Machine Translated by Google

WM_WiFi_free communication interface usage guide

V0.2

WinnerMicro

Beijing Lianshengde Microelectronics Co., Ltd. (winner micro)

Address: Floor 18, Yindu Building, No. 67, Fucheng Road, Haidian District, Beijing

Tel: +86-10-62161900

Company website: www.winnermicro.com

Machine Translated by Google

Document modification record

Version revision time revision record			author review	
V0.1	2019/9/25	[C] Create document	Lilm	
V0.2	2020/7/8	unified font	Cuiyc	

WinnerMicro

Machine Translated by Google

Table of contents

Documentation Modification History 2

Table of contents..... 3

1 Introduction..... 4

1.1 Purpose of writing..... 4

1.2 Intended Reader 4

1.3 Definition of Terms..... 4

1.4 References..... 4

2 Set the channel..... 5

3 Setting the monitor mode..... 5

4 Set receive callback function..... 6

5 Send frame..... 7

WinnerMicro

Machine Translated by Google

1 Introduction

1.1 Purpose of writing

Instruct W60X/W800 related developers to use WiFi free communication interface

1.2 Intended audience

All W60X/W800 related developers

1.3 Definition of terms

none

1.4 References

none

WinnerMicro

Machine Translated by Google

2 Set the channel

When W60X/W800 communicates, only modules in the same channel can communicate with each other.

```
/**
 * @brief          This function is used to change channel actively
 *
 * @param[in]      chanid
 *
 * @return         None
 *
 * @note           Normally, it is just used in listen mode;
 */
void tls_wifi_change_chanel(u32 chanid);
```

Chanid takes a range of 0-13 to represent 1-14 channels. W60X uses 2.4G frequency band, so theoretically supports 1-14 channels, but

Different countries stipulate specific channels that can be used, for example, China can only use channels 1-13.

3 Set the monitor mode

By default, W60X/W800 can only receive packets whose destination address is itself and the broadcast address, so only the W60X/W800

In listening mode, all packets in the air can be received.

Machine Translated by Google


```
/**
 * @brief          This function is used to enable/disable listen mode
 *
 * @param[in]      enable:non-zero, disable: 0
 *
 * @return         None
 *
 * @note           This function used when oneshot start.
 */
void tls_wifi_set_listen_mode(u8 enable);
```

When programming, start the listening mode by calling tls_wifi_set_listen_mode(1), and call tls_wifi_set_listen_mode(0)

Stop listening mode.

4 Set the receive callback function

W60X/W800 can report all received messages to the user for processing.

```
/** callback function of receive Wi-Fi data */
typedef void (*tls_wifi_data_recv_callback)(u8* data, u32 data_len);

/**
 * @brief          This function is used to register recv wifi data callback function
 *
 * @param[in]      callback    point to receive Wi-Fi data function
 *
 * @return         None
 *
 * @note           None
 */
void tls_wifi_data_recv_cb_register(tls_wifi_data_recv_callback callback);
```

Machine Translated by Google


```c
/** 802.11 packet information from the physical layer */
struct tls_wifi_ext_t {
    u8 rssi;    /**< signal strength */
};

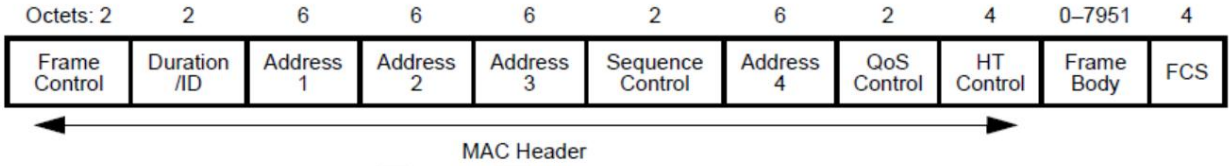
/** callback function of receive Wi-Fi data with some information of the physical layer */
typedef void (*tls_wifi_data_ext_recv_callback)(u8* data, u32 data_len, struct tls_wifi_ext_t *ext);

/**
 * @brief          This function is used to register recv wifi extended
 *                 data callback function
 *
 * @param[in]      callback   point to receive Wi-Fi extended data function
 *
 * @return         None
 *
 * @note           None
 */
void tls_wifi_data_ext_recv_cb_register(tls_wifi_data_ext_recv_callback callback);
```

The difference between these two interfaces is that tls_wifi_data_ext_recv_cb_register not only reports the content of the message, but also adds the

When programming, select a suitable interface according to your needs.

According to the 802.11 protocol, the format of a MAC header is:



The u8* data pointer in the interface points to a complete frame starting from the MAC frame header (excluding FCS content), u32 data_len

Refers to the length of the entire frame (not including the FCS length). FCS error frames will be discarded by the system and will not be reported to the user. When programming

The user needs to parse the frame content according to the protocol.

5 send frame

When users need to send ad hoc frames, they can use the following interface:

Machine Translated by Google


```c
/** transport rate and gain */
struct tls_wifi_tx_rate_t {
    enum tls_wifi_tx_rate tx_rate;    /**< Wi-Fi ransport rate */
    u8 tx_gain;                       /**< Wi-Fi ransport gain,
                                           The caller can get the maximum gain
                                           by using the tls_wifi_get_tx_gain_max function. */
};

/**
 * @brief          This function is used to send an 802.11 frame
 *
 * @param[in]      *mac         mac address, it can be NULL
 * @param[in]      *data        data packet buffer
 * @param[in]       data_len    data packet length
 * @param[in]      *tx          rate and gain, it can be NULL
 *
 * @retval         0            success
 * @retval         other        failed
 *
 * @note     If the @*mac is NULL, @*data should be an entire 802.11 frame.
 *           If the @*mac is not NULL, this function will build an 802.11 frame
 *           with @*mac as destination mac address and @*data as the data body.
 *           If the @*tx is NULL, the packet will be sent at 11B 1Mbps.
 */
int tls_wifi_send_data(struct tls_wifi_hdr_mac_t *mac, u8 *data, u16 data_len, struct tls_wifi_tx_rate_t *tx);
```

When programming, struct tls_wifi_hdr_mac_t *mac is filled with NULL, and u8 *data points to a complete 802.11

frame (excluding FCS content), u16 data_len is the length of this complete 802.11 frame (excluding FCS length),

FCS is automatically calculated by hardware and appended at the end of the frame.

struct tls_wifi_tx_rate_t *tx specifies the packet sending rate and sending gain, and the rate value is filled in according to the enumerated value.

Can,

```c
/**
 * @brief          This function is used to get max gain by rate index
 *
 * @param[in]      tx_rate    rate index (enum)
 *
 * @retval         Max Gain
 *
 * @note           None
 */
u8 tls_wifi_get_tx_gain_max(enum tls_wifi_tx_rate tx_rate);
```

The range of gain value is 0x60-0x78, it is recommended that users directly use tls_wifi_get_tx_gain_max to obtain the corresponding speed

Fill in the gain value of the rate.