GW3323 SDK & environment user manual

Version：V2.0

© Geehy Semiconductor Co., Ltd.

Directory

1 COMPILATION ENVIRONMENT DESCRIPTION 2

1.1 CODEBLOCK(IDE). 2

1.2 DRAG APP.CBP 3

1.3 TOOLCHAIN:. 4

1.4 CP210X_WINDOWS_DRIVERS 4

2 PROJECT INTRODUCTION 6

3 DOWNLOAD INSTRUCTIONS 10

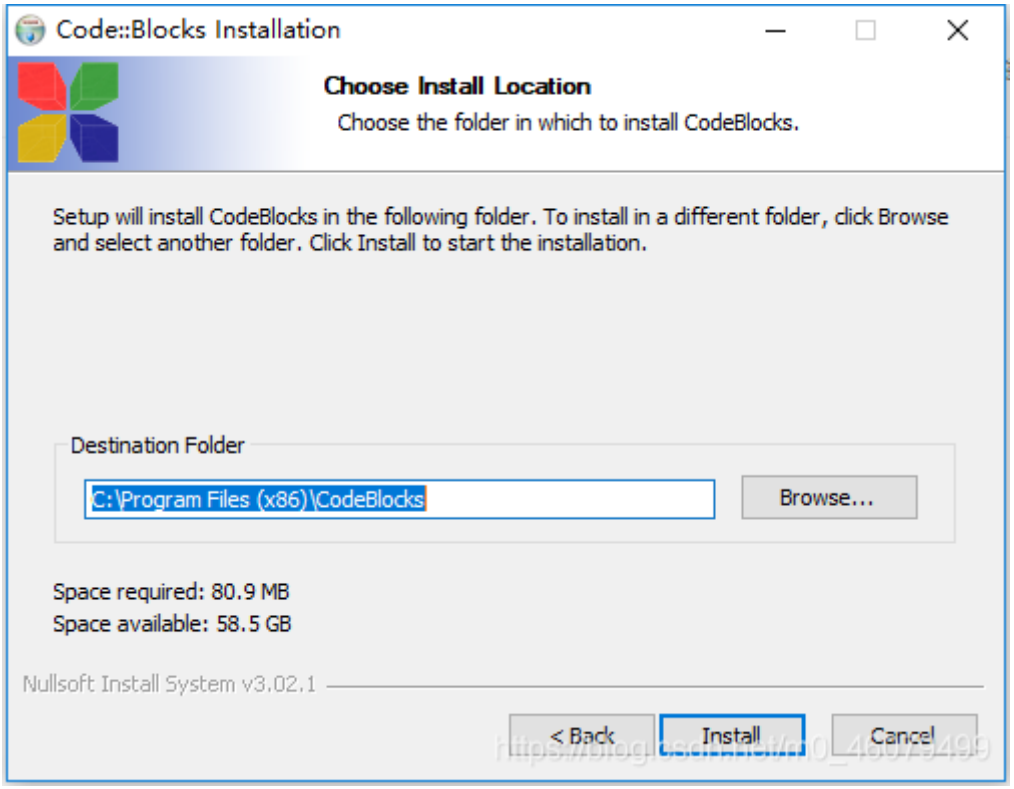
4 VERSION HISTORY 16

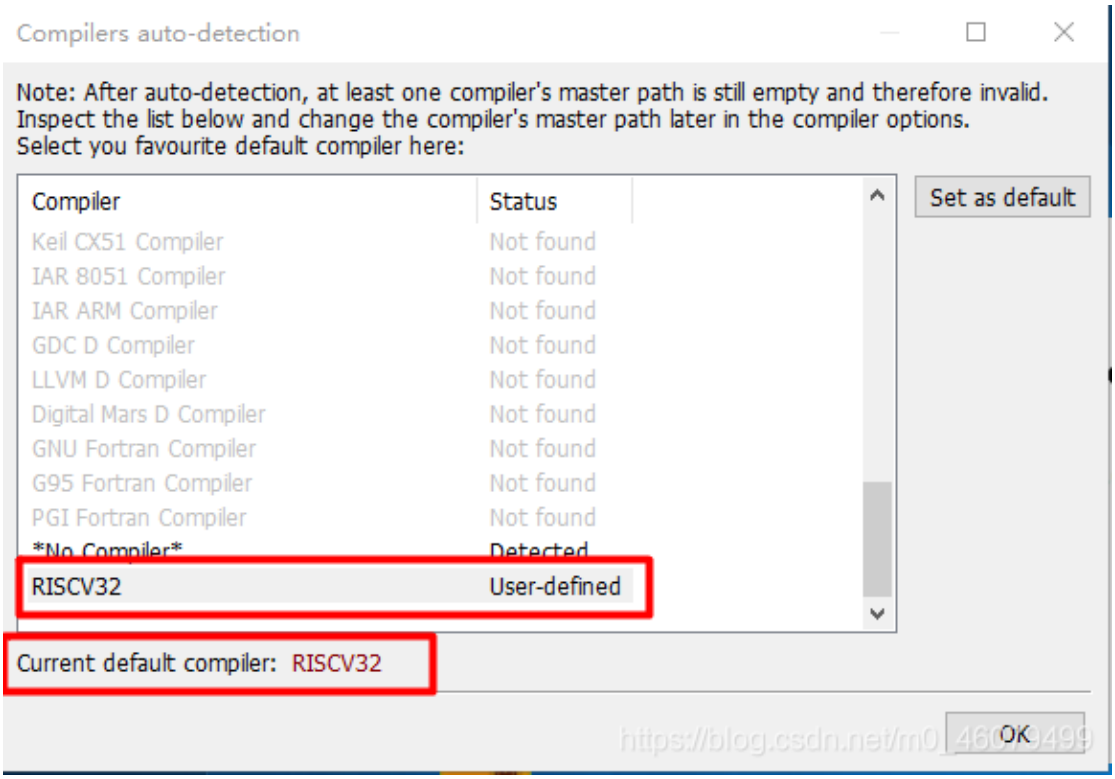
1 Compilation environment description

The development environment for the GW3323 chip is “Codeblocks.exe”. For the installation package, “codeblocks-20.03mingw-setup.exe.”. Install CodeBlocks first, then install RV32-Toolchain (When you install ToolChain, the configuration-related compilation environment is registered with CodeBlocks)

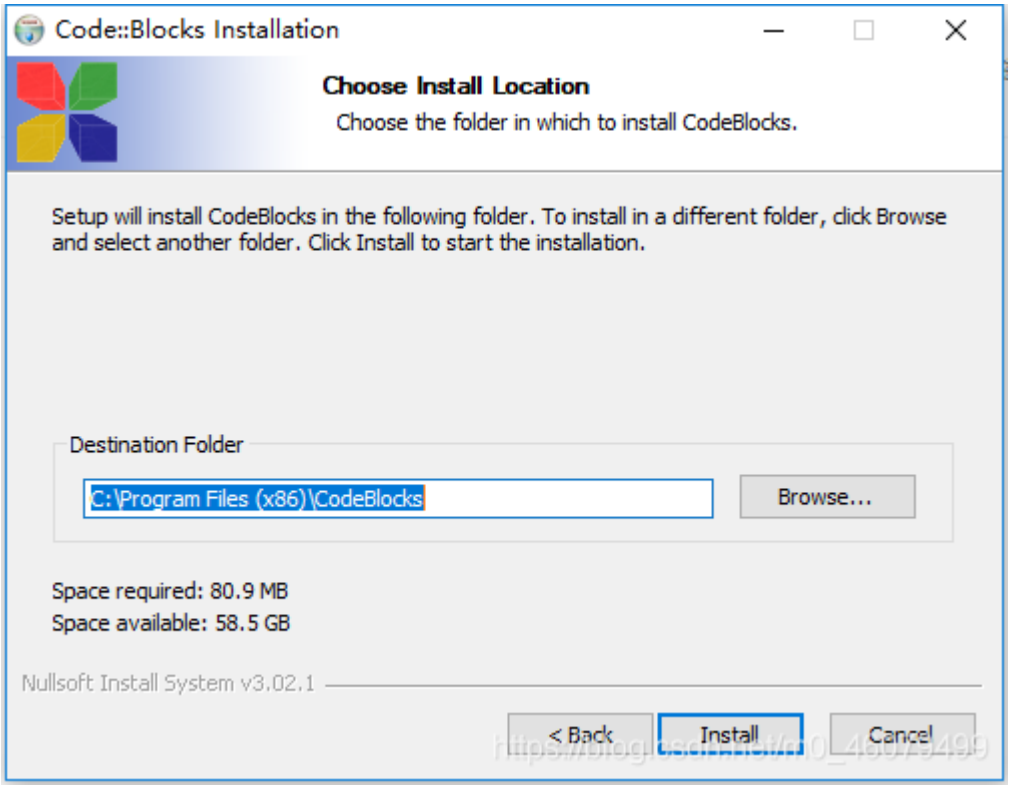
In the development, we generally use serial port printing or GPIO port and logic analyzer for debugging, breakpoint debugging and simulation are not currently supported.

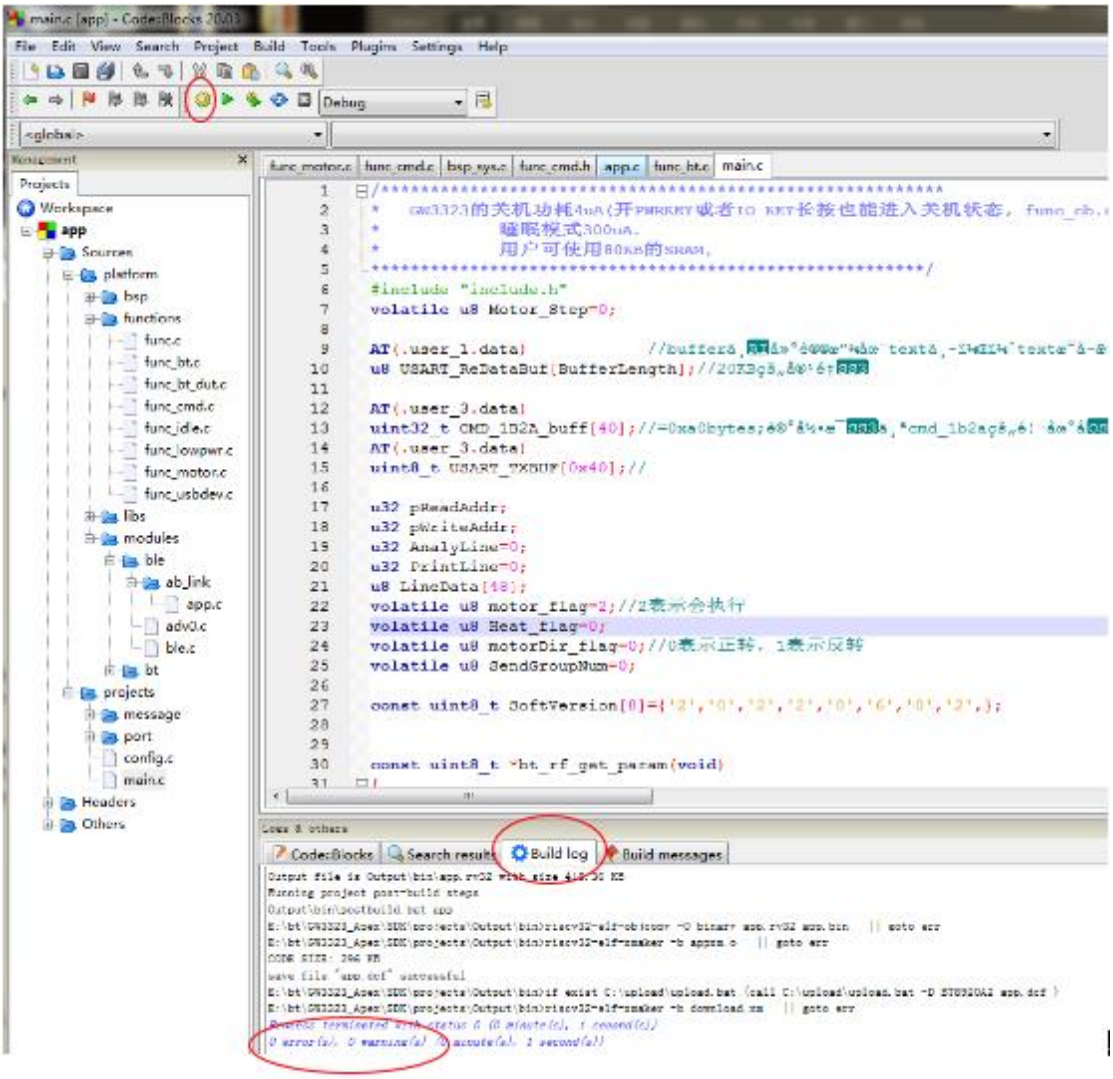
1.1 CodeBlock(IDE): (codeblocks-17.12)Code editor, which calls the tool provided in the ToolChain when compiling the link. Finally, the dcf file for burn is generated.





1.2 Drag app.cbp into the project to open the project for compilation



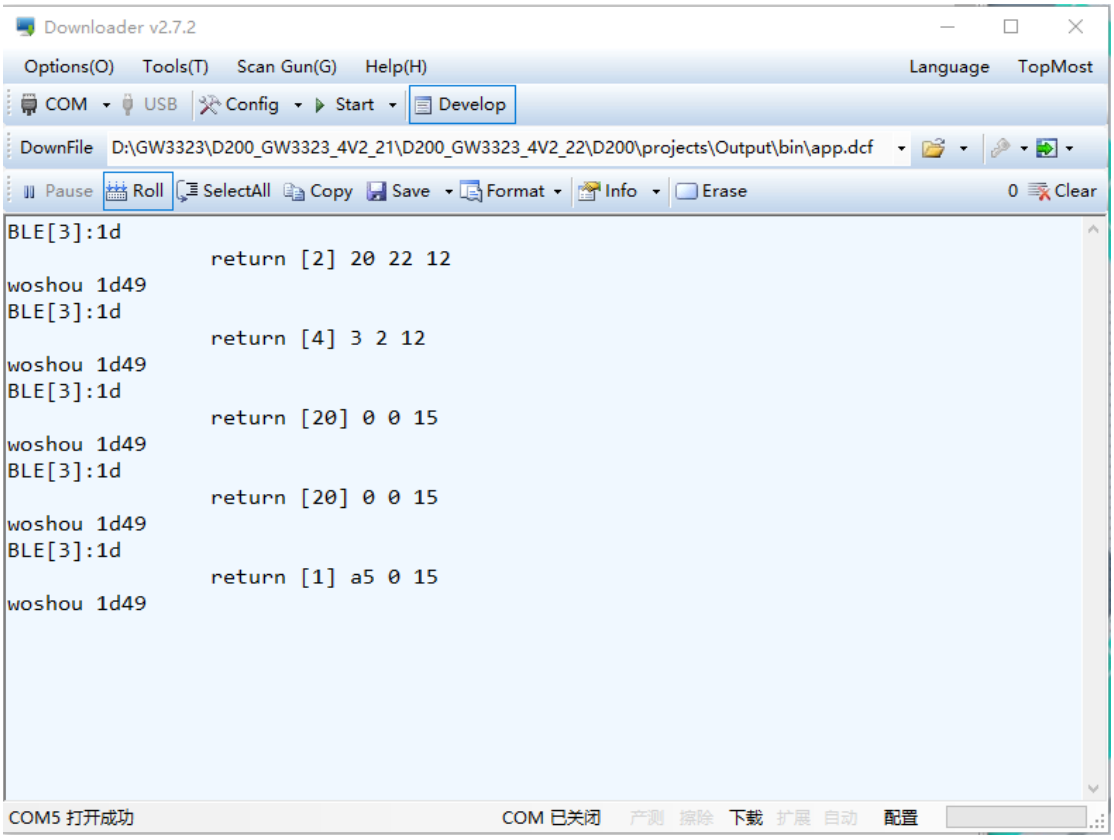


1.3 **ToolChain: (RV32-Toolchain-Setup_vxxx)Contains RISC-V compiler, Bin file conversion tool, etc.**

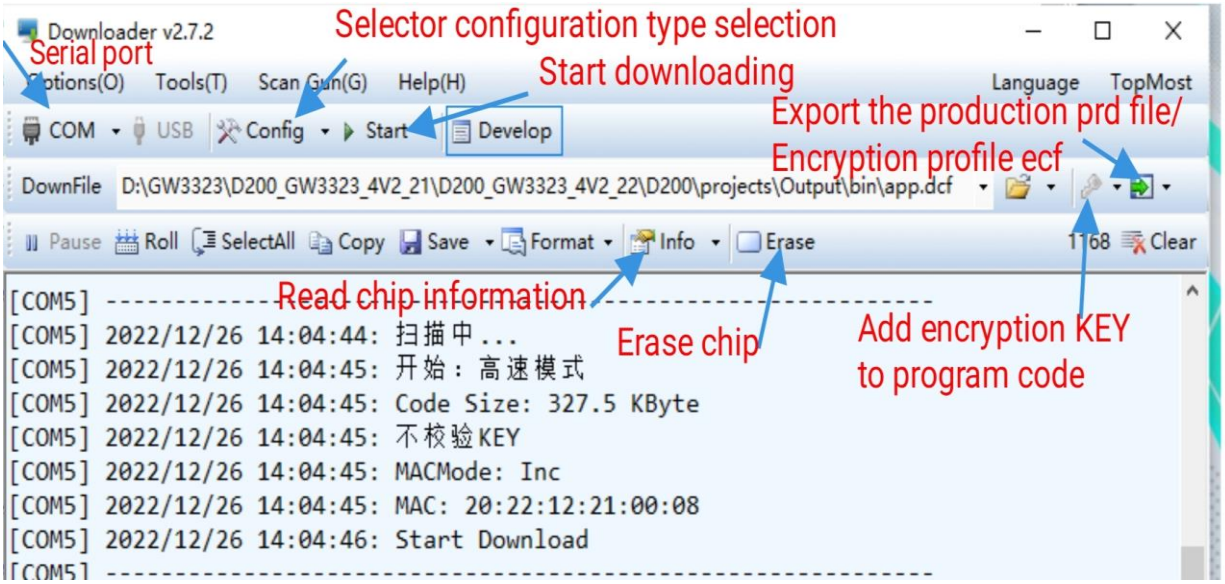
RV32-Toolchain-Setup.exe	2021/12/30 20:48	应用程序	24,977 KB

1.4 **Downloader：the function of serial port printing tool software, can be used for developers debugging，CP210x_Windows_Drivers：the Xlink burner driver.**

CP210x_Windows_Drivers.rar	2022/3/11 17:08	WinRAR 压缩文件	3,656 KB
Downloader_v2.7.2.zip	2022/3/11 16:38	WinRAR ZIP 压缩...	2,488 KB

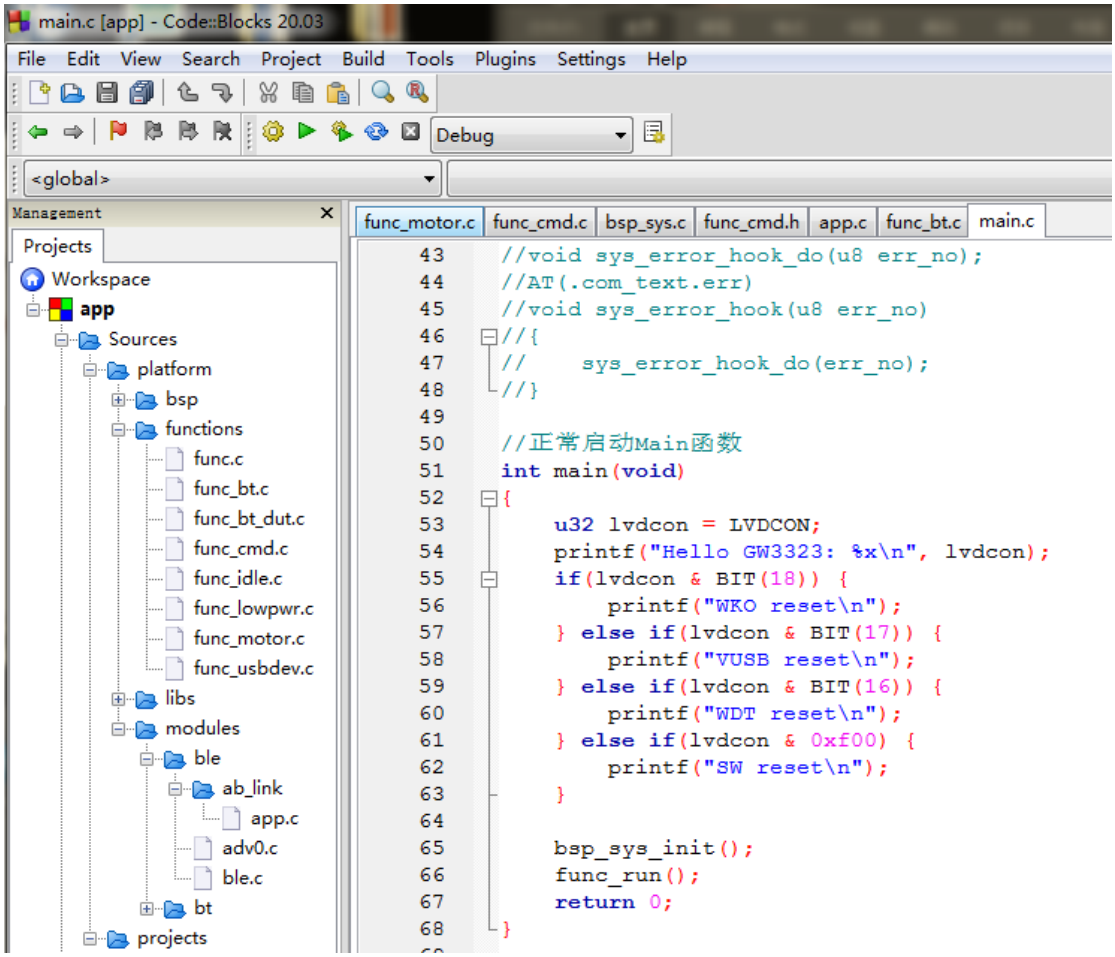


Developers generally select "Develop" in the following figure, so that they can easily view the printing information after downloading.



2 Project Introduction

Put the file "GW3323_SDK_BTtoUart_20221221\app\projects\app.cbp" into the "codeBlocks" development tool.



Initialize first, focus on func_enter();

func_motor.c | func_cmd.c | bsp_sys.c | func_cmd.h | app.c | func_bt.c | main.c | bsp_sys.h | *func.c | func_lowpwr.h

```
105  void func_run(void)
106  {
107      printf("%s\n", __func__);
108      func_bt_chk_off();
109
110      while (1) {
111          func_enter();
112          switch (func_cb.sta) {
113  #if FUNC_BT_EN
114          case FUNC_BT:
115              func_bt();
116              break;
117  #endif
118
119  #if FUNC_IDLE_EN
120          case FUNC_IDLE:
121              func_idle();
122              break;
123  #endif // FUNC_IDLE_EN
124
125  #if FUNC_BT_DUT_EN
126          case FUNC_BT_DUT:
127              func_bt_dut();
128              break;
129
130          case FUNC_BT_FCC:
131              func_bt_fcc();
132              break;
133  #endif // IODM_TEST_MODE
134
135          case FUNC_PWROFF:
136              func_pwroff(1);
137              break;
138
139          default:
140              func_exit();
141              break;
142          }
143      }
144  }
```

func_motor.c | func_cmd.c | bsp_sys.c | func_cmd.h | app.c | func_bt.c | main.c | bsp_sys.h | *func.c | func_lowpwr.h

```
252
253  AT(.text.func.bt)
254  void func_bt(void)
255  {
256      uint16_t i;//,j;
257      uint8_t data;
258      uint8_t x_offset;
259  //    u16 sPrint_Dot=0;
260  //    u16 ccd=0;//
261      u16 Print_Dot=0;
262  //    u16 GroupOffset=0;
263      u8 GroupNum=0;
264  //    u16 PaperAdcRange;//前后差值
265  //    u16 PaperAdcPer;//上一刻的采样值
266      uint8_t Group_printbuf[5][MaxWide];//[0]96点以内;[1]~[2]192点以内,
267      uint16_t Group_HTbuf[5];
268      printf("%s\n", __func__);
269
270      func_bt_enter();
271
272      while (func_cb.sta == FUNC_BT) {
273          func_bt_process();//喂狗，执行SPP或BLE程序，显示蓝牙连接状态
274          WDT_CLR();
275
276          if ( (PrintBuf.PrintTask) )//1点行已处理完
277          {
...
390          else  if( ((TMR3CON&BIT(0))==0) )//if( (motor_flag==2) )//有打
391          {//打印全部执行完成才进入此模块
506
507          func_bt_message(msg_dequeue());
508          func_bt_display();
509      }
510      printf("func_bt_exit!\n");
511      func_bt_exit();
512  }
513
```

The timer operation is in bsp_sys.c

func_motor.c | func_cmd.c | bsp_sys.c | func_cmd.h | app.c | func_bt.c | main.c | bsp_sys.h | *func.c | func_lowpwr.h

```
374  AT(.com_text.isr)
375  void timer3_init(void)
376  {
377      TMR3CON =  BIT(7);                  //Timer overflow interrupt enable
378      TMR3CNT = 0;
379      TMR3PR  = MotorSpeed_Init - 1;          //4步1点行，马达走纸速度达15.6mm/s;
380      TMR3CON |= BIT(2) ;         //Timer works in Counter Mode
381      sys_irq_init(IRQ_TMR3_VECTOR, 1, timer3_isr);
382  }
383
384  #if (TimerPrint == 1)
385  AT(.com_text.isr)
386  void timer4_isr(void)//
387  {
388  }
389  #elif (TimerPrint == 2)
390  AT(.com_rodata.isr)
391  const char str_t4[] = "&\n";//此数据不放在flash中，只能从内存里读取数据，load flash 会复位
392
393  AT(.com_text.isr)
394  FIQ void timer4_isr(void)//快速中断请求，
395  {
396          TMR4CPND = BIT(16);             //Clear Pending
397  //        printk(str_t4);
398          TPH_STB_RESET;
399          TMR4CON &= ~BIT(0);
400  }
401  #endif
402  AT(.com_text.isr)
403  void timer4_init(void)
404  {
405      TMR4CON =  BIT(7);                  //Timer overflow interrupt enable
406      TMR4CNT = 0;
407      #if (TimerPrint == 1)
408      TMR4PR  = 800  - 1;          //500ms, select xosc26_div 1M clk
409      TMR4CON |= BIT(2)| BIT(0);         //Timer works in Counter Mode
410
411      #elif (TimerPrint == 2)
412      TMR4PR  = 1600  - 1;          //25℃时加热1.02ms显影
413      TMR4CON |= BIT(2);         //Timer works in Counter Mode
414      sys_irq_init(IRQ_TMR4_VECTOR, 1, timer4_isr);
415      #endif
416  }
```

The control of SPP Bluetooth is in spp.c

func_motor.c | func_cmd.c | bsp_sys.c | func_cmd.h | app.c | func_bt.c | main.c | bsp_sys.h | *func.c | func_lowpwr.h | ble.c | spp.c

```
78   void spp_connect_callback(void)
79   {
80       printf("--->spp_connect_callback\n");
81   #if BT_SPP_FOT_EN
82       fot_spp_connect_callback();
83   #endif // BT_SPP_FOT_EN
84   }
85
86   void spp_disconnect_callback(void)
87   {
88       printf("--->spp_disconnect_callback\n");
89   #if BT_SPP_FOT_EN
90       fot_spp_disconnect_callback();
91   #endif // BT_SPP_FOT_EN
92   }
93
94   void spp_rx_callback(uint8_t *packet, uint16_t size)
95   {
96   #if BT_SPP_FOT_EN
97       if(fot_app_connect_auth(packet, size)){
98           fot_recv_proc(packet,size);
99           return;
100      }
101  #endif // BT_SPP_FOT_EN
102
103     uint16_t i;
104      uint8_t USARTReDataTemp;
105
106      printf("[%x-%x] (%x)=%x,%x,%x,%x,%x,%x,%x...%x,%x,%x\n'
107      for(i=0;i<size;i++)
108      {
109          USARTReDataTemp = *(packet+i);
110          USART_ReDataBuf[pReadAddr]= USARTReDataTemp;
111  //          printf(" [%x]:%x ",pReadAddr,USART_ReDataBuf[pRe:
```

Controls for BLE Bluetooth are in app.c

func_motor.c | func_cmd.c | bsp_sys.c | func_cmd.h | app.c | func_bt.c | main.c | bsp_sys.h | *func.c | func_lowpwr.h | ble.c | spp.c

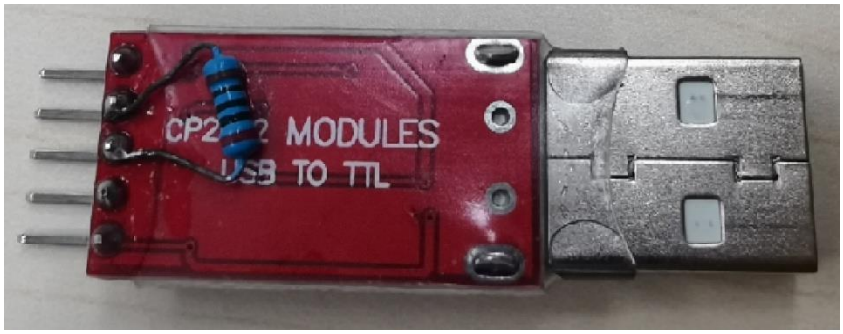
```
189         .type = BLE_GATTS_UUID_TYPE_16BIT,
190         .uuid = app_notify_uuid16,
191     };
192     static gatts_service_base_st gatts_app_notify_base;
193
194
195     //-----------------------------------------------------------------
196     // APP
197     //-----------------------------------------------------------------
198     bool ble_send_packet(u8 *buf, u8 len)
199     {
200         return ble_tx_notify(gatts_tests_base.att_index, buf, len);
201     }
202     //ble每次收0x8c个字节，即140个。
203     static uint8_t gatt_callback_app(u8 *ptr, u8 len)
204     {//cmd_1d0e(退纸并打印);cmd_1b2a...;cmd_1d0c(进纸到撕纸位置)
205     //printf("BLE[%x]:%x  %x  %x  %x  %x  %x  %x  %x  %x  %x  \n",len,*p
206        uint16_t t,j,Line24_oft;
207         uint8_t Rec_No,USARTReDataTemp;
208     //  uint16_t Else_Len;
209
210         for(Rec_No=0;Rec_No<len;Rec_No++)
211         {
212             USARTReDataTemp = *(ptr+Rec_No);
213             USART_ReDataBuf[pReadAddr]= USARTReDataTemp;
214      //        printf(" [%x]:%x ",pReadAddr,USART_ReDataBuf[pReadAddr]);
215             pReadAddr = (pReadAddr+1)%BufferLength;
216             USART_ReceiveCNT++;
217              switch(USART_ReState)
218             {
424     return false;
425     }
426
```

3 Download instructions

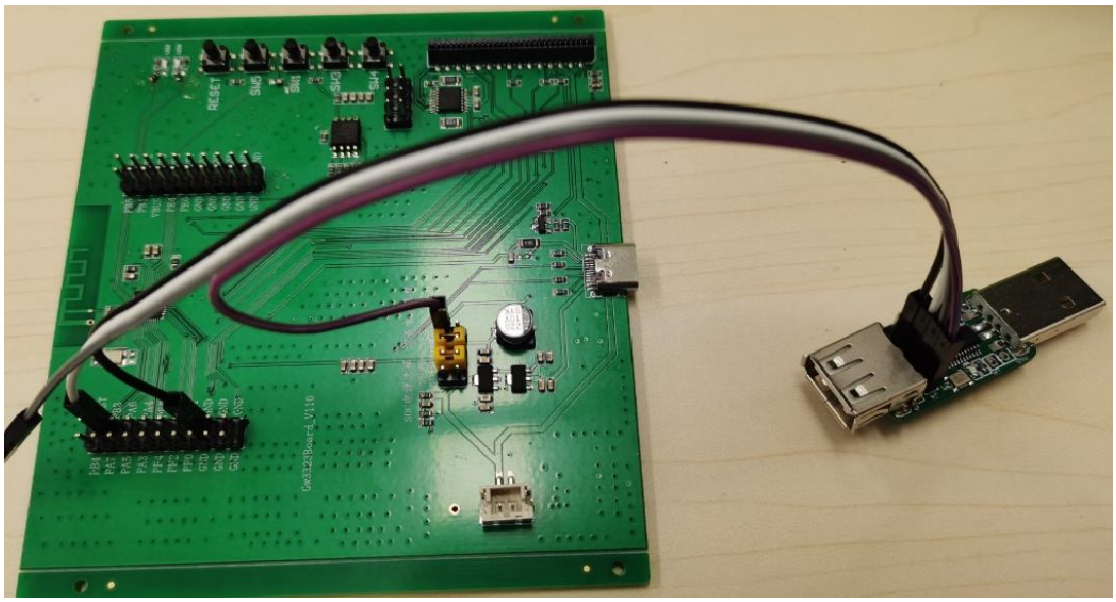
1、 Connect the development board with the serial port module of CP2102 or Geehy serial port module.

Geehy XLink modue: RX---->Link GW3323 PB3;

CP2102 Serial port module： TX -- string 200R -- RX --->Link GW3323 PB3;

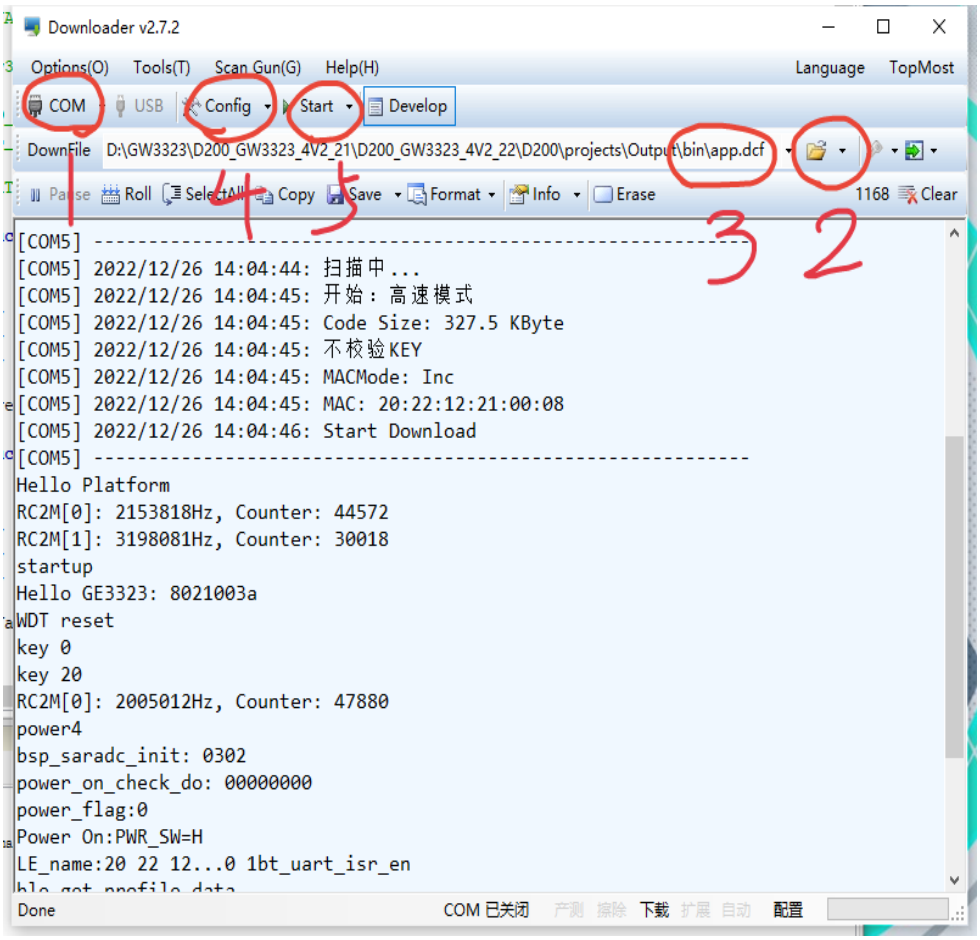


2、 The three wires of the serial port module (GND,RX,3.3V) are connected to the development board (GND,PB3,3.3V), and the other end is connected to the USB port of the computer.



3、 After the serial port is connected, the icon of the serial port turns black, indicating that the serial port is connected. Otherwise, check whether the hardware of the serial port is faulty or the CP210x_Windows_Drivers is incorrectly installed.

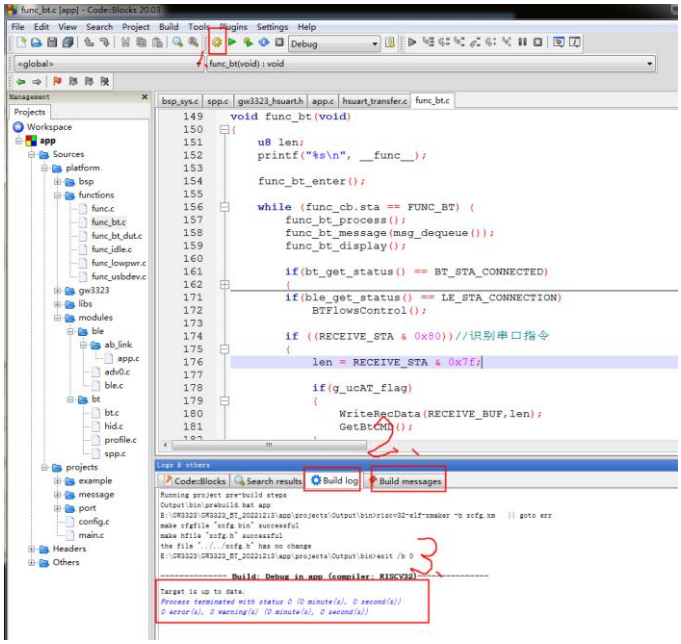
4、 Then select the project burning file. Then select to start burning. As shown below：



4 Example Code presentation

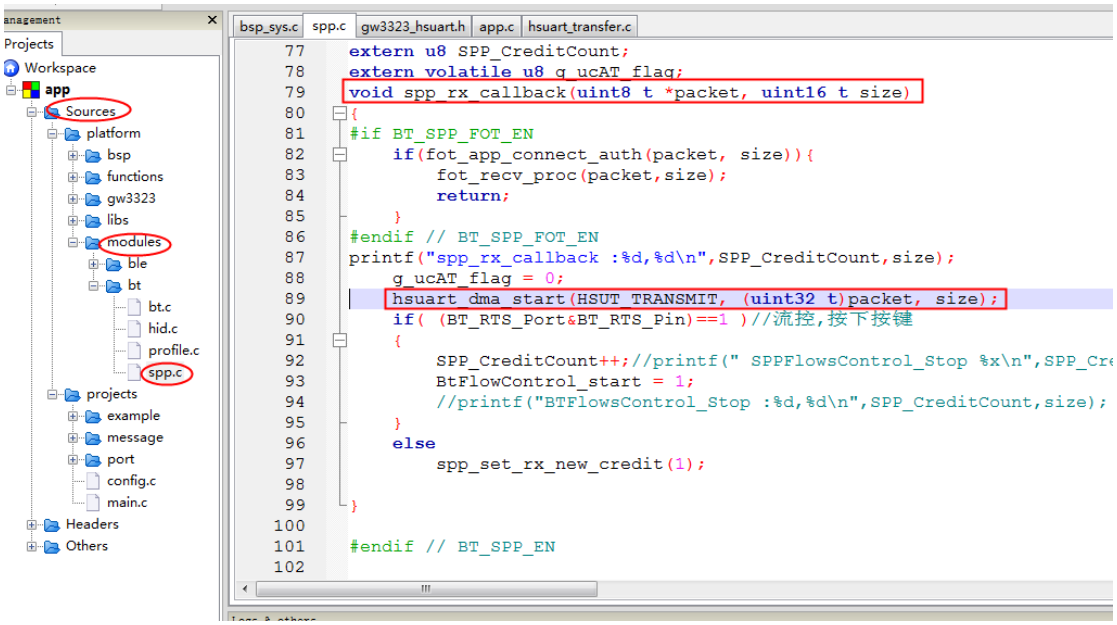
1、The default routine (GW3323_SDK_BTtoUart_20221221.rar) the project is "sending the data received by Bluetooth to the serial port (PB2), and sending the data received by serial port (PB1) to the phone via Bluetooth".

2、If you want to change the project , then burn , must "0 error 0 warning" .

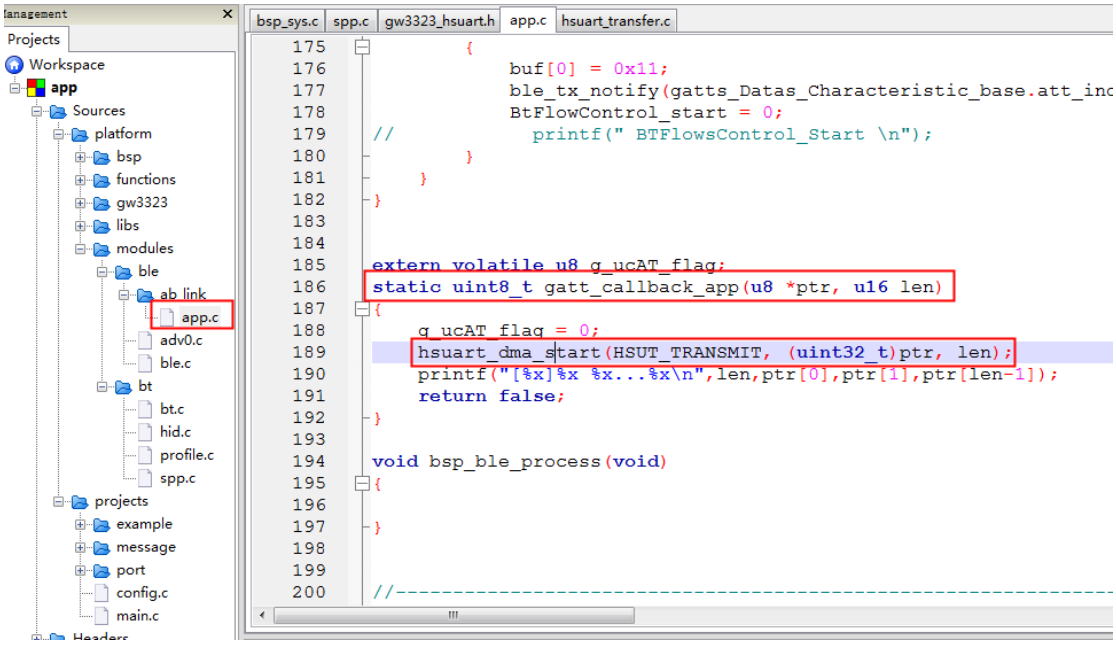


3、Function about Bluetooth

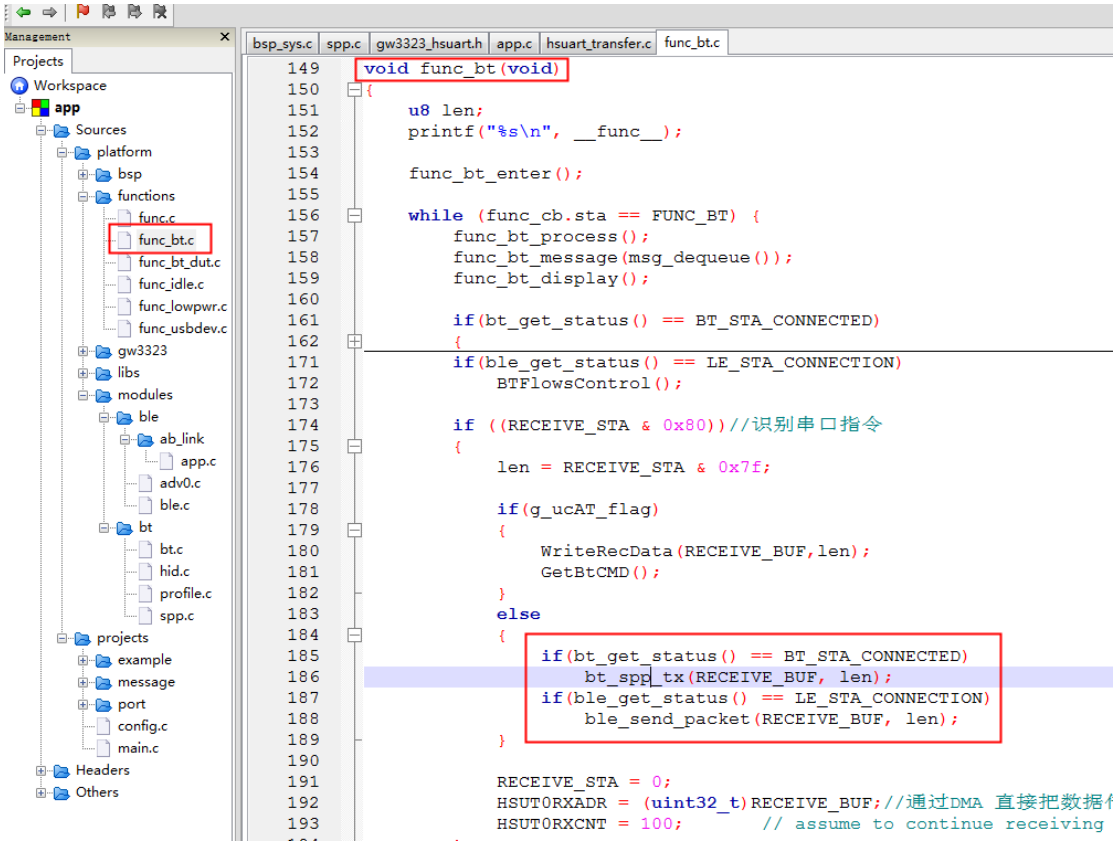
3.1 Send the data by SPP protocol through the serial port (PB2):



3.2 Send the data received by BLE protocol through the serial port (PB2):

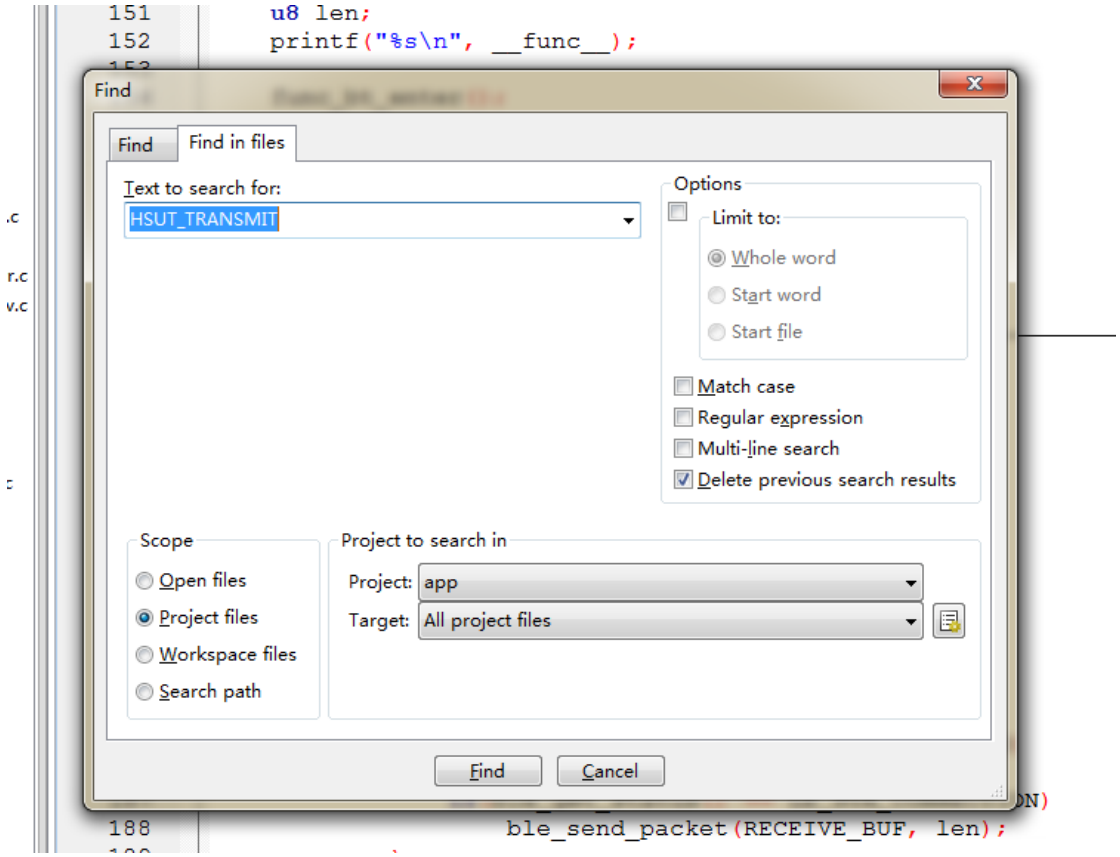


3.3 The data received through the serial port (PB1) is sent to the mobile phone via Bluetooth (SPP\BLE)

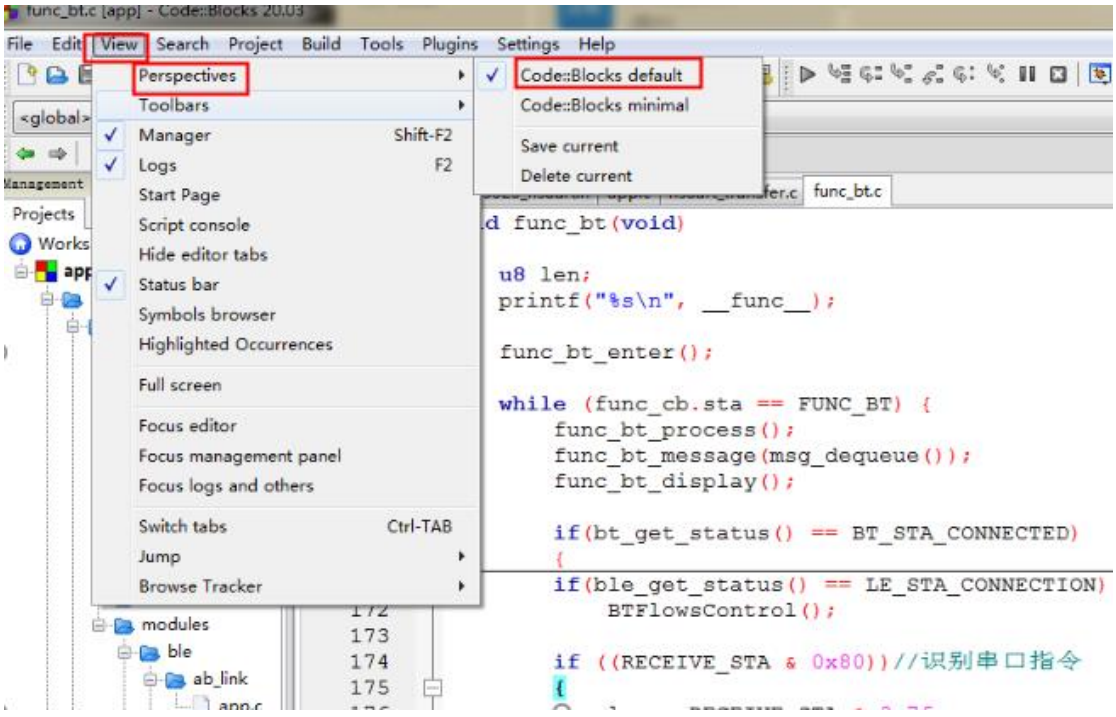


5 Common Settings

1、Find all files (ctrl+F)



2、Recovery window view



6 Version History

Table 1 File version history

DATE	version	Version history
2022.11	1.0	Original
2022.12	2.0	Added "Routine Display" and "Common Settings"

Statement

This document is formulated and published by Geehy Semiconductor Co., Ltd. (hereinafter referred to as "Geehy"). The contents in this document are protected by laws and regulations of trademark, copyright and software copyright. Geehy reserves the right to make corrections and modifications to this document at any time. Please read this document carefully before using Geehy products. Once you use the Geehy product, it means that you (hereinafter referred to as the "users") have known and accepted all the contents of this document. Users shall use the Geehy product in accordance with relevant laws and regulations and the requirements of this document.

1. Ownership

This document can only be used in connection with the corresponding chip products or software products provided by Geehy. Without the prior permission of Geehy, no unit or individual may copy, transcribe, modify, edit or disseminate all or part of the contents of this document for any reason or in any form.

The "极海" or "Geehy" words or graphics with "®" or "™" in this document are trademarks of Geehy. Other product or service names displayed on Geehy products are the property of their respective owners.

2. No Intellectual Property License

Geehy owns all rights, ownership and intellectual property rights involved in this document.

Geehy shall not be deemed to grant the license or right of any intellectual property to users explicitly or implicitly due to the sale or distribution of Geehy products or this document.

If any third party's products, services or intellectual property are involved in this document, it shall not be deemed that Geehy authorizes users to use the aforesaid third party's products, services or intellectual property, unless otherwise agreed in sales order or sales contract.

3. Version Update

Users can obtain the latest document of the corresponding models when ordering Geehy products.

If the contents in this document are inconsistent with Geehy products, the agreement in thesales order or the sales contract shall prevail.

4. Information Reliability

The relevant data in this document are obtained from batch test by Geehy Laboratory or cooperative third-party testing organization. However, clerical errors in correction or errors caused by differences in testing environment may occur inevitably. Therefore, users should understand that Geehy does not bear any responsibility for such errors that may occur in this document. The relevant data in this document are only used to guide users as performance parameter reference and do not constitute Geehy's guarantee for any product performance.

Users shall select appropriate Geehy products according to their own needs, and effectively verify

and test the applicability of Geehy products to confirm that Geehy products meet their own needs, corresponding standards, safety or other reliability requirements. If loses are caused to users due to the user's failure to fully verify and test Geehy products, Geehy will not bear any responsibility.

5. Legality

USERS SHALL ABIDE BY ALL APPLICABLE LOCAL LAWS AND REGULATIONS WHEN USING THIS DOCUMENT AND THE MATCHING GEEHY PRODUCTS. USERS SHALL UNDERSTAND THAT THE PRODUCTS MAY BE RESTRICTED BY THE EXPORT, RE-EXPORT OR OTHER LAWS OF THE COUNTIRIES OF THE PRODUCTS SUPPLIERS, GEEHY, GEEHY DISTRIBUTORS AND USERS. USERS (ON BEHALF OR ITSELF, SUBSIDIARIES AND AFFILIATED ENTERPRISES) SHALL AGREE AND PROMISE TO ABIDE BY ALL APPLICABLE LAWS AND REGULATIONS ON THE EXPORT AND RE-EXPORT OF GEEHY PRODUCTS AND/OR TECHNOLOGIES AND DIRECT PRODUCTS.

6. Disclaimer of Warranty

THIS DOCUMENT IS PROVIDED BY GEEHY "AS IS" AND THERE IS NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

GEEHY WILL BEAR NO RESPONSIBILITY FOR ANY DISPUTES ARISING FROM THE SUBSEQUENT DESIGN OR USE BY USERS.

7. Limitation of Liability

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL GEEHY OR ANY OTHER PARTY WHO PROVIDE THE DOCUMENT "AS IS", BE LIABLE FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, DIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE DOCUMENT (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY USERS OR THIRD PARTIES).

8. Scope of Application

The information in this document replaces the information provided in all previous versions of the document.

© 2022 Geehy Semiconductor Co., Ltd. - All Rights Reserved