MBL Series Evaluation Kit User Manual

New Generation Package Compatible Sub-1G Wireless Module

E30-400MBL-01

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

Disclaimer

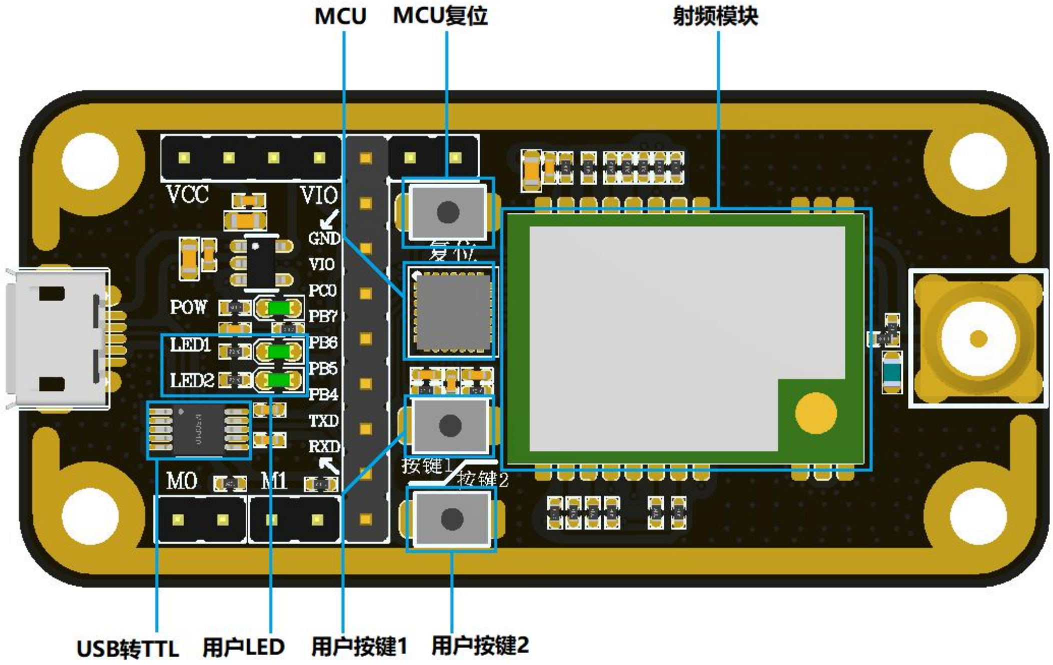
EBYTE reserves all rights to this document and the information contained herein. Products, names, logos and designs described herein may in whole or in part be subject to intellectual property rights. Reproduction, use, modification or disclosure to third parties of this document or any part thereof without the express permission of EBYTE is strictly prohibited.

The information contained herein is provided “as is” and EBYTE assumes no liability for the use of the information. No warranty, either express or implied, is given, including but not limited, with respect to the accuracy, correctness, reliability and fitness for a particular purpose of the information. This document may be revised by EBYTE at any time. For most recent documents, visit www.ebyte.com.

Note:

The contents of this manual are subject to change due to product version upgrades or other reasons. Chengdu Ebyte Electronic Technology Co., Ltd. reserves the right to make changes to the contents of this manual without notice or suggestion. This manual serves only as a user guide and Chengdu Ebyte Electronic Technology Co., Ltd. endeavours to provide accurate information in this manual, but Chengdu Billionaire Electronics Co., Ltd. does not ensure that the contents are completely error-free and that all statements, information and suggestions in this manual do not constitute any express or implied warranty.

Chapter 1 Product Overview

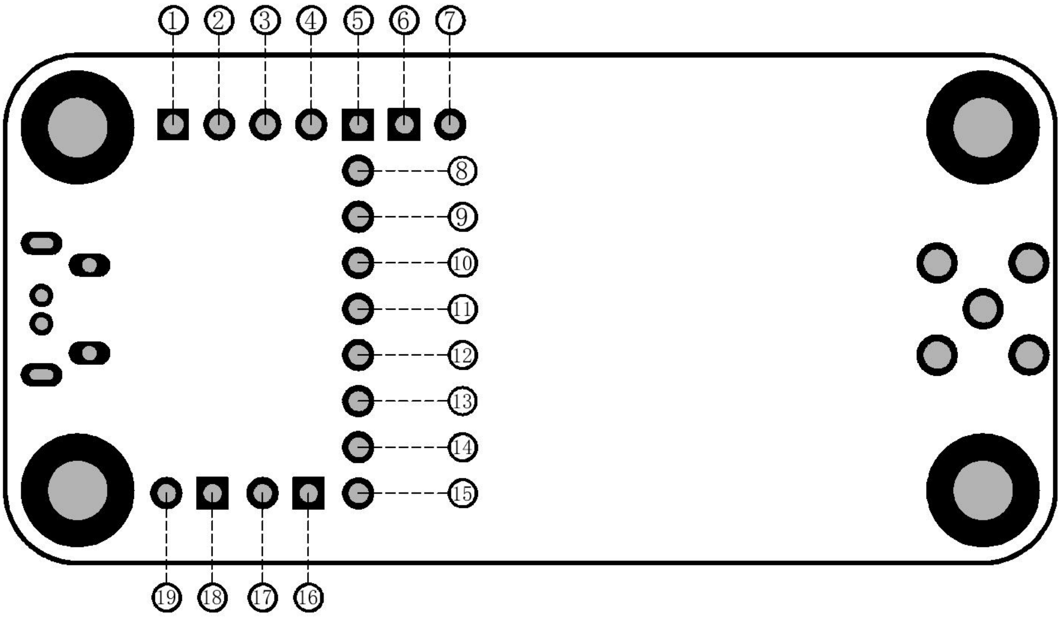


1.1 Brief introduction

MBL series evaluation kits are designed to help users quickly evaluate Ebyte's new generation of package-compatible wireless modules. Most of the pins on the board have been led out to pin headers on both sides, and developers can easily connect various peripheral devices through jumpers according to actual needs.

The kit provides complete software application examples to help customers quickly get started with wireless data communication development. According to customer needs, different types of Sub-1G wireless modules can be mounted on the board. Supported modules are available in pin-compatible packages for quick replacement.

1.2 Size and interface description



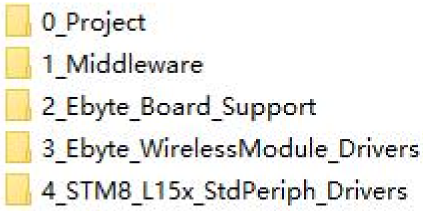
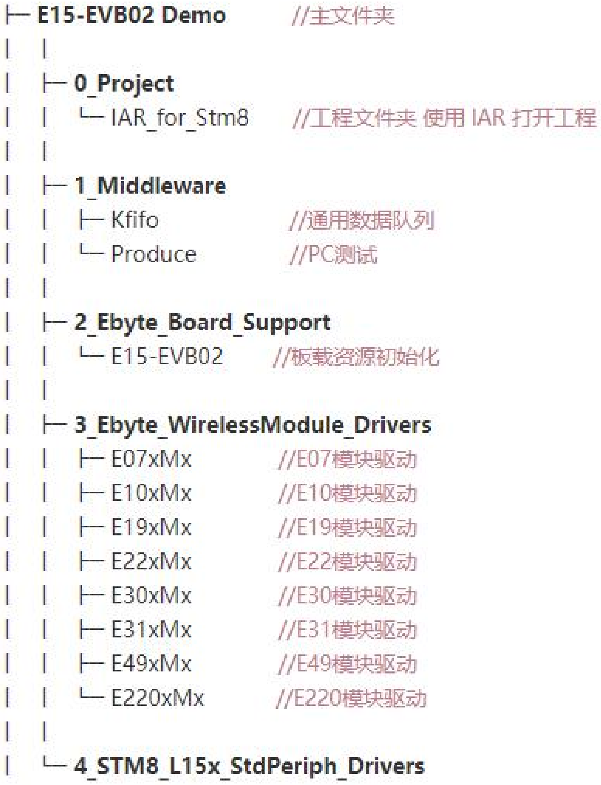
Pin number	Definition	Function Description
1	VCC	Power supply pin: it needs to be short-circuiting with pin 2
2	3.3V	3.3V power pin
3	3.3V	3.3V power pin
4	VIO	MCU power supply pin: it needs to be short-circuiting with pin 3
5	GND	Backplane reference ground
6	REST	MCU external reset pin
7	S WIM	SWIM pin of MCU
8	VIO	MCU power supply pin
9	PC0	Module reset pin
1 0	PB7	Module MISO pin
1 1	PB6	Module MOSI pin
1 2	PB5	Module SCLK pin
1 3	PB4	Module NSS pin
1 4	TXD	MCU serial port TXD
1 5	RXD	MCU serial port RXD
1 6	M1	Module mode switching pin (see module product manual for details)
1 7	GND	Backplane reference ground
1 8	M 0	Module mode switching pin (see module product manual for details)
1 9	GND	Backplane reference ground

1. 3 Support list

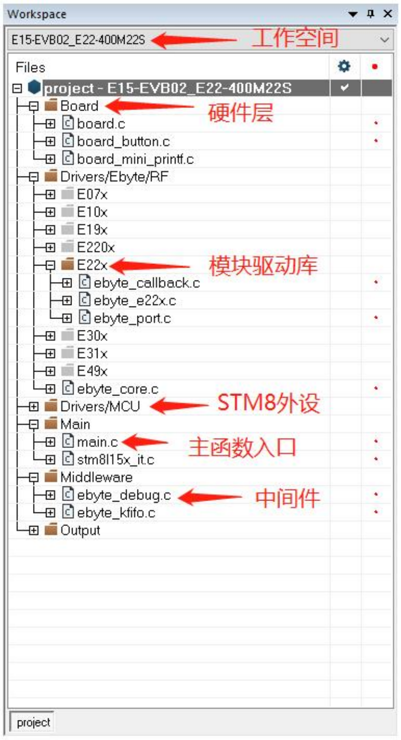
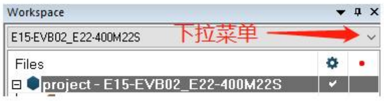
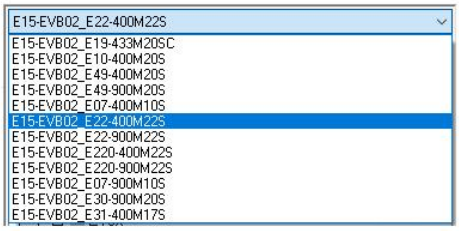
	RF solution	Manufacturer	Module model
1	C C1101	Texas Instruments	E07-400M10S
2	CC1101	Texas Instruments	E07-900M10S
3	SI4438	Silicon Labs	E30-400M20S
4	SI4463	Silicon Labs	E30-900M20S
5	LLCC68	Semtech	E220-400M22S
6	LLCC68	Semtech	E220-900M22S
7	SX1278	Semtech	E32-400M20S
8	SX1276	Semtech	E32-900M20S
9	SX1268	Semtech	E22-400M22S
10	SX1262	Semtech	E22-900M22S
11	AX5243	ON Semiconductor	E31-400M17S
12	LLCC68	Semtech	E220-400MM22S
13	LLCC68	Semtech	E220-900M M 22S

Chapter 2 Software Introduction

2.1 Directory Structure

	Matter	Explanation
1	File Directory	User can download the sample project from the official website, open the directory as shown in the figure below  0_Project 1_Middleware 2_Ebyte_Board_Support 3_Ebyte_WirelessModule_Drivers 4_STM8_L15x_StdPeriph_Drivers
2	Catalog description	User can use the I AR F or STM8 development environment to find the entry file and 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 project

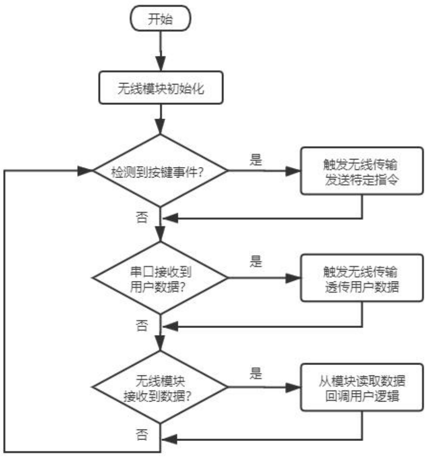
	Matter	Explanation
	Structure	Use IARF or STM8 development environment to open the project and you can see the basic structure  Workspace E15-EVB02_E22-400M22S 工作空间 Files project - E15-EVB02_E22-400M22S Board 硬件层 board.c board_button.c board_mini_printf.c Drivers/Ebyte/RF E07x E10x E19x E220x E22x 模块驱动库 ebyte_callback.c ebyte_e22x.c ebyte_port.c E30x E31x E49x ebyte_core.c Drivers/MCU STM8外设 Main main.c 主函数入口 stm8l15x_it.c Middleware ebyte_debug.c 中间件 ebyte_kfifo.c Output project
	Switch workspace	The global macro definition and file path are defined in the C/C++ Compiler option, which are used to distinguish the driver files of different modules. When switching workspaces, different macro definitions will be used to switch driver files of different modules  Workspace E15-EVB02_E22-400M22S 下拉菜单 Files project - E15-EVB02_E22-400M22S “Exclude from build” attribute of Drivers/Ebyte/RF has changed, that is, select the target module driver folder to participate in the compilation process. Changed the Additional include in the project C/C++ Compiler, that is, specifying the module driver file path. Changed the Defined symbols in the project C/C++ Compiler, that is, defined the global macro definition to help configure the module driver properties.  E15-EVB02_E22-400M22S E15-EVB02_E19-433M20SC E15-EVB02_E10-400M20S E15-EVB02_E49-400M20S E15-EVB02_E49-900M20S E15-EVB02_E07-400M10S E15-EVB02_E22-400M22S E15-EVB02_E22-900M22S E15-EVB02_E220-400M22S E15-EVB02_E220-900M22S E15-EVB02_E07-900M10S E15-EVB02_E30-900M20S E15-EVB02_E31-400M17S

2. 3 main function

The main function entry is in main.c. The demo function process is simplified as follows:

	Matter	Explanation
1	Key Function	If a button is pressed, the command data is sent wirelessly. Essentially sending a specific string "ping" and expecting a response "pong"
2	Serial data to wireless transmission	After the serial port receives the data, it automatically starts wireless transparent transmission of data. Of course, it contains some special command responses, which are mainly used for special tests and can be ignored by users. After the sending is completed, the user function will be automatically called back to process the sending logic by itself.
3	Receive data wirelessly	Generally, the internal status flag of the module is read to determine whether there is data, and the underlying driver will copy the data and pass it to the user's callback function, so as to process the receiving logic by itself

The software process is simplified as shown in the figure below:



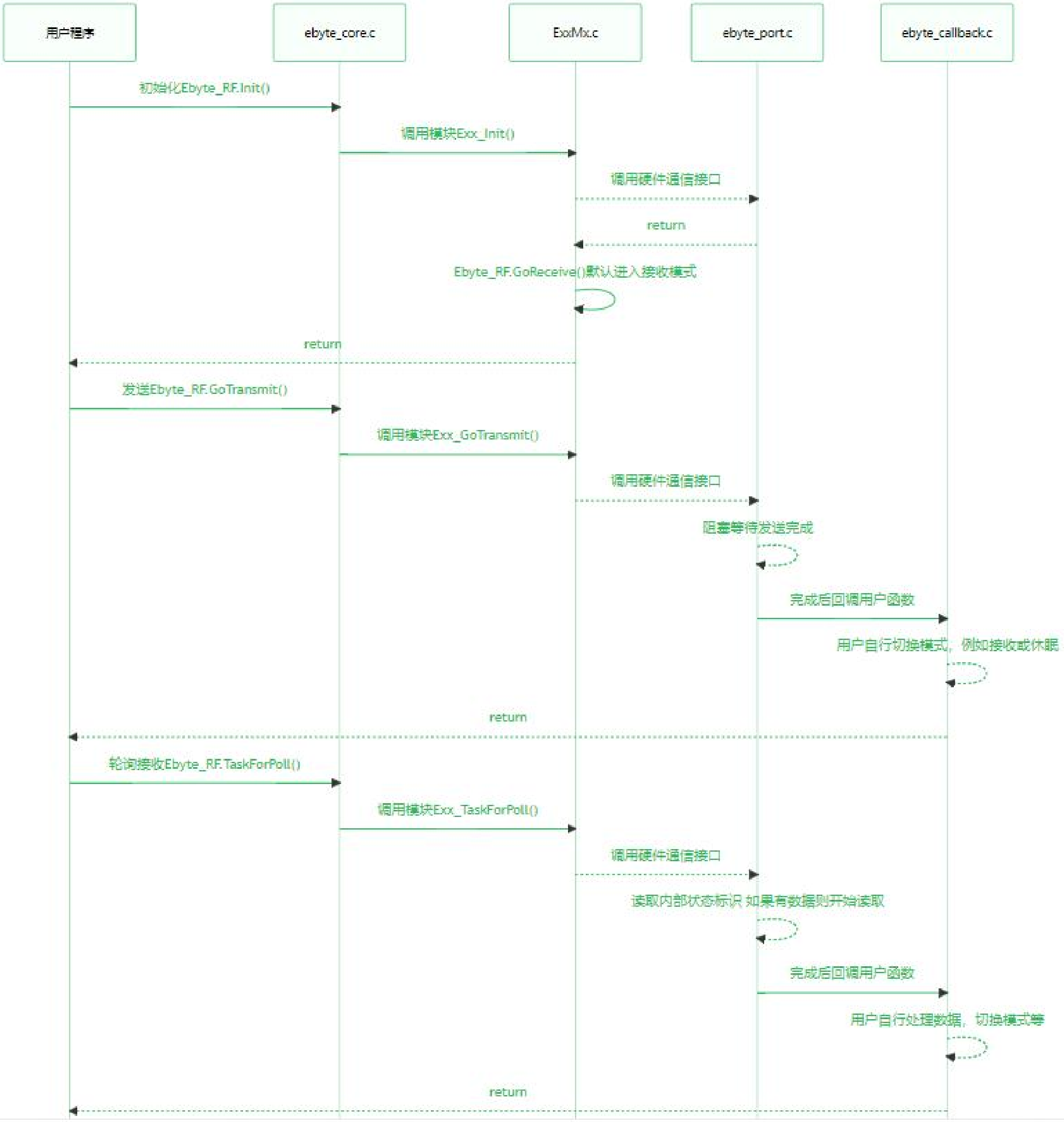
2. 4 Transceiver timing

The wireless module has multiple operating states, and specific functions can only be completed in the corresponding states. From the simplest sending and receiving data, only the sending mode and receiving mode are considered.

	Matter	Explanation
1	Receive mode	After the default initialization is completed, it automatically enters the receiving mode. In essence, the receiving function is called in the initialization to enter the receiving mode. If you need to consider entering other modes after initialization, such as sleep, you can directly replace it with the same type of function Go_xxxxx().
2	Send mode	When calling the send function, the underlying driver actually switches the module into standby mode first, and usually completes the modulation parameter configuration

		in this mode, such as frequency, power, frequency offset, etc. After the parameters are configured correctly, gradually enter some intermediate modes, open the internal FIFO, PA, external XTAL, etc., and the current consumption will gradually increase. Eventually switch into send mode, triggering wireless data transmission. After completion, the module enters the standby mode. In this state, sending and receiving cannot continue, and the user needs to handle the next step in the callback function. When the function is complex, continuous reception or continuous transmission is required, please further switch to other modes according to the characteristics of the chip.

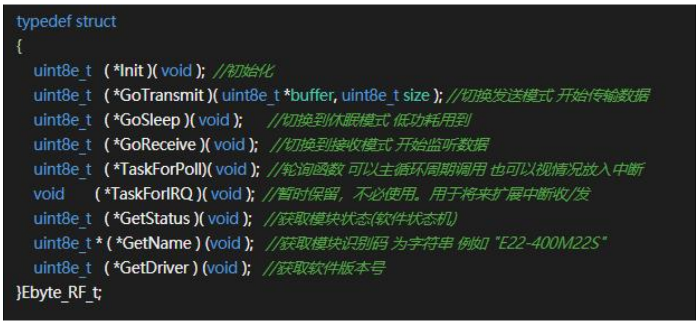
The timing diagram is as follows:



2.5 Programming

<table>
<tr><th></th><th>File</th><th>Key note</th></tr>
<tr><td>1</td><td>ebyte_core.h</td><td>

A module structure is defined, the basic functions are abstracted, and the functions of the underlying module will be bound to the structure. When used in simple sending and receiving applications, there is no need to understand the underlying working details of each module, and the data can be sent and received by directly calling the abstracted function. If you need to customize some functions, you can also consider integrating them into the structure. If you know enough about the functions of the underlying modules, you can also directly remove the ebyte_core.c/h file, and there is no strong coupling between the layers.



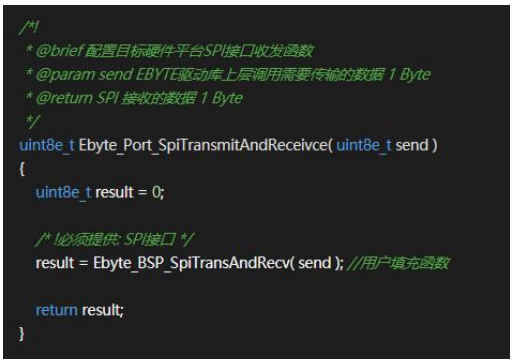
```
typedef struct
{
   uint8e_t  (*Init )( void ); //初始化
   uint8e_t  (*GoTransmit )( uint8e_t *buffer, uint8e_t size ); //切换发送模式 开始传输数据
   uint8e_t  (*GoSleep )( void );     //切换到休眠模式 低功耗用到
   uint8e_t  (*GoReceive )( void );  //切换到接收模式 开始监听数据
   uint8e_t  (*TaskForPoll)( void ); //轮询函数 可以主循环周期调用 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
   uint8e_t  (*GetStatus )( void );  //获取模块状态(软件状态机)
   uint8e_t * (*GetName ) (void );   //获取模块识别码 为字符串 例如 "E22-400M22S"
   uint8e_t  (*GetDriver ) (void );  //获取软件版本号
}Ebyte_RF_t;
```

</td></tr>
<tr><td>2</td><td>ebyte_exx.c</td><td>

It is a specific module driver file, which is generally packaged and does not need to be modified by the user. It only needs to consider how to input and output data from this "box".

</td></tr>
<tr><td>3</td><td>ebyte_port.c</td><td>

It is specially used to bind SPI and GPIO under different hardware platforms , abstracted as the input of "box". Users need to fill the communication interface in their own hardware platform to a fixed position according to the comments. Generally speaking, it is to provide the SPI transceiver function and the level control of the pin. Some modules are slightly special, for example, E49 uses half-duplex SPI, if you are too lazy to write a communication driver, you can directly bind the IO to a fixed position, and leave the rest to the module driver to simulate IO to realize communication. As shown in the figure below, in the comments, it is required to provide the SPI interface position to fill in the specific sending and receiving functions, send the data into the SPI to send data, and return the SPI received data from the result.



```
/*!
 * @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
 */
uint8e_t Ebyte_Port_SpiTransmitAndReceivce( uint8e_t send )
{
   uint8e_t result = 0;

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

   return result;
}
```

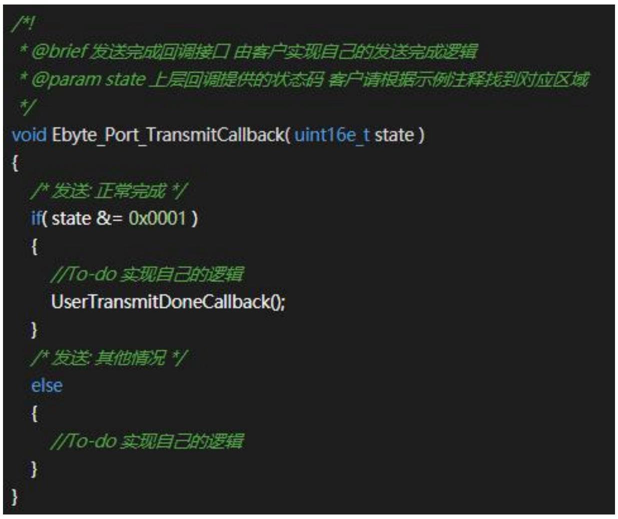
</td></tr>
<tr><td></td><td>ebyte_callback.c</td><td>

It is specially used to bind the user's own sending and receiving logic, abstracted as the output of the "box". Essentially, the module driver is to directly call the user's callback

</td></tr>
</table>

<table>
<tr>
<td></td>
<td></td>
<td>

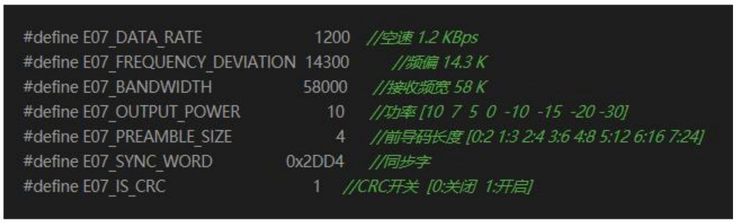
function after confirming that the sending or receiving is complete. As shown in the figure below, just fill in the user's logic function in the To-do prompt position. The state is transmitted by the module driver, and is actually processed by the Exx_GoTransmit() function. When the function is complex, it can be considered to be modified to support more situations.



```
/*!
 * @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
   }
}
```

</td>
</tr>
<tr>
<td></td>
<td>ebyte_exx.h</td>
<td>

Some conventional modulation parameters are defined, generally do not need to be modified, you can adjust them by yourself. Note, please understand the instructions in the comments when modifying. There is a range check for parameters in the module driver, and wrong modulation parameters will cause initialization failure. The following figure shows an example of FSK modulation parameters:

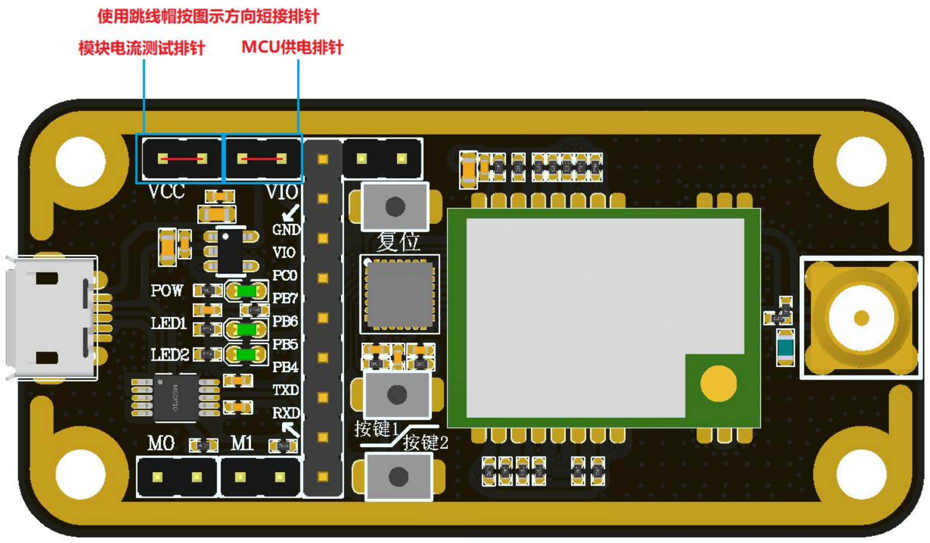
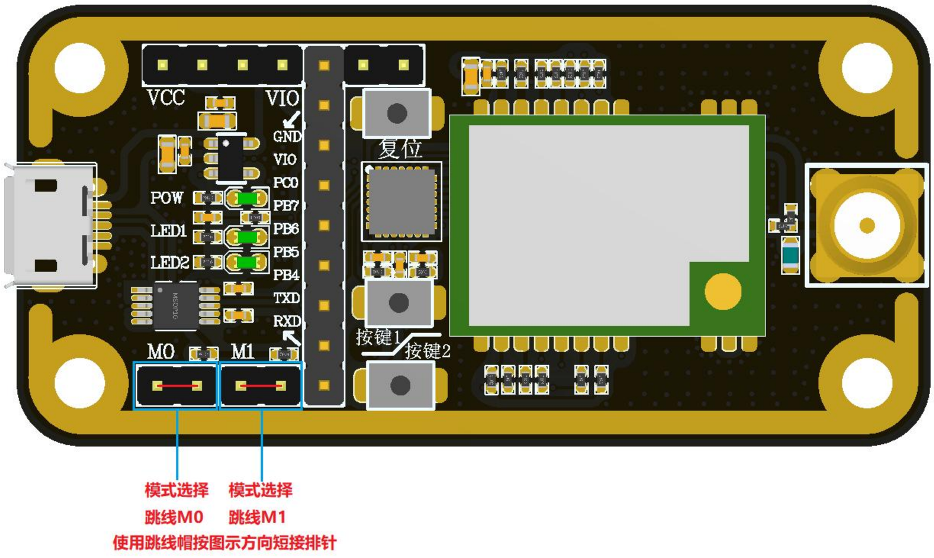


```
#define E07_DATA_RATE                 1200  //空速 1.2 KBps
#define E07_FREQUENCY_DEVIATION  14300     //频偏 14.3 K
#define E07_BANDWIDTH               58000   //接收频宽 58 K
#define E07_OUTPUT_POWER               10   //功率 [10  7  5  0  -10  -15  -20 -30]
#define E07_PREAMBLE_SIZE               4   //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]
```

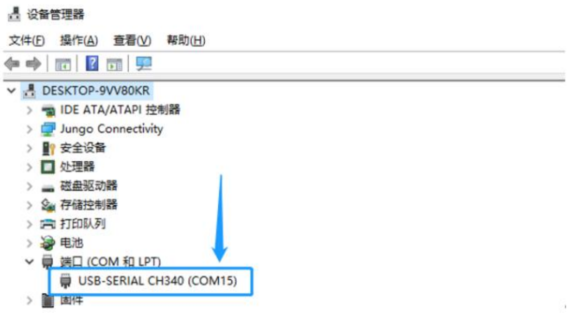
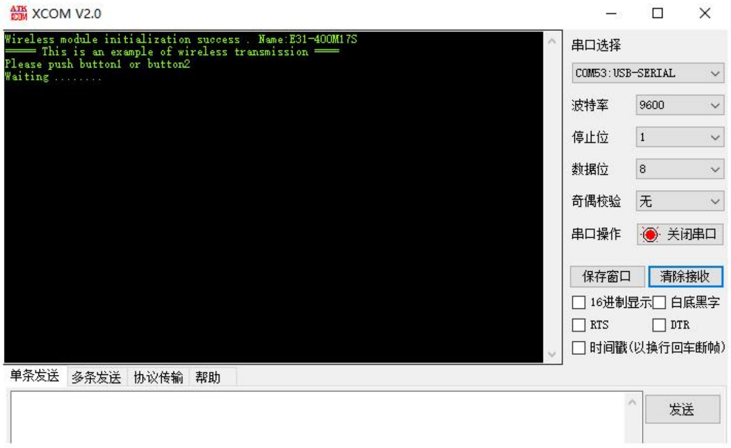
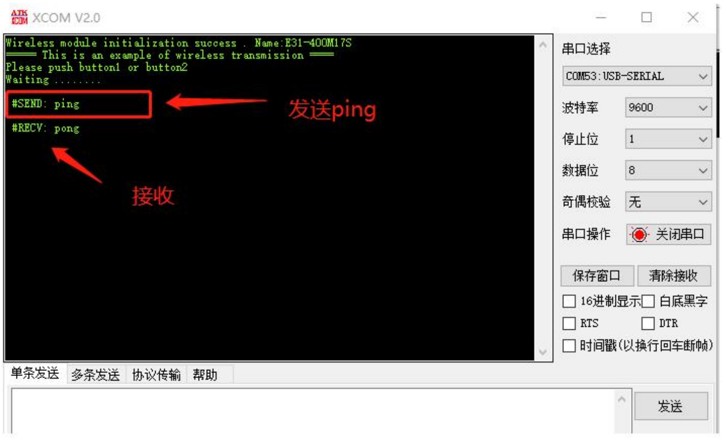
</td>
</tr>
<tr>
<td></td>
<td>board.c</td>
<td>STM8 peripheral initialization, involving SPI, TIMER, GPIO, etc., is strongly coupled with the hardware used.</td>
</tr>
<tr>
<td></td>
<td>board_button.c</td>
<td>The key event queue is a FIFO in terms of data structure . After the timer detects the button, it will store the corresponding event in the queue and wait for the main loop to respond.</td>
</tr>
<tr>
<td></td>
<td>board_mini_printf.c</td>
<td>Simplified printf, although the function has shrunk, it occupies a small volume. The DEBUG macro in the project mainly depends on the mprintf provided by this file.</td>
</tr>
<tr>
<td></td>
<td>ebyte_kfifo.c</td>
<td>Used for serial port data reception, optimized general-purpose FIFO queue, suitable for cache.</td>
</tr>
<tr>
<td></td>
<td>ebyte_debug.c</td>
<td>It is used to connect to a PC for some tests, and generally does not need to be used.</td>
</tr>
<tr>
<td></td>
<td>stm8l15x_it.c</td>
<td>All interrupt function entrances will focus on the interrupt service functions such as serial port, timer, button IO, etc.</td>
</tr>
</table>

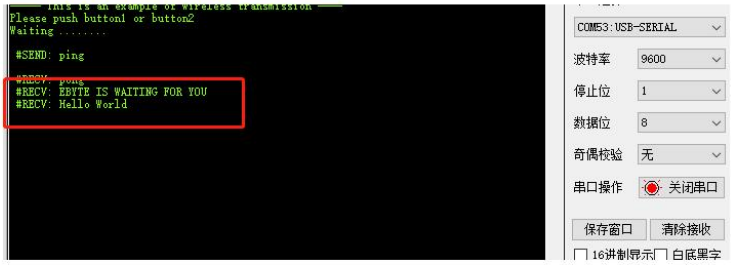
Chapter 3 Quick Demo

3.1 Signal line connection

	Matter	Explanation
1	Power jumper cap	 使用跳线帽按图示方向短接排针 模块电流测试排针 MCU供电排针 VCC VIO GND VIO PC0 PB7 PB6 PB5 PB4 TXD RXD 复位 POW LED1 LED2 按键1 按键2 M0 M1
2	Mode selection jumper cap	 VCC VIO GND VIO PC0 PB7 PB6 PB5 PB4 TXD RXD 复位 POW LED1 LED2 按键1 按键2 M0 M1 模式选择 跳线M0 模式选择 跳线M1 使用跳线帽按图示方向短接排针
3	Auxiliaries	USB cable , antenna, PC , etc.

3.2 Serial port assistant

	Matter	Explanation
1	Device manager View the serial port number	 设备管理器 文件(F) 操作(A) 查看(V) 帮助(H) DESKTOP-9VV80KR IDE ATA/ATAPI 控制器 Jungo Connectivity 安全设备 处理器 磁盘驱动器 存储控制器 打印队列 电池 端口 (COM 和 LPT) USB-SERIAL CH340 (COM15) 固件
2	Serial software	 Wireless module initialization success . Name:E31-400M17S ═══ This is an example of wireless transmission ═══ Please push button1 or button2 Waiting .... 串口选择 COM53:USB-SERIAL 波特率 9600 停止位 1 数据位 8 奇偶校验 无 串口操作 关闭串口 □ 16进制显示 □ 白底黑字 □ RTS □ DTR □ 时间戳(以换行回车断帧) XCOM V2.0 保存窗口 清除接收 单条发送 多条发送 协议传输 帮助 发送
3	Example of push button communication	#RECV Identifier, used only for prompting, indicates the data received by the wireless module. #SEND identifier, only used for prompting, indicating the data sent by the wireless module  Wireless module initialization success . Name:E31-400M17S ═══ This is an example of wireless transmission ═══ Please push button1 or button2 Waiting .... #SEND: ping #RECV: pong 串口选择 COM53:USB-SERIAL 波特率 9600 停止位 1 数据位 8 奇偶校验 无 串口操作 关闭串口 □ 16进制显示 □ 白底黑字 □ RTS □ DTR □ 时间戳(以换行回车断帧) XCOM V2.0 发送ping 接收 保存窗口 清除接收 单条发送 多条发送 协议传输 帮助 发送
4	Serial data transparent transmission	Serial data transparent transmission directly transmits the required content through XCOM

		 ——— This is an example of wireless transmission ——— Please push button1 or button2 Waiting .... #SEND: ping #RECV: pong #RECV: EBYTE IS WAITING FOR YOU #RECV: Hello World COM53:USB-SERIAL 波特率 9600 停止位 1 数据位 8 奇偶校验 无 串口操作 关闭串口 保存窗口 清除接收 □ 16进制显示 □ 白底黑字

Chapter 4 Frequently Asked Questions

4.1 The transmission distance is not ideal

- When there is a straight-line communication obstacle, the communication distance will be attenuated accordingly ;
- Temperature, humidity, and co-channel interference will increase the communication packet loss rate ;
- The ground absorbs and reflects radio waves, and the test effect is poor when it is close to the ground ;
- Seawater has a strong ability to absorb radio waves, so the seaside test results are poor ;
- There are metal objects near the antenna, or placed in a metal case, the signal attenuation will be very serious ;
- The power register is set incorrectly, and the air speed is set too high (the higher the air speed, the closer the distance) ;
- The low voltage of the power supply at room temperature is lower than the recommended value, and the lower the voltage, the lower the output power
- The matching degree between the antenna and the module is poor or the quality of the antenna itself is problematic.

4.2 The module is easily damaged

- Please check the power supply to ensure that it is between the recommended power supply voltage, if it exceeds the maximum value, it will cause permanent damage to the module ;
- Please check the stability of the power supply, the voltage cannot fluctuate greatly and frequently ;
- Please ensure anti-static operation during installation and use, and high-frequency devices are electrostatically sensitive ;
- Please ensure that the humidity during installation and use should not be too high, some components are humidity sensitive devices ;
- If there is no special requirement, it is not recommended to use it at too high or too low temperature.

4.3 The bit error rate is too high

- There is co-channel signal interference nearby, stay away from the source of interference or modify the frequency and channel to avoid interference;
- Unsatisfactory power supply may also cause garbled characters, so be sure to ensure the reliability of the power supply;
- Poor quality or too long extension lines and feeders will also cause high bit error rates.

Revision history

Version	revision date	Revision description	Issued by
1.0	2021-09-22	initial version	JH
1.1	2022-12-29	Modify the schematic diagram of the module and the use manner	HWJ

About Us

Technical support: support@cdebyte.com

Documents and RF Setting download link: https://www.cdebyte.com

Thank you for using Ebyte products! Please contact us with any questions or suggestions: info@cdebyte.com

----

Official hotline:028- 61543675

Web: https://www.cdebyte.com

Address: Building B5, Mould Industrial Park, 199# Xiqu Ave, High-tech Zone, Chengdu, 611731, Sichuan, China

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