h, --help       Prints help information
        --user       Use user bus instead of system bus
    -V, --version    Prints version information

SUBCOMMANDS:
    add-trust-ca                         Add a trusted certificate to certs chain. Usage: >wsbrd_cli add-trust-ca
                                         pem_file
    async-request                        Send an async request to the given meter with destination address
    create-udp-socket                    Create a UDP socket and indicate port number
    get-fhss-channel-mask                Get fhss channel mask array[8]
    get-fhss-timing-configure            Get fhss timing configure such as bc_dwell_interval bc_interval
                                         bc_dwell_interval uc_channel_function bc_channel_function
    get-network-name                     Show wisun network name
    get-network-state                    Show wisun network state
    get-timing-parameters                Get timing parameters, trickle_imin, trickle_imax, trickle_k: and
                                         pan_timeout
    get-wisun-cfg-settings               Get wisun configuration settings
    get-wisun-gtk-active-key-index       Get wisun gtk active key index
    get-wisun-gtk-keys                   Get wisun index gtk keys
    get-wisun-pan-id                     Get Wi-SUN PAN ID
    get-wisun-pan-size                   Get Wi-SUN PAN size
    get-wisun-phy-configs                Show wisun phy configs
    help                                 Prints this message or the help of the given subcommand(s)
    join-multicast-group                 Join a multicast group
    leave-multicast-group                Leave a multicast group
    list-meters                          List registered and async meters
    node-fw-ota                          Start node firmware OTA. Usage: >wsbrd_cli node-fw-ota ota_multicast_addr
                                         ota_filename
    rcp-fw-update                        Start RCP firmware update. Usage: >wsbrd_cli rcp-fw-update rcp_bin_filename
    register-meter                       Register collector to the given meter with destination address
    remove-meter                         Remove registration from the given meter with destination address
    revoke-group-keys                    revoke group keys insert new key then start. Usage: >wsbrd_cli revoke-
                                         group-keys "k0:k2:k3:...:k15" "k0:k2:k3:...:k15"
    send-icmpv6-echo-req                 Send icmpv6 echo request to destination address
    set-edfe-mode                        Set EDFE mode enable(1)/disable(0)
    set-fhss-bc-function                 set fhss broadcast channel function
    set-fhss-channel-mask-f4b            Set first 4 long word(32bit) of fhss channel mask
    set-fhss-channel-mask-l4b            Set last 4 long word(32bit) of fhss channel mask
    set-fhss-timing-configure            Set fhss timing configure such as bc_dwell_interval bc_interval
                                         bc_dwell_interval
    set-fhss-uc-function                 set fhss unicast channel function
    set-icmpv6-body-uint-repeat-times    Set icmpv6 echo request packet body unit repeat_times
    set-icmpv6-body-unit                 Set icmpv6 26-byte body unit of echo request packet(in hex number). Usage:
                                         >wsbrd_cli set-icmpv6-body-unit "body0:body1:...:body25"
```

Figure 4-3. wsbrd_cli Help Dispaly

Details of wsbrd_cli commands are shown below.

4.1.1. General Command

4.1.1.1. help

```
Format: wsbrd_cli help
Function: List all the supporting subcommands with a brief description.
```

4.1.1.2. status

```
Format: wsbrd_cli status
Function: Displays the brief status of the current Wi-SUN network, including the
      current network name, network mode, network level, PAN identifier, PAN size,
      active key list (GAKs), Group Temporary Key (GTKs), and all parent nodes and
      their topological relationships
For example:
```

```
$ wsbrd_cli status
--------------------------------------------------------------
network_name: Wi-SUN test
fan_version: FAN 1.0
domain: NA
mode: 5
class: 3
panid: 0xb5a5
size: SMALL
GAK[0]: a7:6c:4a:e5:f6:41:95:d0:40:d7:e6:91:37:61:03:6e
GAK[1]: 49:a7:17:1a:b3:0f:69:66:ff:d5:b9:e3:5e:e0:4c:44
GAK[2]: 76:fb:e0:62:c4:0b:28:61:43:66:fd:17:5f:4b:46:7c
GAK[3]: 58:03:65:9d:db:4a:6d:4e:53:1f:a4:30:6b:57:de:cf
GTK[0]: bb:06:08:57:2c:e1:4d:7b:a2:d1:55:49:9c:c8:51:9b
GTK[1]: 18:49:83:5a:01:68:4f:c8:ac:a5:83:f3:70:40:f7:4c
GTK[2]: 59:ea:58:a4:b8:83:49:38:ad:cb:6b:e3:88:c2:62:63
GTK[3]: e4:26:b4:91:bc:05:4a:f3:9b:59:f0:53:ec:12:8e:5f
8e:1f:64:5e:40:00:f2:0b
  `- 8c:1f:64:5e:40:bb:00:02
      `- 8e:1f:64:5e:40:00:f0:07
```

4.1.1.3. start-fan10

```
Format: sudo wsbrd_cli start-fan10
Function: Restart border router
```

4.1.1.4. stop-fan10

```
Format: sudo wsbrd_cli stop-fan10
Function: Border router is turned off, while  the wsbrd application is still
     running.
```

4.1.1.5. set-network-name

```
Format: sudo wsbrd_cli set-network-name [network name]
Function: set PAN network name. After this command is enabled, the border router
     will restart the FAN network, and the node route will be re-connected. The
     set network name should not contain Chinese characters.
```

4.1.1.6. set-wisun-pan-id

```
Format: sudo wsbrd_cli set-wisun-pan-id [network ID]
```

```
Function：Set the PAN network identifier. After this command is enabled, the border
     router will restart the FAN network, and the node route will be re-connected.
     The set network name should not contain Chinese characters.
```

4.1.1.7. set-wisun-pan-size

```
Format：sudo wsbrd_cli set-wisun-pan-size [PAN size]
Function：Set the PAN size (that is, the maximum number of nodes). 1 indicates small
     (less than 100 by default), 8 indicates medium (100-800),15 indicates large
     (800-1500), 25 indicates very large (<2500), and 255 indicates Automatic.
     After this command is run, the border router will restart the FAN network,
     and the node router will be re-connected.
```

4.1.1.8. set-wisun-phy-configs

```
Format：sudo wsbrd_cli set-wisun-phy-configs [domain] [class] [mode]
Function：set basic configuration of Wi-SUN network, which includes domain，class
     and mode. After this command is enabled, the configuration of the border router
     is modified and the FAN network is restarted. The node routes that have been
     connected need to be re-connect.
```

4.1.2. Status Command

4.1.2.1. get-network-name

```
Format：wsbrd_cli get-network-name
Function：get Wi-SUN network name
For example：
$ wsbrd_cli get-network-name
--------------------------------------------------------------
Wisun Network Name: Wi-SUN test
```

4.1.2.2. get-wisun-pan-id

```
Format：wsbrd_cli get-wisun-pan-id
Function：get Wi-SUN network PAN ID
for example：
$ wsbrd_cli get-wisun-pan-id
--------------------------------------------------------------
Wi-SUN PAN id: 0xd4ba
```

4.1.2.3. get-wisun-pan-size

```
Format： wsbrd_cli get-wisun-pan-size
Function： get Wi-SUN network PAN size
For example：
$ wsbrd_cli get-wisun-pan-size
--------------------------------------------------------------
Wi-SUN PAN size: SMALL
```

4.1.2.4. get-wisun-pan-size

```
Format： wsbrd_cli get-wisun-phy-configs
Function： get Wi-SUN network which includes basic configurations of domain, mode
     and class
For example：
$ wsbrd_cli get-wisun-phy-configs
fan_version: FAN 1.0
--------------------------------------------------------------
domain: NA
mode: 5
class: 3
```

4.1.2.5. get-fhss-channel-mask

```
Format： wsbrd_cli get-fhss-channel-mask
Function： Read the frequency-hopping channel mask of the Wi-SUN network, 32bit*8
     indicates the usage of the maximum 256 channels, and 1 indicates that the
     channel is in the frequency-hopping pattern.
for example：
$ wsbrd_cli get-fhss-channel-mask
Fhss channel mask:
--------------------------------------------------------------
[
   0xffffffff,
   0xffffffff,
   0xffffffff,
   0xffffffff,
   0xffffffff,
   0xffffffff,
   0xffffffff,
   0xffffffff,
]
```

4.1.2.6. get-fhss-timing-configure

```
Format：wsbrd_cli get-fhss-timing-configure
Function: Read the frequency hopping timing configuration information of Wi-SUN
     network, including broadcast resident interval (BDI), broadcast interval (BI)
     unicast resident interval (UDI), unicast channel function, broadcast channel
     function
For example：
$ wsbrd_cli get-fhss-timing-configure
Fhss timing configure:
--------------------------------------------------------------
uc_dwell_interval:  15
broadcast_interval: 1020
bc_dwell_interval:  255
uc_channel_function:2
bc_channel_function:2
```

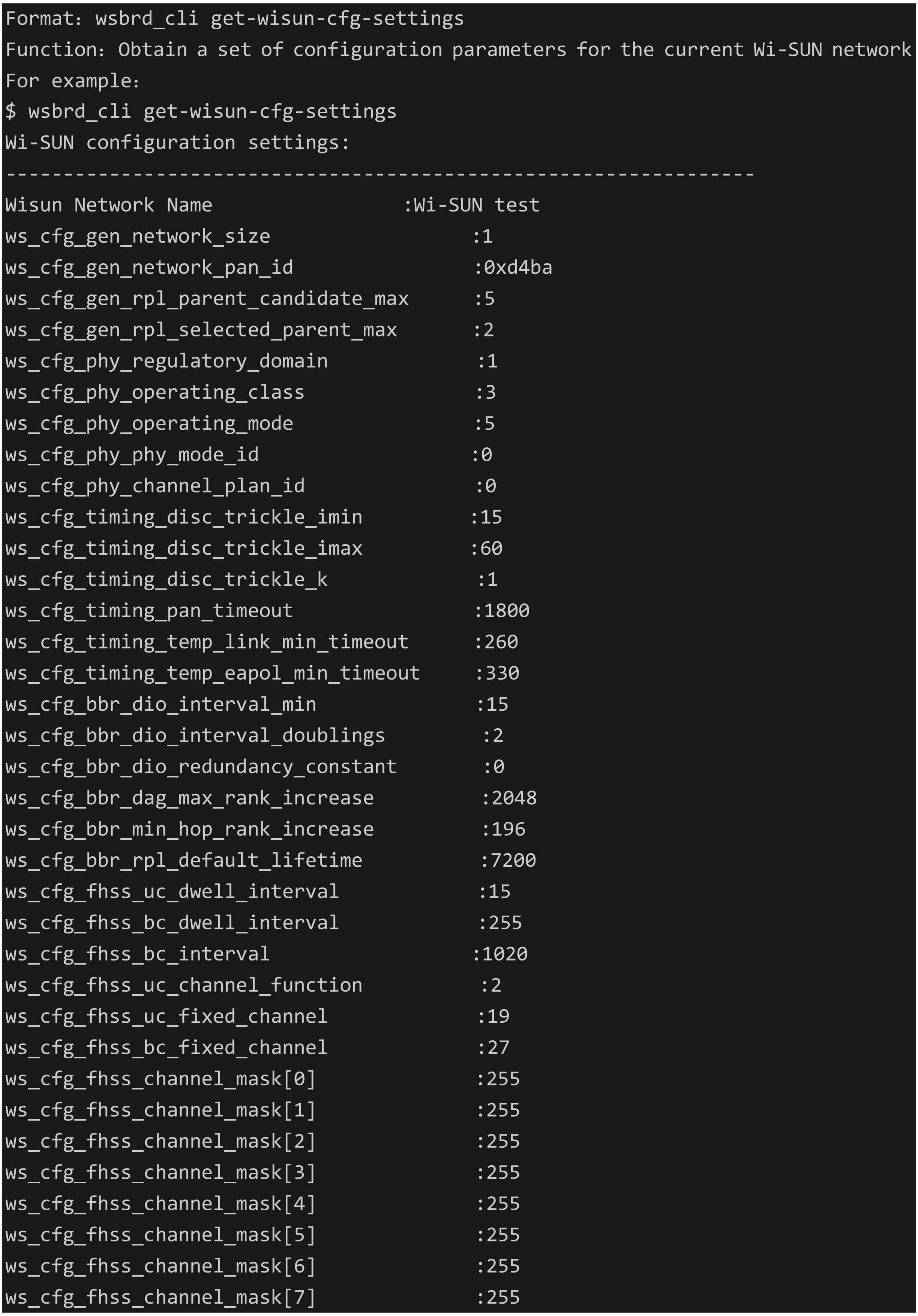
4.1.2.7. get-network-state

```
Format：wsbrd_cli get-network-state
Function：Obtain the link local address and ipv6 address of the current Wi-SUN border
     router.
For example：
$ wsbrd_cli get-network-state
Network state:
--------------------------------------------------------------
IP address[0]: fe80::8c1f:645e:4000:f20b
IP address[1]: fd00:6868:6868:0:8c1f:645e:4000:f20b
```

4.1.2.8. get-timing-parameters

```
Format：wsbrd_cli get-timing-parameters
Function：Obtain the current Wi-SUN network time parameters (in seconds), including
     values of trickle_imin, trickle_imax, trickle_k, and pan_timeout.
For example：
$ wsbrd_cli get-timing-parameters
Timing parameters:
--------------------------------------------------------------
disc_trickle_imin: 15
disc_trickle_imax: 60
disc_trickle_k:    1
pan_timeout:       1800
```

4.1.2.9. get-wisun-cfg-settings



```
Format：wsbrd_cli get-wisun-cfg-settings
Function：Obtain a set of configuration parameters for the current Wi-SUN network
For example：
$ wsbrd_cli get-wisun-cfg-settings
Wi-SUN configuration settings:
-----------------------------------------------------------------
Wisun Network Name                :Wi-SUN test
ws_cfg_gen_network_size                 :1
ws_cfg_gen_network_pan_id                :0xd4ba
ws_cfg_gen_rpl_parent_candidate_max      :5
ws_cfg_gen_rpl_selected_parent_max       :2
ws_cfg_phy_regulatory_domain             :1
ws_cfg_phy_operating_class               :3
ws_cfg_phy_operating_mode                :5
ws_cfg_phy_phy_mode_id                  :0
ws_cfg_phy_channel_plan_id               :0
ws_cfg_timing_disc_trickle_imin         :15
ws_cfg_timing_disc_trickle_imax         :60
ws_cfg_timing_disc_trickle_k             :1
ws_cfg_timing_pan_timeout                :1800
ws_cfg_timing_temp_link_min_timeout      :260
ws_cfg_timing_temp_eapol_min_timeout     :330
ws_cfg_bbr_dio_interval_min              :15
ws_cfg_bbr_dio_interval_doublings         :2
ws_cfg_bbr_dio_redundancy_constant        :0
ws_cfg_bbr_dag_max_rank_increase          :2048
ws_cfg_bbr_min_hop_rank_increase          :196
ws_cfg_bbr_rpl_default_lifetime          :7200
ws_cfg_fhss_uc_dwell_interval            :15
ws_cfg_fhss_bc_dwell_interval            :255
ws_cfg_fhss_bc_interval                 :1020
ws_cfg_fhss_uc_channel_function          :2
ws_cfg_fhss_uc_fixed_channel             :19
ws_cfg_fhss_bc_fixed_channel             :27
ws_cfg_fhss_channel_mask[0]              :255
ws_cfg_fhss_channel_mask[1]              :255
ws_cfg_fhss_channel_mask[2]              :255
ws_cfg_fhss_channel_mask[3]              :255
ws_cfg_fhss_channel_mask[4]              :255
ws_cfg_fhss_channel_mask[5]              :255
ws_cfg_fhss_channel_mask[6]              :255
ws_cfg_fhss_channel_mask[7]              :255
```

```
ws_cfg_mpl_trickle_imin                 :1
ws_cfg_mpl_trickle_imax                 :10
ws_cfg_mpl_trickle_k                    :8
ws_cfg_mpl_trickle_timer_exp             :2
ws_cfg_mpl_seed_set_entry_lifetime       :180
ws_cfg_sectimer_gtk_expire_offset        :43200
ws_cfg_sectimer_pmk_lifetime             :172800
ws_cfg_sectimer_ptk_lifetime             :86400
ws_cfg_sectimer_gtk_new_act_time         :720
ws_cfg_sectimer_revocat_lifetime_reduct   :30
ws_cfg_sectimer_gtk_new_install_req       :80
```

4.1.2.10. get-wisun-gtk-active-key-index

```
Format: wsbrd_cli get-wisun-gtk-active-key-index
Function: Gets the GTK active index of the current Wi-SUN network.
For example:
$ wsbrd_cli get-wisun-gtk-active-key-index
--------------------------------------------------------------
Wi-SUN GTK active key index: 0
```

4.1.2.11. get-wisun-gtk-keys

```
Format: wsbrd_cli get-wisun-gtk-keys
Function: Gets the current Wi-SUN network GTKs list and the current active index
      value.
For example:
$ wsbrd_cli get-wisun-gtk-keys
Wi-SUN GTKs:
--------------------------------------------------------------
Wi-SUN GTK active key index: 0
GTK[0]: bb:06:08:57:2c:e1:4d:7b:a2:d1:55:49:9c:c8:51:9b
GTK[1]: 18:49:83:5a:01:68:4f:c8:ac:a5:83:f3:70:40:f7:4c
GTK[2]: 59:ea:58:a4:b8:83:49:38:ad:cb:6b:e3:88:c2:62:63
GTK[3]: e4:26:b4:91:bc:05:4a:f3:9b:59:f0:53:ec:12:8e:5f
```

4.1.3. Key Configuration Command

4.1.3.1. revoke-group-keys

```
Format: sudo wsbrd_cli revoke-group-keys [GTK] [LGTK]
```

```
Function：Revoke and insert new GTK, LGTK. The network currently does not support
     LGTK and this input parameter is meaningless
For example：
$ sudo wsbrd_cli revoke-group-keys
     “bb:06:08:57:2c:e1:4d:7b:a2:d1:55:49:9c:c8:51:9b”
     “18:49:83:5a:01:68:4f:c8:ac:a5:83:f3:70:40:f7:4c”
```

4.1.3.2. set-wisun-gtk-active-key

```
Format：sudo wsbrd_cli set-wisun-gtk-active-key [gtk_index]
Function：Set the active GTK index. This index value selects the active group among
     the installed GTK key groups.
For example：
$ sudo wsbrd_cli set-wisun-gtk-active-key 1; # active GTK keys of the second group
```

4.1.3.3. set-wisun-gtk-key

```
Format：sudo wsbrd_cli set-wisun-gtk-gtk [gtk_key]
Function：install new temporaryly key(GTK) without activate it
For example：
$ sudo wsbrd_cli set-wisun-gtk-key 18:49:83:5a:01:68:4f:c8:ac:a5:83:f3:70:40:f7:4c
```

4.1.3.4. set-wisun-gtk-time-settings

```
Format：sudo wsbrd_cli set-wisun-gtk-time-settings [Logout Time] [Activation Time]
     [Installation Time] [Maximum Mismatch Value]
Function：Set GTK life cycle related parameters. Currently, this command only passes
     the activation time and installation time (2nd parameter and 3rd parameter),
     the other two parameters don't work.
For example：
$ sudo wsbrd_cli set-wisun-gtk-time-settings 0 720 80 0
```

Lifetime cycle-related parameters such as pmk_lifetime, ptk_lifetime, gtk_new_activation_time (activation time), gtk_new_install_required (installation time), ffn_revocation_lifetime_reduction (revocation time), can be set in the border router configuration file while the parameters set in the file will be loaded into the program configuration when the border router starts up and will remain unchanged.

4.1.3.5. set-wisun-key-lifetime

```
Format：sudo wsbrd_cli set-wisun-key-lifetime [gtk_lifetime] [pmk_lifetime]
     [ptk_lifetime]
Function：set period value of the GTK，PMK and PTK.
For example：
```

```
$ sudo wsbrd_cli set-wisun-key-lifetime 4320 8640 172800; #30 days,2,4 months
```

4.1.4. Frequency Hopping Configured Command

4.1.4.1. set-fhss-bc-function

```
Format： sudo wsbrd_cli set-fhss-bc-function [channel_function] [fixed channel]
     [dwell interval] [broadcast interval]
Function： Set the Wi-SUN network frequency hopping Broadcast channel function (BC
     channel function), fixed channel options, broadcast residential interval, and
     broadcast interval. The channel equation is 0- fixed channel, 1-TR51 frequency
     hopping, 2- direct hash frequency hopping.
```

4.1.4.2. set-fhss-uc-function

```
Format： sudo wsbrd_cli set-fhss-uc-function [channel_function] [fixed channel]
     [dwell interval]
Function： Set the Wi-SUN network frequency-hopping unicast channel equation (UC
     channel function), fixed channel options, and unicast resident interval. The
     channel equation is 0- fixed channel, 1-TR51 frequency hopping, 2- direct hash
     frequency hopping.
```

4.1.4.3. Set FHSS channel mask

```
Format： sudo wsbrd_cli set-fhss-channel-mask-f4b [1st u32] [2nd u32] [3rd u32] [4th
     u32]
     sudo wsbrd_cli set-fhss-channel-mask-l4b [1st u32] [2nd u32] [3rd u32] [4th
     u32]
Function： Set the Wi-SUN frequency hopping FHSS channel mask. The channel mask is
     32*8 bits in total, and each bit controls a channel switch, corresponding to
     eight U32 bits, which are set by the first 4 words and the last 4 words. The
     maximum channels are 256 while in actual use, the channels number will vary
     according to the selection of domain, mode, and class.
```

4.1.4.4. set-fhss-timing-configure

```
Format： sudo wsbrd_cli set-fhss-timing-configure [UDI] [BI] [BDI]
Function： Set time interval of Wi-SUN FHSS, including unicast dwell interval,
     broadcast interval, and broadcast dwell interval. All are measured in
     milliseconds (ms).
```

4.1.5. UDP Command

4.1.5.1. create-udp-socket

```
Format： sudo wsbrd_cli create-udp-socket [UDP port]
Function： Create a UDP socket with specified port number.
```

4.1.5.2. set-udp-dst-port

```
Format： sudo wsbrd_cli set-udp-dst-port [UDP port]
Function： Set the port number of the current UDP socket. The port number is specified
     when a UDP socket is created. Be careful while using this command.
```

4.1.5.3. set-udp-body-unit

```
Format： sudo wsbrd_cli set-udp-body-unit [26-byte UDP body]
Function： The UDO packet data wait to be sent during test is consists of the repeated
     26 bytes plus UDP_tail with 10 bytes.
For example：
$sudo wsbrd_cli set-udp-body-unit 1:2:3:4:...:26
```

4.1.5.4. set-udp-body-uint-repeat-time

```
Format： sudo wsbrd_cli set-udp-body-uint-repeat-time [repeat times]
Function： The outgoing UDO packet data during test is consists of the repeated 26
     bytes plus UDP_tail with 10 bytes. This command sets the repeat times of the
     26-byte of the UDP packet. Notes: 26* repeat times less than MTU (1280)
For example：
$sudo wsbrd_cli set-udp-body-uint-repeat-time 10
```

4.1.5.5. set-udp-tail

```
Format： sudo wsbrd_cli set-udp-tail [10-byte UDP tail]
Function： The outgoing UDO packet data during test is consists of the repeated 26
     bytes. This command sets the 10-byte UDP tail of the packet.
For example：
$sudo wsbrd_cli set-udp-tail 1:2:3:...:10
```

4.1.5.6. socket-udp-sent-to

```
Format： sudo wsbrd_cli socket-udp-sent-to [dest_addr]
Function： Sends the outgoing UDP data packet to the specified target ipv6 address.
For example：
$sudo wsbrd_cli socket-udp-sent-to fd00:6868:6868::8c1f:645e: 4000: f007
```

4.1.5.7. Transmit Python command for the specified UDP packet

This is a more flexible UDP packet transmitting tool based on Python3 support. It can directly create a specified port of UDP socket and specified destination ipv6 address on the network layer of HOPERF Wi-SUN border router, and send the given serial data through the socket

```
Format： python3 tools/udp_packet_send.py [addr] [UDP_PORT] [udp_body]
Function： Sends the given serial data to the specified port of the designed ipv6
     address via UDP socket.
For example：
$python3 tools/udp_packet_send.py fd00:6868:6868::8c1f:645e:4000:f007 5834
     “hello, i am HOPERF Wi-SUN border router...”
```

4.1.6. ICMPv6 Packet Transmitting Command

4.1.6.1. set-icmpv6-id

```
Format： sudo wsbrd_cli set-icmpv6-id [icmpv6_id]
Function： Set the current packet ID of the ICMPv6 echo request
```

4.1.6.2. set-icmpv6-seqnum

```
Format： sudo wsbrd_cli set-icmpv6-seqnum [icmpv6_seqnum]
Function： Set the sequence number of the current ICMPv6 echo request packet
```

4.1.6.3. set-icmpv6-body-unit

```
Format： sudo wsbrd_cli set-icmpv6-body-unit [26-byte UDP body]
Function： The outgoing ICMPv6 data wait to be sent during test is consists of the
     repeated 26 bytes plus ICMPv6_tail with 10 bytes. This command sets the
     outgoing 26-byte packet of the ICMPv6 packet.
For example：
$sudo wsbrd_cli set-icmpv6-body-unit 1:2:3:4:...:26
```

4.1.6.4. set-icmpv6-body-uint-repeat-time

```
Format: sudo wsbrd_cli set-icmpv6-body-uint-repeat-time [重复次数]
Function: The outgoing ICMPv6 packet data during test is consists of the repeated
     26-byte data plus ICMPv6_tail with 10 bytes. This command sets the repeat times
     of the outgoing 26-byte of the ICMPv6 packet. Notes: 26* repeat times less
     than MTU (1280).
For example:
$sudo wsbrd_cli set-icmpv6-body-uint-repeat-time 10
```

4.1.6.5. set-icmpv6-tail

```
Format: sudo wsbrd_cli set-icmpv6-tail [10-byte ICMPv6 tail]
Function: The outgoing ICMPv6 packet data during test is consists of the repeated
     26-byte data. This command sets the outgoing 10-byte ICMPv6 tail of the ICMPv6
     packet.
For example:
$sudo wsbrd_cli set-icmpv6-tail 1:2:3:...:10
```

4.1.6.6. set-icmpv6-mtu-size

```
Format: sudo wsbrd_cli set-icmpv6-mtu-size [icmpv6_mtu]
Function: Set the MTU value of ICMPv6 echo request packet.
For example:
$sudo wsbrd_cli set-icmpv6-mtu-size 1000
```

4.1.6.7. send-icmpv6-echo-req

```
Format: sudo wsbrd_cli send-icmpv6-echo-req [dest_addr]
Function: Transmit the ICMPv6 echo request with the ICMPv6 data packet to the target
     ipv6 address.
For example:
$sudo wsbrd_cli send-icmpv6-echo-req fd00:6868:6868::8c1f:645e:4000: f007
```

4.1.7. Border Router Trickling Set Time Command

```
Format: sudo wsbrd_cli set-timing-parameters [trickle_imin] [trickle_imax]
     [trickle_k] [pan_timeout]
Function: Set trickle timing parameters for Wi-SUN border router. It includes the
     minimum trickle interval time (in seconds), maximum trickle interval time (in
     seconds), trickle redundancy constant and PAN network timing timeout time (in
     seconds).
```

```
For example：
$sudo wsbrd_cli set-timing-parameters 60 960 1 3600
```

4.1.8. Multicast Related Command

4.1.8.1. join-multicast-group

```
Format：sudo wsbrd_cli join-multicast-group [mult_addr]
Function：join the specific multicast group
For example：
$sudo wsbrd_cli join-multicast-group ff15::810a:64d1
```

4.1.8.2. leave-multicast-group

```
Format：sudo wsbrd_cli leave-multicast-group [mult_addr]
Function：leave the specific multicast group
For example：
$sudo wsbrd_cli leave-multicast-group ff15::810a:64d1
```

4.1.9. Firmware Upgrade and OTA Command

HOPERF has implemented local RCP serial firmware upgrade and OTA firmware upgrade for network nodes on its Wi-SUN border router. The upgrade files need to be pre-stored in a known directory of the Raspberry Pi system."

Local RCP Upgrade Command：`sudo wsbrd rcp-fw-update /path/rcp_firmware.hex`

Command for network nodes of the border router：

```
sudo wsbrd node-fw-ota ota_multicast_addr /path/ota_node_firmware.hex
```

5. Wi-SUN meter/collector Example

HOPERF has implemented a set of collector/meter applications based on the Wi-SUN network on border routers and node routers.

As a meter node in the application layer, it will periodically collect local sensor measurements and report them to the collector after registering as the Wi-SUN border router node collector. After knowing the IPv6 address of the node router, the border router can register the specified node to its meter list via a UDP packet.

The Collector can also send an asynchronous read request to the meter to obtain its single measurement result. These commands are wsbrd_cli commands, which are described as follows.

```
register-meter [ipv6_addr]; #Register collector to the given meter with
                             destination address
async-request [ipv6_addr]; #Send an async request to the given meter with
                             destination address
remove-meter [ipv6_addr]; #Remove registration from the given meter with
                             destination address
list-meters;              #List registered and async meters
```

6. Revise History

Table 6-1. Revise Records

Version No.	Chapter	Description	Date
0.1	All	Initial version	2024/08/19

7. Contacts

Shenzhen Hope Microelectronics Co., Ltd.

Address:	30th floor of 8th Building, C Zone, Vanke Cloud City, Xili Sub-district, Nanshan, Shenzhen, GD, P.R. China
Tel:	+86-755-82973805 / 4001-189-180
Fax:	+86-755-82973550
Post Code:	518052
Sales:	sales@hoperf.com
Website:	www.hoperf.com

The information furnished by HOPERF is believed to be accurate and reliable. However, no responsibility is assumed for inaccuracies and specifications within this document are subject to change without notice. The material contained herein is the exclusive property of HOPERF and shall not be distributed, reproduced, or disclosed in whole or in part without prior written permission of HOPERF. HOPERF products are not authorized for use as critical components in life support devices or systems without express written approval of HOPERF. The HOPERF logo is a registered trademark of Shenzhen Hope Microelectronics Co., Ltd. All other names are the property of their respective owners.