



Tyxin Linux WiFi FMAC Driver Development Guide



Zhuhai Taixin Semiconductor Co., Ltd.

TaiXin Semiconductor Co., Limited

3rd Floor, Building 11, Harbor One Science and Technology Park, Zhuhai High-tech Zone

Confidentiality Level	A	Tyxin Linux WiFi FMAC Driver Development Guide	Document No.	
issue date	2025/6/27		File Version	V3.1

Liability and Copyright

Limitation of Liability

The contents of this document are for reference only. Zhuhai Taixin Semiconductor Co., Ltd. (hereinafter referred to as "Taixin") does not guarantee the accuracy or completeness of the information in this document.

No representations or warranties, express or implied, are made and no liability is assumed for the consequences of using the information in this document.

Taixin reserves the right to modify, improve or otherwise change this document without prior notice.

Tysin has no obligation to provide assistance for the application and design verification of customer products. Customers are responsible for the design, verification and testing of their own applications.

And ensure that its application complies with all legal, regulatory and safety-related requirements.

If any damage, expense, loss and/or liability is caused by the customer's failure to comply with this agreement, Taixin shall not bear any responsibility; at the same time, the customer shall pay full

any damages, expenses, losses and/or liabilities caused to Taixin due to Customer's failure to comply with this Agreement.

Copyright Notice

Without the written consent of Taixin, no party may modify, adapt, change, translate or create derivative works based on this document for commercial purposes.

Without the written consent of Taixin, no party shall disclose or distribute any part or all of the source code, SDK, Binary files and object code.

No party may modify, reverse engineer, disassemble, decompile or otherwise attempt to discover any non-source code portion of the SDK, including but Not limited to precompiled binaries and object code.

In addition, any actions that may infringe upon the rights of Tysin or other intellectual property owners are strictly prohibited.

For the subject that commits the above-mentioned infringement, Taixin has the right to take legal action in accordance with the laws of the People's Republic of China or other applicable laws and international treaties.

Necessary legal measures, including but not limited to filing a lawsuit or arbitration against the infringer, or applying for legal enforcement measures.

Zhuhai Taixin Semiconductor Co., Ltd.

September 24, 2024



Zhuhai Taixin Semiconductor Co., Ltd.
TaiXin Semiconductor Co., Limited

3rd Floor, Building 11, Harbor One Science and Technology Park, Zhuhai High-tech Zone

Copyright infringement is subject to prosecution

Copyright © 2025 by TaiXin Semiconductor All rights reserved

Confidentiality Level	A	Tyxin Linux WiFi FMAC Driver Development Guide	Document No.	
issue date	2025/6/27		File Version	V3.1


Revision History


date	Version	Description	Revised by
2025/6/27	V3.1	Modify the description of wakeup_io; Modify the description of sysdbg; Added get mode/key_mgmt/bss_bw; Modify the set txpower range; Modify the description of set tx_mcs; Added description of wake-up reason; Delete the description of the device model in the server keep-alive command; Corrected fonts; This document is used to support both 1.x and 2.x SDKs.	WY
2025/3/24	V3.0.1	Fixed typo in wpa_psk;	WY
2025/2/26	V3.0	Added description of SDK V2.x;	WY
2024/8/12	V2.18.2	Corrected the typo in super_pwr description;	WY
2024/6/20	V2.18.1	Modify the type of get sta_list;	WY
2024/5/5	V2.18	Added mode3 description of dcdc13;	WY
2024/4/18	V2.17	Add get signal interface; Modify the description of AP low power consumption (PS_mode4); Modify the description of AP low power wake-up reason;	WY
2024/2/8	V2.16	Modify the description of AP low power consumption; Added description of relay; Added the description that this document only supports 1.x SDK;	WY
2023/12/8	V2.15	Added description of superpwr=2;	WY
2023/11/29	V2.14.1	Adjust the description of country_region/scan/paired_stas;	WY
2023/11/27	V2.14	Added country_region and modified the scan command;	ZEL
2023/10/3	V2.13	Modify the description of wkreason and reassoc_wkhost;	WY
2023/9/20	V2.12	Modify the description of ps_mode and wakeup_io;	WY
2023/9/3	V2.11	Modify the description of pa_pwrctrl_dis;	WY
2023/8/31	V2.10	Modify the firmware download error message description	DY


	Zhuhai TaiXin Semiconductor Co., Ltd. TaiXin Semiconductor Co., Limited	3rd Floor, Building 11, Harbor One Science and Technology Park, Zhuhai High-tech Zone
---	--	---


Copyright infringement is subject to prosecution

Copyright © 2025 by TaiXin Semiconductor All rights reserved

Confidentiality Level	A	Tyxin Linux WiFi FMAC Driver Development Guide	Document No.																																													
issue date	2025/6/27		File Version	V3.1																																												
<table><tr><td>2023/7/28</td><td>V2.9</td><td>Modify the description of the roaming</td><td>DY</td></tr><tr><td>2023/5/23</td><td>V2.8</td><td>Modify multicast mode parameter description</td><td>WY</td></tr><tr><td>2023/5/17</td><td>V2.7</td><td>Modify the roaming interface description</td><td>DY</td></tr><tr><td>2023/4/20</td><td>V2.6</td><td>Added fwinfo interface description Modify the radio_onoff interface description</td><td>DY</td></tr><tr><td>2023/4/13</td><td>V2.5</td><td>Added description of get conn_state</td><td>DY</td></tr><tr><td>2023/4/7</td><td>V2.4</td><td>Add get bssid</td><td>DY</td></tr><tr><td>2023/4/6</td><td>V2.3</td><td>Modify the description of the roaming Corrected the typo in superpwr</td><td>WY</td></tr><tr><td>2023/3/7</td><td>V2.2.1</td><td>Delete several read command parameters</td><td>WY</td></tr><tr><td>2023/2/17</td><td>V2.2</td><td>Fix description of sleep and wake Add wakeup command description</td><td>WY</td></tr><tr><td>2023/2/6</td><td>V2.1</td><td>Added description of dtim and autosleep Added a hyperlink to the standby parameter</td><td>WY</td></tr><tr><td>2023/1/12</td><td>V2.0</td><td>Initial release</td><td>DY</td></tr></table>					2023/7/28	V2.9	Modify the description of the roaming	DY	2023/5/23	V2.8	Modify multicast mode parameter description	WY	2023/5/17	V2.7	Modify the roaming interface description	DY	2023/4/20	V2.6	Added fwinfo interface description Modify the radio_onoff interface description	DY	2023/4/13	V2.5	Added description of get conn_state	DY	2023/4/7	V2.4	Add get bssid	DY	2023/4/6	V2.3	Modify the description of the roaming Corrected the typo in superpwr	WY	2023/3/7	V2.2.1	Delete several read command parameters	WY	2023/2/17	V2.2	Fix description of sleep and wake Add wakeup command description	WY	2023/2/6	V2.1	Added description of dtim and autosleep Added a hyperlink to the standby parameter	WY	2023/1/12	V2.0	Initial release	DY
2023/7/28	V2.9	Modify the description of the roaming	DY																																													
2023/5/23	V2.8	Modify multicast mode parameter description	WY																																													
2023/5/17	V2.7	Modify the roaming interface description	DY																																													
2023/4/20	V2.6	Added fwinfo interface description Modify the radio_onoff interface description	DY																																													
2023/4/13	V2.5	Added description of get conn_state	DY																																													
2023/4/7	V2.4	Add get bssid	DY																																													
2023/4/6	V2.3	Modify the description of the roaming Corrected the typo in superpwr	WY																																													
2023/3/7	V2.2.1	Delete several read command parameters	WY																																													
2023/2/17	V2.2	Fix description of sleep and wake Add wakeup command description	WY																																													
2023/2/6	V2.1	Added description of dtim and autosleep Added a hyperlink to the standby parameter	WY																																													
2023/1/12	V2.0	Initial release	DY																																													
<div><div><p>TaiXin Semiconductor</p></div><div><p>Zhuhai Taixin Semiconductor Co., Ltd.</p><p>TaiXin Semiconductor Co., Limited</p></div><div><p>3rd Floor, Building 11, Harbor One Science and Technology Park, Zhuhai High-tech Zone</p></div></div> <div>Copyright infringement is subject to prosecution</div> <div>Copyright © 2025 by TaiXin Semiconductor All rights reserved</div>																																																

Confidentiality Level	A	Tyxin Linux WiFi FMAC Driver Development Guide	Document No.	
issue date	2025/6/27		File Version	V3.1
Table of contents				
Tyxin Linux WiFi FMAC Driver Development Guide..... 1				
1. Overview..... 1				
2. Linux Kernel Compilation Configuration..... 1				
3. WiFi Driver Development and Usage Instructions.....2				
3.1. hgic_fmac 2				
3.1.1. hgic_fmac driver file..... 2				
3.1.2. hgic_fmac loading process.....3				
3.1.3. hgpriv configuration instructions.....4				
Basic Network Parameters.....5				
1. hgpriv hg0 set mode=xx5				
2. hgpriv hg0 set ssid=xx..... 5				
3. hgpriv hg0 set key_mgmt=xx 5				
4. hgpriv hg0 set wpa_psk=64_hexchar5				
5. hgpriv hg0 set pairing=xx6				
6. hgpriv hg0 set bss_bw=xx6				
7. hgpriv hg0 set chan_list=freq0,freq1,...,freqN..... 6				
8. hgpriv hg0 set freq_range=start,end,bw.....6				
9. hgpriv hg0 set country_region=xx7				
10. hgpriv hg0 set acs=1,107				
		Zuhai Taixin Semiconductor Co., Ltd. TaiXin Semiconductor Co., Limited	3rd Floor, Building 11, Harbor One Science and Technology Park, Zhuhai High-tech Zone	
Copyright infringement is subject to prosecution				
Copyright © 2025 by TaiXin Semiconductor All rights reserved				

Confidentiality Level	A	Tyxin Linux WiFi FMAC Driver Development Guide	Document No.	
issue date	2025/6/27		File Version	V3.1
<div>Debug Commands..... 7</div> <div>1. hgpriv hg0 set loaddef=0/1 7</div> <div>2. hgpriv hg0 set dbginfo=0/18</div> <div>3. hgpriv hg0 set sysdbg=type, 0/1/28</div> <div>4. hgpriv hg0 set atcmd=at+xx8</div> <div>5. hgpriv hg0 scan=0/1/2?8</div> <div>Status Query Commands.....9</div> <div>1. hgpriv hg0 get conn_state9</div> <div>2. hgpriv hg0 get module_type9</div> <div>3. hgpriv hg0 get mode 10</div> <div>4. hgpriv hg0 get ssid.....10</div> <div>5. hgpriv hg0 get key_mgmt.....10</div> <div>6. hgpriv hg0 get bss_hw.....10</div> <div>7. hgpriv hg0 get center_freq 10</div> <div>8. hgpriv hg0 get sta_list 11</div> <div>9. hgpriv hg0 get scan_list11</div> <div>10. hgpriv hg0 get disassoc_reason.....11</div> <div>11. hgpriv hg0 get bgrssi=chan_index11</div> <div>12. hgpriv hg0 get bssid (supported only by SDK V1.x).....11</div> <div>13. hgpriv hg0 get signal 12</div>				
		Zhuhai Taixin Semiconductor Co., Ltd. TaiXin Semiconductor Co., Limited	3rd Floor, Building 11, Harbor One Science and Technology Park, Zhuhai High-tech Zone	
Copyright © 2025 by TaiXin Semiconductor All rights reserved				

Confidentiality Level	A	Tyxin Linux WiFi FMAC Driver Development Guide	Document No.	
issue date	2025/6/27		File Version	V3.1
<div>14. hgpriv hg0 get fwinfo 12</div> <div>Network Advanced Parameters..... 12</div> <div>1. hgpriv hg0 set txpower=xx12</div> <div>2. hgpriv hg0 set super_pwr=xx 12</div> <div>3. hgpriv hg0 set tx_mcs=xx13</div> <div>4. hgpriv hg0 set agg_cnt=xx13</div> <div>5. hgpriv hg0 set max_txcnt=xx13</div> <div>6. hgpriv hg0 set ap_hide=113</div> <div>7. hgpriv hg0 set bss_max_idle=xx.....13</div> <div>8. hgpriv hg0 set disassoc_sta=f8:de:09:96:8f:28 14</div> <div>9. hgpriv hg0 set unpair=f8:de:09:96:8f:2814</div> <div>10. hgpriv hg0 set conn_paironly=1/0 14</div> <div>11. hgpriv hg0 set paired_stas=mac1,mac2,... 14</div> <div>12. hgpriv hg0 set pair_autostop=xx 15</div> <div>13. hgpriv hg0 set acktmo=xx15</div> <div>14. hgpriv hg0 set channel=xx15</div> <div>Low power consumption related parameters..... 15</div> <div>1. hgpriv hg0 set sleep=1 15</div> <div>2. hgpriv hg0 set ps_mode=xx16</div> <div>3. hgpriv hg0 set ap_psmode=1 16</div>				
		Zhuhai Taixin Semiconductor Co., Ltd. TaiXin Semiconductor Co., Limited	3rd Floor, Building 11, Harbor One Science and Technology Park, Zhuhai High-tech Zone	
Copyright © 2025 by TaiXin Semiconductor All rights reserved				

Confidentiality Level	A	Tyxin Linux WiFi FMAC Driver Development Guide	Document No.	
issue date	2025/6/27		File Version	V3.1

4. hgpriv hg0 set wakeup_io=1,0.....16

5. hgpriv hg0 set wkio_mode=xx 17

6. hgpriv hg0 set wkhost_reason=2,7,8,12,14 17

7. hgpriv hg0 get wkreason 17

8. hgpriv hg0 set dcdc13=xx 18

9. hgpriv hg0 set pa_pwrctl_dis=xx 19

10. hgpriv hg0 set dtim_period=xx 19

11. hgpriv hg0 set autosleep_time=xx 20

12. hgpriv hg0 set wait_psmode=0/120

13. hgpriv hg0 set ps_connect=60,420

14. hgpriv hg0 set dis_psconnect=121

15. hgpriv hg0 set standby_rfm_idx,wakeuptime.....21

16. hgpriv hg0 set apreset_time=xx 21

17. hgpriv hg0 set wkdata_save=1 21

18. hgpriv hg0 set heartbeat=ip_str,port,hb_interval,hb_tmo 21

19. hgpriv hg0 set heartbeat_resp 22

20. hgpriv hg0 set wakeup_data 22

21. hgpriv hg0 set wkdata_mask 22

22. hgpriv hg0 set wakeup=mac_addr 23

23. hgpriv hg0 set reassoc_wkhost=123

	Zhuhai Taixin Semiconductor Co., Ltd. TaiXin Semiconductor Co., Limited	3rd Floor, Building 11, Harbor One Science and Technology Park, Zhuhai High-tech Zone
---	--	---

Confidentiality Level	A	Tyxin Linux WiFi FMAC Driver Development Guide	Document No.	
issue date	2025/6/27		File Version	V3.1

Multicast Mode Parameters (Supported only by SDK V1.x)..... 23

1. hgpriv hg0 set join_group=group_addr,aid 23

2. hgpriv hg0 set mcast_txparam=dupcnt,tx_bw,tx_mcs,clearch...24

Relay Parameters (Note that the definitions of SDK V1.x and V2.x are different).....24

1. hgpriv hg0 set r_ssid=xx24

2. hgpriv hg0 set r_psk=64_hexchar24

Roaming Parameters (Note that SDK V2.x standard protocol does not support)..... 25

1. hgpriv hg0 set roaming=onoff,0,threshold,rssi_diff,rssi_int25

Dual Antenna Parameters..... 25

1. hgpriv hg0 set ant_auto=1 25

2. hgpriv hg0 set ant_sel=0/125

Other Parameters..... 26

1. hgpriv hg0 set auto_chswitch=xx26

2. hgpriv hg0 set auto_save=xx26

3. hgpriv hg0 save..... 26

4. hgpriv hg0 set dhcpc=1 26

5. hgpriv hg0 set reset_sta=mac_addr 27

6. hgpriv hg0 set radio_onoff=x27

3.1.4. Proc fs interface.....27

/proc/hgicf/status (read-only).....27

1. Overview

Taixin Linux WiFi FMAC driver supports WiFi protocol stack running inside WiFi module, the host does not need WiFi Protocol stack.

The FMAC driver supports the AH module and low power mode. The corresponding WiFi firmware is v2.xx5 type.

2. Linux Kernel compilation configuration

The FMAC driver supports both SDIO and USB interfaces. The following functions need to be enabled when compiling the kernel:

1. According to the interface you choose to use (sdio/usb), open the corresponding support module

1) sdio interface: open *Device Driver* → *MMC/SD/SDIO card support* and the corresponding mmc host driver.

2) USB interface: open *Device Driver* → *USB support*, and the corresponding usb host driver.

Note: If the following error occurs when compiling the driver: `mmc_card_disable_cdUndefinederror`, please open `if_sdio.c` `mmc_card_disable_cd` definition code, as shown below:

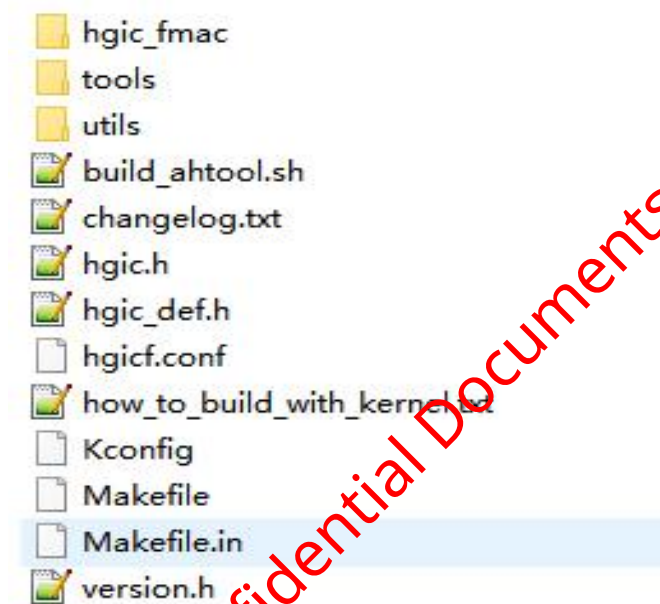
```
5:
6: #define FUNC_DEV(f)    (&((f) %dev))
7:
8: #define SDIO_CAP_IRQ(func) ((func)->card->host->caps & MMC_CAP_SDIO_IRQ)
9: #define SDIO_CAP_POLL(func) ((func)->card->host->caps & MMC_CAP_NEEDS_POLL)
10: #define HOST_SPI_CRC(func, crc) (func)->card->host->use_spi_crc=crc
11:
12: //define mmc_card_disable_cd(c) (1)
13: #define hgic_card_disable_cd(func)    mmc_card_disable_cd((func)->card)
14: #define hgic_card_set_highspeed(func) mmc_card_set_highspeed((func)->card)
15: #define hgic_host_is_spi(func)        mmc_host_is_spi((func)->card->host)
16: #define hgic_card_cccr_widebus(func)  ((func)->card->cccr.low_speed && !(func)->card->cccr.high_speed)
17: #define hgic_card_cccr_highspeed(func) ((func)->card->cccr.high_speed)
18: #define hgic_host_highspeed(func)    ((func)->card->host->caps & MMC_CAP_SD_HIGHSPEED)
19: #define hgic_host_supp_4bit(func)    ((func)->card->host->caps & MMC_CAP_4_BIT_DATA)
20: #define hgic_card_highspeed(func)    mmc_card_highspeed((func)->card)
21: #define hgic_func_rca(func)          ((func)->card->rca)
22: #define hgic_card_max_clock(func)   ((func)->card->cis.max_dtr)
```

3. WiFi Driver Development and Usage Instructions

3.1. hgic_fmac

3.1.1. hgic_fmac driver files

The hgic_fmac driver is released as source code, and users can compile it by themselves:



When compiling the fmac driver, you can choose to support sdio or usb interface. Execute the make command to view the compilation instructions.

```
-----
usage:
make smac      : compile SMAC driver. support sdio/usb interface. generate hgics.ko
make smac_usb  : compile SMAC driver. only support usb interface. generate hgics.ko
make smac_sdio: compile SMAC driver. only support sdio interface. generate hgics.ko

make fmac      : compile FMAC driver. support sdio/usb interface. generate hgicf.ko
make fmac_usb  : compile FMAC driver. only support usb interface. generate hgicf.ko
make fmac_sdio: compile FMAC driver. only support sdio interface. generate hgicf.ko

make clean
-----
```

Auxiliary tools and driver encapsulation APIs are provided in the tools/test_app directory.

Execute build_ahtool.sh to compile auxiliary tools, such as hgpriv, hgicfwait.

3.1.2. hgic_fmac loading process

1. Load the fmac driver: insmod hgicf.ko.

When loading the driver, you can specify the following parameters as needed:

- **ifname:** Specify the network interface name. The driver creates the hg0 interface by default (hg0 is used in the following examples).

Such as: `insmod hgicf.ko ifname="wlan%d"`, the driver will create a wlanx interface (x is %d The obtained index).

- **conf_file:** Used to specify the loading parameter configuration file. The default value of this parameter is `/etc/hgicf.conf`. There are two usages:

Usage:

- 1) The `conf_file` parameter value is **Specific file path**, For example:

`insmod hgicf.ko conf_file=/etc/hgicf.conf` At this time, the driver will load the specified parameter file.

This method can be used for network card solutions.

- 2) The `conf_file` parameter value is **Directory Path**, for example: `insmod hgicf.ko conf_file=/etc`

The driver will then **Read** from directory `/etc/hg0.conf` parameter file. This method is required for dual network card solutions.

For example, in the same system 2 Network cards: hg0 and hg1 In this way, the driver will

Read `/etc/hg0.conf` and `/etc/hg1.conf` As hg0 and hg1 Parameters.

- **fw_file:** Specify the AH module firmware name. The driver loads `hgicf.bin` by default. For details, see [3.3 Firmware](#)

[download](#) chapter

Such as: `insmod hgicf.ko fw_file=xxxx.bin`

Note: The `fw_file` parameter value can only be a file name and cannot contain a path.

2. Configure the hg0 interface: set the IP address and bring up the interface.

3. Use the hgpriv tool to configure parameters (you can also use the encapsulation API)

The most basic parameter settings are as follows:

- `hgpriv hg0 set freq_range=9080,9240,8` # Set the AH module operating frequency range and bandwidth.
- `hgpriv hg0 set bss_bw=8` # Set bss bandwidth to 8M, optional 1/2/4/8, same as above bandwidth

To

- hgpriv hg0 set tx_mcs=255#Set tx mcs to automatic mode
- hgpriv hg0 set key_mgmt=NONE#Turn off encryption.
- hgpriv hg0 set ssid=ah_test_ssid#Set SSID
- hgpriv hg0 set mode=ap#Set working mode ap

Parameter setting order: set frequency/bandwidth information, set encryption method/password, and finally set working mode.

4. Driver configuration file [Optional]

If you need to quickly load parameters at startup, you can create a parameter configuration file for the hgicf driver and set it when loading the driver.

conf_file parameter, the driver will automatically load the setting parameters.

The contents of the parameter configuration file are the parameters of the hgpriv command.

For example:

hgpriv hg0 set mode=ap, the corresponding content of the configuration file is: mode=ap

hgpriv hg0 set ssid=ap_ssid, the corresponding content of the configuration file is: ssid=ap_ssid

Sample Files:

```
freq_range=9080,9240,8
bss_bw=8
tx_mcs=25
acs=1,10
key_mgmt=NONE
ssid=ah_test_ssid
mode=ap
```

3.1.3. hgpriv Configuration Instructions

illustrate: The hgpriv tool is recommended only for manual input. The code uses the one provided by the driver: Encapsulation API, you can directly

Integrated into the application, more convenient to use.

1. hgpriv hg0 set mode=xx

Set the working mode of the WiFi module, AP or STA or Group

- mode=ap : Work in AP mode
- mode=sta : Work in STA mode
- mode=group: works in broadcast group mode(Not supported in V2.x)
- mode=apsta: Works in relay mode. The device in relay mode acts as both a STA to connect to the upper AP level and an AP to

Provide connection services for other STAs.

[Please set other networking parameters first, and then set the mode parameter]

2. hgpriv hg0 set ssid=xx

Set the SSID of the AH module. The maximum length is 32 characters.

3. hgpriv hg0 set key_mgmt=xx

Set the encryption mode of the WiFi module

- NONE : Disable encryption
- WPA-PSK: Enable encryption. (need to set wpa_psk)

4. hgpriv hg0 set wpa_psk=64_hexchar

Set the WiFi module encryption psk, the psk value is 64 hex characters.

This character can be generated with the wpa_passphrase tool, for example:

```
# wpa_passphrase hgic_ah_test 12345678
network={
    ssid="hgic_ah_test"
    #psk="12345678"
    psk=baa58569a9edd7c3a55e446bc658ef76a7173d023d256786832474d737756a82
}
```

The red line indicates the generated PSK.

[If you do not want to use wpa_passphrase, you can generate 64 hex characters between 0 and f yourself]

5. hgpriv hg0 set pairing=xx

Set the start/stop of the one-touch pairing function of the WiFi module. This function is used to control two WiFi modules to pair automatically.

- pairing=1 : Enable pairing
- pairing=0 : stop pairing

For detailed instructions, please see [One-click pairing](#) Section.

6. hgpriv hg0 set bss_bw=xx

Set the BSS bandwidth of the WiFi module. Valid values are: 1/2/4/8

7. hgpriv hg0 set chan_list=freq0,freq1,...,freqN

Set the frequency list of the WiFi module to customize the working frequency. You can set non-continuous frequencies in units of 0.1MHz. Supports up to 16 frequencies.

8. hgpriv hg0 set freq_range=start,end,bw

Set the AH module operating frequency range. The unit of start and end is 0.1MHz, and the unit of bw is Mhz. For example
freq_range=9080,9160,8:

- 9080: starting center frequency, 908M
- 9240: End center frequency, 924M
- 8 : bss bandwidth, 8M [This value needs to be consistent with bss_bw]

Note that the channels generated by this command are continuous without gaps. If you want to generate discontinuous channels, you can only use the following chan_list. If chan_list is used, the setting of freq_range will not be considered. chan_list has a higher priority than freq_range.

high.

9. hgpriv hg0 set country_region=xx

Set the country code of the AH module. After successful setting, the module will switch to the corresponding country/region standard according to the bss_bw parameter and country code.

For detailed working frequency, please refer to the document "Tyxin 802.11AH Frequency Setting Instructions".

Currently AH firmware supports US (United States), EU (European Union), KR (Korea), SG (Singapore), AU (Australia), NZ (New Zealand), ID (Indonesia), JP (Japan), MY (Malaysia), TH (Thailand) and other ten countries/regions

Click Settings.

Note that this command is mutually exclusive with the set chan_list and set freq_range commands. Use the latter two commands to set the frequency.

Click to clear the country code field.

10. hgpriv hg0 set acs=1,10

Set the automatic frequency selection function of the WiFi module. When this function is enabled, the AH module will automatically select the frequency with the least interference within the specified range.

The smaller frequency point is used as the working frequency point. Generally, it is only used in AP mode.

Parameter Description:

- 1: turns on the function (0: turns off the function)
- 10: The frequency monitoring time of the automatic frequency selection function is 10ms, which generally does not need to be adjusted.

Debug commands

1. hgpriv hg0 set loaddef=0/1

loaddef=1: restore default parameters, which will erase the parameter area in the WiFi module flash.

The block will restart.

loaddef=0: Do not restart after executing the recovery parameters.

2. hgpriv hg0 set dbginfo=0/1

Enable or disable firmware debug information output.

When this function is enabled, the firmware debugging information will be output to the WiFi driver and printed out.

Test information for problem analysis.

hgpriv hg0 set dbginfo=1 # Enable firmware debugging information output

hgpriv hg0 set dbginfo=0 # Disable firmware debugging information output

3. hgpriv hg0 set sysdbg=type, 0/1/2

Enable or disable certain types of firmware debug information output.

Type represents the category of debug information with the following:

- heap: Heap information, default = 0, closed; =1, open;
- top: CPU usage information of each thread, default = 0 is off; =1 is on;
- wnb : WiFi protocol stack information (only for SDK V1.x), default = 0 to disable; = 1 to enable;
- lmac: lmac layer information, default = 2 for simplified information; =0 for disabled; =1 for complete information;
- umac : umac layer information (only for SDK V2.x), default = 0 to disable; = 1 to enable;

4. hgpriv hg0 set atcmd=at+xx

Send AT commands to the module through the main control.

5. hgpriv hg0 scan=0/1/2?

Execute this command in STA mode to scan surrounding AP information.

- scan=0: stop scanning;
- scan=1: start scanning (do not clear the previously scanned AP list before starting, it will be cached for 10 seconds);
- scan=2: Start scanning (clear the previously scanned AP list before starting).

Status query command

1. hgpriv hg0 get conn_state

Query the connection status of the WiFi module.

Return value: 0 or 9.

```
enum hgicf_hw_state {
    HGICF_HW_DISCONNECTED    = 0,
    HGICF_HW_DISABLED        = 1,
    HGICF_HW_INACTIVE        = 2,
    HGICF_HW_SCANNING        = 3,
    HGICF_HW_AUTHENTICATING  = 4,
    HGICF_HW_ASSOCIATING     = 5,
    HGICF_HW_ASSOCIATED      = 6,
    HGICF_HW_4WAY_HANDSHAKE  = 7,
    HGICF_HW_GROUP_HANDSHAKE = 8,
    HGICF_HW_CONNECTED       = 9;
};
```



2. hgpriv hg0 get module_type

Get the AH module type information. The application sets relevant parameters adaptively according to the module type.

utils/ah_freqinfo.c defines the frequency information/transmit power of each region. The application can integrate this part of the code.

Adaptively set frequency/transmit power parameters according to different regions.

The return value of this command is a 16-bit number, including two fields: Type (lower 8 bits), Saw (higher 8 bits)

```
struct hgic_module_hwinfo{
    union{
        struct{
            unsigned char type;
            unsigned char saw:1, rev:7;
        };
        unsigned short v;
    };
};
```

Type has the following meanings:

- 1: 700M module
- 2: 900M module
- 3: 860M module

Saw has the following meanings:

- 0: without saw
- 1: With saw

3. hgpriv hg0 get mode

For the definition of mode, please refer to [set mode](#).

4. hgpriv hg0 get ssid

Get ssid.

5. hgpriv hg0 get key_mgmt

To obtain the encryption method, query set key_mgmt.

6. hgpriv hg0 get bss_bw

Get bss_bw.

7. hgpriv hg0 get center_freq

Returns the center frequency of the current channel in 100kHz.

8. hgpriv hg0 get sta_list

View the currently connected sta information: aid, mac address, ps (sta sleep status, =1 means sleep), rssi, evm, tx_snr (rssi - bgrssi counted by the other device, fed back over the air), rx_snr (rssi - bgrssi counted by this device)

Execute this command on the AP side to obtain the connected STA information.

Execute this command on the STA side to view the information of the AP to which the STA is currently connected.

9. hgpriv hg0 get scan_list

After executing the scan command, you can use this command to obtain the scanned AP list

10. hgpriv hg0 get disassoc_reason

Reasons for connection failure:

- 0: Connection successful
- 1: Password or SSID error
- 17: The number of STAs connected to the AP is full
- 93: Insufficient AP Memory resources
- 255: AP not found

11. hgpriv hg0 get bgrssi=chan_index

Returns the bgrssi corresponding to the corresponding chan_index.

chan_index: The specified channel, starting from 1.

12. hgpriv hg0 get bssid (supported only by SDK V1.x)

In STA mode, query the connected AP MAC address and return the STA's AID.

Output format: MAC, AID

For example: 00:11:22:33:44:55,1

13. hgpriv hg0 get signal

Get the signal strength RSSI.

Notice: Only STA can use this interface; if AP uses this interface, the RSSI read will be inaccurate.

14. hgpriv hg0 get fwinfo

Get firmware information.

Notice: This command will print garbled characters when executed manually. The application uses the encapsulated API: `hgpriv_get_fwinfo` to obtain

Firmware information.

Manually execute `cat /proc/hgicf/status` to view the version information.

Network Advanced Parameters

1. hgpriv hg0 set txpower=xx

Set the maximum transmit power (in dBm, in 1dB increments), the value range is 1 ~ 20, if it is out of the range, it will return to the default value.

The recognized value is 20.

2. hgpriv hg0 set super_pwr=xx

Set whether to enable the super power function of the AH module.

After this function is enabled, the AH module will increase the transmission power during long-distance communication, up to 25dbm.

It is enabled by default in the AH mode and disabled by default in the AH test mode.

`hgpriv hg0 set super_pwr=0` # Disable super power

`hgpriv hg0 set super_pwr=1` # Enable super power, up to 25dbm (supports relatively low rates)

`hgpriv hg0 set super_pwr=2` # Enable super power, up to 22dbm (supports higher than 25dbm)

Higher speed, but closer distance)

3. hgpriv hg0 set tx_mcs=xx

Set the mcs of AH module TX. Valid values are: 0/1/2/3/4/5/6/7/10/255

Setting 0/1/2/3/4/5/6/7/10 indicates fixed MCS gear, among which 10 only supports 1M;

Setting 255 means automatic adjustment, and the AH module will enable the tx rate automatic adjustment function.

The default value is 255. Generally, you can keep the default value.

4. hgpriv hg0 set agg_cnt=xx

Set the maximum number of frame aggregations, the default is 16. Setting a smaller number of aggregations will reduce the average traffic, but setting more than 16 will

Make the traffic fluctuation larger. It is recommended to keep the default value.

5. hgpriv hg0 set max_txcnt=xx

Set the maximum number of retransmissions of the WiFi frame; 0 is invalid, N means a maximum of N times in total, the default is 7 times.

6. hgpriv hg0 set ap_hide=1

Setting the AP hiding function is only valid in AP mode. After setting the AP hiding function, STA will not be able to find the AP through scanning.

7. hgpriv hg0 set bss_max_idle=xx

Set the BSS max idle time in S.

sta must send 1 packet to the AP during the max idle time to maintain the connection with the AP.

If the sta message is received, it is considered that the sta is offline.

Note that this parameter will affect the sleep power consumption of sta. The smaller the setting, the greater the power consumption. The default value of 300 seconds corresponds to 200uA@DTIM10.

8. hgpriv hg0 set disassoc_sta=f8:de:09:96:8f:28

Disconnect the specified STA. AP can use this command to actively disconnect STA. However, when STA is in normal mode, disconnect

It will automatically reconnect after that.

9. hgpriv hg0 set unpair=f8:de:09:96:8f:28

Unpair the module with the specified mac. This command can be executed on both AP and STA.

10. hgpriv hg0 set conn_paironly=1/0

After setting conn_paironly, AP will only allow STAs in the pairing list to connect. STAs not in the pairing list will

Even if the correct SSID and password are set, it is still impossible to connect, so the AP restricts the connection of STA, which is similar to the whitelist.

- conn_paironly=1 : Only stas in the pairing list are allowed to establish connections.
- conn_paironly=0: Allow all STAs to initiate connections. Only the correct SSID and password are required to successfully connect.

The default value of this parameter is 0.

11. hgpriv hg0 set paired_stas=mac1,mac2,...

During the pairing process, the module will automatically generate STA information and form a STA list. If the module has flash, the STA list

It will be saved in flash, and the module will automatically load the STA list when it restarts. If the module is not mounted in flash, the STA list cannot be saved.

The module is lost after reboot.

For APs without flash, you can use set paired_stas to reset the pairing list after powering on.

paired_stas, and conn_paironly=1 is set at the same time, the effect is: only these STAs are allowed to connect to the AP,

Other STAs cannot connect even if they have the correct SSID and password. AP can limit STA connections, which is similar to the whitelist.

The maximum number of STAs that can be set with this command is limited by the maximum number of STAs supported by the firmware; the MAC addresses are in English.

For example:

```
iwpriv hg0 set paired_stas=00:11:22:33:44:55,00:12:34:56:78:99
```

The STA list set in this example contains 2 STAs, whose MAC addresses are: 00:11:22:33:44:55 and 00:12:34:56:78:99.

You need to set conn_paironly=1 first, then set paired_stas, and finally set the ap/sta mode.

12. hgpriv hg0 set pair_autostop=xx

Set whether the AH module automatically stops pairing after pairing is successful.

By default, the AH firmware will not stop automatically. You need to manually execute hgpriv hg0 set pairing=0 to stop pairing.

hgpriv hg0 set pair_autostop=1 # Enable pairing auto-stop

hgpriv hg0 set pair_autostop=0 # Disable pairing auto-stop

13. hgpriv hg0 set acktmo=xx

Set the value of the WiFi protocol parameter ack timeout of the AH module in microseconds. The default value is 0.

This parameter needs to be set only for 1km communication. The calculation formula is $10 \times (\text{distance in kilometers} - 1)$, for example, for 2km, set:

acktmo=10 #ack timeout value increases by 10us

14. hgpriv hg0 set channel=xx

Set the channel index of a fixed working frequency of an AH module. The value starts from 1.

Low power consumption related parameters

1. hgpriv hg0 set sleep=1

Control the WiFi module to enter sleep mode.

hgpriv hg0 set sleep=1 //Control the WiFi module to enter sleep mode

hgpriv hg0 set sleep=0 // Clear the sleep flag recorded by the WiFi driver. When entering sleep mode, the WiFi driver will

Record the current sleep state of WiFi, which will block all access to the WiFi module. You can continue to access after clearing this mark

WiFi module.

2. hgpriv hg0 set ps_mode=xx

Set the WiFi module sleep mode:

- 0: The sleep mode is not set, and the effect is the same as mode 3.
- 1: The module keeps alive with the server when entering sleep (the module keeps alive with the server itself, which requires secondary development).
- 2: When the module enters sleep mode, it keeps the server alive (AP replaces the module to keep the server alive, and the sleep power consumption is lower than mode 1

Low. Only the AH module supports this mode, and only supports UDP protocol).

- 3: When the module enters sleep mode, it only maintains a connection with the AP. Any unicast packet can wake up the module (sleep power consumption is different from mode 2).
- 4: The module enters sleep mode and keeps itself alive only with the AP. It can only be awakened by executing the wakeup command from the AP (the power consumption of sleep mode is similar to that of the module). Same as formula 2).

- 6: The module enters sleep mode and does not keep alive. It can only be awakened by pulling wakeup_io (not supported by firmware prior to 1.6.x).

3. hgpriv hg0 set ap_psmode=1

When using AP low power consumption, both the AP and STA should set this parameter to help the STA quickly connect to the AP when it restarts.

Note that when using AP low power consumption, this parameter needs to be set to 1 on both the AP and STA.

4. hgpriv hg0 set wakeup_io=1,0

Set the wake-up Pin for the host to wake up the AH-TXW8301, and the trigger edge: 0: rising edge (positive pulse), 1: falling edge (negative pulse

rush).By default, MCLR is used as the wake-up pin. Note that MCLR can only be woken up by a negative pulse with a width of 500uS.

If you need to use a positive pulse to wake up, or the wake-up pulse width is difficult to control to 500uS, you need to set other idle IO as the wake-up Pin.

Note that other IOs except MCLR can only be awakened by positive pulses, not negative pulses. In addition, use IOB as the wake-up pin.

Compared with using MCLR and IOA, it will consume an additional 20uA of sleep power.

The corresponding serial numbers of IO are: IOA0~31: 0~31, IOB0~7: 32~39, MCLR: 51.

Example:

```
hgpriv hg0 set wakeup_io=34,0
```

Set the wake-up IO to IOB2 and positive pulse to wake up.

5. hgpriv hg0 set wkio_mode=xx

Set the working mode of AH wakeup Host IO (IOB0). Default is pulse mode.

- 1: Level mode, the WiFi maintains a high level when it is operating normally and a low level when it is sleeping.
- 0: Pulse mode, when the WiFi module wakes up the main control, it pulls up a 2ms pulse signal.

6. hgpriv hg0 set wkhost_reason=2,7,8,12,14

Set which wake-up reasons need to wake up the master control.

For the explanation of wake-up reasons, refer to the following [wake reason](#).

7. hgpriv hg0 get wkreason

Query the reason why the WiFi module is awakened (the query is valid in the connected state). Return value meaning:

STA wake-up reason:

- 1: Timer wakeup in non-connected state (this reason generally does not need to set the wakeup master)
- 2: Unicast TIM wake-up, active wake-up by the AP or unicast wake-up by other devices (usually requires setting the wake-up master)
- 3: Broadcast TIM wake-up, wake-up by broadcast data (this reason has been deleted)
- 4: Button IO wake-up (MCLR pin by default) (If it is pulled by the master, you don't need to wake up the master)
- 5: Beacon lost wakeup (this reason generally does not require setting the wakeup master)
- 6: AP restart detection wakeup (this reason generally does not need to set the wakeup master)
- 7: Heartbeat timeout wake-up (used in sleep mode 2)
- 8: Wake-up packet wake-up (used in sleep mode 2)
- 9: MCLR IO reset wakeup (MCLR is pulled for more than 2ms in sleep state, causing reset. This reason generally does not need to be set.

Wake up the master control)

- 10: LVD low power reset (this reason does not wake up the main control by default)
- 11: PIR IO wake-up (requires special hardware circuit support, otherwise no need to set)
- 12: AP API wakeup, the AP executes wnb_wakeup_sta or hgic_iwpriv_wakeup_sta (usually
Need to set wake-up)
- 13: During STA sleep, AP detects that STA is offline (this reason generally does not need to set the master to wake up)
- 14: Wake up when connected to AP in STANDBY state (only need to be set after using the STANDBY function)
- 16: Low power wake-up (needs customer circuit support, otherwise no need to set)
- 17~19 can be used by customers for expansion;

AP wake-up reason:

- 20: AP wakes up due to failure to enter sleep mode
- 21: STA disconnection causes AP wakeup
- 22: AP wakes up due to receiving STA data
- 23: Pairing wake-up

8. hgpriv hg0 set dcdc13=xx

Set whether the AH module uses an external 1.3V DCDC power supply or an internal LDO power supply.

hgpriv hg0 set dcdc13=0 #Using internal LDO, no external DCDC power supply is required, normal power consumption and sleep power consumption are

Higher, **Use this when you don't need to save power**; Usually the firmware defaults, and you need to configure dcdc13 through the interface when you need to save power

For some of the following patterns;

hgpriv hg0 set dcdc13=1 #Use external DCDC (hardware circuit must have 1.3V DCDC) to VDD13A and

VDD13D power supply, low power consumption in normal and sleep mode (about 40% lower than LDO power supply); **This is the default recommended setting for power saving.**

hgpriv hg0 set dcdc13=2 #sleep when using external DCDC (hardware circuit must have 1.3V DCDC)

VDD13A and VDD13D power supply, Low sleep power consumption (same as dcdc13=1) ; Use internal LDO to provide VDD13A during normal operation

The power supply is VDD13D (the software increases the output voltage of LDO), and the normal power consumption is high, which is equivalent to LDO power supply.

The power consumption is high, so this mode is not recommended.

`hgpriv hg0 set dcdc13=3` #Use external DCDC (hardware circuit must have 1.3V DCDC) to supply VDD13D

In normal mode, VDD13A uses internal LDO for power supply, while VDD13D still uses internal LDO for power supply.

When using external DCDC power supply, the power consumption is lower than that of mode 2, but higher than that of mode 1 (reduced by about 20% relative to LDO power supply);

It can not only ensure RF performance, but also save power consumption to a certain extent. If it is found that the DCDC used causes poor RF EVM, you can consider adopting this option.

This parameter must be set according to the actual hardware circuit. Otherwise, if it is set to a non-zero value and there is no external DCDC,

The device cannot be turned on now; if it is set to 0 but there is an external DCDC, it may not save power.

9. `hgpriv hg0 set pa_pwrctl_dis=xx`

Set the PA power supply control logic switch when the module is in sleep mode.

`hgpriv hg0 set pa_pwrctl_dis=0` (default value), PA power supply control logic is not turned off, PA does not supply power during sleep;

It needs to be used with hardware, that is, the IOA30 is added to control the peripheral circuit of RF power down in sleep mode;

`hgpriv hg0 set pa_pwrctl_dis=1`, turn off the PA power supply control logic, PA always supplies power;

In fact, the default value of this parameter is 1.

10. `hgpriv hg0 set dtim_period=xx`

Set the dtim period of the WiFi module in sleep mode in milliseconds. WiFi will be called up regularly at the specified dtim period.

Check the connection with AP and receive AP wake-up. Dtim is set on the AP side.

The setting value for DITM10 is 1000 (default value), the setting value for DITM20 is 2000, and so on.

This parameter affects the sleep power consumption and wake-up time. The larger the DTIM value, the lower the sleep power consumption and the longer the wake-up time.

The smaller the value, the greater the SLEEP power consumption and the shorter the wake-up time.

If you want to use a value less than 1000, that is, the keep-alive period is less than 1S, in addition to setting dtim_period, you also need to set

beacon_int is required. Please consult FAE for details.

11. hgpriv hg0 set autosleep_time=xx

Set the module auto sleep time in seconds, the default is 10 seconds, the maximum value is 32 seconds; set to -1 to turn it off

auto sleep. Setting 0 also means the default value of 10 seconds.

Auto sleep means that if the module does not receive a command from the master control within a specified time after it is powered on, it will assume that the master control is not powered on and automatically enter sleep mode.

sleep.

Example:

```
hgpriv hg0 set autosleep_time=20
```

Set the auto sleep time to 20 seconds.

12. hgpriv hg0 set wait_psmode=0/1

Set the sleep mode in non-connected state. The default value is 0 for ps connect and 1 for standby mode.

Just set STA.

13. hgpriv hg0 set ps_connect=60,4

When the STA's WiFi module is disconnected in sleep mode, it will wake up and reconnect to the AP. If the connection fails, the WiFi module will

Enter PS Connect mode: cyclic sleep/wakeup/reconnection. The sleep in the middle is to prevent the power consumption from being too high due to constant reconnection.

The default PS Connect behavior is: sleep interval is 1 minute, and after n failures, it will increase by sleep n*1 minutes.

- The first connection failed, sleep for 1 minute
- The second connection failed, sleep for 2 minutes
- The third connection failure sleeps for 3 minutes
- 4th connection failure sleep 4 minutes

After the sleep time increases 4 times, it wraps back to the first interval and repeats regularly.

Example:

hgpriv hg0 set ps_connect=30,4

Set the sleep interval of ps connect to 30 seconds, with a maximum increment of 4 times.

14. hgpriv hg0 set dis_psconnect=1

If you do not want the STA to automatically enter sleep mode when not connected to an AP, you can turn off the PS Connect mode.

In this scenario, you hope that the device will not go into sleep mode when it is not connected.

Since the dormant STA will not report information to the master after disconnecting, it is recommended that low-power devices do not

Turn off PS Connect mode, otherwise if you are disconnected, you will have to keep reconnecting to the AP. If you can't connect, the battery will be consumed quickly.

15. hgpriv hg0 set standby=chn_idx,wakeuptime

Set the frequency (starting from 1) and wake-up interval (in ms) in standby mode.

Just set STA.

16. hgpriv hg0 set aplast_time=xx

Set the time (in seconds) to detect AP loss when the module is in sleep mode. If the module does not receive the AP beacon within the specified time,

The AP is considered lost. The default value is 10 seconds.

17. hgpriv hg0 set wkdata_save=1

Set whether the module saves the wake-up data sent by the server.

After the wakeup data is saved, the module cache wakeup data can be read through the /proc/hgic/wkdata_buff interface after the main control is turned on.

The module can cache up to 4 wake-up data.

18. hgpriv hg0 set heartbeat=ip_str,port,hb_interval,hb_tmo

Set the heartbeat server's IP address/port number/heartbeat cycle/heartbeat timeout.

Example: `hgpriv hg0 set heartbeat=192.168.1.1,6000,60,300`

Set the heartbeat server IP address to 192.168.1.1, port number to 6000, heartbeat period to 60 seconds, and heartbeat timeout to 300 seconds.

This command can be used to set parameters only when `ps_mode=2` is set. Before entering low power SLEEP

Before sending a heartbeat packet, the heartbeat application must send at least one heartbeat packet to the server. The WiFi module will capture the heartbeat packet content during the sending process.

After entering SLEEP mode, the WiFi module automatically sends a heartbeat packet to the server to keep it alive.

19. `hgpriv hg0 set heartbeat_resp`

Set the content of the server heartbeat response. Used for WiFi module response matching to determine whether it has been received.

The server's response to determine if the heartbeat is lost.

This command can only be used using the API: `hgic_iwpriv_set_heartbeat_resp_data` to set it up.

This parameter needs to be set only when `ps_mode=2`.

20. `hgpriv hg0 set wakeup_data`

Set the wake-up packet content when the server wakes up the device. Used by the WiFi module to match the wake-up packet content and determine whether it needs to be woken up.

Awake.

This command can only be used using the API: `hgic_iwpriv_set_wakeup_data` to set it up.

This parameter needs to be set only when `ps_mode=2`.

21. `hgpriv hg0 set wkdata_mask`

Set the wakeup packet content matching mask when the server wakes up the device.

This command can only be used using the API: `hgic_iwpriv_set_wkdata_mask` to set it up.

```
int hgic_iwpriv_set_wkdata_mask(char *ifname, int offset/*from IP hdr*/, char
* mask, int mask_len)
```

Mask is a bitmap, supporting up to 8 bytes, and the length of the wake-up data that can be matched is 64 bytes.

shift.

22. hgpriv hg0 set wakeup=mac_addr

Execute the wakeup command on the master to wake up the sta with the specified mac address. For example:

```
-hgpriv hg0 set wakeup=00:11:22:33:44:55
```

If sta does not exist or is not in the dormant state, the return value is -1.

23. hgpriv hg0 set reassoc_wkhost=1

When the module detects an AP abnormality in sleep mode, it will restart scanning to connect to the AP. By default, the module will not notify the main control that a reconnection has occurred.

AP situation. Set reassoc_wkhost=1, and the module will notify the main control after reconnecting to AP.

Multicast Mode Parameters(Only supported by SDK V1.x)

1.hgpriv hg0 set join_group=group_addr,aid

After setting the working mode of the WiFi module to group, you can use this command to set the WiFi module to join a multicast network.

After joining the multicast network, the WiFi module will only receive data in the multicast network. All data communications are carried out using the multicast address.

line of communication.

If the working mode is set to group, but no multicast network is added, all data communications are carried out in the form of broadcast

Send and receive.

Note that the JOINGROUP command can only be set after the GROUP mode is set.

Parameter Description:

- group_addr: The address of the multicast network to be joined.
- aid: The AID of the device in the multicast network. AID valid values: 1~N (N is the maximum value supported by the firmware).

The AID of each device in the network should be unique.

- Set valid AID: The WiFi module will periodically send heartbeats in the multicast network to announce to other WiFi modules

Own existence.

- Setting invalid AID: The WiFi module will not send heartbeats and will not notify other WiFi modules.

If AID is set to 0, the maximum number of STAs supported by the firmware will not be limited.

Example:

```
hgpriv hg0 set join_group=11:22:33:44:55:66,3
```

2.hgpriv hg0 set mcast_txparam=dupcnt,tx_bw,tx_mcs,clearch

Set multicast communication TX parameters

- dupcnt: The number of times multicast data is resent is n. By default, it is sent only once. After setting this parameter, each multicast data will be resent.

Repeat n times.

- tx_bw: Set the multicast data tx bandwidth:
- tx_mcs: Set multicast data mcs.
- Clearch: Set whether to clear the channel before sending multicast data. This parameter is currently invalid and can be set to 0.

Relay parameters(Note that the definitions of SDK V1.x and V2.x are different)

1. hgpriv hg0 set r_ssid=xx

V2.x: The ssid parameter is used to connect to the upper-level AP, and the r_ssid parameter is used as the SSID of the relay hotspot.

V1.x: The ssid parameter is used as the ssid of the relay hotspot, and r_ssid is used to connect to the upper-level AP.

2. hgpriv hg0 set r_psk=64_hexchar

The key value is 64 hex characters and its usage requirements are the same as wpa_psk parameters.

V2.x: The psk parameter is used to connect to the upper-level AP, and the r_psk parameter is used as the password for the relay hotspot.

V1.x: The psk parameter is used as the password for the relay hotspot, and r_psk is used to connect to the upper-level AP.

Roaming parameters (note that SDK V2.x standard protocol does not support)

1. hgpriv hg0 set roaming=onoff,0,threshold,rssi_diff,rssi_int

Enable or disable the roaming function. This command contains 5 parameters:

Parameter 1 onoff: The value is 0 (off) or 1 (on), indicating whether the roaming function is enabled. The default value is off.

Parameter 2: The value is 0. Deprecated .

Parameter 3 threshold: Set the signal strength threshold for roaming switching. The default value is -60, which means the signal strength of the AP is lower than -60dbm.

When the STA starts to look for an AP with a stronger signal, it can generally be increased to -50.

Parameter 4 rssi_diff: roaming detection signal difference. The AP is considered suitable for switching only when the difference reaches this value. The default difference is 12db.

That is to say, the newly discovered AP signal strength is considered to need to be switched only when it is 12db stronger than the current AP signal.

Parameter 5 rssi_interval: roaming rssi monitoring period, the default is 5, which means 5 beacon periods.

Switching speed.

STA needs to set this parameter, but AP does not need to set it.

Dual antenna parameters

1. hgpriv hg0 set ant_auto=1

Use dual antennas to automatically select mode.

2. hgpriv hg0 set ant_sel=0/1

Manually select Antenna 0 or Antenna 1.

When selecting manually, you need to turn off the automatic selection mode.

Other parameters

1. hgpriv hg0 set auto_chswitch=xx

Set the module's automatic frequency hopping function to be on or off (on by default).

auto_chswitch=1 Enable

auto_chswitch=0 Disable

The automatic frequency hopping function means that when the AP detects interference in the current channel during operation, it will automatically hop to another relatively clean channel.

When the AP is frequency hopping, it will notify the STA to switch the frequency together, but it cannot notify the STA in the sleep state to switch the frequency.

After hopping to another frequency, the STA in sleep will detect AP timeout and restart to scan and connect to AP again.

Go to sleep again.

2. hgpriv hg0 set auto_save=xx

Set whether the AH module automatically saves parameters (when the AH module is configured with nor flash).

After the automatic save function is turned off, only hgpriv hg0 save will save the parameters (the parameter values of this command are saved immediately).

hgpriv hg0 set auto_save=0 # Disable automatic saving, need to be used with the following save command

hgpriv hg0 set auto_save=1 # Enable autosave (default)

3. hgpriv hg0 save

This command has no parameters. Its function is to set the AH module to save parameters (when the AH module is configured with nor flash).

The AH firmware will automatically save parameters by default; when modifying parameters, it will automatically save them to flash if a parameter change is detected.

If auto_save is set to 0, then call this save function when you need to save the parameters.

4. hgpriv hg0 set dhcpc=1

Enable the DHCP Client function inside the module. The module will obtain an IP address in advance after it is powered on, and can be used directly after the main control is powered on.

The module obtains the IP address, saving the time of the main control to apply for the IP address.

5. hgpriv hg0 set reset_sta=mac_addr

Resets the remote AH module of the specified MAC address.

6. hgpriv hg0 set radio_onoff=x

Used to control the wifi radio on/off.

hgpriv hg0 set radio_onoff=0 Turn off the Wi-Fi radio.

hgpriv hg0 set radio_onoff=1 Turn on the Wi-Fi radio.

hgpriv hg0 set radio_onoff=2 Turn on wifi radio RX and turn off TX

3.1.4. Proc fs interface

The hgic_fmac driver provides the proc fs interface, through which applications can interact with the driver.

The tools/test_app directory provides iwpriv.c, which provides a packaging API that can be integrated into the application.

Use the API directly to interact with the driver.

/proc/hgicf/status (read-only)

cat /proc/hgicf/status can check the driver running status, including firmware information and driver data cache information.

This interface returns version information in string form, where the format of fw info is as follows:

fw info:2.4.1.5, svn version:35402

The meaning of each field:

- 2: Major version number
- 4: Minor version number
- 1: Patch version number
- 5: fmac project type

-35402: Patch code version number (this version number is incremented and is unique)

/proc/hgicf/ota (write-only)

Module firmware OTA function interface, through which the firmware in the module can be upgraded. If the module does not have Flash,

If the firmware is downloaded and run, this interface is invalid.

Directions:

1. Put the firmware.bin in the /lib/firmware directory;
2. echo -n firmware.bin > /proc/hgicf/ota, then start the OTA upgrade function (you can also execute API:

hgic_proc_ota), the whole process may take 24s (tested with a firmware size of 320K).

If the module has a Flash but it is blank, you can also use the OTA function to burn the firmware: first download the firmware through the interface and run

Run it, and then use OTA to burn the flash.

/proc/hgicf/fwevnt (read-only)

Driver event reading interface. By cyclically reading this interface, you can receive messages from the driver and firmware.

The demo code is provided in the tools/test_app/hgicf.c file.

/proc/hgicf/iwpriv (read-only)

Driver parameter setting interface, which is used by hgpriv.c and iwpriv.c in the tools/test_app directory.

tools/test_app/iwpriv.c provides a driver command encapsulation API. Applications can integrate this file and directly use the API

The calling method takes parameters.

tools/test_app/hgpriv.c is a manual command tool that can be used to manually execute driver parameter commands.

3.1.5. Driver event messages

The WiFi module will generate various event notifications to the main control during operation. By reading the /proc/hgicf/fwevnt interface,

To receive event messages generated by the firmware. tools/test_app/hgicf.c provides demo code.

Refer to hgicf.c.

Common event descriptions are as follows:

- HGIC_EVENT_SCANNING = 5

This event is generated when the WiFi module enters the scanning state.

- HGIC_EVENT_SCAN_DONE = 6

This event is generated when the WiFi module scan is completed.

- HGIC_EVENT_TX_BITRATE = 7

During operation, the WiFi module will evaluate the maximum transmission capacity based on the current wireless transmission and reception conditions, and generate a

The application can make some adjustments based on the information fed back by this event, such as adjusting the video bit rate.

- HGIC_EVENT_PAIR_START = 8

This event is generated when the WiFi module enters the configuration state

- HGIC_EVENT_PAIR_SUCCESS = 9

This event is generated when the WiFi module is paired successfully, and will continue to be generated until pairing stops.

This event will report the current paired sta mac address.

- HGIC_EVENT_PAIR_DONE = 10

This event is generated when the WiFi module stops pairing. In this event, all currently paired sta mac lists are reported.

surface.

- HGIC_EVENT_CONNECT_START = 11

This event is generated when the WiFi module starts connecting.

- HGIC_EVENT_CONNECTED = 12

This event is generated when the WiFi module is successfully connected.

- HGIC_EVENT_DISCONNECTED = 13

This event is generated when the WiFi module is disconnected.

- HGIC_EVENT_SIGNAL = 14

This event is generated when the wireless signal of the WiFi module changes. The event will report the current rssi and evm information.

- HGIC_EVENT_REQUEST_PARAM = 16

This event is generated when WiFi as STA cannot connect to AP. The application can reset the parameters.

- HGIC_EVENT_CUSTOMER_MGMT = 19,

This event is generated when the WiFi module receives a custom management frame. This event reports the content of the custom management frame.

- HGIC_EVENT_DHCPC_DONE = 21

This event is generated when the WiFi module successfully executes DHCP to request an IP address.

- HGIC_EVENT_CONNECT_FAIL = 22

This event is generated when the WiFi module fails to connect. The event will report the reason for the connection failure.

- HGIC_EVENT_CUST_DRIVER_DATA = 23

This event is generated when the WiFi module sends custom driver data to the main control.

- HGIC_EVENT_UNPAIR_STA = 24

This event is generated when the WiFi module receives the unpairing request from the other party.

- HGIC_EVENT_EXCEPTION_INFO = 27

The WiFi module reports module abnormality information.

3.1.6. One-click pairing

The Tysin FMAC driver supports one-key pairing networking, simplifying the AP and STA networking parameter settings.

When starting pairing, the AP usually needs to set parameters such as SSID/password first. If no password is set, when starting pairing

The AP will automatically generate a random password for each STA.

During the network configuration process, the STA will obtain the SSID and password information from the AP, and the AP and STA will record each other's MAC address, and establish a paired STA list.

When the module has flash, the above information will be automatically saved to the flash.

When the module has no flash, the above information will be lost after restart. The SSID/password/MAC address can be read by the master control when pairing is completed.

The address and other information are saved.

illustrate:

1. After one-touch pairing is completed, you need to execute `set pairing=0` to stop pairing before entering the connection state.
2. One-touch pairing only supports pairing of 2 devices at the same time. If you need to pair multiple devices, please connect the STA devices in sequence.

For example: when 1 to 4 network, please pair 4 STA devices with AP in turn.

The device is paired with the AP at the same time.

3. When the number of paired STAs reaches the maximum value, the AP will choose to overwrite the disconnected STA information.

If the STA is covered, pairing will fail.

This can be controlled using the `hgpriv` command or the API: `hgic_iwpriv_set_pairing`:

`hgpriv hg0 set pairing=1` #Start pairing. The other device involved in pairing also needs to enter pairing mode.

`hgpriv hg0 set pairing=0` #Stop pairing and exit pairing mode. If pairing is successful, the WiFi module will automatically establish a connection.

During the pairing process, the firmware will generate some events, `hgic.c` will read the events, and the application can process the events.

For example, when pairing is successful, the SSID, password and other information are read and saved on the main control end.

After pairing is successful, the system will not automatically exit the pairing state by default. If you want to automatically exit the pairing state after success, you can configure

`pair_autostop` (see `hgpriv` advanced networking parameters);

If pairing is unsuccessful, it will not time out automatically. You need to manually execute `hgpriv hg0 set pairing=0` to stop pairing.

3.1.7. STA low power consumption process

The Taixin AH module supports low-power SLEEP mode. After entering SLEEP, the AH module will close the connection with the Host controller.

If the Host tries to access the AH module again, the access will fail.

There are 2 ways to use SLEEP mode:

1. In SLEEP mode, only keep alive with AP

When the SLEEP device does not need to keep alive with the remote server, it only needs to keep alive with the AP.

The SLEEP process is relatively simple, follow the steps below:

1) Enter low power consumption: hgpriv hg0 set sleep=1 or execute API: hgic_iwpriv_sleep

2) Wireless wake-up: Execute the API hgic_iwpriv_wakeup_sta on the AP side to wake up the STA.

3) Local wake-up: The master (or button) pulls down the MCLR pin of the AH module to wake up the AH module.

The pulse should not be less than 100uS and not greater than 1mS. The recommended value is 500uS.

4) WiFi module wakes up the main control: Use IOB0 to wake up the main control. IOB0 has two working modes, the default is pulse mode,

When waking up the main control, IOB0 is pulled high for 2ms to generate a pulse signal. Use the hgpriv wkio_mode parameter command to modify IOB0

working mode.

2. Keep alive with remote server during SLEEP

When the SLEEP device needs to keep alive with the remote server, the AH module will communicate with the remote server in the SLEEP state.

The server performs heartbeat keep-alive. Please follow the steps below:

1) Set the heartbeat server's IP address, port number, heartbeat packet period, and heartbeat timeout

Set the IP address and port number of the heartbeat server to the AH module. The AH module will automatically capture the heartbeat during the communication process.

After entering SLEEP, the AH module will replace the device to communicate with the heartbeat server for heartbeat keepalive.

It will be awakened and the main control will be awakened through IOB0.

Use the API: hgic_iwpriv_set_heartbeat to set it. The example is as follows:

```
hgic_iwpriv_set_heartbeat("hg0", inet_addr("192.168.0.1"), 60002, 60, 300);
```

The heartbeat server IP is 192.168.0.1, the port number is 60002, the heartbeat packet period is 60 seconds, and the heartbeat timeout is

300 seconds.

2) Set the heartbeat response content sent by the heartbeat server

Set the heartbeat response content to the AH module. In SLEEP mode, the AH module communicates with the heartbeat server.

When a heartbeat interaction occurs, the AH module will identify the heartbeat response based on the data packet content.

If no response is received after sending the heartbeat, the AH module will continue to send heartbeats to the server the next time it wakes up.

If no response is received for 7 consecutive heartbeat packets, it is considered disconnected from the server and the master control will be notified of the network abnormality through IO.

Use API: hgic_iwpriv_set_heartbeat_resp_data to set the heartbeat response content.

3) Set the content of the wake-up packet sent by the heartbeat server

Set the wake-up packet content of the heartbeat server to the AH module. After the AH module recognizes the wake-up packet, it will call the

The main control is woken up, and the AH module itself is also woken up.

Use API: `hgic_iwpriv_set_wakeup_data` to set the wakeup packet content.

4) Before the device enters SLEEP mode, it ensures that it exchanges heartbeat information with the heartbeat server at least once.

Since the AH module captures the heartbeat packet during the communication process, it must send at least one heartbeat packet before entering SLEEP.

The AH module can capture the heartbeat packet. If the AH module fails to capture the heartbeat packet, the AH module will not

Keep alive with remote server.

5) Remote Wake-up

The remote server sends a wake-up packet to the device, and the WiFi module wakes up the device after receiving and identifying it.

6) Local wake-up

The WiFi module can be woken up by pulling down the MCLR pin of the WiFi module by the master control (or button).

Less than 100uS, not greater than 1mS, recommended value is 500uS.

7) WiFi module wakes up the main control

The WiFi module uses IOB0 to wake up the main control. IOB0 is in pulse mode by default. When the WiFi module wakes up the main control, a pulse will be generated.

A 2ms high level pulse signal. The working mode of IOB0 can be modified by `hgpriv wkio_mode` parameter.

After the machine is turned on, you can query the wakeup reason through the API: `hgic_iwpriv_get_wkreason`.

3. Exception handling process

1) AP abnormal restart:

When the WiFi module in Sleep state finds that the AP restarts abnormally, the WiFi module will restart and communicate with the AP again.

After the connection is successful, it will automatically enter the sleep state for 10 seconds. During this process, the main control will not be notified to wake up.

Sleeping is the action that the module takes when no interactive command from the master is detected. The module thinks that the master is in sleep mode.

The sleep time can be modified via `iwpriv hg0 set autosleep_time`.

2) AP shutdown:

After disconnection in sleep mode, the WiFi module will restart and reconnect to the AP. If the connection fails, the WiFi module will

Enter PS Connect mode or standby mode: cycle sleep/wake up/reconnect.

By setting [wait_psmode](#) To choose whether to enter PS Connect mode (wait_psmode=0, default value)

Still in Standby mode (wait_psmode=1).

- PS Connect Mode: The default PS Connect behavior is: sleep interval is 1 minute, and the number of failed attempts increases.

Increase sleep time by n*1 minutes

- The first connection failed, sleep for 1 minute
- The second connection failed, sleep for 2 minutes
- The third connection failure sleeps for 3 minutes
- 4th connection failure sleep 4 minutes

After the sleep time increases by 4 times, it wraps back to the first interval and repeats this cycle to avoid frequent

Reconnection increases power consumption.

After the connection is successful, it will automatically enter the sleep state 10 seconds later, and the main control will not be notified to wake up during this process.

This parameter can be modified through the iwpriv ps_connect command.

PS Connect mode can be turned off using the iwpriv dis_psconnect command.

Note that during the PS connect process, the module will restart cyclically, so the power consumption is relatively high.

- Standby mode: A new unconnected sleep mode. Compared to PS Connect, Standby mode allows STA

Get lower power consumption when not connected and connect to the AP faster after the AP is powered on.

STA needs to set [Standby Parameters](#) : The sleep channel index (numbered from 1), and the wake-up interval (ms as units).

The following is the corresponding relationship between wake-up time and standby sleep current:

Wake-up time (S)	1	3	5	10
Standby sleep current (uA)	500	300	250	200

The AP needs to specify the same sleep channel as the STA after powering on (through [set channel](#) to specify).

You can consider setting the standby channel to the channel that the AP used before shutting down (through [get center freq](#)).

3) Network disconnection:

If the WiFi module or AP in Sleep state finds that the heartbeat timeout occurs, it means that the network is abnormal.

The WiFi module will restart and wake up the main control. After the main control is turned on, try to connect to the server again to confirm whether the network is connected.

The control is handed over to the master controller, who decides the subsequent process.

3.1.8. AP low power consumption process

The Taixin AH module supports AP low power mode. The host sets the AH module to enter AP low power mode.

After entering the mode, the master controller can choose to enter standby mode or power off to reduce overall power consumption.

In AP low power mode, the AH module does not need to be awakened by IO, but can be awakened by reinitializing the interface.

When you need to use AP low power consumption, you need to set `hgpriv hg0 set ap_psmode=1` for both AP and STA; and

Set `PS_MODE=4`.

General process:

Enter AP low power operation steps:

1. Turn off data communication: Before entering AP low power consumption, please turn off all data communications.
2. Enter low power consumption: Execute API: `hgic_iwpriv_sleep("hg0", 1)` to notify the AH module to enter low power consumption mode.

There are two situations that trigger the exit of AP low power mode:

1. **Master trigger:** The master controller sends a command through the interface to wake up the module;

2. **STA trigger:** When the STA sends data to the AP, it will trigger the AP to exit the low power mode. Or use

API: `hgic_iwpriv_wakeup_sta` interface wakes up the AP. After the AP receives the wake-up packet, the AH module has not yet been actually awakened.

Use IOB0 to wake up the master, and then the master fully wakes up the module through the interface. IOB0 uses the pulse mode by default to wake up the master

When IOB0 is pulled high for 2ms, a pulse signal is generated. The `hgpriv wkio_mode` command can be used to modify the working mode of IOB0.

Please refer to the instructions for using this command. Note that the module has not yet fully awakened, and the sdio interface needs to be initialized after the main control is turned on.

After the main control is turned on, you can query the wake-up reason through the API: `hgic_iwpriv_get_wkreason`.

For detailed description of wake-up reasons, please refer to [get wkreason](#) illustrate.

Exception handling:

1. After the AP enters low power consumption mode, the STA loses connection

In AP low power mode, the AP cannot detect the disconnected connection status of the STA in time.

2. AP fails to enter low power mode

When the AP enters low power mode, it will notify all connected STAs. If the STA loses power or loses signal at this time,

If the AP cannot notify the STA, the AP will enter low power mode and fail. After the failure, the main control will be awakened, and the main control will query after being awakened.

The wake-up reason can notify the AH module again to force it into low power mode.

3. It is not currently supported for AP and STA to enter low-power sleep mode at the same time

3.1.9. Relay Function Instructions

Refer to the relevant interface setting instructions of the relay.

3.1.10. Instructions for using the roaming function (note that the standard protocol of SDK V2.x is not supported)

AH-STA mode supports roaming between multiple APs.

How to use the roaming function:

1. Enable roaming: STA use [hqpri v hq0 set roaming](#) Enable roaming. AP does not need to enable roaming.

2.roamingAP parameter settings:

1) Set the same SSID for each AP, or the last 3 digits of the SSID are different, for example: roaming_ssid_001, roaming_ssid_002.

2) Set the same password for each AP

3.**roaming**If APs need to communicate, they need to be interconnected using a wired network.

3.1.11. Driver auxiliary module

In order to simplify the use of fmac drivers in applications, the fmac driver package provides driver auxiliary modules:

tools/test_app/iwpriv.c. This file implements the encapsulation of the hgpriv command and proc interface of the fmac driver, and implements

A series of encapsulated APIs can be directly integrated into applications. Applications access fmac drivers in the form of APIs, which can simplify application

These wrapper APIs implement the parsing of the output information of the hgpriv/proc interface.

-hgpriv command wrapper API

The hgpriv command encapsulation APIs are shown below. The usage of these APIs is consistent with the corresponding hgpriv commands.

But there are 2 differences:

- Added new ifname parameter: This parameter is the AH network interface name, such as "hg0"
- The mac parameter is a 6-byte char array, while the mac parameter when using the hgpriv command is a MAC address string.

3.1.12. Firmware abnormal information

This section introduces the exception debugging information in EVENT, which can be read through /proc/hgic/fwevnt.

hgicf.c contains the example code:

```
break;
case HGIC_EVENT_EXCEPTION_INFO:
    exp = (struct hgic_exception_info *)data;
    switch(exp->num){
        case HGIC_EXCEPTION_TX_BLOCKED:
            printf("wireless tx blocked, maybe need reset wifi module*\r\n");
            break;
        case HGIC_EXCEPTION_TXDELAY_TOOLONG:
            printf("wireless txdelay too long, %d:%d:%d *\r\n",
                exp->info.txdelay.max, exp->info.txdelay.min, exp->info.txdelay.avg);
            break;
        case HGIC_EXCEPTION_STRONG_BGRSSI:
            printf("detect strong background noise. %d:%d:%d *\r\n",
                exp->info.bgrssi.max, exp->info.bgrssi.min, exp->info.bgrssi.avg);
            break;
        case HGIC_EXCEPTION_TEMPERATURE_OVERTOP:
            printf("chip temperature too overtop: %d *\r\n", exp->info.temperature.temp);
            break;
        case HGIC_EXCEPTION_WRONG_PASSWORD:
            printf("password maybe is wrong *\r\n");
            break;
    }
    break;
```

TXDELAY_TOOLONG

- Print: *wireless txdelay too long, max : min : avg *
- Description: tx delay is too long, reporting when it exceeds 500ms

STRONG_BGRSSI

- Print: *detect strong background noise. max : min : avg *
- Note: Background noise is poor, exceeding -85dbm;

TEMPERATURE_OVERTOP

- Print: chip temperature too overtop
- Description: The temperature is overheated, exceeding 85 degrees Celsius;

WRONG_PASSWORD

- Print: password maybe is wrong
- Description: The other party's encryption error caused the connection to be disconnected, and the main control had an error prompt. It is necessary to check the encryption-related parameters;

TX_BLOCKED

- Print: wireless tx blocked
- Description: The send queue is blocked;

3.1.13. Interface test mode

hgic_fmac provides a sdio/usb interface test mode, which can test the stability of the sdio/usb interface.

Specifying the if_test parameter when building hgicf.ko can start the interface test.

- insmod hgicf.ko if_test=1 #Start the one-way test of the interface, and the test data is sent from the host to the ah module
- insmod hgicf.ko if_test=2 #Start the interface bidirectional test and test the bidirectional data transmission.

3.2. Firmware Download

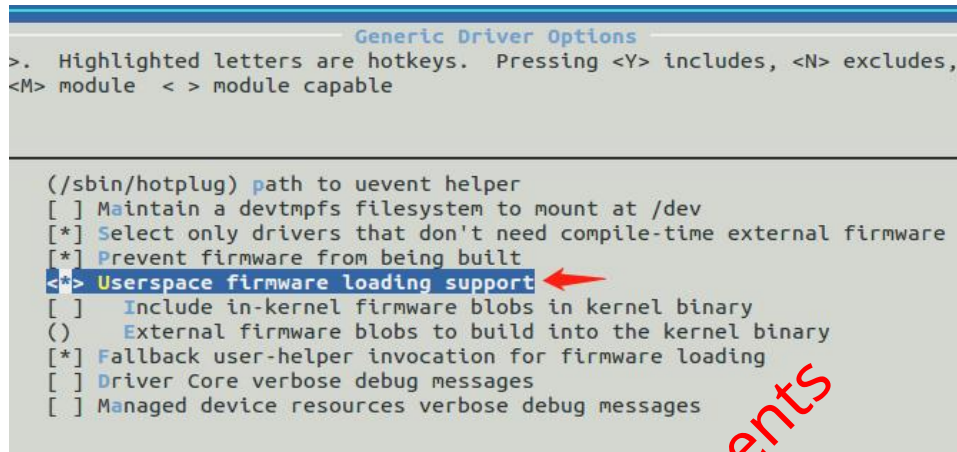
The WiFi module supports downloading firmware through the SDIO/USB interface. The firmware download function is related to the Linux system.

The usage of the Linux kernel may be different. The WiFi firmware should be placed in the firmware loading directory supported by the system, usually

It is /lib/firmware. If you encounter problems loading the firmware, please check the following aspects:

1. Kernel compilation configuration

The kernel needs to support CONFIG_FW_LOADER (Devices Drivers->Generic Driver Options)



After modifying this option, the WiFi driver needs to be recompiled.

2. View the firmware directory supported by the kernel

Look at the fw_path array in the firmware_class.c file.

```
9:
10: /* direct firmware loading support */
11: static char fw_path_para[256];
12: static const char * const fw_path[] = {
13:     fw_path_para,
14:     "/lib/firmware/updates/" UTS_RELEASE,
15:     "/lib/firmware/updates",
16:     "/lib/firmware/" UTS_RELEASE,
17:     "/lib/firmware"
18: };
19:
```

Note:

- 1) Some firmware_class.c files in older kernel versions may not have the fw_path array.
- 2) The default firmware directory of the Android system is /vendor/firmware or /etc/firmware or /firmware/image

3. Busybox supports mdev

Check whether the /sbin/mdev file exists.

4. View /proc/sys/kernel/hotplug event handler

cat /proc/sys/kernel/hotplug Check whether the event handler is specified. If not, you need to initialize the system

Execute: echo /sbin/mdev > /proc/sys/kernel/hotplug

5. The USB interface sends an empty packet

Some USB Host controllers do not support sending empty packets, which will cause firmware download failure. If you encounter similar problems as below, please make sure

Check whether the USB Host supports sending empty packets, or try adding

-DCONFIG_USB_ZERO_PACKET

```
07] Leave hgic_bootdl_download
96] enter hgic_bootdl_download
75] hgic_get_fw_dl_addr:150::hgic_get_fw_dl_addr:Check Para:download addr:20000000
18] hgic_get_fw_aes_en:114::hgic_get_fw_aes_en:Check Para:aes_en:0
45] hgic_get_fw_crc_en:132::hgic_get_fw_crc_en:Check Para:crc_en:1
72] hgic_get_fw_local_crc32:204::hgic_get_fw_local_crc32:Check Para:local_crc:0
02] hgic_get_fw_code_offset:186::hgic_get_fw_code_offset:Check Para:code_offset:3072
30] hgic_bootdl_parse_fw:184::firmware hdr len : 3072
51] hgic_bootdl_parse_fw:185::firmware run addr : 20000000 内核已读取固件数据
74] hgic_bootdl_parse_fw:186::firmware size : 134160
96] hgic_bootdl_parse_fw:187::firmware aes_en:0,crc_en:1
29] hgic_bootdl_send_fw:215::send fw data error, no resp!
63] hgic_bootdl_download:412::hgic_bootdl_send_fw error!
88] hgic_bootdl_download:427::Release boot download firmware... 固件下载失败
92] Leave hgic_bootdl_download
25] hgic_bootdl_send_cmd_tmo:252::cmd: 0, no response!!

#FH8852
#ARCH := arm
#COMPILER := arm-fullhan-linux-gcc-libcgnueabi-
#LINUX_KERNEL_PATH := $(CURRENT_PATH)/../linux-3.0.8
#CFLAGS += -DFH8852 -DCONFIG_USB_ZERO_PACKET
```

6. The fw_file parameter cannot contain a path

7. Common errors when downloading firmware

3) If the firmware cannot be found, please confirm it according to the above steps. In most cases, it is because the firmware is not in the correct location or the kernel is not installed.

Enable the firmware loading function.

4) Error code returned by firmware data download

- 1: Illegal command
- 2: Illegal address (read or write address error)
- 3: Illegal length
- 4: No permission (incorrect boot enter key)
- 5: Command verification failed

- 6: Data failed (crc check error)

- 7: Communication timeout

8. Maximum packet length of SDIO interface

During the firmware download phase, the WiFi driver defaults to a maximum packet length of 32704 bytes. Some SD Host controllers may

It does not support sending such long data. If you encounter problems downloading the firmware, you can try to modify the maximum packet length. The code needs to be modified at the location

As shown in the figure below (if_sdio.c):

```

:   sdiodev->trans_cnt_addr = SDIO_TRANS_COUNT_ADDR;
:   sdiodev->int_status_addr = SDIO_INIT_STATUS_ADDR;
:   //sdiodev->status = SDIO_STATUS_STOP;
:   sdiodev->bus.type = HGIC_BUS_SDIO;
:   sdiodev->bus.driv_tx_headroom = SDIO_TX_HEADROOM;
:   sdiodev->bus.tx_packet = hgic_sdio_tx_packet;
:   sdiodev->bus.tx_ctrl = hgic_sdio_tx_ctrl;
:   sdiodev->bus.is_busy = hgic_sdio_check_busy;
:   #ifdef CONFIG_SDIO_REINIT
:   sdiodev->bus.reinit = hgic_sdio_reinit;
:   #endif
:   sdiodev->bus.bootdl_pktlen = 32704;
:   sdiodev->bus.bootdl_cksum = HGIC_BUS_BOOTDL_CHECK_0XFD;
:   sdiodev->bus.probe = probe_hdl;
:   sdiodev->bus.dev_id = DEVID_ID(id);
:   sdiodev->bus.blk_size = SDIO_BLOCK_SIZE;
:   sdiodev->rx_retry = SDIO_CAP_IRQ(func);
:   init_completion(&sdiodev->busy);
:
:   if (!SDIO_CAP_POLL(func)) {
:       set_bit(HGIC_BUS_FLAGS_NOPOLL, &sdiodev->bus.flags);
:   }

```

```

48:
49: static struct hgic_bus hgic_ifbus_sdio = {
50:     .type = HGIC_BUS_SDIO,
51:     .driv_tx_headroom = SDIO_TX_HEADROOM,
52:     .tx_packet = hgic_sdio_tx_packet,
53:     .reinit = hgic_sdio_reinit,
54:     .bootdl_pktlen = 32704,
55:     .bootdl_cksum = HGIC_BUS_BOOTDL_CHECK_0XFD,
56: };
57:
58: static int hgic_sdio_enable(struct sdio_func_t *func)
59: {

```

If you encounter the following error when downloading the firmware: The WiFi driver prompts that the firmware data has been downloaded successfully, but cmd 4

fail. You can try to modify the maximum sdio packet length.

```

hgic_bootdl_download: Cmd run failed:-1
hgic_bootdl_download:425::Release boot download firmware...
leave hgic_bootdl_download
enter hgic_bootdl_download
hgic_get_fw_dl_addr:150::hgic_get_fw_dl_addr:Check Para:download addr:20001000
hgic_get_fw_aes_en:114::hgic_get_fw_aes_en:Check Para:aes_en:0
hgic_get_fw_crc_en:132::hgic_get_fw_crc_en:Check Para:crc_en:1
hgic_get_fw_local_crc32:204::hgic_get_fw_local_crc32:Check Para:local crc:0
hgic_get_fw_code_offset:186::hgic_get_fw_code_offset:Check Para:code offset:8192
hgic_bootdl_parse_fw:186::firmware hdr len : 8192
hgic_bootdl_parse_fw:187::firmware run addr : 20001000
hgic_bootdl_parse_fw:188::firmware size : 335888
hgic_bootdl_parse_fw:189::firmware aes_en:0,crc_en:1
hgic_bootdl_send_fw:212::Send fw data success!
hgic_bootdl_send_fw:212::Send fw data success!
hgic_bootdl_send_fw:212::Send fw data success!
hgic_bootdl_send_fw:212::Send fw data success!
hgic_bootdl_send_fw:212::Send fw data success!
hgic_bootdl_send_fw:212::Send fw data success!
hgic_bootdl_send_fw:212::Send fw data success!
hgic_bootdl_send_fw:212::Send fw data success!
hgic_bootdl_send_fw:212::Send fw data success!
hgic_fwctrl_send_data:220::timeout, ctrl->rxq:0
hgic_bootdl_send_cmd_tmo:255::cmd: 4, no response!!
hgic_bootdl_download: Cmd run failed:-1
hgic_bootdl_download:425::Release boot download firmware...
leave hgic_bootdl_download
enter hgic_bootdl_download
hgic_get_fw_dl_addr:150::hgic_get_fw_dl_addr:Check Para:download addr:20001000

```

The reason for this problem may be related to the maximum block count supported by the sd host. The default setting of the WiFi driver is

The block size is 64 bytes, and the maximum packet length is 512 blocks. So increasing the block size should also solve this problem.

question.

Tysin Confidential Documents