

RAK411 SPI-WIFI Module

Programming Manual v1.7

Shenzhen Rakwireless Technology Co., Ltd

www.rakwireless.com info@rakwireless.com

© 2015 Rakwireless all rights reserved.

Mentioned in this document , the actual company and product names, trademarks are their respective owners.

After update the new version, this document without prior notice.

Content

| | |
|-----------------------------------|----|
| 1 Overview | 1 |
| 1.1 Module Introduction | 1 |
| 1.2 Device Features | 1 |
| 1.3 Key Applications | 2 |
| 2 Functional Description | 3 |
| 2.1 HW Interface | 3 |
| 2.2 Wireless Driver | 3 |
| 2.3 TCP/IP | 3 |
| 2.4 Power Consumption | 3 |
| 3 SPI Interface | 4 |
| 3.1 Hardware Connection | 4 |
| 3.2 SPI Timing Diagram | 4 |
| 3.3 Interrupt Pin | 5 |
| 3.4 SPI Frame Format | 5 |
| 3.4.1 Command frame header | 5 |
| 3.4.2 Read Status | 6 |
| 3.4.3 Read Data | 7 |
| 3.4.4 Write Data | 8 |
| 3.4.5 Command example | 9 |
| 3.4.6 Status Register | 12 |
| 3.4.7 Error code | 12 |
| 3.5 Boot | 13 |
| 3.6 Power Mode | 13 |
| 3.7 Operational Process | 14 |
| 4. AT Command | 16 |
| 4.1 Module Management Commands | 17 |
| 4.1.1 Initializing Module | 17 |
| 4.1.2 Checking Software Version | 18 |
| 4.1.3 Setting Power Mode | 18 |
| 4.1.4 Reading Module Status | 19 |
| 4.1.5 Reset | 19 |
| 4.1.6 Module firmware upgrade | 20 |
| 4.2 Network Operation Commands | 21 |
| 4.2.1 Scanning Wireless Network | 21 |
| 4.2.2 Getting Scanned Information | 22 |
| 4.2.3 Setting Password | 23 |

| | |
|--|----|
| 4.2.4 Setting Channel | 23 |
| 4.2.5 Connecting Wireless Network | 24 |
| 4.2.6 Configuring Static IP Address | 25 |
| 4.2.7 Setting DHCP Mode | 26 |
| 4.2.8 Connecting Network by Easyconfig | 27 |
| 4.2.9 Connecting Network by WPS | 28 |
| 4.2.10 Getting Network Connection Status | 29 |
| 4.2.11 Querying module IP information | 30 |
| 4.2.12 Getting Network Signal Strength | 31 |
| 4.2.13 DNS | 31 |
| 4.2.14 PING | 32 |
| 4.2.15 AP Network Advanced Settings | 33 |
| 4.2.16 Setting Network Listening Intervals | 34 |
| 4.2.17 Disconnecting Current Wireless Network | 34 |
| 4.3 Socket Operation Commands | 35 |
| 4.3.1 TCP Server | 35 |
| 4.3.2 TCP Client | 36 |
| 4.3.3 UDP Client | 37 |
| 4.3.4 UDP Server | 38 |
| 4.3.5 UDP multicast | 39 |
| 4.3.6 Query TCP server connections | 40 |
| 4.3.7 Closing Socket | 41 |
| 4.3.8 Sending Data | 42 |
| 4.3.9 Receiving Data | 43 |
| 4.4 Save Parameters Commands | 45 |
| 4.4.1 Storing Network Configuration Parameters | 45 |
| 4.4.2 Storing Current Network Parameters | 46 |
| 4.4.3 Modifying Web Parameters | 47 |
| 4.4.4 Enabling Automatic Connection | 48 |
| 4.4.5 Starting Web Server | 49 |
| 4.4.6 Getting Saved Network Parameters | 50 |
| 5 Sales and Service | 54 |
| 6 Revision History | 55 |

1 Overview

1.1 Module Introduction

RAK411 module is a Wi-Fi module that fully compliant with IEEE 802.11b/g/n wireless standards, with internally integrated TCP / IP protocol stack, supporting numerous protocols such as ARP, IP, ICMP, TCP, UDP, DHCP CLIENT, DHCP SERVER, DNS and other etc. It supports AP mode, Station mode and Ad-hoc and mode. Users can easily and quickly use it to networking and data transmission. Through SPI interface, the module' s maximum transmission rate is up to 2Mbps.

RAK411 supports storing parameters, and by the customer commands it determines whether to enable automatic networking to realize easy networking and reduce time for system to networking. The module has built-in WEB server, supporting wireless network parameters configuration, supporting wireless firmware upgrade. It also supports WPS and EasyConfig one-key networking, significantly reducing software development effort.

RAK411 has four power management modes, among which the minimum standby power consumption is 2uA, fully meet customer' s requirement for low power design.

1.2 Device Features

- Support IEEE 802.11b/g/n wireless standards
- Support four-wire SPI interface
- Support SPI Clock up to Maximum 16Mhz
- Minimalist hardware peripheral circuit design
- Support Station, Ad-hoc and AP modes
- Support DHCP SERVER / DHCPCLIENT
- Support OPEN, WEP, WPA-PSK, WPA2-PSK and WPS encryption
- Support TCP, UDP protocols, with maximum 8 UDP/TCP connections
- Support webpage-based parameter configuration
- Support WPS and EasyConfig one-key to network connection
- Support parameter storage, customer orders loading after boot
- Support parameters store in Deep Sleep State, with connection time as fastest as 300m
- Support wireless upgrade firmware

- On-board ceramic antenna or U.FL antenna connector
- Operating voltage: 3.3V
- 4 kinds power working modes, with minimum power consumption as 1-2uA
- Small package size: 28.75mmX23.14mmX3.40mm
- FCC, RoHS and CE compliant

1.3 Key Applications

- Portable products
- Home appliances and electrical appliances
- Industrial sensors
- Sales terminals
- Buildings automation
- Logistics and freight management
- Home security and automation
- Medical applications, such as patient monitoring, medical diagnostics
- Metering (stop timing, measuring instruments, meters, etc.)

2 Functional Description

2.1 HW Interface

- Support clock 16MHz Maximum
- Interface actual throughput up to 2Mbps
- Four-wire SPI interface, support SPI data interrupt pin

2.2 Wireless Driver

- Compliant with IEEE 802.11b/g/n standards
- Support AP and STA Mode
- Support WEP, WPA/WPA2-PSK encryption
- Fast networking, allowing module to be added to network within 1 sec after power up
- Support WPS and EasyConfig one-key to network connection
- Support wireless configuration and firmware upgrade

2.3 TCP/IP

- DHCP Client and Server features
- DNS Client and Server functions
- TCP Client, TCP Server, UDP Client, UDP Server and Multicast functions
- 8-way socket applications

2.4 Power Consumption

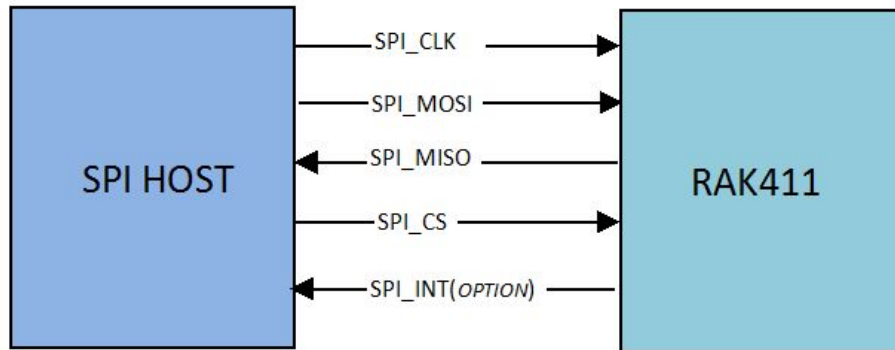
The module supports four power consumption modes:

- Full speed working mode, with approx 80mA average power consumption, peak current less than 310mA
- Power-saving mode, with approx 10mA average power consumption, peak current <310mA, DTIM = 100ms
- Deep sleep mode, with approx 5mA average power consumption, peak current <310mA, DTIM = 100ms
- Standby mode, with power consumption <2uA

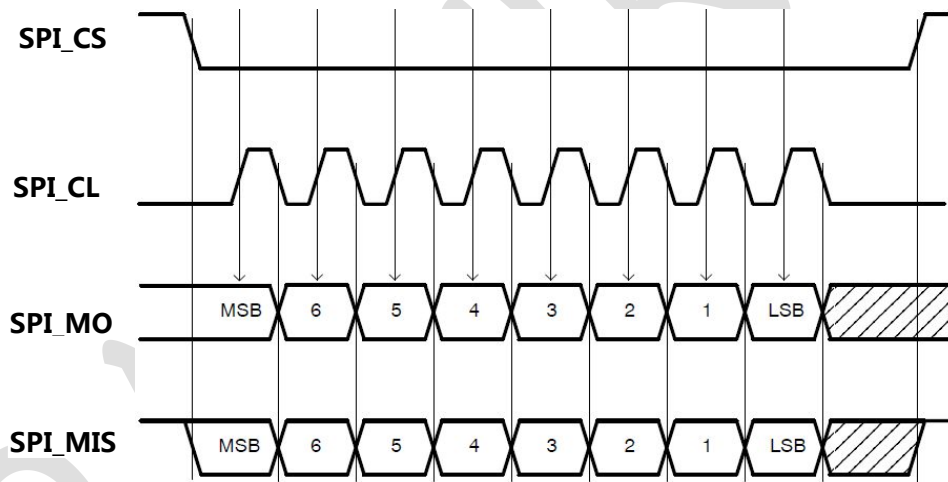
3 SPI Interface

RAK411 communicates with the host through a standard 4-wire SPI interface. SPI clock supports maximum 16MHZ and optional SPI-INT pin. The SPI interface configuration diagram is as follows:

3.1 Hardware Connection



3.2 SPI Timing Diagram



CPOL = 0-----SCK is idle in low level voltage
CPHA = 0-----Data is latched on clock rising edge, while
transmitted on clock falling edge
MSB_FIRST-----MSB is first sent8 BIT
MODE-----Data length is 8
bitsCS-----Slave selective signal is effective low

3.3 Interrupt Pin

RAK411 provides an optional INT pin, so that the host can quickly respond to module data requests via the INT pin. The normal INT pin is low level voltage, when there is abnormal data to be sent, the module pulls INT to high level. After the host receives the rising edge, the module can read the data by sending read frames directly. After reading a package, INT pin goes low. If there is data in the module needs to be sent to the host, the module will once again pull up INT pin.

3.4 SPI Frame Format

The SPI basic operations are divided into three categories: read status, write data, and read data.

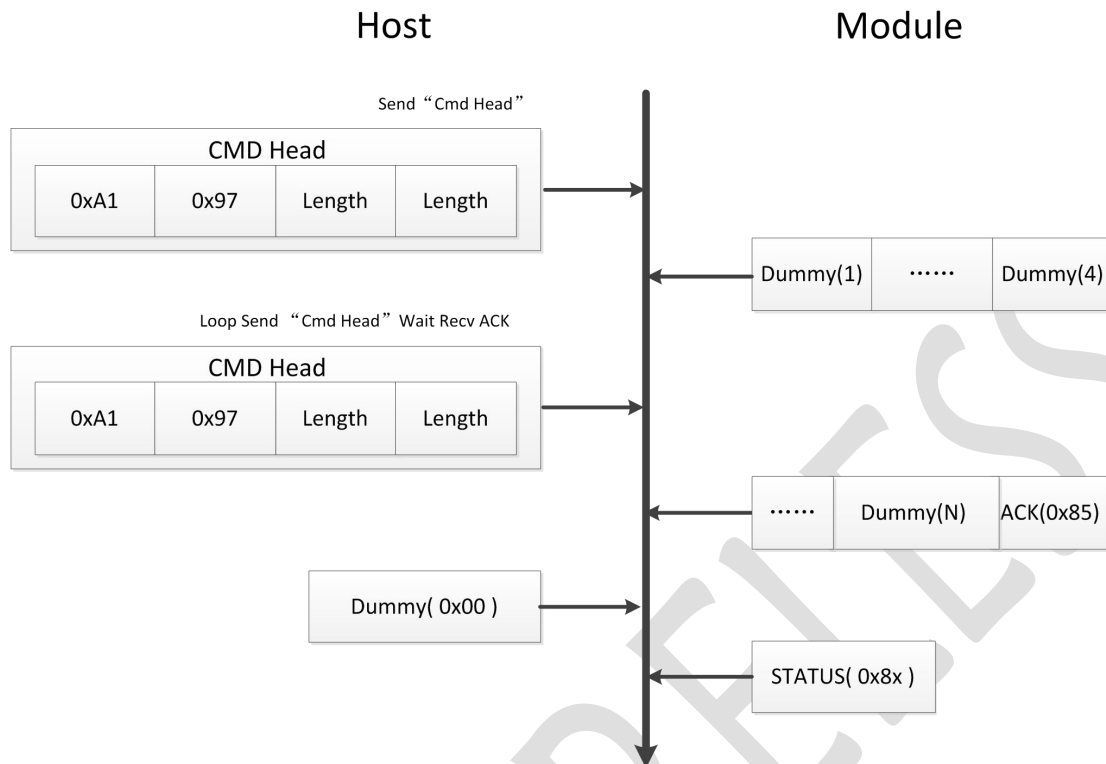
3.4.1 Command frame header

CMD Head:

| | | | |
|-----|------|--------|--------|
| CMD | 0x97 | Length | Length |
|-----|------|--------|--------|

The frame head is composed of three parts, CMD, Length, 0x97.

| Field | Bytes | Value | Instruction |
|--------|-------|-----------|---|
| CMD | 1 | 0xA0-0xCD | Command code for all commands |
| 0x97 | 1 | 0x97 | Fixed |
| Length | 2 | 0-1416 | The command and the length of the data, the low byte in the front |



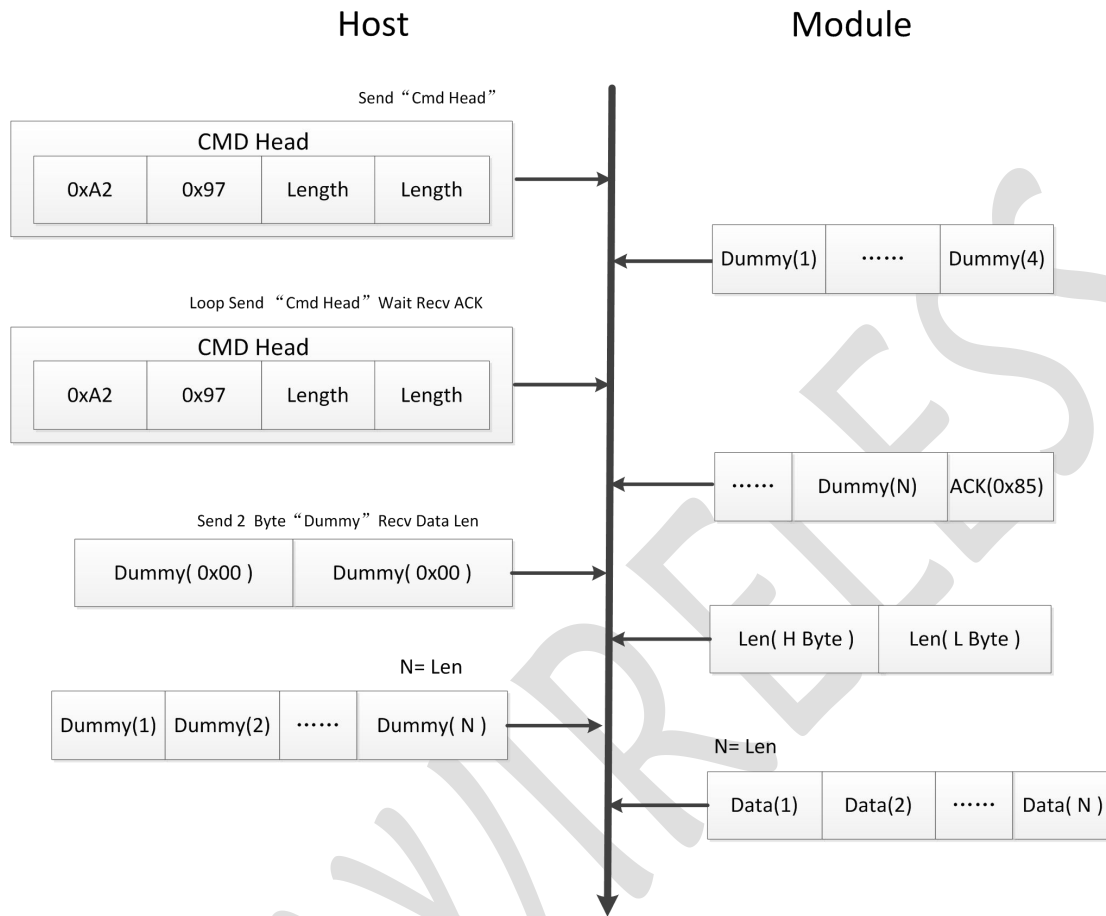
Description:

- 1.The Host end loop sends the Head CMD to the module, and the sending process requires a query if there is received ACK (0x85) .
- 2.If received CMD then stop the current Head ACK send.
- 3.Send 1 Dummy bytes for the subsequent status status.
- 4.Status is the status byte, the detail 3.4.6 status register.

CMD Head :

- 1.CMD fill in the read status command (0xA1).
- 2.0x97 fixed.
- 3.Length should fill in 0 because the current command does not include any parameters.

3.4.3 Read Data



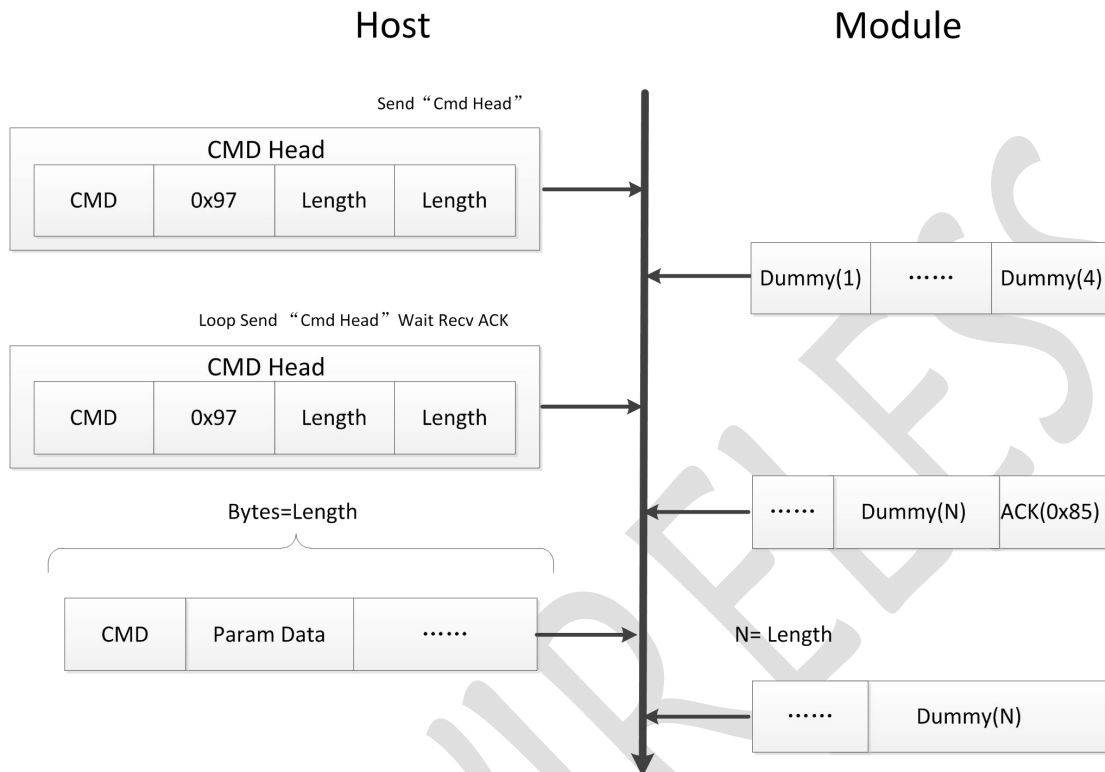
Description:

- 1.The Host end loop sends the Head CMD to the module, and the sending process requires a query if there is received ACK (0x85) .
- 2.If received CMD then stop the current Head ACK send.
- 3.Send 2 bytes Dummy for receiving len, the high byte in the front.
- 4.After the value of the len, the Dummy bytes of len are sent to receive the following data.

CMD Head :

- 1.CMD fill in the read status command (0xA2).
- 2.0x97 fixed.
- 3.Length should fill in 0 because the current command does not include any parameters.

3.4.4 Write Data



Description:

- 1.The Host end loop sends the Head CMD to the module, and the sending process requires a query if there is received ACK (0x85) .
- 2.If received CMD then stop the current Head ACK send.
- 3.Send the current command command code and the parameters of the data, detailed reference 4 "command Daquan" .

CMD Head :

- 1.CMD fill in the command code to send, such as scanning the wireless network command to fill in 0xA3, obtain the software version number command to fill in 0xBE.
- 2.0x97 fixed.
- 3.Length should fill in the corresponding command parameter length, len calculation method is sum of the command parameters in bytes, for example, scanning wireless network command to "CMD" + "channel" + "SSID" =4+4+32=40, access to software version number command CMD = 4. Detailed

reference "4.2.1 scanning wireless network" and "4.1.2 query software version" command.

3.4.5 Command example

Set the network channel command, channel set to 7

CMD Head:

| | | | |
|------|------|------|------|
| 0xAB | 0x97 | 0x08 | 0x00 |
|------|------|------|------|

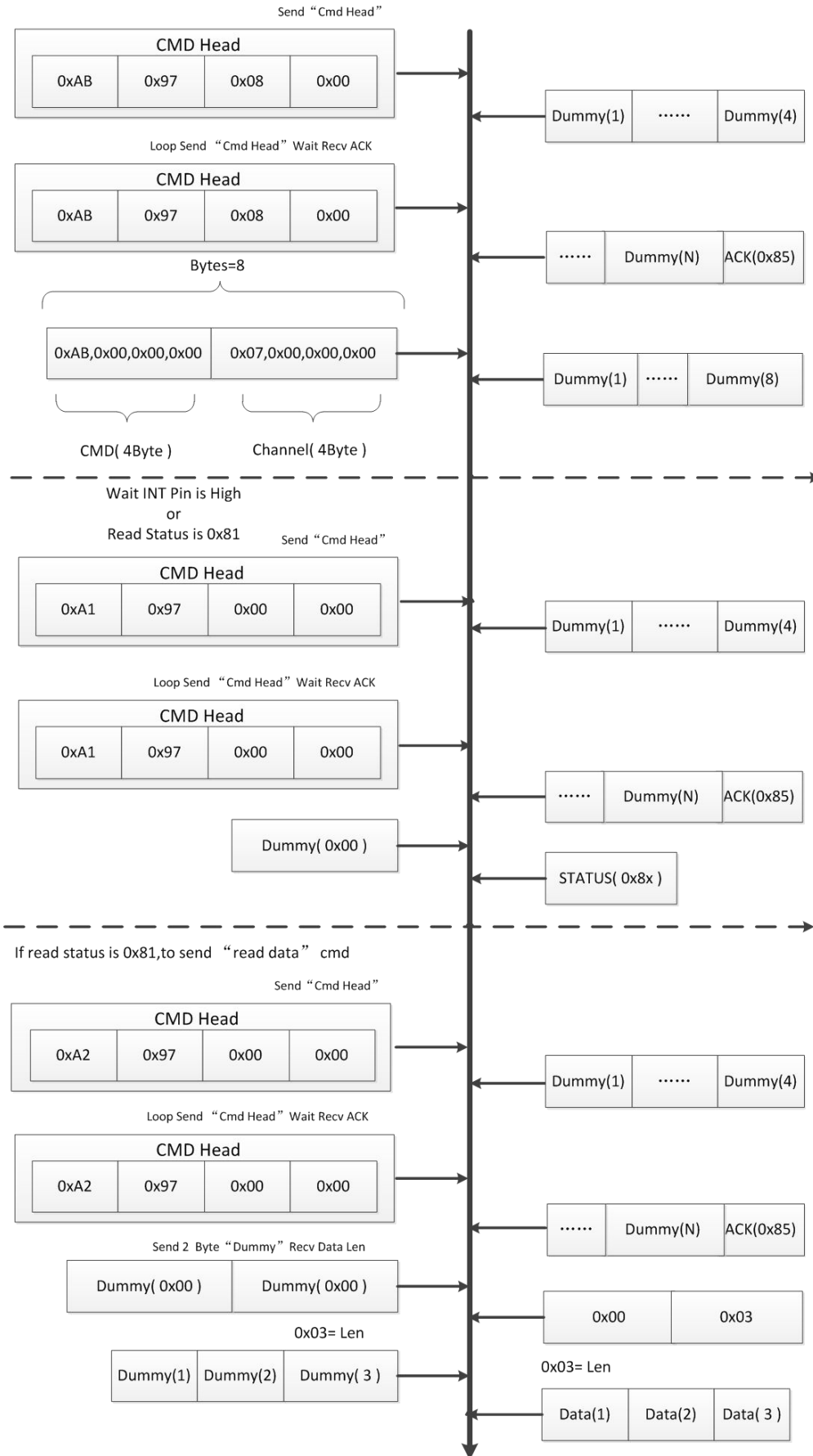
- 1.CMD to fill in the corresponding set channel command code 0xAB.
- 2.0x97 fixed.
- 3.Length the length of the corresponding commands is required.. Set the parameters for the channel command to include the CMD field (4Byte) and the channel field (4Byte), where 0x08 should be filled. Detailed reference to the 4.2.4 set up network channel command.

A complete command flow is divided into three parts:

- 1.Need to send host to write commands to the module. Command for "setting up network channel " .
- 2.Host terminal is required to wait for the INT pin is high, or read the status display data need to read.
- 3.The host is required to send the read data command to the module, and the module returns the command to the host.Finally, the host determines whether the command executes successfully by verifying the returned data.

Host

Module



Description:

- 1.The Host terminal loop sends the Head CMD to the module of the "set channel" command code 0xAB, and the sending process requires a query whether received ACK (0x85).
- 2.If received CMD then stop the current Head ACK send.
- 3.The command code for the current command and the parameter data (8Byte) of the belt.. Detailed reference "4.2.4 network channel " .
- 4.Need to wait for the execution of the command, and determine whether the execution is successful. You can judge the INT pin is high, or read the status command, read to Data_flag=1 (0x81), to determine whether the module data need to read.Detail 3.4.6 status register.
- 5.The Host end loop sends the read status command code CMD to the Head 0xA1 to the module, and the sending process requires a query if there is received ACK (0x85).
- 6.If received CMD then stop the current Head ACK send.
- 7.Send 1 Dummy bytes for the subsequent status status.
- 8.Determine the status byte Data_flag=1, then read the data read the command, or read the state repeatedly.
- 9.Host end loop sends the read data command code CMD to the Head 0xA2 to the module, and the sending process is required to query if there is received ACK (0x85).
- 10.If received CMD then stop the current Head ACK send.
- 11.Send 2 Dummy bytes for receiving length, high byte before. The received length should be 0x03 because the return value of the setup channel command is only 3 bytes. Detailed reference to the " 4.2.4 set up network channel command" .
- 12.After obtaining the value of length equal to 0x03, send Dummy 0x03 bytes for receiving subsequent data. Of the received data, the first two bytes are code code, and the last byte bit command is executed by the status code, and then the command is executed successfully. Detailed reference to the " 4.2.4 set up network channel command" .

3.4.6 Status Register

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|------------|---------|---------|---------|---------|---------|-------------|-----------|
| SPI_STATUS | Reserve | Reserve | Reserve | Reserve | Reserve | Buffer_full | Data_flag |

Description:

Here is the byte of status read, the last two bits representative the data type of module feedback.

Bit7: SPI_STATUS

Bit6: reserve

Bit5: reserve

Bit4: reserve

Bit3: Send_full, Indicates that the module is unable to respond to the command at this time, except for reading status and reading data

Bit 2 Upgrade_err, Upgrade error, need to reset the module

Bit1: Buffer_full=1, The internal data cache area of module is full, valid when SPI_STATUS is 1

Bit0: Data_flag=1, Module has data to inform host (response of command or received data), valid when SPI_STATUS is 1

3.4.7 Error code

When the command frame error, will prompt the error code returned,detailed as follows:

| Code | Description |
|-------|--|
| -1 | Parameter input error (parameter is unable to identify / missing parameter / command is too long / other illegal parameters) |
| -11 | System error (restart module) |
| -12 | Fatal error (contact manufacturer) |
| Other | Please refer to the following commands |

3.5 Boot

RAK411's reboot time is about 210ms. After a normal start, the host sends the initialization command, and the module will return start information:

ASCII----- Welcome to RAK411

HEX----- 57 65 6C 63 6F 6D 65 20 74 6F 20 52 41 4B 34 31 31

3.6 Power Mode

RAK411 supports four power modes, shown as the following table:

| Mode | Control Part | Wireless Part | Wake-up Type | Min Power Consumption (AP) |
|------|--------------|---------------|-----------------------|----------------------------|
| 0 | Normal | Normal | No need | 80mA |
| 1 | Sleep_Mode | Power_Save | No need | 10mA |
| 2 | Deep_Sleep | Power_Save | SPI wake up | 5mA |
| 3 | Deep_Sleep | Shut_down | SPI wake up, Reset | 2uA |

pwrmode=0-----Mode 0

Module works under the maximum performance, control part and wireless part are fully opened.

pwrmode=1-----Mode 1

The control part enters into shallow sleep, the wireless part maintains the current connection status, and enters a low-power mode. The communication is normal, but this will reduce performance of module, the speed of sending and receiving is reduced.

pwrmode=2-----Mode 2

The control part enters into a deep sleep, the wireless part maintains the current wireless connection status, and enters a low-power mode. The remote data or host initiates communications to wake up control part, then enters into mode 1. If no sending and receiving data, it automatically enters into Mode 2.

pwrmode=3-----Mode 3

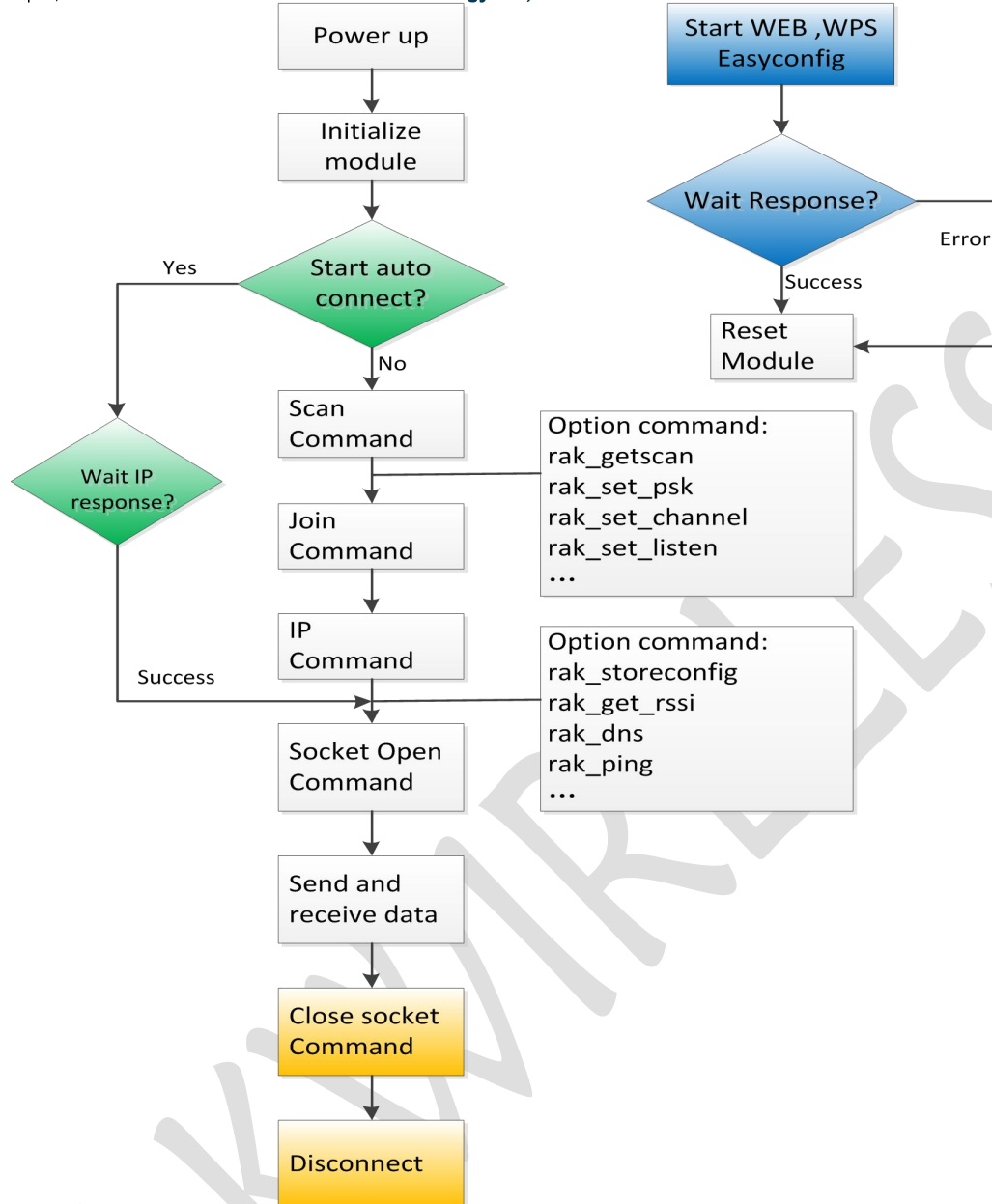
When enters into this mode, module firstly saves current connective status to RAM, and shuts down the power of wireless part, then control part enters into deep state. In this state, module cannot respond to any command or wireless data, lowering consumption to minimum. User can initiate communication or reset module. It enters into mode 0 by default after start.

3.7 Operational Process

RAK411 SPI command operation feature completes a few basic steps of WIFI communication, including network scanning, joining network and obtaining an IP address, and eventually establishing Socket communication. RAK411 provides a variety of convenient operation to implement networking, so that customers can easily complete the network configuration, and concentrate on the management of socket and their own data protocols.

To realize automatic networking management, customers can take advantage of WEB, WPS and EasyConfig configuration features. The module will automatically store parameters after a successful configuration, and these automatic networking commands can be used any time, letting the module automatically complete networking operation, and returning the results.

The basic operation of the process is as follows:



4. AT Command

The SPI commands are divided into four parts: module management commands, network operations commands, socket operation commands, and parameters storing commands, shown as the followings:

| Command | Description |
|-----------------------------------|--|
| Module Management Commands | |
| rak_sys_init | Initialize module, read boot information |
| rak_get_version | Check software version |
| rak_setpwrmode | Set module power mode |
| rak_read_status | Read module status information |
| rak_reset | Reset module |
| rak_set_upgrade | Firmware upgrade for modules |
| Network Operation Commands | |
| rak_scan | Scan wireless networks |
| rak_getscan | Reads a specified number of scan results |
| rak_set_psk | Set network password |
| rak_set_channel | Set network channel |
| rak_connect | Connect wireless network |
| rak_set_ipstatic | Configure Static IP Address |
| rak_ipconfig_dhcp | Setting DHCP Mode |
| rak_easy_config | Connecting network by Easyconfig |
| rak_wps | Connecting Network by WPS |
| rak_get_con_status | Get network connection status |
| rak_ipconfig_query | Check module IP information |
| rak_get_rssi | Get network signal strength of module |
| rak_dns | DNS |
| rak_ping | Ping hosts in the network |
| rak_apconfig | AP Network Advanced Settings |
| rak_set_listen | Set network listening intervals |
| rak_disconnect | Disconnect the current wireless network |
| Socket Operation Commands | |
| rak_udp_client | Establish UDP client |
| rak_udp_server | Establish UDP Server |
| rak_tcp_client | Establish TCP client |
| rak_tcp_server | Establish TCP Server |

| | |
|---------------------------------|--|
| rak_socket_close | Close an opened socket handle |
| rak_get_tcps | Gets the number and the information for the tcp sever to connect to client |
| rak_send_data | Send data to an opened socket handle |
| rak_read_data | Read command returns / network data / network information |
| Save parameters commands | |
| rak_storeconfig_data | Store network configuration parameters |
| rak_storeconfig | Store the current network parameters |
| rak_web_store | Store web server built-in parameters |
| rak_auto_connect | Enable automatically connecting |
| rak_start_web | Start web server and configure module |
| rak_get_storeconfig | Get saved network parameters |
| rak_get_webconfig | Get web server built-in parameters |

4.1 Module Management Commands

4.1.1 Initializing Module

Command:

rak_sys_init

Syntax:

uint32_t cmd;

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| Cmd | 4 | 0xA0,0x00,0x00,0x00 | Command Code |

Description:

It is used to initialize module, and read boot information.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-------------------|--------------------|
| <CODE> | 2 | 0xA0,0x00 | Response Code |
| <DATA> | 17 | Welcome to RAK411 | Welcome string |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.1.2 Checking Software Version

Command:

```
rak_get_version
```

Syntax:

```
uint32_t cmd;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| Cmd | 4 | 0xBE,0x00,0x00,0x00 | Command Code |

Parameter:

It is used to check module versions, including versions of host and WLAN.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xBE,0x00 | Response Code |
| <HOST_FW> | 8 | ASCII | Host version |
| <-> | 1 | - | Version delimiter |
| <WLAN_FW> | 6 | ASCII | Wlan version |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.1.3 Setting Power Mode

Command:

```
rak_setpwrmode
```

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t powermode;
}rak_pwr_mode_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| Cmd | 4 | 0xB8,0x00,0x00,0x00 | Command Code |

| | | | |
|-----------|---|------|------------|
| powermode | 4 | 0--3 | Power mode |
|-----------|---|------|------------|

Parameter:

It is used to set module power mode. Detailed reference "3.6 power mode" .

Return Value: N/A

4.1.4 Reading Module Status

Command:

rak_read_status

Syntax:

uint32_t cmd;

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| Cmd | 4 | 0xA1,0x00,0x00,0x00 | Command Code |

Parameter:

It is used to read the data of module status register.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-------|--|
| <STATUS> | 1 | 0x81 | There are data to read |
| | | 0x82 | Data buffer is full |
| | | 0x84 | Upgrade error, need to reset the module |
| | | 0x88 | Indicates that the module is unable to respond to the command at this time, except for reading status and reading data |

4.1.5 Reset

Command:

rak_reset

Syntax:

```
uint32_t cmd;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| Cmd | 4 | 0xBF,0x00,0x00,0x00 | Command Code |

Parameter:

It is used to resets the entire module via command.

Return Value: N/A

4.1.6 Module firmware upgrade

Command:

```
rak_reset_upgrade
```

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t File_len;
    uint16_t Frame_len;
    char buffer[1400];
}rak_upgrad_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|----------------------------|---|
| cmd | 4 | 0xC8,0x00,0x00,0x00 | Command Code |
| File_len | 4 | | Number of bytes of whole firmware file |
| Frame_len | 2 | 0-1400 , Recommend 1000 | Number of firmware data bytes sent by the current command |
| buffer | 1400 | | Firmware data |

Description:

Use the command to upgrade the module firmware. A firmware file is required to generate an array of bytes by means of the tool, and then load in the host code (approximately 500KByte). After the host code through this command array data packets are sent into the internal module. At the same time to judge the status byte of the module upgrade err bit is set. If the set is said data error and need to reset module. After the completion of the firmware data is complete, the module will be firmware upgrade, after the upgrade will restart. The host side need to query whether the INT pin is high to determine whether the module has been restarted (the entire process is about 50 seconds), and then determine whether the upgrade is successful by verifying the version number. Return Value: N/A

4.2 Network Operation Commands

4.2.1 Scanning Wireless Network

Command:

rak_scan

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t channel;
    charss id[32];
} rak_scan_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------------------|--|
| Cmd | 4 | 0xA3,0x00,0x00,0x00 | Command Code |
| channel | 4 | 0—13 | Scanning specified channel(s), scan all channels if value is 0 |
| Ssid | 32 | Wireless Network Name | Specified SSID, scan all channels if value is null |

Parameter:

It is used to scan wireless networks via command.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--|
| <CODE> | 2 | 0xA3,0x00 | Response Code |
| <AP_NUM> | 2 | 0x08,0x00 | Scan to the number of wireless networks, up to 8 |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.2.2 Getting Scanned Information

Command:

rak_getscan

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t scan_num;
}rak_getscan_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--|
| Cmd | 4 | 0xA4,0x00,0x00,0x00 | Command Code |
| Scan_num | 4 | 1—10 | The amount of network information is obtained if the scan_num is greater than the actual number of scans, then the number of network information is returned |

Description:

It is used to get the scanned network information.

Return Value:

| Parameter | Bytes | Value | Description | | | | | | | |
|------------|-------|---------------|------------------------------------|----------|----------|------|------|------|------|------|
| <CODE> | 2 | 0xA4, 0x00 | Response Code | | | | | | | |
| <CHANNEL> | 1 | 1--14 | Channel | | | | | | | |
| <RSSI> | 1 | -99--0 | Channel intensity (negative value) | | | | | | | |
| <SEC_MODE> | 2 | | Bit7 | Bit 6 | Bit 5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| | | | Wp | Wp | We | 802. | Psk | Wep | Tkip | Ccmp |

| | | | | | | | | | | |
|------------|----|----|--------------------|---|---|----|--|--|--|--|
| | | | a2 | a | p | 1x | | | | |
| <SSID_LEN> | 2 | | Length of SSID | | | | | | | |
| <SSID> | 32 | | SSID | | | | | | | |
| <BSSID> | 6 | | BSSID | | | | | | | |
| <STATUS> | 1 | 0 | Command successful | | | | | | | |
| | | -2 | Command failed | | | | | | | |

4.2.3 Setting Password

Command:

rak_set_psk

Syntax:

```
typedef struct {
uint32_t cmd;
char psk[64];
}rak_psk_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|------------------|
| Cmd | 4 | 0xA5,0x00,0x00,0x00 | Command Code |
| Psk | 64 | | Network password |

Description:

It is used to set network password.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xA5,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.2.4 Setting Channel

Command:

rak_set_channel

Syntax:

```
typedef struct {
```

```
uint32_t cmd;

uint32_t channel;

}rak_channel_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|----------------------------|
| Cmd | 4 | 0xAB,0x00,0x00,0x00 | Command Code |
| channel | 4 | 0--13 | SettingAP / Ad-Hoc channel |

Description:

It is used to set network channel.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xAB,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.2.5 Connecting Wireless Network

Command:

```
rak_connect
```

Syntax:

```
typedef struct {
uint32_t cmd;
uint32_t mode;
char ssid[32];
}rak_conn_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|---------------------|
| Cmd | 4 | 0xA6,0x00,0x00,0x00 | Command Code |
| mode | 4 | 0=station | Select network mode |
| | | 1=ap | |
| | | 2=ad-hoc | |
| Ssid | 32 | ASCII | SSID |

Description:

It is used to connect an AP/ADHOC network or establish a specified AP Network.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|----------------------|
| <CODE> | 2 | 0xA6,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Cannot find the SSID |
| | | -3 | Command failed |

4.2.6 Configuring Static IP Address

Command:

`rak_set_ipstatic`

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t addr;
    uint32_t mask;
    uint32_t gw;
    uint32_t dnssvr1;
    uint32_t dnssvr2;
}rak_ipstatic_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xAD,0x00,0x00,0x00 | Command Code |
| addr | 4 | 0xC0,0xA8,0x07,0x01 | IP address |
| mask | 4 | 0xFF,0xFF,0xFF,0x00 | subnet mask |
| gw | 4 | 0xC0,0xA8,0x07,0x01 | gateway |
| dnssvr1 | 4 | 0xC0,0xA8,0x07,0x01 | DNS server 1 |
| dnssvr2 | 4 | 0x00,0x00,0x00,0x00 | DNS server 2 |

Description:

This command is used to assign static IP address for module.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xAD,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.2.7 Setting DHCP Mode

Command:

rak_ipconfig_dhcp

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t mode;
} rak_ipdhcp_t ;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|------------------|
| Cmd | 4 | 0xAC,0x00,0x00,0x00 | Command Code |
| mode | 4 | 0= DHCP CLIENT | Select DHCP Mode |
| | | 1= DHCP SERVER | |

Description:

This command is used to set DHCP working mode.

Return Value:

DHCP SERVER

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xAC,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

DHCP CLENT:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|----------------------------|
| <CODE> | 2 | 0xAC,0x00 | Response Code |
| <MAC> | 6 | | MAC address |
| <ADDR> | 4 | | IP address |
| <MASK> | 4 | | Subnet mask |
| <GW> | 4 | | Gateway |
| <DNS1> | 4 | | DNS server 1 |
| <DNS2> | 4 | | DNS server 2 |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |
| | | -4 | Get ip information failure |

4.2.8 Connecting Network by Easyconfig

Command:

`rak_easy_config`

Syntax:

`uint32_tcmd;`

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xC2,0x00,0x00,0x00 | Command Code |

Description:

Through this command to open the module EasyConfig function, with the mobile phone APP software to complete the module automatically join the designated network. After successfully, the module automatically saves current network parameters, with the `rak_storeconfig` command.

Return Value:

| Parameter | Bytes | Value | Description |
|------------|-------|---------------|---|
| <CODE> | 2 | 0xC2, 0x00 | Response Code |
| <SSID> | 32 | ASCII | SSID |
| <SEC_MODE> | 1 | | Bit7 Bit6 Bit5 Bit4 Bit3 Bit2 Bit1 Bit0 |
| | | | Wpa Wpa Wep 802. Psk Wep Tkip Ccm 2 1x p |
| PSK | 64 | ASCII | Network password |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | cannot find the ap |
| | | -3 | Failed to be added in router |
| | | -4 | Failed to get dynamic IP |
| | | -6 | Easy config failed |

4.2.9 Connecting Network by WPS

Command:

rak_wps

Syntax:

uint32_t cmd;

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xC3,0x00,0x00,0x00 | Command Code |

Description:

Open module WPS function via this command. After the success of WPS, the module automatically saves the current network parameters, with the rak_storeconfig command.

Return Value:

| Parameter | Bytes | Value | Description | | | | | | | |
|-----------|-------|-----------|------------------------|------|------|--------|------|------|------|------|
| <CODE> | 2 | 0xC3,0x00 | Response Code | | | | | | | |
| <SSID> | 32 | ASCII | SSID | | | | | | | |
| SEC_MODE | 1 | | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| | | | Wpa2 | Wpa | Wep | 802.1x | Ps | Wep | Tkip | Ccm |
| Psk | 64 | ASCII | Network password | | | | | | | |
| <STATUS> | 1 | 0 | Command successful | | | | | | | |
| | | -2 | Cannot find AP | | | | | | | |
| | | -3 | Join the router failed | | | | | | | |
| | | -4 | Ip failed to get | | | | | | | |
| | | -5 | Wps failed | | | | | | | |

4.2.10 Getting Network Connection Status

Command:

`rak_get_constatus`

Syntax:

`uint32_t cmd;`

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xA7,0x00,0x00,0x00 | Command Code |

Description:

It is used to get module network status.

If the module is working in Station mode, this command is used to get wireless network connection status.

If the module is working in AP mode, this command is used to determine the device's connection status.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xA7,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.2.11 Querying module IP information

Command:

`rak_ipconfig_query`

Syntax:

`uint32_t cmd;`

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xAE,0x00,0x00,0x00 | Command Code |

Description::

It is used to get module IP information, including MAC address, IP address, subnet mask, gateway, and DNS server.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|---------------------------------|
| <CODE> | 2 | 0xAE,0x00 | Response Code |
| <MAC> | 6 | | MAC address |
| <ADDR> | 4 | | IP address |
| <MASK> | 4 | | Subnet mask |
| <GW> | 4 | | Gateway |
| <DNS1> | 4 | | DNS Server 1 |
| <DNS2> | 4 | | DNS Server 2 |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | To obtain IP information failed |

4.2.12 Getting Network Signal Strength

Command:

`rak_get_rssi`

Syntax:

`uint32_t cmd;`

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xA9,0x00,0x00,0x00 | Command Code |

Description:

It is used to get the current signal strength of the network.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|---------------------------------|
| <CODE> | 2 | 0xA9,0x00 | Response Code |
| <RSSI> | 2 | -99--0 | current signal strength |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | To obtain IP information failed |

4.2.13 DNS

Command:

`rak_dns`

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t name[42];
}rak_dns_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xAF,0x00,0x00,0x00 | Command Code |
| name | <42 | ASCII | Domain |

Description:

It is used to convert domain to the corresponding IP address, the domain must be configured available DNS server.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|------------------------|
| <CODE> | 2 | 0xAF,0x00 | Response Code |
| <ADDR> | 4 | | Ip address |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | DNS resolution failure |

4.2.14 PING

Command:

rak_ping

Syntax:

```

Typedef struct {
uint32_t cmd;
uint32_t hostaddr;
uint32_t count;
uint32_t size;
}rak_ping_t;
  
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|-------------------------------|
| Cmd | 4 | 0xB0,0x00,0x00,0x00 | Command Code |
| hostaddr | 4 | String | Specified host |
| count | 2 | | Number of packets |
| size | 2 | 1--1400 | Packet size, maximum 1400byte |

Description:

It is used to run the ping command.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-------|-------------|
|-----------|-------|-------|-------------|

| | | | |
|----------|---|-----------|--------------------|
| <CODE> | 2 | 0xB0,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Cannot access host |

4.2.15 AP Network Advanced Settings

Command:

rak_apconfig

Syntax:

```

typedef struct {
    uint32_t cmd;
    uint8_t hidden;
    uint8_t countryCode[3];
}rak_apconfig_t;
  
```

Parameter:

| Parameter | Bytes | Value | Description |
|--------------|-------|---------------------|-------------------------|
| cmd | 4 | 0xAA,0x00,0x00,0x00 | Command Code |
| hidden | 1 | 0 | Network name is visible |
| | | 1 | Network name is hidden |
| Country Code | 3 | Country code | Country code, e.g. CN |

Description:

It is used to set up parameters for a wireless access point, such as the country code, whether network name is hidden or not.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xAA,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.2.16 Setting Network Listening Intervals

Command:

`rak_set_listen`

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t time;
}rak_beacon_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|---|
| cmd | 4 | 0xA8,0x00,0x00,0x00 | Command Code |
| time | 4 | 20--1000 | Need to refer to the wireless router settings for specific parameters |

Description:

It is used to set module beacon interval in Station mode.

Note:

In power saving mode, reducing power consumption can be realized via increasing parameter values, but by this way it may cause delay in receiving wireless data.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xA8,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.2.17 Disconnecting Current Wireless Network

Command:

rak_disconnect

Syntax:

```
uint32_t cmd;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| Cmd | 4 | 0xB7,0x00,0x00,0x00 | Command Code |

Description:

It is used to disconnect the current wireless network.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|---------------------------------|
| <CODE> | 2 | 0xB7,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Current network is disconnected |

4.3 Socket Operation Commands

4.3.1 TCP Server

Command:

```
rak_tcp_server
```

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint16_t dummy;
    uint16_t port;
}rak_server_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|-------------------|
| cmd | 4 | 0xB4,0x00,0x00,0x00 | Command Code |
| dummy | 2 | | Invalid data |
| port | 2 | 1-65535 | Local port number |

Description:

Module as a TCP server and create a listening port, if the operation is successful, the module will return a hexadecimal identifier (Socket ID), is used to manage the connection. Establish a TCP server can connect up to seven clients. Establish a TCP server on a different port, allows you to create up to four. Close TCP server identifier to connect to the TCP sever the client connection will be closed.

Return Value:

| Parameter | Bytes | Value | Description |
|---------------|-------|-----------|------------------------------|
| <CODE> | 2 | 0xB4,0x00 | Response Code |
| <SOCKET_FLAG> | 2 | 8--11 | Socket identifier |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Failed to create |
| | | -3 | Failed to bind |
| | | -4 | Target port connection error |

4.3.2 TCP Client

Command:

rak_tcp_client

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t dest_addr;
    uint16_t dest_port;
    uint16_t local_port;
}rak_client_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------------|
| cmd | 4 | 0xB3,0x00,0x00,0x00 | Command Code |
| dest_addr | 4 | | Target IP address |
| dest_port | 2 | 1-65535 | Target port number |

| | | | |
|------------|---|---------|-------------------|
| local_port | 2 | 0-65535 | Local port number |
|------------|---|---------|-------------------|

Description:

This command is to create a TCP CLIENT and connect with the remote TCP SERVER, if the operation is successful, the module will return a hexadecimal identifier that is used to manage the connection. This command can create up to eight connections. Port numbers are in sorted in ascending order.

Return Value:

| Parameter | Bytes | Value | Description |
|---------------|-------|-----------|------------------------------|
| <CODE> | 2 | 0xB3,0x00 | Response Code |
| <SOCKET_FLAG> | 2 | 0--7 | Socket identifier |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Failed to create |
| | | -3 | Failed to bind |
| | | -4 | Target port connection error |

4.3.3 UDP Client

Command:

rak_udp_client

Syntax:

```
typedef struct {
  uint32_t cmd;
  uint32_t dest_addr;
  uint16_t dest_port;
  uint16_t local_port;
}rak_client_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|-------------------|
| cmd | 4 | 0xB3,0x00,0x00,0x00 | Command Code |
| dest_addr | 4 | | Target IP address |

| | | | |
|------------|---|---------|--------------------|
| dest_port | 2 | 1-65535 | Target port number |
| local_port | 2 | 0-65535 | Local port number |

Description:

This command is to create a UDP port on the module and set remote IP address and port number, if you create successful, the module will return a hexadecimal identifier that is used to manage the connection. This command can create up to eight connections. Port numbers are in sorted in ascending order.

Return Value:

| Parameter | Bytes | Value | Description |
|---------------|-------|-----------|------------------------------|
| <CODE> | 2 | 0xB1,0x00 | Response Code |
| <SOCKET_FLAG> | 2 | 0--7 | Socket identifier |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Failed to create |
| | | -3 | Failed to bind |
| | | -4 | Target port connection error |

4.3.4 UDP Server

Command:

rak_udp_server

Syntax:

```
typedef struct {
  uint32_t cmd;
  uint16_t dummy;
  uint16_t port;
}rak_server_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|-------------------|
| cmd | 4 | 0xB2,0x00,0x00,0x00 | Command Code |
| dummy | 2 | | Invalid data |
| port | 2 | 1-65535 | Local port number |

Description:

Returns a hexadecimal after this command is to create a UDP listening locally on the specified port, create success identifier (Socket ID), waiting to receive the remote port data. If the remote port to send data to this port, the received data will be included in each other's IP and port information; if other modules need to reply to a message, you need to specify each other's IP and port number information when sending data, only reply. UDP server applications more flexible, capable of receiving unicast and broadcast messages and can send data to the specified IP and port initiative.

Note: The port number LSB first.

Return Value:

| Parameter | Bytes | Value | Description |
|---------------|-------|-----------|------------------------------|
| <CODE> | 2 | 0xB2,0x00 | Response Code |
| <SOCKET_FLAG> | 2 | 0--7 | Socket identifier |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Failed to create |
| | | -3 | Failed to bind |
| | | -4 | Target port connection error |

4.3.5 UDP multicast

Command:

rak_multicast

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t dest_addr;
    uint16_t dest_port;
    uint16_t local_port;
}rak_client_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|------------|-------|---------------------------|-----------------------------|
| cmd | 4 | 0xC4,0x00,0x00,0x00 | Command code |
| dest_addr | 4 | 224.0.0.1-239.255.255.255 | Target multicast IP address |
| dest_port | 2 | 1-65535 | Target port number |
| local_port | 2 | 1-65535 | Local port number |

Description:

This command is to create a UDP multicast socket on the module, you can specify the multicast IP multicast, in the group for data communications. Port number is low before.

Return Value:

| Parameter | Bytes | Value | Description |
|---------------|-------|-----------|----------------------------------|
| <CODE> | 2 | 0xC4,0x00 | Response code |
| <SOCKET_FLAG> | 2 | 0--7 | Socket identifier |
| <STATUS> | 1 | 0 | Execute successfully |
| | | -2 | Create failure |
| | | -3 | Failure of binding |
| | | -4 | The target port connection error |

4.3.6 Query TCP server connections

Command:

rak_get_tcps

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t tcpsever_flag;
}rak_tcpsever_status_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|-------|-------------|
|-----------|-------|-------|-------------|

| | | | |
|---------------|---|---------------------|--------------------------------|
| cmd | 4 | 0xCD,0x00,0x00,0x00 | Command Code |
| tcpsever_flag | 4 | 8-11 | Query corresponding TCP server |

Description:

By identifier corresponding server, TCP server queries the number of client connections, and client information.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------------|-------|-----------|---|
| <CODE> | 2 | 0xCD,0x00 | Response Code |
| <TCP_NUM> | 1 | | The maximum number of connections TCP Client 7 (0-7) |
| <TCP_FLAG> | 1 | | TCP client identifier |
| <DEST_TCP_PORT> | 2 | | TCP port number of the client |
| <DEST_TCP_IP> | 4 | | TIP address of the TCP client |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Specifies the identifier does not exist |
| | | -3 | The check fails |

4.3.7 Closing Socket

Command:

rak_socket_close

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint16_t dummy;
    uint16_t flag;
}rak_close_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|-------------------|
| cmd | 4 | 0xB5,0x00,0x00,0x00 | Command Code |
| dummy | 2 | | Invalid data |
| flag | 2 | 0-11 | Socket identifier |

Description:

It is used to close the already opened socket identifier, Close port monitoring or connection.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|-------------------------------|
| <CODE> | 2 | 0xB5,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Specified port does not exist |
| | | -3 | Disable to close |

4.3.8 Sending Data

Command:

rak_send_data

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t dest_addr;
    uint16_t dest_port;
    uint16_t socket_flag;
    uint16_t len;
    Char buffer[1400];
}rak_send_t;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|---------|---------------------|-------------------|
| cmd | 4 | 0xB6,0x00,0x00,0x00 | Command Code |
| dest_addr | 4 | | Target IP address |
| dest_port | 4 | 1-65535 | Target port |
| flag | 2 | 0-7 | Socket identifier |
| len | 2 | 1--1400 | Data length |
| buffer | 1--1400 | | Data |

Description:

This command is used to send data to the target (port identifier), the maximum data length is 1400, buffer can be data in any format, module will send data without any treatment. If the connection is a TCP connection, then the destination IP and destination port can be omitted, entered with value 0. When the connection is UDP, if not specified, the value can be 0. If it needs to send data to specified target as LUDP, fill in the target IP, and target port number. Port numbers are in sorted in ascending order.

Return Value:N/A

4.3.9 Receiving Data

Command:

`rak_rcv_data`

Syntax:

`uint32_t cmd;`

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xA2,0x00,0x00,0x00 | Command Code |

Description:

This command is to read data commands of module. The results can be command results, or the network data and connection information. The data type can be viewed by CODE.

Return Value:

<CODE> =0xC8 Receiving data

Receiving data from network

| Parameter | Bytes | Value | Description |
|---------------|-------|-----------|-----------------|
| <CODE> | 2 | 0xC8,0x00 | Response Code |
| <SOCKET_FLAG> | 2 | 0-7 | Port descriptor |
| <DATA_LEN> | 2 | | Data length |
| <IP_PORT> | 2 | | Port number |
| <IP_ADDR> | 4 | | IP address |

| | | |
|--------|--------|------|
| <DATA> | 1-1400 | Data |
|--------|--------|------|

<CODE> =0xC9 socket connected

| Parameter | Bytes | Value | Description |
|---------------|-------|-----------|-----------------|
| <CODE> | 2 | 0xC9,0x00 | Response Code |
| <SOCKET_FLAG> | 2 | 0-7 | Port descriptor |
| <dummy> | 2 | | Invalid data |
| <IP_PORT> | 2 | | Port number |
| <IP_ADDR> | 4 | | IP address |

<CODE> =0xCA socket disconnected

| Parameter | Bytes | Value | Description |
|---------------|-------|-----------|-----------------|
| <CODE> | 2 | 0xCA,0x00 | Response Code |
| <SOCKET_FLAG> | 2 | 0-7 | Port descriptor |
| <dummy> | 2 | | Invalid data |
| <IP_PORT> | 2 | | Port number |
| <IP_ADDR> | 4 | | IP address |

<CODE> =0xCB network connected

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|---------------|
| <CODE> | 2 | 0xCB,0x00 | Response Code |
| <STATUS> | 1 | 0 | Successful |

<CODE> =0xCC network disconnected

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|---------------|
| <CODE> | 2 | 0xCC,0x00 | Response Code |
| <STATUS> | 1 | 0 | Successful |

Note:

RAK411 modules within the network to automatically connect, disconnect from the network will notify the host. After disconnecting the network module, the internal reconnection interval is: 1s, 2s, 4s, 4s, 4s, 4s....

4.4 Save Parameters Commands

4.4.1 Storing Network Configuration Parameters

Command:

rak_storeconfig_data

Syntax:

```
typedef struct {
    uint32_t cmd;
    uint32_t feature_bitmap;
    uint8_t net_type;
    uint8_t channel;
    uint8_t sec_mode;
    uint8_t dhcp_mode;
    char ssid[32];
    char psk[64];
    ip_param_t ip_param;
    ap_config_t ap_config;
}config_t
```

Parameter:

| Parameter | Bytes | Value | Description |
|----------------|-------|---------------------|--------------------------|
| cmd | 4 | 0xC0,0x00,0x00,0x00 | Command Code |
| feature_bitmap | 4 | 0x00,0x00,0x00,0x00 | Switching features |
| net_type | 1 | 0 | Station |
| | | 1 | Ap |
| | | 2 | Ad-hoc |
| channel | 1 | 0-13 | Channel |
| sec_mode | 1 | 0 | Network is not encrypted |
| | | 1 | Network is encrypted |
| dhcp_mode | 1 | 0 | STA:Dhcp client |
| | | 1 | STA:ip static |
| ssid | 32 | ASCII | Network identifier |
| psk | 65 | ASCII | Network key |

| | | | |
|-----------|----|--|------------------------|
| dummy | 2 | | Null data |
| ip_param | 20 | | IP parameters |
| ap_config | 4 | | AP advanced parameters |

Description:

It is used to save user parameters, including password, SSID, IP address, and scan information.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xC0,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.4.2 Storing Current Network Parameters

Command:

rak_storeconfig

Syntax:

uint32_t cmd;

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xC1,0x00,0x00,0x00 | Command Code |

Description:

It is used to save user parameters. Parameters can be successfully saved only after correctly performing the commands scan, connect and get IP.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xC1,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

4.4.3 Modifying Web Parameters

Command:

rak_web_store

Syntax:

```
typedef struct {
    uint32_t cmd;
    config_t params;
    char user_name[17];
    char user_psk[17];
}web_t
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|------------------------------|
| Cmd | 4 | 0xC5,0x00,0x00,0x00 | Command Code |
| params | 132 | | Network parameters |
| User_name | 17 | ASCII | Web authentication user name |
| User_psk | 17 | ASCII | Web authentication password |

Description:

It is used to save the network parameters used to initiate network.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--------------------|
| <CODE> | 2 | 0xC5,0x00 | Response Code |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed |

Related parameters :

Param

| Parameter | Bytes | Value | Description |
|----------------|-------|---------------------|-----------------------|
| feature_bitmap | 4 | 0x00,0x00,0x00,0x00 | Characteristic switch |

| | | | |
|-----------|----|-------|------------------------------|
| net_type | 1 | 0 | Station |
| | | 1 | Ap |
| | | 2 | Ad-hoc |
| channel | 1 | 0-13 | Channel |
| sec_mode | 1 | 0 | The network is not encrypted |
| | | 1 | Network encryption |
| dhcp_mode | 1 | 0 | STA:Dhcp client |
| | | 1 | STA:ip static |
| ssid | 33 | ASCII | Network identifier |
| psk | 65 | ASCII | Network key |
| dummy | 2 | | Null data |
| ip_param | 20 | | IP parameter |
| ap_config | 4 | | AP advanced parameters |

Ip_param:

| Parameter | Bytes | Value | Description |
|-----------|-------|-------|--------------|
| addr | 4 | | IP address |
| mask | 4 | | Subnet mask |
| gw | 4 | | Gateway |
| dnssvr1 | 4 | | DNS server 1 |
| dnssvr2 | 4 | | DNS server 2 |

Ap_config

| Parameter | Bytes | Value | Description |
|-------------|-------|--------------|---------------------------------|
| hidden | 1 | 0 | Network name is visible |
| | | 1 | Network name is hidden |
| countrycode | 3 | Country code | National code, such as China CN |

4.4.4 Enabling Automatic Connection

Command:

rak_auto_connect

Syntax:

```
uint32_t cmd;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xC6,0x00,0x00,0x00 | Command Code |

Description:

Use the saved network parameters to enable automatic networking. Automatically run internal scan, join and IP setting, and then return IP allocation results.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|-------------------------------|
| <CODE> | 2 | 0xC6,0x00 | Response Code |
| <MAC> | 6 | | MAC address |
| <ADDR> | 4 | | IP address |
| <MASK> | 4 | | Subnet mask |
| <GW> | 4 | | Gateway |
| <DNS1> | 4 | | DNS Server 1 |
| <DNS2> | 4 | | DNS Server 2 |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Specified SSID is not found |
| | | -3 | Failed to join router |
| | | -4 | Failed to allocate IP address |

4.4.5 Starting Web Server

Command:

```
rak_start_web
```

Syntax:

```
uint32_t cmd;
```

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xC7,0x00,0x00,0x00 | Command Code |

Description:

It is used to start the embedded WEB service. Module will start the WEB with default parameters, typically in AP mode. When user is added, user can use the browser to configure the module parameters for wireless modules or wireless firmware upgrade.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|-----------|--|
| <CODE> | 2 | 0xC7,0x00 | Response Code |
| <STATUS> | 1 | 0 | Configuration is successful or upgrade is successful |
| | | -2 | Configuration timeout |

4.4.6 Getting Saved Network Parameters

Command:

rak_get_storeconfig

Syntax:

uint32_t cmd;

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xb9,0x00,0x00,0x00 | Command Code |

Description:

This command is used to save network parameters.

Return Value:

| Parameter | Bytes | Value | Description |
|------------------|-------|-----------|---------------------|
| <CODE> | 2 | 0xb9,0x00 | Response Code |
| <feature_bitmap> | 4 | | Featured parameters |
| <net_type> | 1 | 0 | Station |
| | | 1 | Ap |
| | | 2 | Ad-hoc |

| | | | |
|-------------|----|------|----------------------------|
| <CHANNEL> | 1 | 0-13 | Channel |
| <sec_mode> | 1 | 0 | Network is not encrypted |
| | | 1 | Network is encrypted |
| <dhcp_mode> | 1 | 0 | STA:DHCP client |
| | | 1 | STA:ip static |
| <ssid> | 33 | | Network identifier |
| <psk> | 65 | | Network key |
| <DUMMY> | 2 | | Null data |
| <ip_param> | 20 | | IP parameters |
| <ap_config> | 4 | | AP parameters |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed, or timeout |

Related parameters :

Ip_param:

| parameter | Bytes | Value | Description |
|-----------|-------|-------|-------------|
| addr | 4 | | IP address |
| mask | 4 | | Subnet mask |
| gw | 4 | | Gateway |
| dnssvr1 | 4 | | DNS serve 1 |
| dnssvr2 | 4 | | DNS serve 2 |

Ap_config

| parameter | Bytes | Value | Description |
|-------------|-------|--------------|------------------------------------|
| hidden | 1 | 0 | Network name is visible |
| | | 1 | Network name is hidden |
| countrycode | 3 | Country code | National code, such as China CN |

4.4.7 Getting Web Server Built-in Parameters

Command:

Rak_get_webconfig

Syntax:

uint32_t cmd;

Parameter:

| Parameter | Bytes | Value | Description |
|-----------|-------|---------------------|--------------|
| cmd | 4 | 0xba,0x00,0x00,0x00 | Command Code |

Description:

This command is available to save network parameters.

Return Value:

| Parameter | Bytes | Value | Description |
|-----------|-------|--------------------|----------------------------|
| <CODE> | 2 | 0xba,0x00 | Response Code |
| params | 132 | | Network parameters |
| User_name | 17 | | User name |
| User_psk | 17 | Network encryption | Password |
| <STATUS> | 1 | 0 | Command successful |
| | | -2 | Command failed, or timeout |

Related parameters :

Param

| Parameter | Bytes | Value | Description |
|----------------|-------|---------------------|------------------------------|
| feature_bitmap | 4 | 0x00,0x00,0x00,0x00 | Characteristic switch |
| net_type | 1 | 0 | Station |
| | | 1 | Ap |
| | | 2 | Ad-hoc |
| channel | 1 | 0-13 | Channel |
| sec_mode | 1 | 0 | The network is not encrypted |
| | | 1 | Network encryption |
| dhcp_mode | 1 | 0 | STA:Dhcp client |
| | | 1 | STA:ip static |
| ssid | 33 | ASCII | Network identifier |
| psk | 65 | ASCII | Network key |
| dummy | 2 | | Null data |
| ip_param | 20 | | IP parameter |
| ap_config | 4 | | AP advanced parameters |

Ip_param:

| parameter | Bytes | Value | Description |
|-----------|-------|-------|-------------|
| addr | 4 | | IP address |
| mask | 4 | | Subnet mask |
| gw | 4 | | Gateway |
| dnssvr1 | 4 | | DNS serve 1 |
| dnssvr2 | 4 | | DNS serve 2 |

Ap_config

| parameter | Bytes | Value | Description |
|-------------|-------|-------------|------------------------------------|
| hidden | 1 | 0 | Network name is visible |
| | | 1 | The network name is not visible |
| countrycode | 3 | Countrycode | National code, such as China CN |

5 Sales and Service

Shanghai

FAE mailbox: allan.jin@rakwireless.com

Tel : 021- 61553990

Fax : 021- 54721038

Address: Room B205, Green light kechuang garden, 2588 Lane, Hongmei South road,
Minhang District, Shanghai

Shenzhen

FAE mailbox: steven.tang@rakwireless.com

Tel : 0755-26506594

Fax: 0755-86152201

Address: Room 802, Yongfu building, No.1s06, Yongfu road, Baoan District , Shengzhen

6 Revision History

| Version | Author | Modification | Date |
|---------|------------|---|------------|
| V1.0 | | Initial Draft | 2014-03-11 |
| V1.1 | | Modify draft and release | 2014-03-28 |
| V1.2 | | Modify command returns status, data structure | 2014-06-09 |
| V1.3 | | Update the contact way, Update the document format | 2014-08-22 |
| V1.4 | | 1. Add a new command rak_tcpserver_status (check TCP server connections) and description 2. Modify Easyconfig, WPS Command Description 3. Modify Itcp command description; 8-11 4. Modify Iudp Command Description 5. Modify Multicast Command Description 6. Modify the closed SOCKET command description; 0-11 7. Remark Network Status (4.3.9) | 2014-09-17 |
| V1.5 | | Modify the frame format description, increase the upgrade firmware command, modify some of the structure | 2015-05-20 |
| V1.6 | | Initial Draft | 2015-06-01 |
| V1.7 | Shi Feifei | Update the contact way, Update the document format | 2016-03-02 |