# Protocal between ESP WIFI and Host MCU V1.0

Content

[Protocal between ESP WIFI and Host MCU V1.0 1](#_Toc56267911)

[Communication data frame format 2](#_Toc56267912)

[ESP WIFI to Host MCU 2](#_Toc56267913)

[Upload network configurations and firmware versions 2](#_Toc56267914)

[Upload Gcode commands 3](#_Toc56267915)

[Upload file first fragment 4](#_Toc56267916)

[Upload file data fragments 5](#_Toc56267917)

[Upload list of searched WiFi hotspot. 5](#_Toc56267918)

[Host MCU to ESP WIFI 6](#_Toc56267919)

[Configure the network 6](#_Toc56267920)

[Gcode reply message 7](#_Toc56267921)

[Exception information 7](#_Toc56267922)

[Configure the cloud information 7](#_Toc56267923)

[Unbind the user to wifi module 7](#_Toc56267924)

[Scan the wifi hot spots 8](#_Toc56267925)

[Wifi connect 8](#_Toc56267926)

## Communication data frame format

|  |  |  |  |
| --- | --- | --- | --- |
| **Segment** | **Offset(bytes)** | **Length(bytes)** | **Definition** |
| Head | 0 | 1 | Message start flag: 0xa5 |
| Type | 1 | 1 | Message Type, details see bellow |
| DataLen | 2 | 2 | Data length of “Data” segment, no more than 1024-5=1019 |
| Data | 4 | dataLen | Data |
| Tail | 4+dataLen | 1 | Message end flag: 0xfc |
| Padding | 5+dataLen | 1019-dataLen | **Only use on file data tranferring. Every fragment Data should be fixed 1024 bytes, so if the data is not enough, so add zeros to padding.** |

Data between Wifi and Host MCU using the packages of hex format. In both directions, using the following format:

TIPS:

1. Before ESP WiFi send the message to host MCU, it should wait the IO4 to be low level, it hints that the host MCU is ready to receive.
2. When the other side receive the message, there is not necessary to reply, except for the Gcode command.

## ESP WIFI to Host MCU

### Upload network configurations and firmware versions

**Type = 0x00**

**Data:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Segment** | **Offset**  **(bytes)** | **Length**  **(bytes)** | **Definition** |
| IP | 0 | 4 | LAN IP, Eg,192.168.1.123 |
| Port | 4 | 2 | Port of cloud service |
| State | 6 | 1 | Network state: 0x0a: link ok  0x05：link fail 0x0e: exception（module ID not valid) |
| Mode | 7 | 1 | Network mode： 0x01:AP 0x02:Client 0x03:AP+Client（not ready now） |
| Wifi\_name\_len | 8 | 1 | Length of wifi name |
| Wifi\_name | 9 | Wifi\_name\_len | Wifi name string, no more than 32characters |
| Wifi\_key\_len | 9+wifi\_name\_len | 1 | Length of wifi password |
| Wifi\_key | 10+wifi\_name\_len | Wifi\_key | Wifi password string, no more than 64 characters |
| Cloud\_state | 10+wifi\_name\_len+wifi\_key | 1 | Cloud link state： 0x12: cloud enable，connected，user binded  0x11: cloud enable，connected，not binded  0x10：cloud enable，not connected  0x00: cloud disable |
| Cloud\_host\_len | 11+wifi\_name\_len+wifi\_key\_len | 1 | Length of Cloud\_host |
| Cloud\_host | 12+wifi\_name\_len+wifi\_key\_len | Cloud\_host\_len | Url of cloud server, no more than 96 characters |
| Cloud\_port | 12+wifi\_name\_len+wifi\_key\_len+cloud\_host\_len | 2 | Port of cloud server |
| Id\_len | 14+wifi\_name\_len+wifi\_key\_len+cloud\_host\_len | 1 | Length of Id\_string |
| Id\_string | 15+wifi\_name\_len+wifi\_key\_len+cloud\_host\_len | id\_len | ID string of the wifi module, using to be identified by the cloud server |
| Ver\_len | 15+wifi\_name\_len+wifi\_key\_len+cloud\_host\_len+id\_len | 1 | Length of the version of wifi firmware |
| Ver\_string | 16+wifi\_name\_len+wifi\_key\_len+cloud\_host\_len+id\_len | ver\_len | Wifi firmware version string |

**Reply: no**

**This message should be uploaded every 10s**

### Upload Gcode commands

**Type = 0x01**

**Data=$Gcode Comand String + ”\r\n”**

**Reply = “ok\r\n” (Every command will be replied with “ok”, and also addition necessary information.)**

There also some customize commands of mks wifi, which maybe not the command commands with Marlin:

**M20 xxx**：Get the printer file list, and the Host MCU will return the gcode files and folders under the file path of the “xxx” layer (the tentative character string does not exceed 1024 bytes). If it is a folder, it will automatically add the .DIR suffix. The following format is returned:

*Begin file list*

*abc.gcode*

*123.gcode*

*temp.DIR*

*End file list*

**M23 xxx.gcode**：select the gcode file

**M24**：If printing has not started, this command is to start printing the file, if printing is suspended, this command is to resume printing

**M25**: pause printing

**M26**：stop printing

**M27**：To get the printing progress, Host MCU would reply with this format: "M27 30\r\n" (30%)

**M991**：like M105, get the temperature. Host MCU would reply as this format：”T:%d /%d B:%d /%d T0:%d /%d T1:%d /%d @:0 B@:0\r\n”

**M992**：Get the time has print, Host MCU would reply as this format：”M992 10:30:20\r\n”

**M994**： Get the name and size of the file being printing, Host MCU would reply as this format: “M994 abc.gcode;size”（size means the actually size in bytes of file）

**M997**：Get the current printer state, Host MCU would reply according different states:

IDLE: “M997 IDLE\r\n”

PRINTING: “M997 PRINTING\r\n”

PAUSED: “M997 PAUSE\r\n”

**M998**: switch the filesystem to USB or Sdcard

**M115**: Get the firmware information of mainboard, Host MCU would reply in this format: “FIRMWARE\_NAME:MKS Robin Nano V2.1.0\r\n”

**Reply: according to above**

### Upload file first fragment

**Type = 0x02**

**Data:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Segment** | **Offset**  **(bytes)** | **Length**  **(bytes)** | **Definition** |
| fileNameLen | 0 | 1 | The length of filename to be transferred, no more than 255 |
| fileLen | 1 | 4 | The length of file to be transferred, no more than 2^32 |
| fileName | 5 | fileNameLen | The name string of file to be transferred |

**Reply: no**

### Upload file data fragments

**Type = 0x03**

**Data:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Segment** | **Offset**  **(bytes)** | **Length**  **(bytes)** | **Definition** |
| fragment | 0 | 4 | The file fragment number, begin from 0. 0~14th bits indicate the fragment number, the 15th bit indicates whether it is the last fragment. |
| fileData | 4 | Length of fragment file data | The raw file data of this fragment. The length should be the file data actually read out. |

**Reply: no**

### Upload list of searched WiFi hotspot.

**Type = 0x04**

**Data:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Segment** | **Offset**  **(bytes)** | **Length**  **(bytes)** | **Definition** |
| hot spotNum | 0 | 1 | Number of scanned WiFi hotspot, no more than 20 |
| hot spot Len1 | 1 | 1 | Length of 1st hot spot name, no more than 32 |
| hot spot Str1 | 2 | hot spot Len1 | 1st Hotspot Name String |
| hot spot Rssi1 | hot spot Len1+2 | 1 | 1st hot spot signal Strength, It’s a negative number, the higher it is, the higher the intensity. |
| hot spot Len2 | hot spot Len1+3 | 1 | Length of 2nd hot spot name, no more than 32 |
| hot spot Str2 | hot spot Len1+4 | hot spot Len2 | 2nd Hotspot Name String |
| hot spot Rssi2 | hot spot Len1+hot spot Len2+4 | 1 | 2nd hot spot signal Strength, It’s a negative number, the higher it is, the higher the intensity. |
| hot spot LenN | hot spot Len1+hot spot Len2+……+hot spot Len(N-1)+2\*N-1 | 1 | Length of the “N” hot spot name, no more than 32 |
| hot spot StrN | hot spot Len1+hot spot Len2+……+hot spot Len(N-1)+2\*N | hot spot LenN | The “N” Hot spot Name String |
| hot spot RssiN | hot spot Len1+hot spot Len2+……+hot spot LenN+2\*N | 1 | The “N” hot spot signal Strength, It’s a negative number, the higher it is, the higher the intensity. |

**Reply: no**

**When the wifi module receive the “scan wifi hotspots” command, it will make the scanning and feekback this message.**

## Host MCU to ESP WIFI

### Configure the network

**Type = 0x00**

**Data:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Segment** | **Offset**  **(bytes)** | **Length**  **(bytes)** | **Definition** |
| mode | 0 | 1 | 0x01:AP 0x02:Client 0x03:AP+Client(not used yet) |
| wifi\_name\_len | 1 | 1 | Length of wifi name |
| wifi\_name | 2 | wifi\_name\_len | Wifi name string, no more than 32 characters |
| wifi\_key\_len | 2+wifi\_name\_len | 1 | Length of wifi password |
| wifi\_key | 3+wifi\_name\_len | wifi\_key | Wifi password string, no more than 64 characters |

**Reply: no**

### Gcode reply message

**Type = 0x02**

**Data: the raw string to be replied**

**Reply: no**

**When Host MCU receive gcode from esp wifi, it replies “ok\r\n”, and also the file list…using this type of message**

### Exception information

**Type = 0x03**

**Data:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Segment** | **Offset**  **(bytes)** | **Length**  **(bytes)** | **Definition** |
| Exception code | 0 | 1 | 0x01: file transfer error 0x02: file transfer ok  Just now only these two codes |

**Reply: no**

### Configure the cloud information

**Type = 0x04**

**Data:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Segment** | **Offset**  **(bytes)** | **Length**  **(bytes)** | **Definition** |
| cloud\_enable | 0 | 1 | 0x0a: enable the cloud service 0x05: disable the cloud service |
| cloud\_host\_len | 1 | 1 | Length of Cloud\_host |
| cloud\_host | 2 | cloud\_host\_len | Url of cloud server, no more than 96 characters |
| cloud\_port | cloud\_host\_len+2 | 2 | Port of cloud server |

**Reply: no**

### Unbind the user to wifi module

**Type = 0x05**

**Data: no data**

**Reply: no**

### Scan the wifi hot spots

**Type = 0x06**

**Data: no**

**Reply: no**

### Wifi connect

**Type = 0x09**

**Data:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Segment** | **Offset**  **(bytes)** | **Length**  **(bytes)** | **Definition** |
| Operation code | 0 | 1 | 0x01: Connect to the wifi hotspot being configured(The WiFi module should determines that it is not connected and takes action) 0x02: Disconnect linking current WiFi hotspot  0X3: Forget the password of current Wifi |

**Reply: no**