Conectar y Listo - Serial Bluetooth

Contents

[hide]

Introduction

Grove - Serial Bluetooth is an easy to use module compatible with the existing Grove Base Shield, and designed for transparent wireless serial connection setup. The serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR(Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Bluecore 04-External single chip Bluetooth system with CMOS technology and with AFH(Adaptive Frequency Hopping Feature).It has the smallest footprint of 12.7mm x 27mm. Hope it will simplify your overall design/development cycle.

Model:WLS31746P (LCBTCY6)

Features

- Fully Qualified Bluetooth V2.0+EDR 3Mbps Modulation.
- Low Power Operation.
- PIO control.
- Grove interface
- Integrated PCB antenna.
- Baud rate can select
- Auto-connect the last device on power
- Permit matched device connect
- Auto-reconnect in 30 min when disconnected as a result of beyond the range of connection.

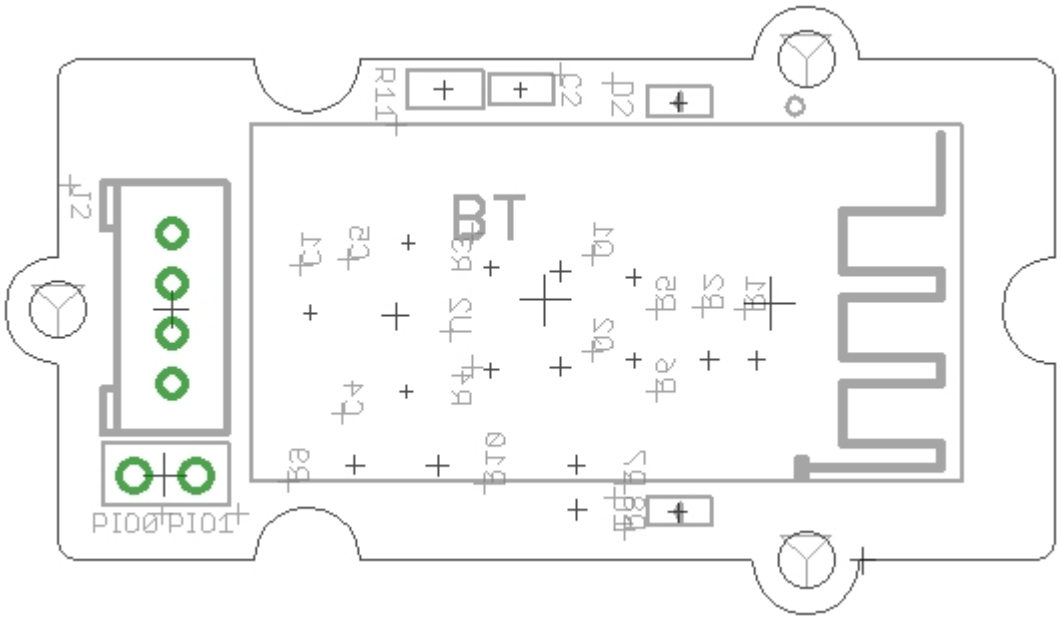
Specification

Item	Typical	Unit
Voltage	5.0	VDC
Data Rate	2	Mbps
RF Transmit Power (Max)	+4	dBm
Sensitivity	-80	dBm

Cautions

- While using with Seeeduino / Arduino, set the operation voltage to 5V. Else use a proper logic level converter.
- While using with UartSBee, set the operation voltage to 5V
- Command to change baud rate is persistent even after reset. Hence remember the baud rate for next use.

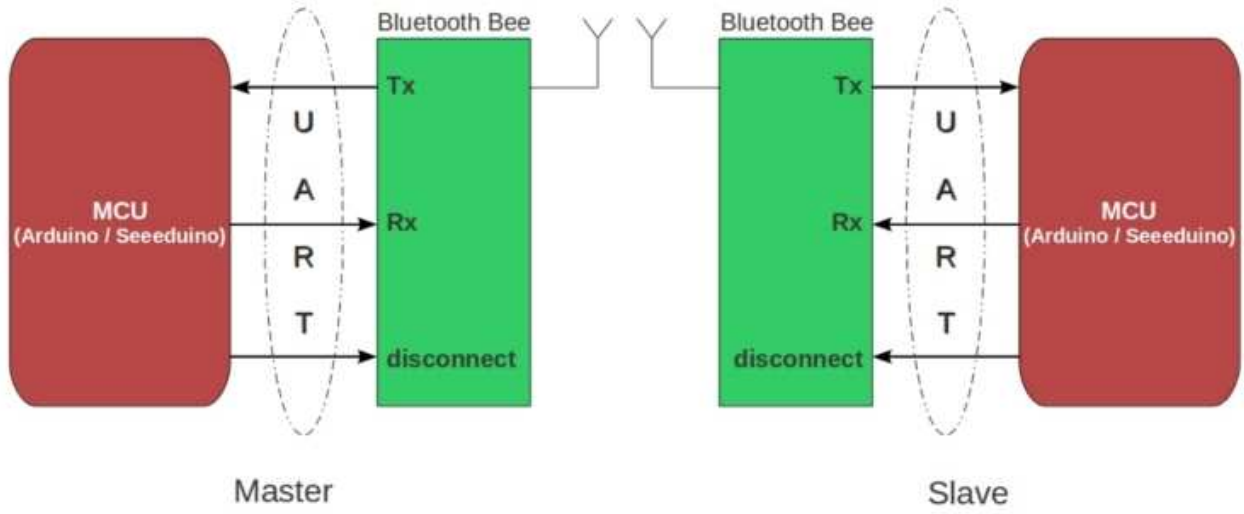
Interface Function



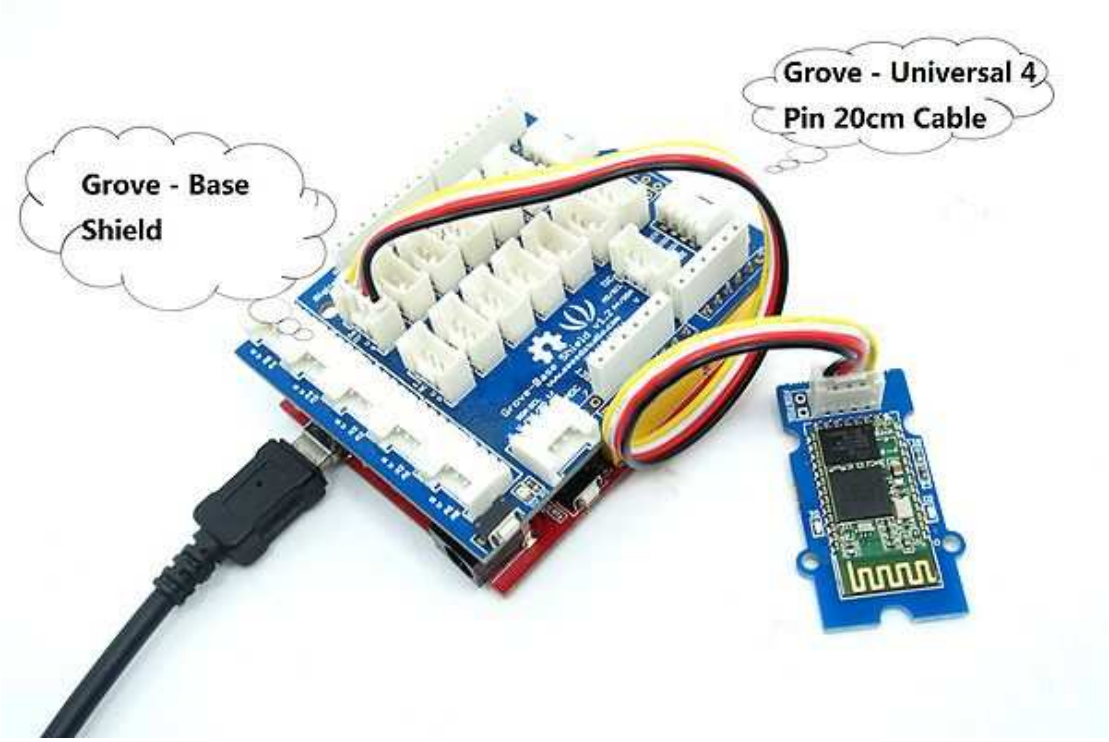
Pad Type	Description
GND	Ground port
PIO1	Status instruction port PIO1: low-disconnected, high-connected
PIO0	When a rising pulse is detected in PIO0, device will be disconnected
RX	UART Data input
TX	UART Data output
VCC	Designed for 5V(+)supply using IC MIC5205_3.3 for Bluetooth module power supply

Usage

Two Bluethooth module work sketch map as show below:



- Connect the module to D6e Port of Grove - Base Shield:



- Download the demo code, upload the skech to Arduino and reset it. When it at pairing status ,the green and red led blinks in interval ,and only the green led blinks 1 time/2s after connecions .When it fails or disconnects the green led blinks 2 times/s, check if the hardware connection and baud rate is right.And please do remember the baud rate every time you set.

```
'''This code work as Master'''
// Please note that this module can't be used to connected your
Phone/PC as Master

#include <SoftwareSerial.h>   //Software Serial Port

#define RxD 6
#define TxD 7

String retSymb = "+RTINQ=";//start symble when there's any return
String slaveName = ";SeeedBTSlave";//Set the Slave name ,caution that
';'must be included
int nameIndex = 0;
int addrIndex = 0;

String recvBuf;
String slaveAddr;

String connectCmd = "\r\n+CONN=";

SoftwareSerial blueToothSerial(RxD,TxD);

void setup()
{
  Serial.begin(9600);
  pinMode(RxD, INPUT);
  pinMode(TxD, OUTPUT);
  setupBlueToothConnection();
  //wait 1s and flush the serial buffer
  delay(1000);
  Serial.flush();
  blueToothSerial.flush();
}

void loop()
{
  char recvChar;
  while(1){
    if(blueToothSerial.available()){//check if there's any data sent
from the remote bluetooth shield
      recvChar = blueToothSerial.read();
      Serial.print(recvChar);
    }
```

```
    if(Serial.available()){//check if there's any data sent from the
local serial terminal, you can add the other applications here
      recvChar  = Serial.read();
      blueToothSerial.print(recvChar);
    }
 }
}

void setupBlueToothConnection()
{
  blueToothSerial.begin(38400); //Set BluetoothBee BaudRate to default
baud rate 38400
  blueToothSerial.print("\r\n+STWMOD=1\r\n");//set the bluetooth work
in master mode
  blueToothSerial.print("\r\n+STNA=SeeedBTMaster\r\n");//set the
bluetooth name as "SeeedBTMaster"
  blueToothSerial.print("\r\n+STPIN=0000\r\n");//Set Master
pincode"0000",it must be same as Slave pincode
  blueToothSerial.print("\r\n+STAUTO=0\r\n");// Auto-connection is
forbidden here
  delay(2000); // This delay is required.
  blueToothSerial.flush();
  blueToothSerial.print("\r\n+INQ=1\r\n");//make the master inquire
  Serial.println("Master is inquiring!");
  delay(2000); // This delay is required.

  //find the target slave
  char recvChar;
  while(1){
    if(blueToothSerial.available()){
      recvChar = blueToothSerial.read();
      recvBuf += recvChar;
      nameIndex = recvBuf.indexOf(slaveName);//get the position of
slave name
      //nameIndex -= 1;//decrease the ';' in front of the slave name,
to get the position of the end of the slave address
      if ( nameIndex != -1 ){
        //Serial.print(recvBuf);
        addrIndex = (recvBuf.indexOf(retSymb,(nameIndex -
retSymb.length()- 18) ) + retSymb.length());//get the start position
of slave address
        slaveAddr = recvBuf.substring(addrIndex, nameIndex);//get the
string of slave address
        break;
```

```
      }
    }
  }
  //form the full connection command
  connectCmd += slaveAddr;
  connectCmd += "\r\n";
  int connectOK = 0;
  Serial.print("Connecting to slave:");
  Serial.print(slaveAddr);
  Serial.println(slaveName);
  //connecting the slave till they are connected
  do{
    blueToothSerial.print(connectCmd);//send connection command
    recvBuf = "";
    while(1){
      if(blueToothSerial.available()){
        recvChar = blueToothSerial.read();
        recvBuf += recvChar;
        if(recvBuf.indexOf("CONNECT:OK") != -1){
          connectOK = 1;
          Serial.println("Connected!");
          blueToothSerial.print("Connected!");
          break;
        }else if(recvBuf.indexOf("CONNECT:FAIL") != -1){
          Serial.println("Connect again!");
          break;
        }
      }
    }
  }while(0 == connectOK);
}
```

This code work as Slave

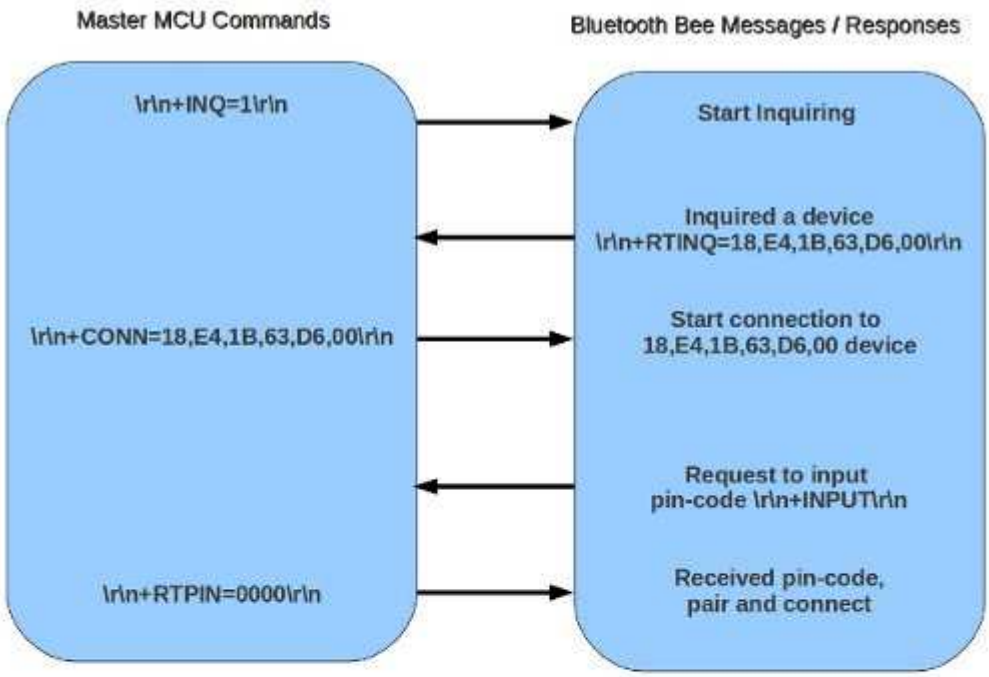
```
#include <SoftwareSerial.h> //Software Serial Port
#define RxD 6
#define TxD 7

SoftwareSerial blueToothSerial(RxD,TxD);
```

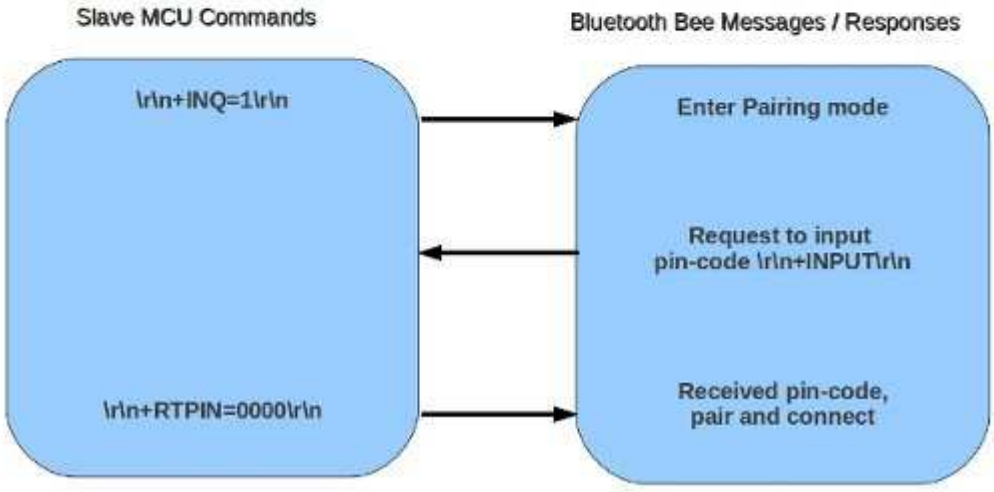
```
void setup()
{
 Serial.begin(9600);
 pinMode(RxD, INPUT);
 pinMode(TxD, OUTPUT);
 setupBlueToothConnection();
}

void loop()
{
 char recvChar;
 while(1){
 if(blueToothSerial.available()){//check if there's any data sent from
the remote bluetooth shield
 recvChar = blueToothSerial.read();
 Serial.print(recvChar);
 }
 if(Serial.available()){//check if there's any data sent from the
local serial terminal, you can add the other applications here
 recvChar = Serial.read();
 blueToothSerial.print(recvChar);
 }
 }
}

void setupBlueToothConnection()
{
 blueToothSerial.begin(38400); //Set BluetoothBee BaudRate to default
baud rate 38400
 blueToothSerial.print("\r\n+STWMOD=0\r\n"); //set the bluetooth work
in slave mode
 blueToothSerial.print("\r\n+STNA=SeeedBTSlave\r\n"); //set the
bluetooth name as "SeeedBTSlave"
 blueToothSerial.print("\r\n+STPIN=0000\r\n");//Set SLAVE
pincode"0000"
 blueToothSerial.print("\r\n+STOAUT=1\r\n"); // Permit Paired device
to connect me
 blueToothSerial.print("\r\n+STAUTO=0\r\n"); // Auto-connection should
be forbidden here
 delay(2000); // This delay is required.
 blueToothSerial.print("\r\n+INQ=1\r\n"); //make the slave bluetooth
inquirable
 Serial.println("The slave bluetooth is inquirable!");
 delay(2000); // This delay is required.
```

```
 blueToothSerial.flush();
}
```

Reference

Flowchat





Commands to change default configuration

1. Set working MODE

\r\n+STWMOD=0\r\n	Set device working mode as client (slave). Save and Rest.
\r\n+STWMOD=1\r\n	Set device working mode as server (master). Save and Rest.

Note: \r\n is necessary for operation and the value of are **0x0D 0x0A** in Hex. **\r** and **\n** represent **carriage-return** and **line-feed**(or next line),

2.Set BAUDRATE

\r\n+STBD=115200\r\n	Set baudrate 115200. Save and Rest.
Supported baudrate: 9600, 19200,38400,57600,115200,230400,460800.	

3. Set Device NAME

\r\n+STNA=abcdefg\r\n	Set device name as “abcdefg”. Save and Rest.

4. Auto-connect the last paired device on power

\r\n+STAUTO=0\r\n	Auto-connect forbidden. Save and Rest.
\r\n+STAUTO=1\r\n	Permit Auto-connect. Save and Rest.

5. Permit Paired device to connect me

\r\n+STOAUT=0\r\n	Forbidden. Save and Rest.
\r\n+STOAUT=1\r\n	Permit. Save and Rest.

6. Set PINCODE

\r\n+STPIN=2222\r\n	Set pincode “2222”, Save and Rest.

7. Delete PINCODE(input PINCODE by MCU)

\r\n+DLPIN\r\n	Delete pincode. Save and Rest.

8. Read local ADDRESS CODE

\r\n+RTADDR\r\n	Return address of the device.

9. Auto-reconnecting when master device is beyond the valid range (slave device will auto-reconnect in 30 min when it is beyond the valid range)

\r\n+LOSSRECONN=0\r\n	Forbid auto-reconnecting.
\r\n+LOSSRECONN=1\r\n	Permit auto-reconnecting.

Commands for Normal Operation:

1. Inquire

a) Master	
\r\n+INQ=0\r\n	Stop Inquiring
\r\n+INQ=1\r\n	Begin/Restart Inquiring
b) Slave	
\r\n+INQ=0\r\n	Disable been inquired
\r\n+INQ=1\r\n	Enable been inquired

When **+INQ=1** command is successful, the red and green LEDS blink alternatively.

2. Bluetooth module returns inquiring result

\r\n+RTINQ=aa,bb,cc,dd,ee,ff;name\r\n	Serial Bluetooth device with the address “aa,bb,cc,dd,ee,ff” and the name “name” is inquired

3. Connect device

\r\n+CONN=aa,bb,cc,dd,ee,ff\r\n	Connect to a device with address of "aa,bb,cc,dd,ee,ff”

4. Bluetooth module requests inputting PINCODE

\r\n+INPIN\r\n

5. Input PINCODE

\r\n+RTPIN=code\r\n	
Example: RTPIN=0000	Input PINCODE which is four zero

6. Disconnect device Pulling PIO0 high will disconnect current working Bluetooth device.

7. Return status \r\n+BTSTA:**xx**\r\n

xx status:

- 0 - Initializing
- 1 - Ready
- 2 - Inquiring
- 3 - Connecting
- 4 - Connected

(**Note:** This is not a command, but the information returned from the module after every command)