MBL Series Evaluation Kit User Manual

New Generation Package Compatible Sub-1G Wireless Module

E32-900MBL-01

成都亿佰特电子科技有限公司
Chengdu Ebyte Electronic Technology Co.,Ltd.

Disclaimer and Copyright Notice

The information in this document, including the URL address for reference, is subject to change without notice. this document is provided "as is" without warranty of any kind, including any warranties of merchantability, fitness for a particular purpose or non-infringement, and any warranties referred to elsewhere in any proposal, specification or sample. this document disclaims all liability, including liability for infringement of any patent rights arising out of the use of the information in this document. No license, express or implied, to use any intellectual property is granted herein by estoppel or otherwise.

All test data in this document are from EBYTE'S laboratory tests and actual results may vary slightly.

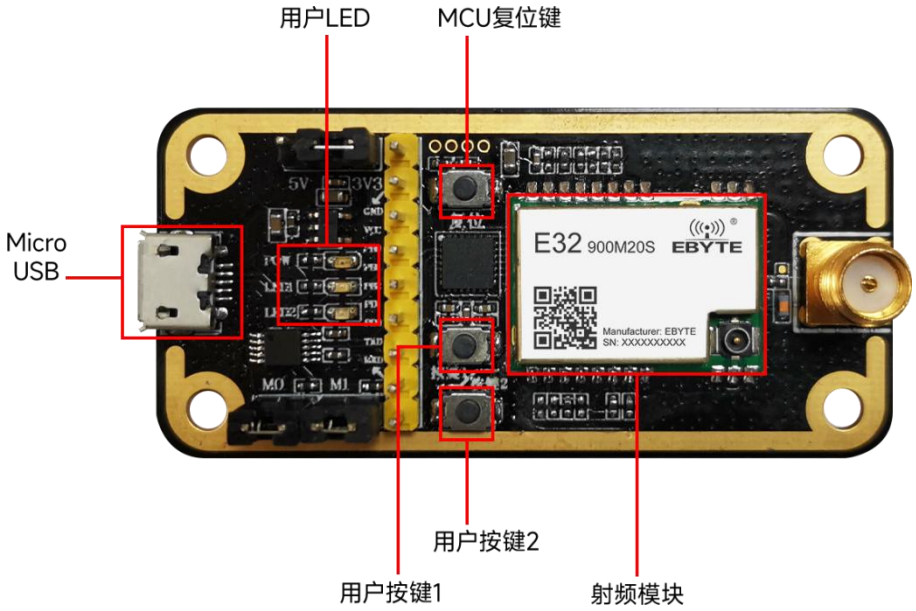
All trade names, trademarks and registered trademarks mentioned herein are the property of their respective owners and are hereby declared.

The final right of interpretation belongs to Chengdu Ebyte Electronic Technology Co.,Ltd.

Notice:

The contents of this manual are subject to change due to product version upgrade or other reasons.Chengdu Ebyte Electronic Technology Co.,Ltd. reserves the right to make changes to the contents of this manual without any notice or prompting. Chengdu Ebyte Electronic Technology Co.,Ltd. makes every effort to provide accurate information in this manual, but does not warrant that the contents of this manual are error-free, nor do all statements, information and recommendations in this manual constitute any express or implied warranty.

Ⅰ Product Introduction

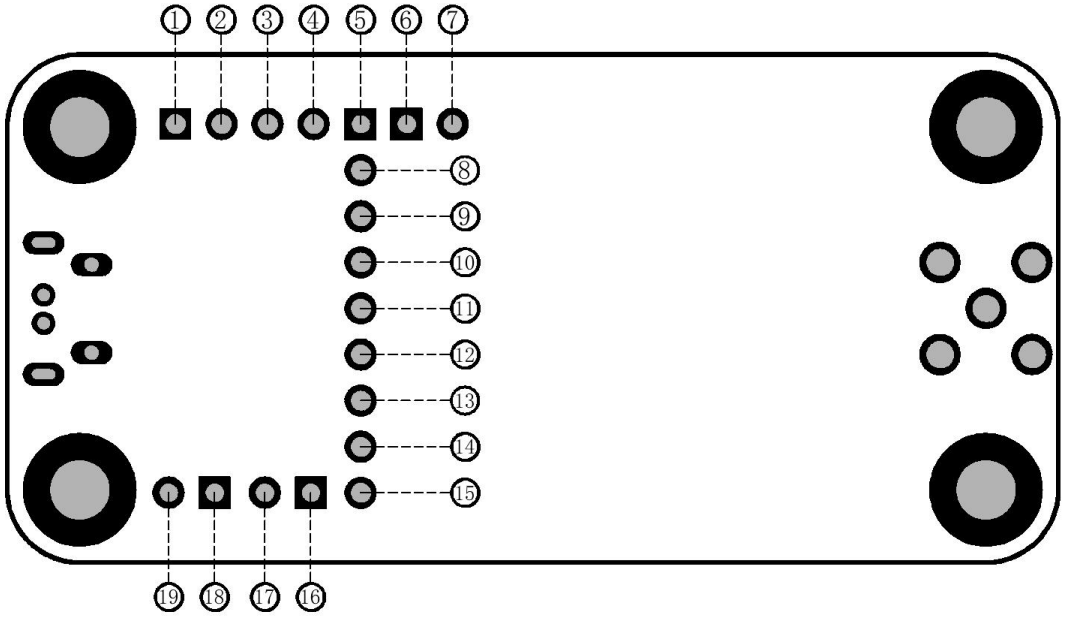


1.1 Product Description

The MBL series evaluation kits are designed to help users quickly evaluate the new generation of package-compatible wireless modules from EBST. Most of the pins on the board are already pinned out to both sides of the row of pins, so developers can easily connect a variety of peripheral devices through jumpers according to their actual needs.

The kit provides complete software application examples to help customers quickly get started with wireless data communication development. Different types of Sub-1G wireless modules can be on-board according to customer requirements. The supported modules are available in pin-compatible packages and can be quickly replaced.

1.2 Size, interface description



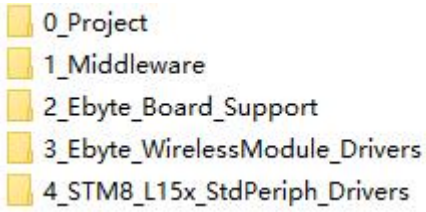
Pin Serial Number	Definition	Function Description
1	VCC	Module power supply pin, need to short with pin 2 to power the module
2	3.3V	3.3V electrical lead pin
3	3.3V	3.3V electrical lead pin
4	VIO	MCU power supply pin, need to short with pin 3 to power MCU
5	GND	Base plate reference ground
6	REST	MCU external reset pins
7	SWIM	SWIM pins of MCU
8	VIO	MCU power supply pins
9	PC0	Module reset pins
10	PB7	Module MISO Pinout
11	PB6	Module MOSI Pinout
12	PB5	Module SCLK pins
13	PB4	Module NSS Pinout
14	TXD	MCU serial port TXD
15	RXD	MCU serial port RXD
16	M1	Module mode switching pins (see module product manual for details)
17	GND	Base plate reference ground
18	M0	Module mode switching pins (see module product manual for details)
19	GND	Base plate reference ground

1.3 Support List

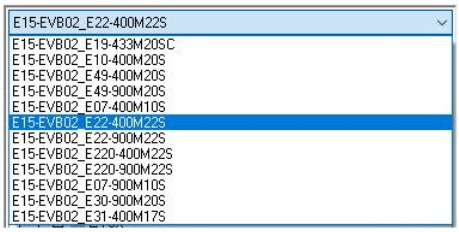
	RF chip	Manufacturer	Module Model
1	CC1101	Texas Instruments	E07-400M10S
2	CC1101	Texas Instruments	E07-900M10S
3	SI4438	Silicon Labs	E30-400M20S
4	SI4463	Silicon Labs	E30-900M20S
5	LLCC68	Semtech	E220-400M22S
6	LLCC68	Semtech	E220-900M22S
7	SX1278	Semtech	E32-400M20S
8	SX1276	Semtech	E32-900M20S
9	SX1268	Semtech	E22-400M22S
10	SX1262	Semtech	E22-900M22S
11	AX5243	ON Semiconductor	E31-400M17S
12	LLCC68	Semtech	E220-400MM22S
13	LLCC68	Semtech	E220-900MM22S

Ⅱ Software Introduction

2.1 Catalog Structure

	Matters	Description
1	File Directory	You can download the sample project from the official website and open the directory as shown below  0_Project 1_Middleware 2_Ebyte_Board_Support 3_Ebyte_WirelessModule_Drivers 4_STM8_L15x_StdPeriph_Drivers
2	Catalog Description	You can use the IAR For STM8 development environment to find the entry file to open the project **E15-EVB02 Demo** //主文件夹 **0_Project** IAR_for_Stm8 //工程文件夹 使用 IAR 打开工程 **1_Middleware** Kfifo //通用数据队列 Produce //PC测试 **2_Ebyte_Board_Support** E15-EVB02 //板载资源初始化 **3_Ebyte_WirelessModule_Drivers** E07xMx //E07模块驱动 E10xMx //E10模块驱动 E19xMx //E19模块驱动 E22xMx //E22模块驱动 E30xMx //E30模块驱动 E31xMx //E31模块驱动 E49xMx //E49模块驱动 E220xMx //E220模块驱动 **4_STM8_L15x_StdPeriph_Drivers**

2.2 IAR Engineering

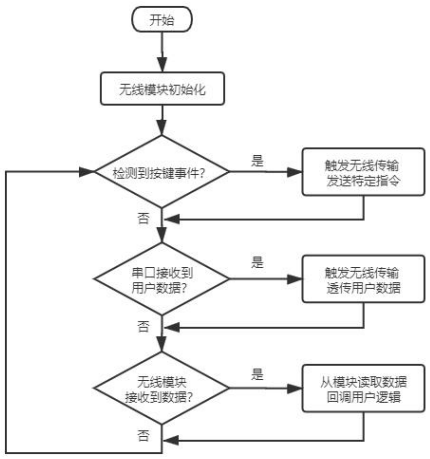
	Matters	Description
	Engineering Structure	Using the IAR For STM8 development environment, you can open the project and see the basic structure 
	Switching workspaces	Global macro definitions and file paths are defined in the C/C++ Compiler options to distinguish between the driver files of different modules. When switching workspaces, different macro definitions will be used, thus switching the driver files of different modules  Changed the Exclude from build attribute of Drivers/Ebyte/RF to select the target module driver folder for the compilation process. Changed Additional include in Project C/C++ Compiler, which specifies the module driver file path. Changed Defined symbols in Project C/C++ Compiler to define global macro definitions to help configure module driver properties. 

2.3 Main Functions

main.c is the main function entry. The flow of the demo function is simplified as follows.

	Matters	Description
1	Key Function	If a key is pressed, the command data is sent wirelessly. In essence, it is sending a specific string "ping" and expecting to receive a response "pong".
2	Serial data to wireless transmission	After the serial port receives the data, it automatically starts to transmit the data wirelessly, which of course contains some special command responses, mainly for special tests, which the user can ignore. After the transmission is completed, it will automatically call back the user function and thus handle the transmission logic by itself.
3	Receive data wirelessly	Generally it reads the module internal state marker to determine if there is data, and the underlying driver will copy the data and pass it to the user callback function, thus handling the receiving logic itself

The software process is simplified as shown in the following diagram.



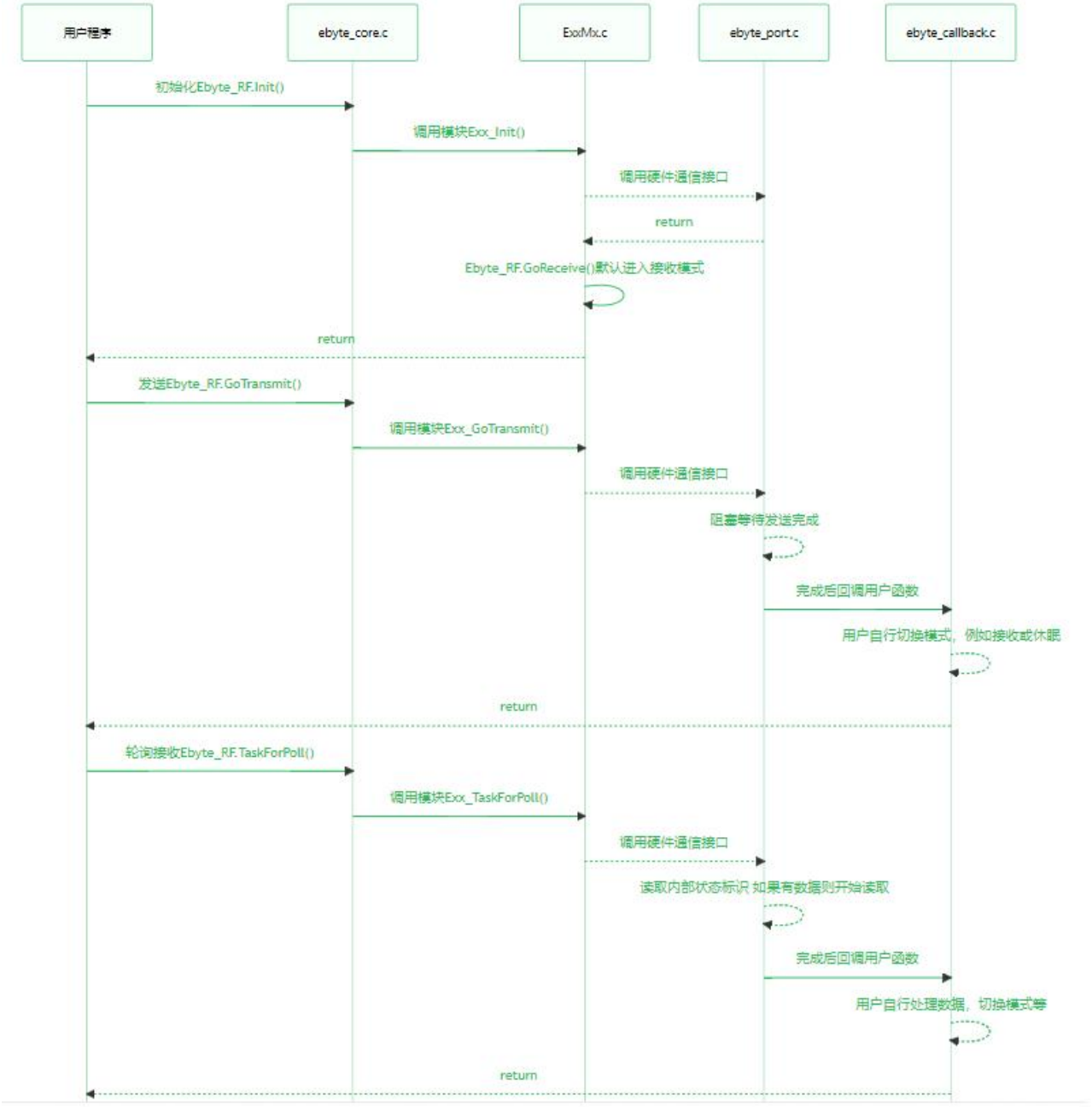
2.4 Sending and Receiving Timing

Wireless modules exist in multiple operating states and can only perform specific functions in the corresponding states. In terms of the simplest sending and receiving data, only the transmitting mode and receiving mode are considered.

	Matters	Description
1	Receiving Mode	It automatically enters receive mode after the default initialization is completed. In essence, this means that the receive function is called during initialization and thus enters receive mode. If you want to consider entering other modes after initialization, such as sleep, you can directly replace the function Go_xxxxx() with the same type.
2	Sending Mode	When the transmit function is called, the underlying driver actually switches the module into standby mode first, where the configuration of modulation parameters, such as frequency, power, frequency bias, etc., is usually done. After the parameters are configured correctly, it gradually enters some intermediate

		modes, turning on the internal FIFO, PA, external XTAL, etc., and the current consumption gradually climbs. Finally switch into transmit mode to trigger wireless data transmission. After completion, the module enters standby mode, this state can not continue to send and receive, and the user needs to handle the next mode in the callback function by himself. When the function is complicated and continuous receiving or continuous transmitting is needed, please further switch to other modes according to the chip characteristics.

The timing diagram is shown below.

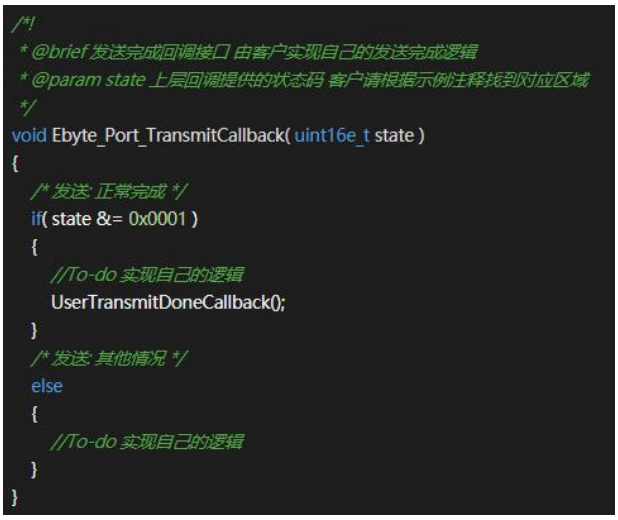


2.5 Programming

	Documents	Key Notes
1	ebyte_core.h	A module structure is defined that abstracts the basic functionality to which the functions of the underlying module will be bound. When used for simple sending and receiving applications, it is not necessary to understand the underlying details of each module's work, and the abstracted functions can be called directly to start sending and receiving data. If you need to customize some functions, you can also consider integrating them into this structure. If you know enough about the functions of the underlying modules, you can also remove the ebyte_core.c/h file, and there is no strong coupling between the layers.
2	ebyte_exx.c	It is a specific module driver file, which is generally encapsulated and does not require user changes, only how to input and output data from this "box".
3	ebyte_port.c	Specifically designed to bind SPI and GPIO on different hardware platforms, abstracted as "box" inputs. Users need to populate their hardware platforms with communication interfaces in fixed locations according to the comments. In general, it is to provide the SPI send/receive function and the pin level control. Some modules are slightly special, such as the E49 using half-duplex SPI, if you are too lazy to write the communication driver, then directly bind the IO to a fixed location, and leave the rest to the module driver to simulate their own IO to achieve communication. As shown in the figure below, in the comments required to provide the SPI interface location to fill in the specific send and receive functions, from send to pass the SPI send data, by the result to return the SPI receive data.
	ebyte_callback.c	It is specifically designed to bind the user's own sending and receiving logic, abstracted as the output of a "box". Essentially, the module driver calls the user's callback function directly after determining whether sending or receiving is complete.

```
typedef struct
{
    uint8_t   ( *Init )( void );  //初始化
    uint8_t   ( *GoTransmit )( uint8_t *buffer, uint8_t size ); //切换发送模式 开始传输数据
    uint8_t   ( *GoSleep )( void );     //切换到休眠模式 低功耗用到
    uint8_t   ( *GoReceive )( void );  //切换到接收模式 开始监听数据
    uint8_t   ( *TaskForPoll)( void ); //轮询函数 可以主循环周期调用 也可以视情况放入中断
    void      ( *TaskForIRQ )( void ); //暂时保留，不必使用。用于将来扩展中断收/发
    uint8_t   ( *GetStatus )( void );  //获取模块状态(软件状态机)
    uint8_t * ( *GetName ) (void );  //获取模块识别码 为字符串 例如 "E22-400M22S"
    uint8_t   ( *GetDriver ) (void );  //获取软件版本号
}Ebyte_RF_t;
```

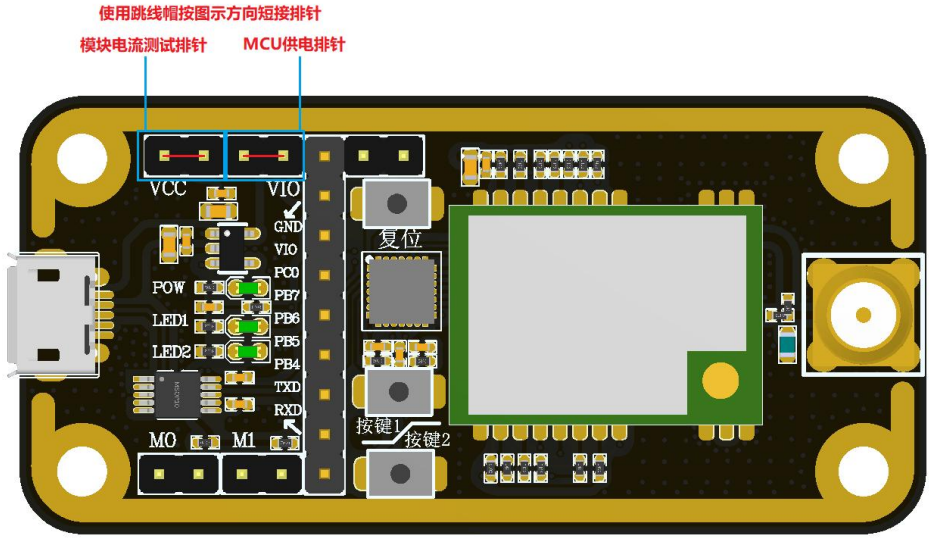
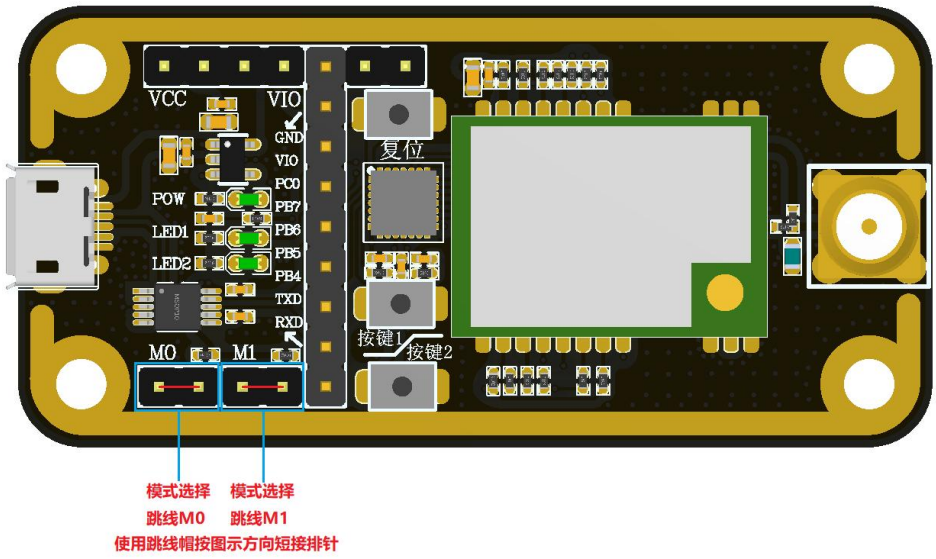
```
/*!
 * @brief 配置目标硬件平台SPI接口收发函数
 * @param send EBYTE驱动库上层调用需要传输的数据 1 Byte
 * @return SPI 接收的数据 1 Byte
 */
uint8_t Ebyte_Port_SpiTransmitAndReceivce( uint8_t send )
{
    uint8_t result = 0;

    /* 必须提供 SPI接口 */
    result = Ebyte_BSP_SpiTransAndRecv( send ); //用户填充函数

    return result;
}
```

		As shown below, the user's logic is populated in the To-do prompt position. state is passed from the module driver and actually handled by the Exx_GoTransmit() function, which can be modified to support more cases when the functionality is complex.
		 
		/*!
		* @brief 发送完成回调接口 由客户实现自己的发送完成逻辑
		* @param state 上层回调提供的状态码 客户请根据示例注释找到对应区域
		*/
		void Ebyte_Port_TransmitCallback(uint16e_t state)
		{
		/* 发送 正常完成 */
		if(state &= 0x0001)
		{
		//To-do 实现自己的逻辑
		UserTransmitDoneCallback();
		}
		/* 发送 其他情况 */
		else
		{
		//To-do 实现自己的逻辑
		}
		}

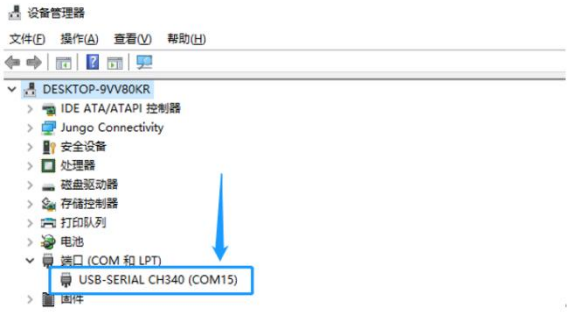
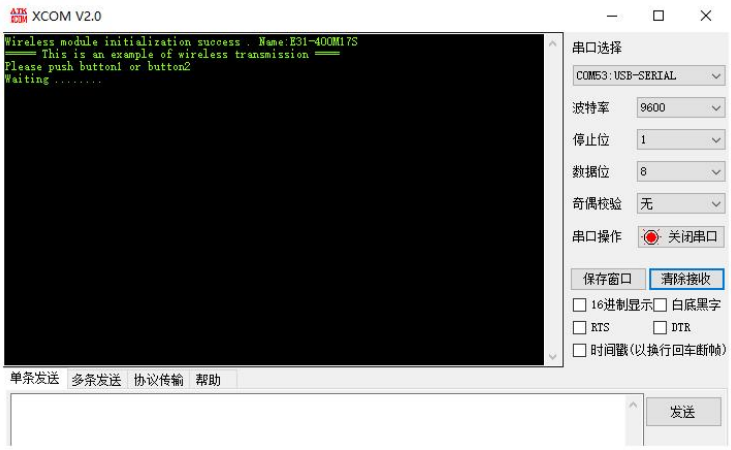
	ebyte_exx.h	Some general modulation parameters are defined, which usually do not need to be modified and can be adjusted in them by yourself. Note, please understand the comments when modifying, there is a range check for the parameters in the module driver, wrong modulation parameters will lead to initialization failure. The following is an example of FSK modulation parameters.  #define E07_DATA_RATE 1200 //空速 1.2 KBps #define E07_FREQUENCY_DEVIATION 14300 //频偏 14.3 K #define E07_BANDWIDTH 58000 //接收频宽 58 K #define E07_OUTPUT_POWER 10 //功率 [10 7 5 0 -10 -15 -20 -30] #define E07_PREAMBLE_SIZE 4 //前导码长度 [0:2 1:3 2:4 3:6 4:8 5:12 6:16 7:24] #define E07_SYNC_WORD 0x2DD4 //同步字 #define E07_IS_CRC 1 //CRC开关 [0:关闭 1:开启]
	board.c	STM8 peripheral initialization, involving SPI, TIMER, GPIO, etc., strongly coupled to the hardware used.
	board_button.c	The keystroke event queue, in terms of data structure, is a FIFO, and the timer detects the keystroke and stores the corresponding event in the queue waiting for the main loop to respond.
	board_mini_printf.c	A simplified printf, with reduced functionality but a small footprint. The DEBUG macro in the project relies heavily on the mprintf provided by this file.
	ebyte_kfifo.c	For serial data reception, optimized for general-purpose FIFO queues, suitable for caching.
	ebyte_debug.c	Used to connect to a PC for some tests, generally not required.
	stm8l15x_it.c	All interrupt functions are entered here, and the interrupt service functions for serial, timer, key IO, etc. are concentrated here.

Ⅲ Quick Demo

3.1 Signal cable connection

	Matters	Description
1	Power Jumper Cap	
2	Mode selection jumper cap	
3	Assistance	USB cable, antenna, PC, etc.

3.2 Serial Assistant

	Matters	Description
1	Device Manager View Serial Port Number	
2	Serial Software	
3	Key communication example	The #RECV identifier, used only for hints, indicates data received by the wireless module. The #SEND identifier, used only for hints, indicates data sent by the wireless module 
4	Serial Data Transit	Serial data pass-through Direct transmission of the required content via

		XCOM 

Ⅳ Frequently Asked Questions

4.1 Unsatisfactory transmission distance

- A corresponding attenuation of communication distance when linear communication barriers exist.
- Temperature, humidity, and co-channel interference, which can lead to higher communication packet loss rates.
- The ground absorbs and reflects radio waves, and the test effect is poor near the ground.
- seawater has a very strong ability to absorb radio waves, so the seaside test effect is poor.
- metal objects near the antenna, or placed in a metal shell, signal attenuation will be very serious.
- Wrong setting of power register, too high setting of air rate (the higher the air rate, the closer the distance).
- the low voltage of the power supply at room temperature is lower than the recommended value, the lower the voltage the less power is generated
- The use of antenna and module match the degree of poor or antenna itself quality problems.

4.2 Module is vulnerable to damage

- Please check the power supply to ensure that it is between the recommended supply voltages, as exceeding the maximum will cause permanent damage to the module.
- Please check the stability of the power supply, the voltage should not fluctuate significantly and frequently.
- Please ensure that the installation and use process anti-static operation, high-frequency devices electrostatic sensitivity.
- Please ensure that the installation and use process humidity should not be too high, some components are humidity-sensitive devices.
- If there is no special demand is not recommended to use at too high or too low temperature.

4.3 BER is too high

- Nearby interference with the same frequency signal, away from the source of interference or modify the frequency, channel to avoid interference.
- unsatisfactory power supply may also cause garbled codes, be sure to ensure the reliability of the power supply.
- Poor quality or too long extension cable or feeder line may also cause high BER.

Revision History

Versions	Revision Date	Revision Notes	Maintainer
1.0	2021-09-22	Initial Version	JH
1.1	2022-12-29	Modify the module diagram and how to use it	HWJ

About us

Sales Hotline: 4000-330-990 Company Phone: 028-61543675

Technical support: support@cdebyte.com Official website: https://www.ru-ebyte.com

Address: Building B5, No.199 West Avenue, High-tech West District, Chengdu, Sichuan Province

成都亿佰特电子科技有限公司
Chengdu Ebyte Electronic Technology Co.,Ltd.