Introduction

The Bluetooth Shield integrates a Serial Bluetooth module. It can be easily used with Arduino/Seeedstudio for transparent wireless serial communication. You can choose two pins from Arduino D0 to D7 as Software Serial Ports to communicate with Bluetooth Shield (D0 and D1 is Hardware Serial Port). The shield also has two Grove connectors (one is Digital, the other is Analog) for you to install Grove modules.

Features

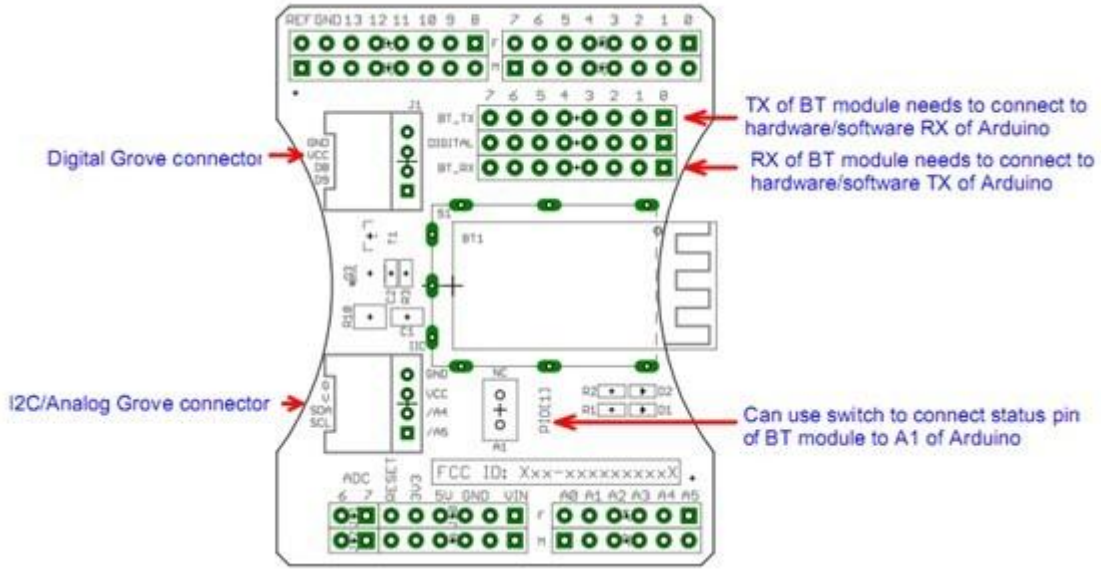
- Seeeduino/Arduino compatible
- Up to10m communication distance in house without obstacle
- UART interface (TTL) with programmable baud rate (SPP firmware installed)
- Default Baud rate: 38400, Data bits: 8, Stop bit: 1, Parity: No parity
- Default PINCODE:"0000"
- A full set of configuration commands
- On board PCB Antenna
- FCC ID Certificated

Specification

Item	Min	Typical	Max	Unit
Voltage	2.8	3.3	3.5	VDC

Current	3	/	50	mA
Communication Distance(in house)	/	/	10	m
Protocol	Bluetooth V2.0 with SPP firmware			/
Interface	Uart Serial Port(TTL)			/
Supported Baudrate	9600, 19200, 38400, 57600, 115200, 230400, 460800			bps
ESD contact discharge	±4			KV
ESD air discharge	±8			/
Dimension	57.4x45.3x19.4			mm
Net Weight	10±2			g

Interface funcion



Pad Type	Description
PIO1	Status instruction port of Bluetooth module can be read by Arduino A1 port: low-disconnected, high-connected.
BT_RX	UART Data input of Bluetooth module.
BT_TX	UART Data output Bluetooth module.

Two Grove connectors	One is Digital (D8 and D9), the other is I2C/Analog (A4 and A5).

Usage

Hardware Installation

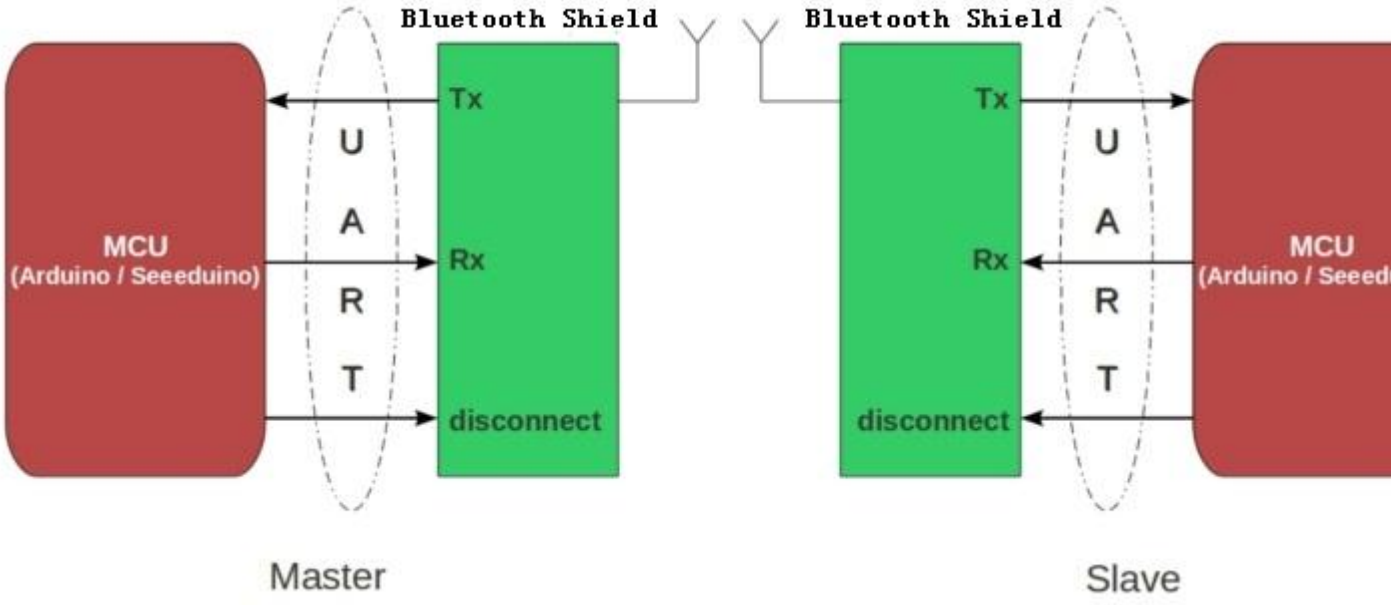
Connecting to Arduino with Bluetooth Shield

Connect **Bluetooth Shield** to Seeeduino / Arduino.

Software Instruction

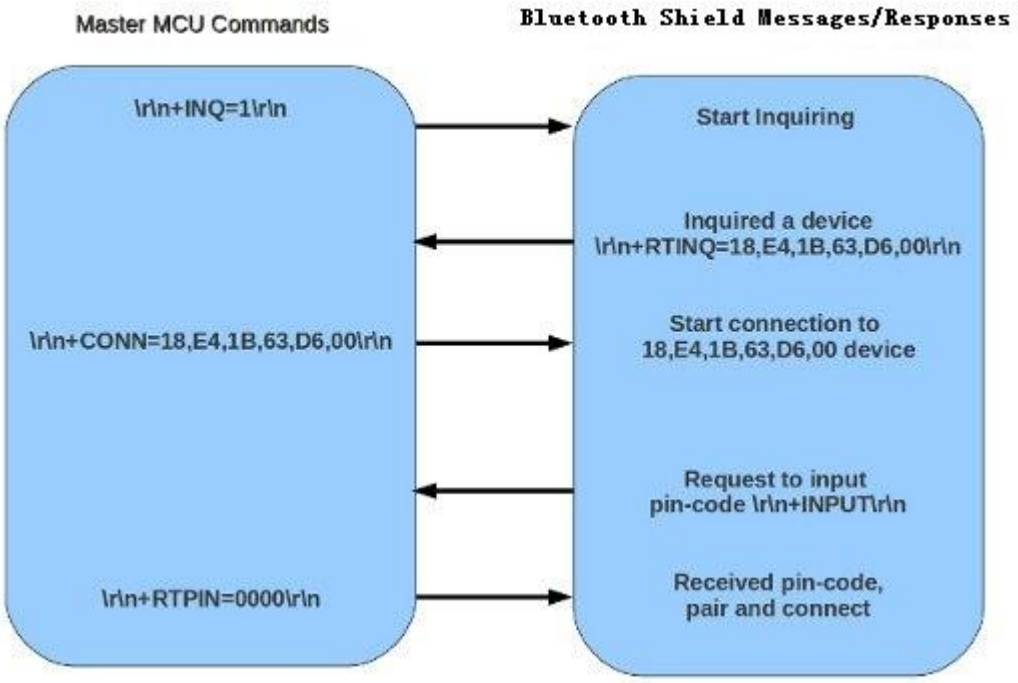
Working Sketch Map

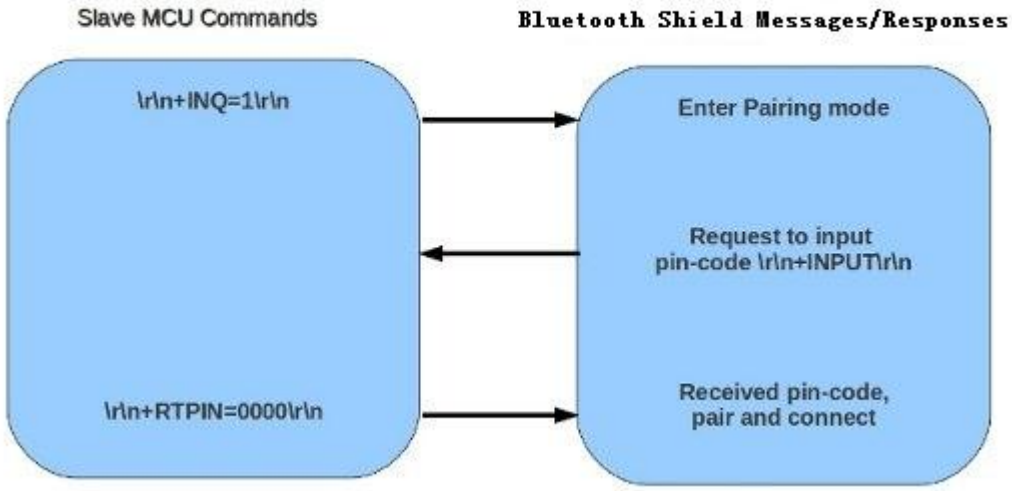
The following sketch presents an overview of **Bluetooth Shield** operation in master and slave mode.



Flowchat

The following flowchart gives a quick start guide to **Bluetooth Shield** programming.





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	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)

Programming

The following sketch configures **Bluetooth Shield** as **Slave Device** and waits for connection request from PC or other master device. Bluetooth Shield is connected to Seeeduino via Stem - Basic Shield as shown above.

```
/*
BluetoothBee Demo Code
2010 Copyright (c) Seeed Technology Inc.  All right reserved.

Author: Visweswara R

This demo code is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
```

```
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301
USA

For more details about the product please check
http://www.seeedstudio.com/depot/

*/


/* Upload this sketch into Seeeduino and press reset*/

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

#define DEBUG_ENABLED  1

NewSoftSerial blueToothSerial(RxD,TxD);

void setup()
{
    pinMode(RxD, INPUT);
    pinMode(TxD, OUTPUT);
    setupBlueToothConnection();

}

void loop()
{

  if(blueToothSerial.read() == 'a')

  {
    blueToothSerial.println("You are connected to Bluetooth Shield");
    //You can write you BT communication logic here
  }


}

void setupBlueToothConnection()
```

```
{
    blueToothSerial.begin(38400); //Set BluetoothBee BaudRate to default
baud rate 38400
    delay(1000);
    sendBlueToothCommand("\r\n+STWMOD=0\r\n");
    sendBlueToothCommand("\r\n+STNA=SeeeduinoBluetooth\r\n");
    sendBlueToothCommand("\r\n+STAUTO=0\r\n");
    sendBlueToothCommand("\r\n+STOAUT=1\r\n");
    sendBlueToothCommand("\r\n +STPIN=0000\r\n");
    delay(2000); // This delay is required.
    sendBlueToothCommand("\r\n+INQ=1\r\n");
    delay(2000); // This delay is required.

}

//Checks if the response "OK" is received.
void CheckOK()
{
  char a,b;
  while(1)
  {
    if(int len = blueToothSerial.available())
    {
    a = blueToothSerial.read();

    if('O' == a)
    {
      b = blueToothSerial.read();
      if('K' == b)
      {
        break;
      }

    }
   }
  }

  while( (a = blueToothSerial.read()) != -1)
  {
    //Wait until all response chars are received
  }
}

//Send the command to Bluetooth Shield
void sendBlueToothCommand(char command[])
```

```
{
    blueToothSerial.print(command);
    CheckOK();
}
```

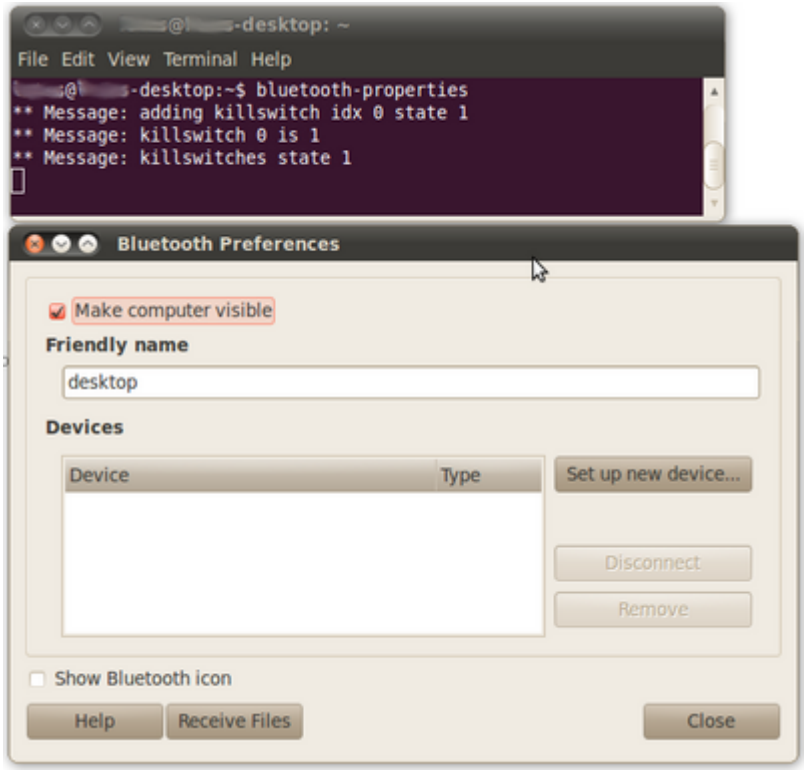
Connecting Bluetooth Shield to PC (via Bluetooth Dongle) under GNU/Linux

This procedure demonstrates how **Bluetooth Shield** can be connected to PC under GNU/Linux OS. An USB Bluetooth Dongle is used at PC side to communicate with **Bluetooth Shield**. The above sketch is uploaded to Seeeduino.

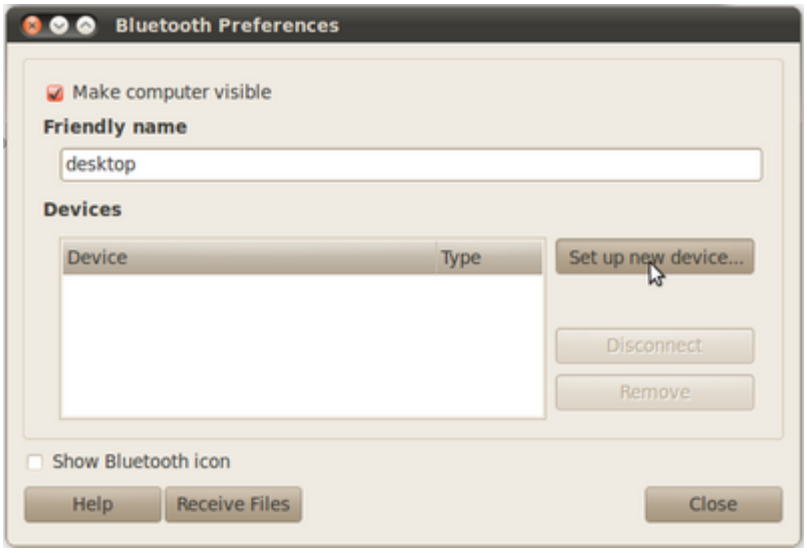
- Install gnome-bluetooth

```
sudo apt-get install gnome-bluetooth
```

- Open **bluetooth-properties** application from shell



- Click **Set up new device**



and click **Forward**

- Open **PIN options...**

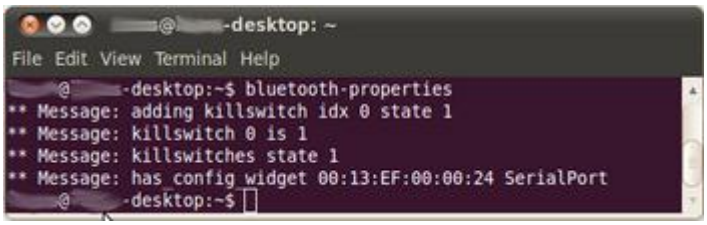
- Set Fixed PIN **0000**. 0000 is the default pin used in the above sketch.

- Device **Setup Window** opens

and

Setup Completed dialog opens. Click **Close**.

- The address of the **Bluetooth Shield** **00:13:EF:00:00:24** is displayed in shell.

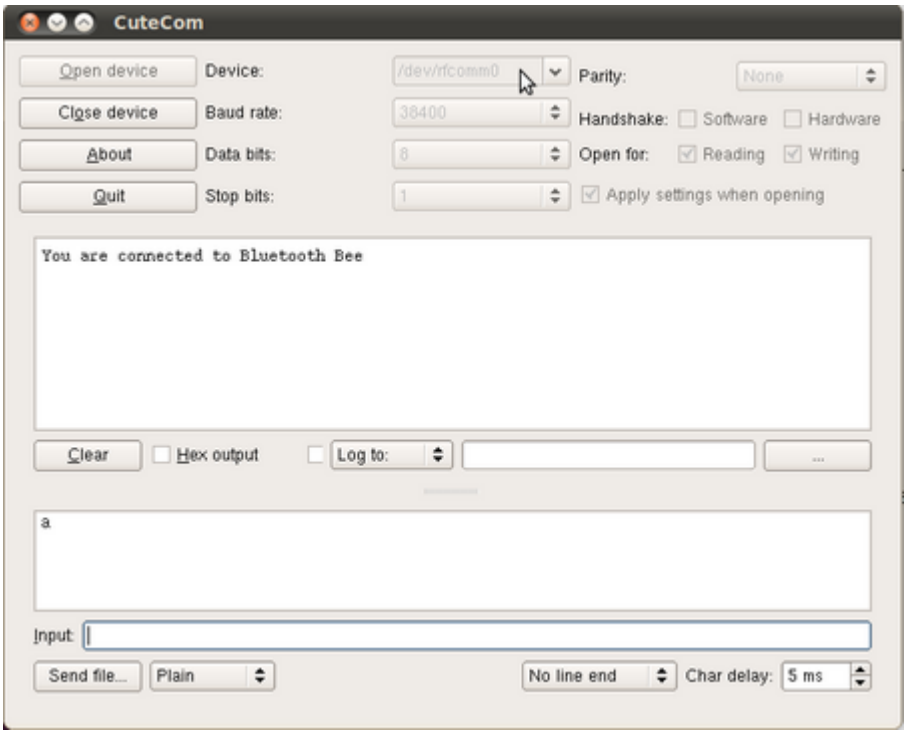


- Bind the **Bluetooth Shield** to rfcomm port. Here the address of **Bluetooth Shield** is bound to a serial port device **/dev/rfcomm0**

```
user@user-desktop:~$ sudo rfcomm bind 0 00:13:EF:00:00:24 1
```

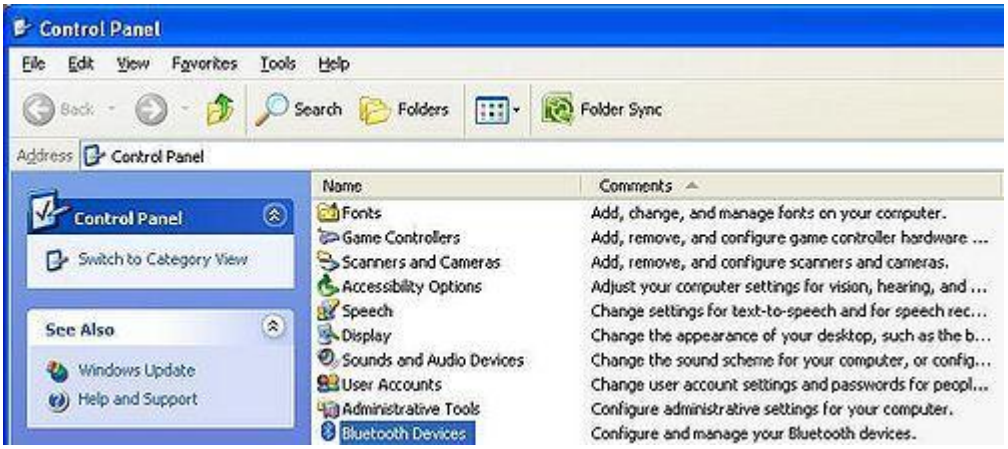
```
user@user-desktop:~$ ls /dev/rfcomm*

/dev/rfcomm0
```

- This **/dev/rfcomm0** serial port can be accessed by any serial port terminal like **cutecom**.
  - Open **/dev/rfcomm0** with baud rate:**38400**, Databits: **8**, Stopbits: **1** and **No Flow Control**
  - Send character 'a'
  - Seeeduino + Bluetooth Shield will reply with "**You are connected to Bluetooth Shield**"

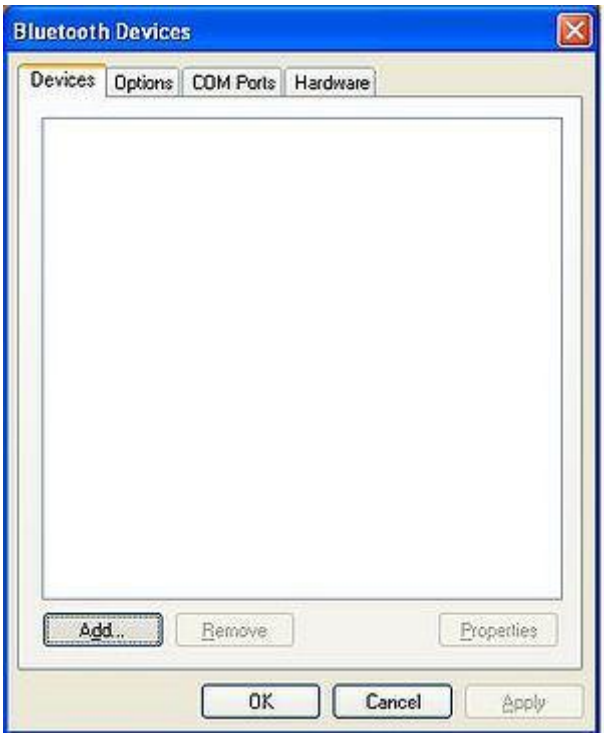


Connecting Bluetooth Shield to PC (via Bluetooth Dongle) under Windows

- Install the Microsoft Bluetooth default drivers. Open **Control Panel** -> **Bluetooth Devices**



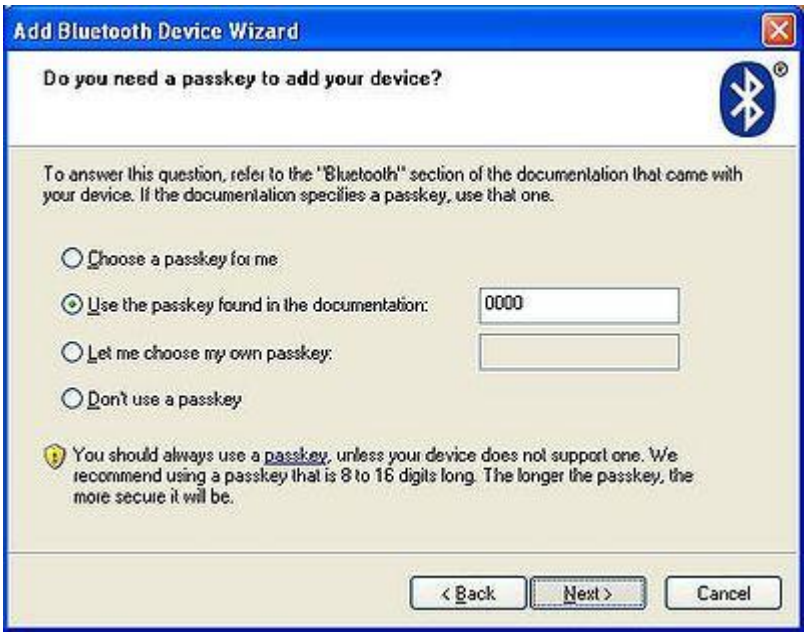
- Click **Add** button.



- Check **My devices is set up and ready to be found** and click **Next** button

- Select the "**SeeedBlueToothBee**" device and click **Next**.

- Select **Use the passkey found in the documentation** and enter **0000**

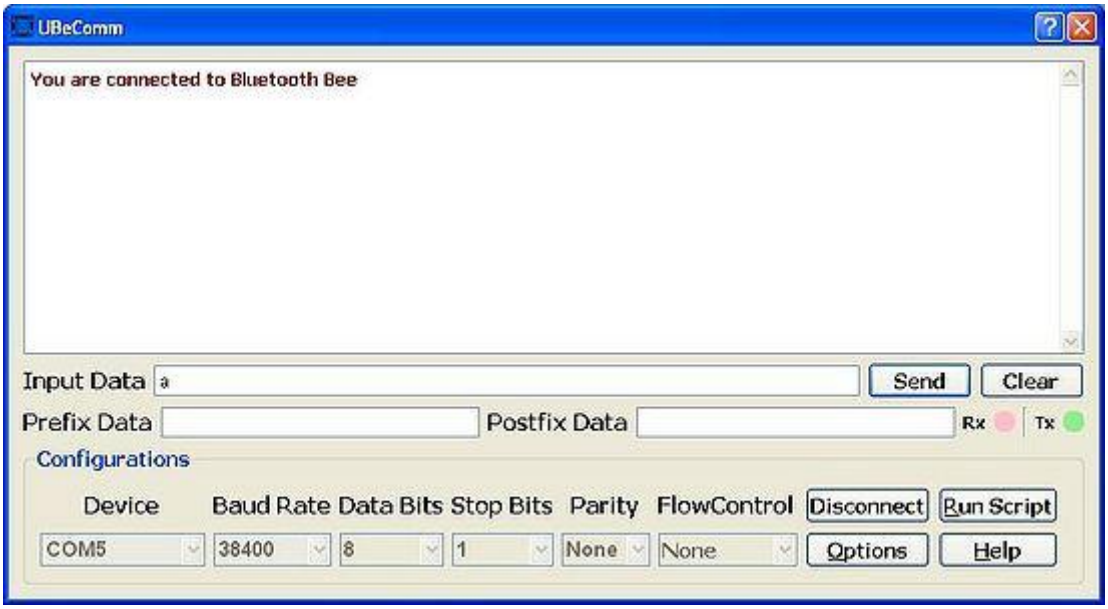


- Passkeys are exchanged and an outgoing serial port **COM5** is assigned for our communication.

-
- A task-bar balloon shows that a new Bluetooth Serial Port link is added.

-
- **COM5** is assigned for communication. This port should be used to communicate PC with Bluetooth Shield.

- This **COM5** serial port can be accessed by any Serial Port terminal.
  - Open **COM5** with baud rate:**38400**, Databits: **8**, Stopbits: **1** and **No Flow Control**
  - Send character 'a'
  - Seeeduino + Bluetooth Shield will reply with "**You are connected to Bluetooth Shield**"



Connecting Bluetooth Shield to PC using UartSBee in Master Mode

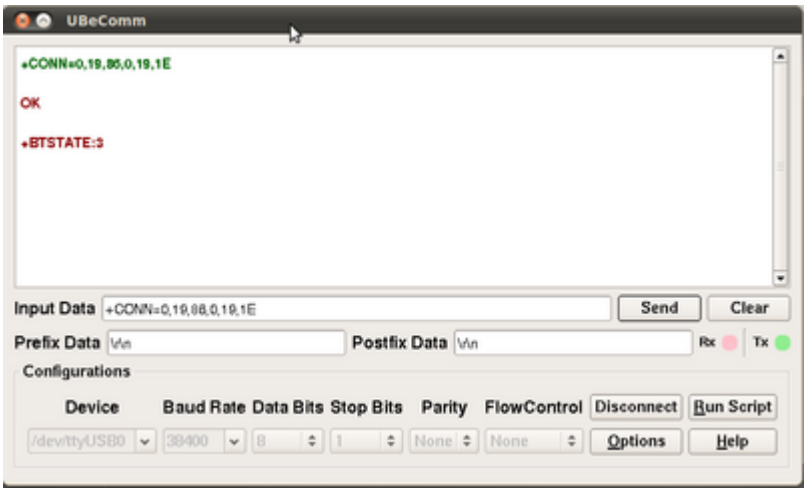
This demo uses hardware arrangement described in Hardware Installation - UartSBee. Connect UartSbee to PC using a mini USB cable.

- Open a serial terminal and connect to UartSBee serial port device like **COM1** in Windows or **/dev/ttyUSB0** in GNU/Linux with baud rate:**38400**, Databits: **8**, Stopbits: **1** and **No Flow Control**

- Send **\r\n+STWMOD=1\r\n** command. This configures the **Bluetooth Shield** in master mode.

- Send **\r\n+INQ=1\r\n** command. **Bluetooth Shield** searches available Bluetooth devices in neighborhood and list the address of the devices. In this case with **+RTINQ=0,19,86,0,19,1E;desktop**, where **desktop** is a PC with Bluetooth interface. While inquiring the red and green led blinks alternatively.

- Send **\r\n+CONN=0,19,86,0,19,1E\r\n** where **0,19,86,0,19,1E** is the address of desktop.



- Enter the pin 0000 at PC side and complete the connection.