

Arduino and FSC-DB008 Programming Tutorial

Contents

1.	Introduction.....	1
2.	Hardware Connection.....	1
3.	Software Instructions.....	3
	3.1 Overview.....	3
	3.2 Source Code.....	3
4.	Program Initialization.....	6
	4.1 Methods.....	6
	4.2 Function Test.....	7
5.	Q&A.....	9

1. Introduction

In this tutorial, you can learn how to use an Arduino UNO board to communicate with the FSC-DB008. FSC-DB008 enables the Arduino developers to evaluate Feasycom Bluetooth audio modules (e.g. FSC-BT966) in a very convenient way.

The FSC-DB008 only occupies ~10 and ~11 for UART communication, while all other I/Os are free to use by developers.

2. Hardware Connection

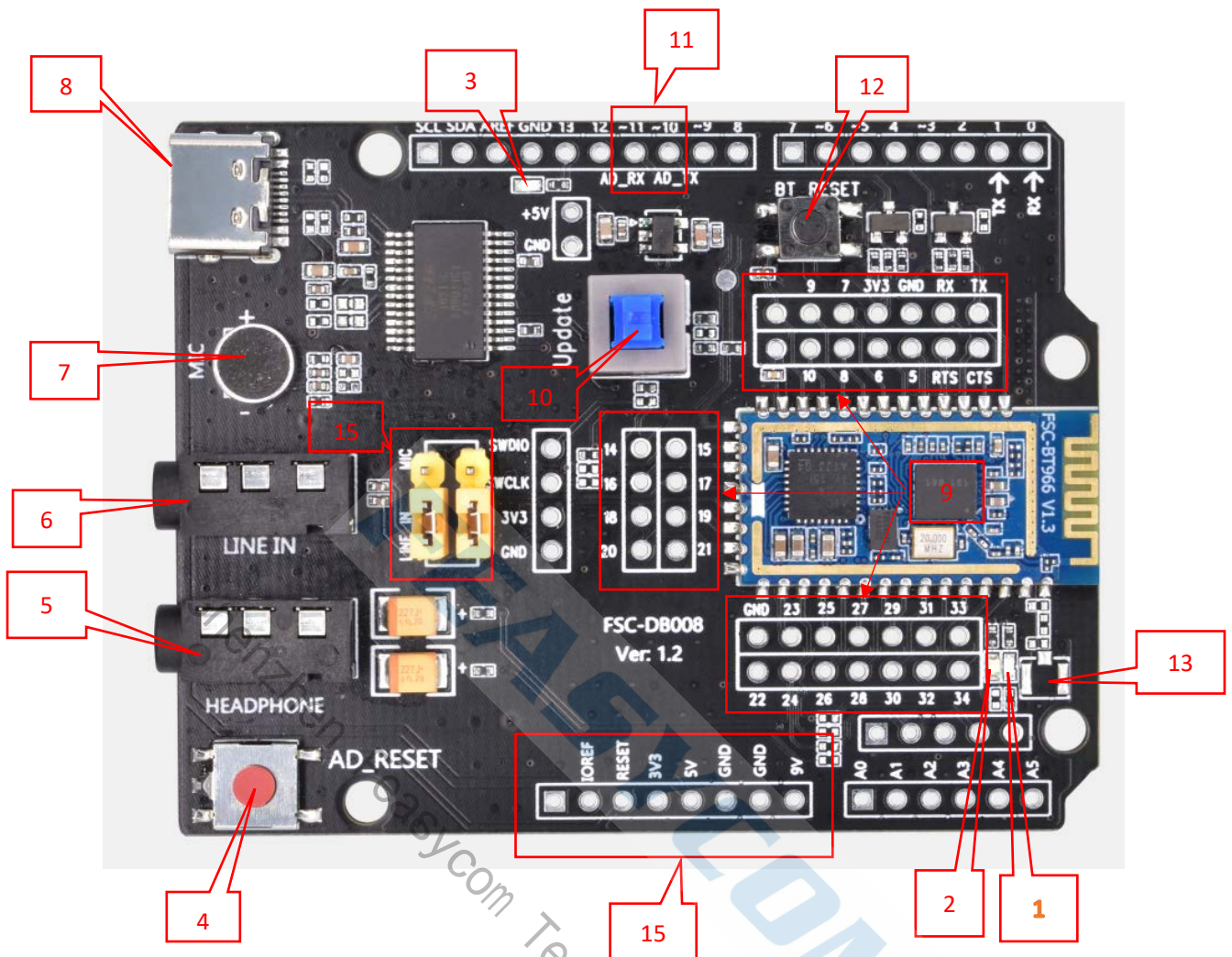


Figure 1. Product View

- 1, Red Bluetooth status LED will always be on when device connected.
- 2, The blue Bluetooth working LED flashes when the device is not connected, and it will always be on when device connected.
- 3, Blue Arduino reset LED.
- 4, Arduino reset button.
- 5, Headphone
- 6, Line in
- 7, MIC
- 8, Type-C power
- 9, Module pin
- 10, Download Key (BT966 use only)
- 11, Arduino UART-TX/RX
- 12, BT reset
- 13, RF(external)
- 14, Arduino Power
- 15, MIC/Line_IN select

3. Software Instructions

3.1 Overview

This example is based on the self-contained example named "SoftwareSerialExample" in Arduino UNO IDE.

This example implemented two functions:

1. Use the Serial Monitor (Tools > Serial Monitor) or smartphone send '1' to turn on the LED and send '0' to turn off.
2. Send AT commands to read or write the configuration of FSC-BT826 module.

3.2 Source Code

```
4. #include <SoftwareSerial.h>
5.
6. #define VERSION "Software Version: 1.0.1"
7.
8. #define LED_PIN 7
9. #define GetArrayNum(array) (sizeof(array)/sizeof(array[0]))
10.
11. SoftwareSerial mySerial(10, 11); // RX, TX
12. String str;
13.
14. void setup()
15. {
16.   pinMode(LED_PIN, OUTPUT);
17.   digitalWrite(LED_PIN, LOW);
18.   // Open serial communications and wait for port to open:
19.   Serial.begin(57600);
20.   while (!Serial) {
21.     ; // wait for serial port to connect. Needed for native USB port only
22.   }
23.   Serial.println("\r\nHello Bluetooth!\r\n");
24.   Serial.println(VERSION);
25.
26.   //get and set the data rate for the SoftwareSerial port
27.   getmySerialbaudrate();
28. }
29.
30. void loop()
31. {
32.   //delay(100);
```

```

33. // run over and over
34. if (mySerial.available())
35. {
36.     str = mySerial.readString();
37.     settledstatus(str);
38.     Serial.print(str);
39. }
40.
41. //delay(100);
42. if (Serial.available())
43. {
44.     str = Serial.readString();
45.     str+="\r\n";
46.     do
47.     {
48.         if(setledstatus(str)) break;
49.         if(setmySerialbaudrate(str))
50.         {
51.             Serial.print("\r\nOK\r\n");
52.             break;
53.         }
54.         mySerial.print(str);
55.     }while(0);
56. }
57. }
58.
59. int settledstatus(String str)
60. {
61.     if(str.startsWith("1"))
62.     {
63.         digitalWrite(LED_PIN, HIGH);
64.         return 1;
65.     }
66.     else if(str.startsWith("0"))
67.     {
68.         digitalWrite(LED_PIN, LOW);
69.         return 1;
70.     }
71.     return 0;
72. }
73.
74. int setmySerialbaudrate(String str)
75. {
76.     if(str.startsWith("AT+BAUD="))

```

```

77. {
78.     String baud,cmd;
79.     baud = str.substring(strlen("AT+BAUD="),str.length()-2); //\r\n
80.     cmd = "AT+BAUD="+baud+"\r\n";
81.     mySerial.print(cmd);
82.     mySerial.end();
83.     delay(100);
84.     mySerial.begin(baud.toInt());
85.     while(!mySerial)
86.     {
87.         Serial.print("\r\nwait");
88.     }
89.     return 1;
90. }
91. else return 0;
92. }
93.
94. void getmySerialbaudrate(void)
95. {
96.     unsigned long baudrate_table[]= {2400,4800,9600,19200,38400,57600,115200};
97.     unsigned long tick=0;
98.
99.     for(int i=0;i<GetArrayNum(baudrate_table);i++)
100.    {
101.        mySerial.begin(baudrate_table[i]);
102.        mySerial.print("AT\r\n");
103.        tick = millis(); //time out
104.        while(millis() - tick < 200)
105.        {
106.            if(mySerial.available())
107.            {
108.                str = mySerial.readString();
109.                if(str.indexOf("OK")>0)
110.                {
111.                    if(baudrate_table[i] > 57600)
112.                    {
113.                        // baudrate is too high for arduino software uart.
114.                        setmySerialbaudrate("AT+BAUD=57600\r\n");
115.                        i = 5; // 57600
116.                    }
117.
118.                    Serial.print("\r\nmySerialbaudrate=");
119.                    Serial.print(baudrate_table[i]);

```

```

120.         Serial.print("\r\n");
121.         return ;
122.     }
123.     break;
124. }
125. }
126. mySerial.end();
127. delay(50);
128. }
129.
130. //never run here if correctly
131. Serial.print("\r\nCannot get correct baudrate\r\n");
132. return ;
133. }

```

4. Program Initialization

There is a software serial port class in file <SoftwareSerial.h>, we needed to contain this headfile and declare a SoftwareSerial object.

In setup() method, we made some initialization:

1. Set baudrate of serial port and start the communication between your computer and Arduino board.
2. Iterate over baudrates to get the UART baudrate of the FSC-DB008 and start the communication between Arduino board and FSC-DB008.

Open Arduino IDE, open Serial Monitor (Tools > Serial Monitor), paste the source code in the Arduino IDE, compile and upload, when initialization succeed, you can see some messages printed on the Serial Monitor, Figure 3.

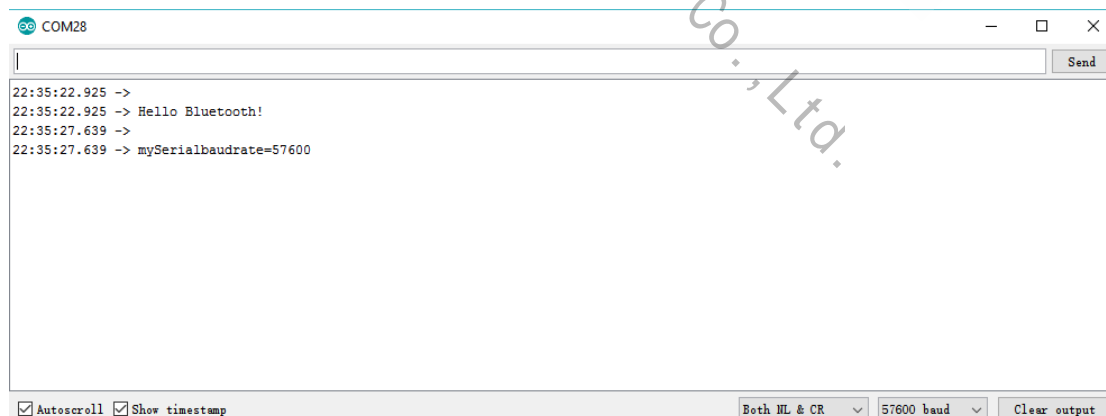


Figure 3. Initialization Succeed

4.1 Methods

In *loop()* method, the program will scan two serial ports whether there is data

come-in over and over again. If there is incoming data, method ***mySerial.available()*** or ***Serial.available()*** will return true. We used a String object named ***str*** which declared at the beginning to get the incoming data. Method ***setledstatus()*** is used to set ledstatus. Method ***setmySerialbaudrate()*** is used to reset the baudrate of the software serial port.

4.2 Function Test

After the program has been uploaded into the Arduino board, we can start test our function:

1. All AT commands should be ended up with New Line and Carriage Return, so select ***Both NL & CR*** at the bottom right of the Serial Monitor, choose the 57600 baudrate, below steps we will use the Serial Monitor to send AT commands to the Bluetooth module.
2. Send "AT+NAME" command (Figure 4), it will return the name of the Bluetooth module and you can use your smartphone to scan nearby Bluetooth devices to check the name.
3. Send "AT+NAME=HELLO" command and check it by your smartphone like step 2.
4. Send "AT+BAUD=38400" command to modify the baudrate of the software serial port. Send "AT+BAUD" command to check the baudrate whether it was modified successfully (Figure 5).
5. Install the app named ***FeasyBlue*** from App Store (e.g. Google Play) on your smartphone. Open FeasyBlue app, click the device named "HELLO" and establish connection (Figure 6).
6. Send "hello" from app (Figure 6), you can get the message in Serial Monitor (Figure 7).
7. Send "1" from app to turn on the LED and send "0" to turn off.

More AT commands can be accessed from module-related programming user guide.

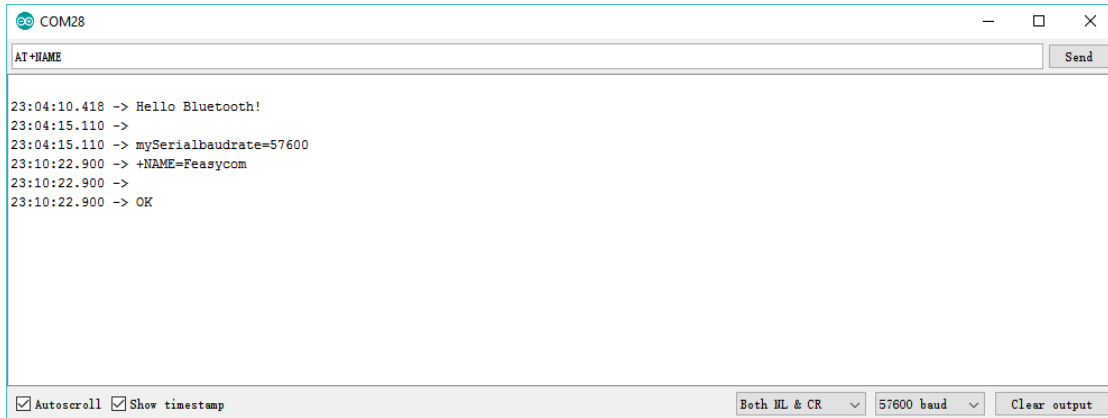


Figure 4. Send AT+NAME

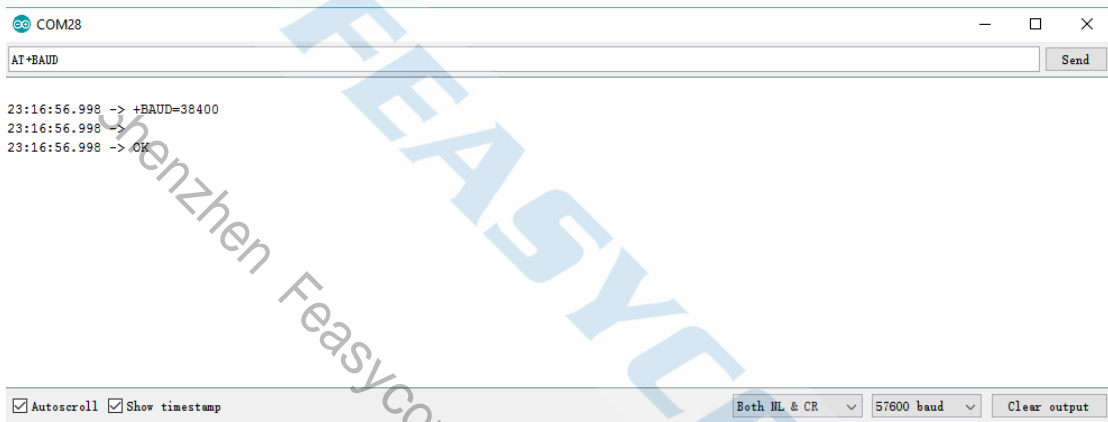


Figure 5. Send AT+BAUD

Shenzhen Feasycom Technology Co., Ltd.

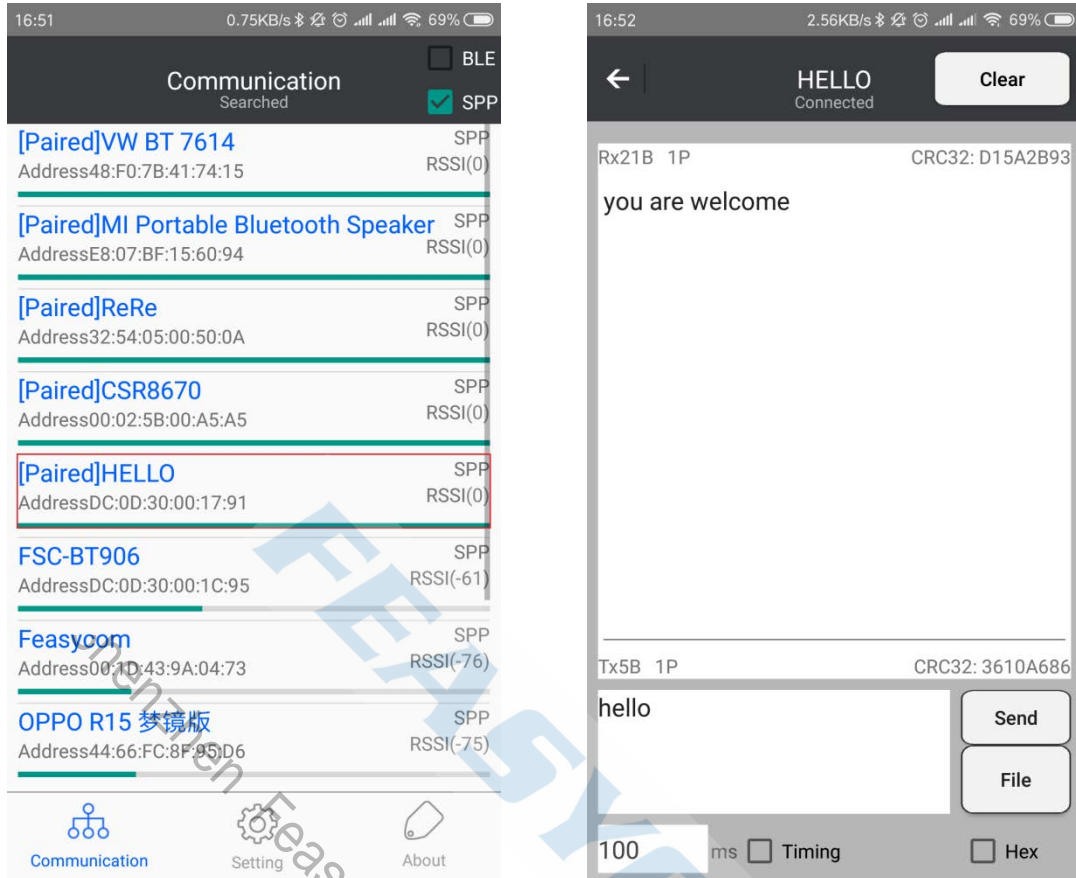


Figure 6. Connect and Transmission with FeasyBlue

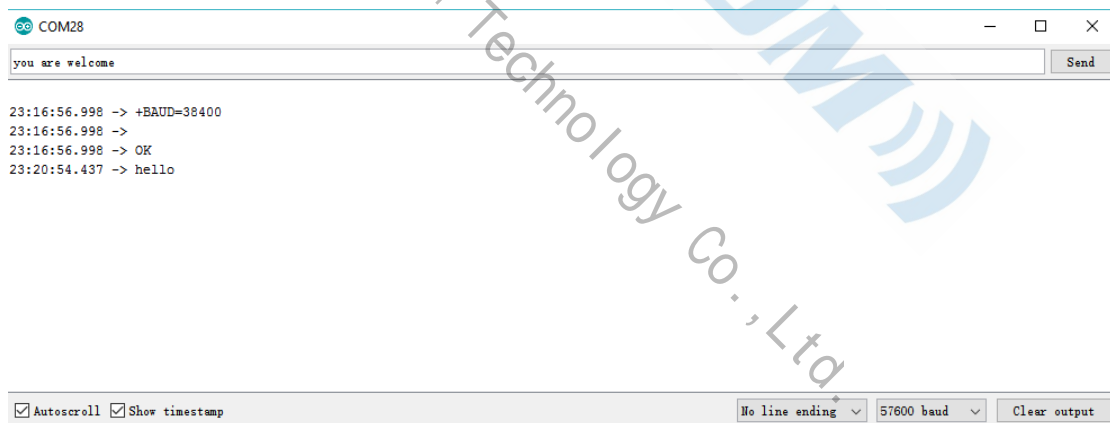


Figure 7. Transmission with Serial Monitor

5. Q&A

5.1 Why the Serial Monitor print "Cannot get correct baudrate"?

A: Please check whether the hardware connection between Arduino board and FSC-BT966 is correct. When FSC-BT966 is working normally, a yellow led on the module will twinkle. The software serial port of Arduino board can not support 115200 baudrate on RXD line, but FSC-BT966 default settings is 115200. You

need to modified the baudrate to 57600 by "AT+BAUD=57600" when the Serial Monitor prints "mySerialbaudrate=115200".

5.2 Why the program upload failed?

A: Please check if there is any software has occupied your serial port. If occupied, close the software.

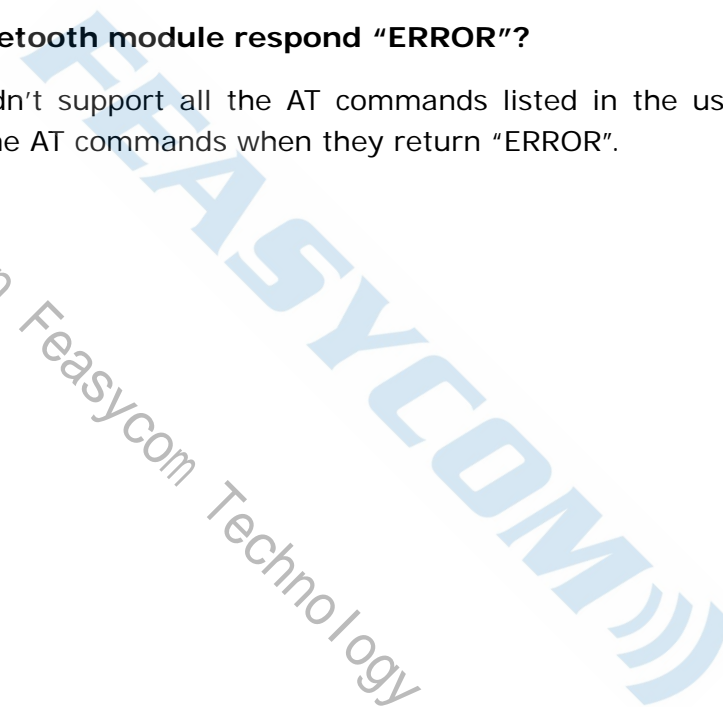
5.3 Why can't i get the response when i send AT command ?

A: Please check whether the AT command you sent is correct, . Illegal command will never be responded to. All AT commands should be ended up with New Line and Carriage Return, so select **Both NL & CR** at the bottom right of the Serial Monitor.

5.4 Why the bluetooth module respond "ERROR"?

A: FSC-BT966 didn't support all the AT commands listed in the user guide. Please ignore some AT commands when they return "ERROR".

Shenzhen Feasycom Technology Co., Ltd.

The logo for Feasycom, featuring the word "FEASYCOM" in a large, bold, blue sans-serif font. To the right of the text is a stylized blue icon of a signal tower or antenna with three curved lines representing signal waves.