Lab 8. Communications between Arduino and Android via Bluetooth

Dr. X. Li

xhli@citytech.cuny.edu

Dept. of Computer Engineering Technology

New York City College of Technology

In this class, we will learn how to connect Arduino to an Android smartphone via Bluetooth, and use a Bluetooth app on Android to control Arduino. The techniques introduced in this hand-out can be also applied to iOS smartphone or tablet. You only need to find a proper app for the iOS device.

To connect Arduino with Android through Bluetooth, you need

1. An Arduino board (e.g., Uno)
2. A Bluetooth serial board capable to be connected to the Arduino (e.g., Bluetooth Mate Silver by Sparkfun)
3. An Android device with Bluetooth (e.g., an Android smartphone or tablet)
4. A Bluetooth app on Android device (e.g., "Connection Terminal" from Google Play)

If you use iOS mobile device, you need to search and download a proper Bluetooth app from Apple app store.

1. Arduino Side:

Hardware Setup

The following figure shows the Bluetooth serial board made by Sparkfun, named as "Bluetooth Mate Silver". It uses a bluetooth IC, "RN-42", from Roving Networks Inc.

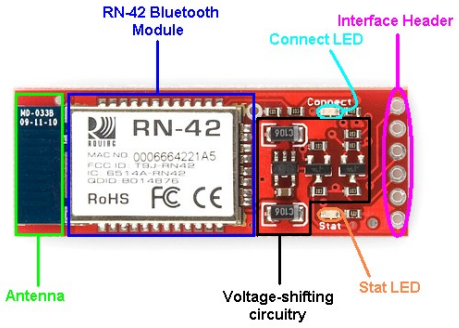

Figure Copyright by Sparkfun.com

As you can see, a major chunk of the Mate board is Roving Network's **RN-42 Bluetooth IC** module. The RN-42 is the brain of the device, it's what handles all the complicated Bluetooth protocol, and it's what you'll talk to over a hard-wired serial connection. In the figure, the blue

rectangle next to the RN-42 is an **antenna**. Just keep in mind that the antenna is there, and don't try to interfere with it. Next to the RN-42 there are a number of other components, including voltage regulators, transistors, resistors, etc. which are used to regulate the voltages, introduce input and output pins for RN-42. The voltages that go into the RN-42 can't stray too far from 3.3V, so we've put all that level-regulating circuitry on there so you can use it at a more Arduino-accessible 5V. Enveloped by the level-regulating stuff are two **LEDs** that indicate the status of the Bluetooth Mate. The green 'Connect' LED will illuminate when a wireless connection is formed, and the red 'Stat' LED blinks at varying speeds to indicate the state of the RN-42 as in the following table. Of course, different Bluetooth board may have different LED indications.

Mode	**Stat LED Blink Rate**
Configuration	10 times per second
Startup/Config Timer	2 times per second
Discoverable/Inquiring/Idle	1 per second
Connected	Solid

Finally, the six-pin header of the Bluetooth Mate has everything you'll need to power the device, and send and receive data. Labels on the bottom of the board read 'GND', 'CTS-I', 'VCC', 'TX-O', 'RX-I', and 'RTS-O'. GND and VCC are your power pins, RTS-O and CTS-I are flow control signals, which can really be ignored in most applications. And the most importantly, **RX-I and TX-O** are the receive and transmit pins, in which **"I"** means input and "**O**" means output.

In general, the 4 pins you need to use are: **VCC, GND, RX-I and TX-O.** Connect GND, VCC, RX-I and TX-O of the bluetooth board to the appropriate Arduino ports.

Note that: theoretically you can connect RX-I and TX-O pins to any digital ports of Arduino, but some Bluetooth boards may require or recommend to be connected to Arduino's RX and TX ports. Read its datasheet for detailed instructions.

Arduino Sketch

A library named as "SoftwareSerial" is provided with Arduino IDE for you to program communication via two digital pins to a Bluetooth device connected to Arduino. "SoftwareSerial" library provides several functions to receive and transmit data through the connected Bluetooth board.

The functions in "SoftwareSerial" library available for Bluetooth communications are:

- SoftwareSerial() //To define a Bluetooth software serial communication
- available() //To detect any data being received by Bluetooth
- begin() //To set the Bluetooth baud rate (data transmission rate)
- overflow()
- peek()
- read() //To read in data received by Bluetooth RX pin

- print() //To print data to Bluetooth TX pin. Similar to Serial.print().
- println() //To print data to Bluetooth TX pin. Similar to Serial.println().
- listen()
- write() //To output data **as raw bytes** to Bluetooth TX pin

Important:

```
SoftwareSerial mySoftSerial =  SoftwareSerial(rxPin, txPin);
```

In which parameters are:

`rxPin`: The pin on which Arduino receives serial data. It should be connected to the TX pin of external Bluetooth device!

`txPin`: The pin on which Arduino transmit serial data. It should be connected to the RX pin of external Bluetooth device!

Note:

- If using multiple software serial ports, only one can receive data at a time.
- If you use Mega or Mega 2560, only the following ports can be used for RX: 10, 11, 12, 13, 14, 15, 50, 51, 52, 53, A8 (62), A9 (63), A10 (64), A11 (65), A12 (66), A13 (67), A14 (68), A15 (69).

Before you start writing the code, you need to make sure you set the SoftwareSerial baud rate to match the baud rate of your bluetooth board. See the datasheet of your Bluetooth board for default baud rate.

The follows are a sample sketch to receive and transmit data via the Bluetooth mate connected to an Arduino Uno. Please read it thoroughly and understand every single line.

```
#include <SoftwareSerial.h>  //To use the SoftwareSerial libary

SoftwareSerial myBluetooth(4, 5); //Define a Bluetooth software serial that use
                    //  Arduino's port4 as rxPin and port5 as txPin, i.e., connects
                    //  Bluetooth's TX pin to Arduino port4, RX pin to Arduino port5.

void setup()
{
  Serial.begin(9600);            //Setup baud rate for Serial Monitor
  Serial.println("Hello, Serial Monitor is activated!");

  pinMode(4, INPUT);      //Set Arduino's port4 as INPUT
  pinMode(5, OUTPUT);     //Set Arduino's port5 as OUTPUT

  /*The data rate for the SoftwareSerial port needs to match the data rate for your
  bluetooth board (that is usually stated in the datasheet).*/
  myBluetooth.begin(115200);  //Set Bluetooth baud rate. Check the datasheet!!
  myBluetooth.println("Hello, Bluetooth is activated!");
}

void loop()
{
  if (myBluetooth.available()) //Detect any data being received by Bluetooth
  //Use "read()" to read in data received by Bluetooth RX pin, then use
  //Use Serial.println() to display the data on the Serial Monitor
    Serial.write(myBluetooth.read());

  if (Serial.available())      //Ready to send data from Serial Monitor
  //Read the data typed in Serial Monitor, then send to Bluetooth TX pin.
    myBluetooth.write(Serial.read());
```



```
}
```

2. Android Side

Hardware Setup

Just as you connect any Bluetooth device (e.g., a Bluetooth earphone) to your smartphone, first you need to scan, pair, and connect the Bluetooth device from your smartphone. Turn on the Bluetooth and Arduino, scan and pair the bluetooth device. If you are not sure which device is your Bluetooth on Arduino, check the MAC address. Note: you may skip this step if you use one of the Android apps in the follows.

If you are using the Sparkfun Bluetooth Mate board, when you successfully connect your Android device to it, the blinking red LED marked "Stat" (red arrow in the above picture of the breadboard Arduino) will turn off and the green LED marked "Connect" (green arrow in the above picture) will turn on. Check the datasheet of your Bluetooth board and see the correct LED indication when it is successfully connected.

Software Setup

You need a program on Android that can connect to the Bluetooth board on Arduino and host an connection interface. I used ***Connection Terminal***, ***BlueTerm*** from Google Play. Many other Bluetooth app are available online too, such as ***BlueBots***, ***Arduino Uno Communicator***.

Use the app to simply send a data to the Bluetooth board connected to Arduino. You may also send a data from Arduino back to your smartphone.

Note: You may have to use an app, or send some data to the Bluetooth board in order to connect it. Without using any app, or without sending/receiving any data, you cannot really connect the Bluetooth board.

3. Another Approach:

Besides using SoftwareSerial, you can use Serial communications (including functions of `Serial.available( ), Serial.read( ), Serial.println( ),` ...) to communicate with a Bluetooth. You only need to directly connect a Bluetooth board to Arduino's port 0 and 1.

Note: You need to connect Arduino port 0 to the RX pin of your Bluetooth board, and Arduino port 1 to Bluetooth's TX pin. This connection mechanism has been exactly indicated by the labels next to Arduino port 0 and 1 (pay close attention to the small arrows next to the port)!

After you directly connect your Bluetooth board to Arduino port 0 and 1, the programming is the same simple as using Serial communication functions. You will see that the Bluetooth app on your smartphone acts exactly as a duplicate of Serial Monitor.

```
void setup() {
        Serial.begin(9600);     //Set Serial Monitor baud rate to 9600 bps
        Serial.println("Activated!");
}
```



```
void loop() {
        if (Serial.available() > 0) {
             incomingByte = Serial.read();   // read the incoming byte
             Serial.println(incomingByte);   // display what you got
             … …
```

4. Lab 8:

Connect your Bluetooth board to Arduino. Use an app on your smartphone (Android) to communicate with Arduino via Bluetooth. Write a sketch to achieve the following task:

1. Type the number '1' on Android device, send it to Arduino via Bluetooth, Arduino receives the number '1' and displays it on Serial Monitor.
2. Arduino reads in a char from Serial Monitor or any sensor, send it to Android device via Bluetooth.

Show the instructor the transmitted/received data on both Serial Monitor and your Android device. In the lab report, include the circuitry or wiring schematics, your complete sketch and relevant results (such as a screenshot or picture of the data being displayed on Serial Monitor and your Android device).

Due date of Lab8 Report: TBA on the Blackboard.

Reminder: Started from the next class, please bring everything you need for your project. We will work on the project in the class.

References:

1. https://www.sparkfun.com/products/12576
2. https://playground.arduino.cc/Learning/Tutorial01
3. https://www.arduino.cc/en/Reference/SoftwareSerial
4. https://www.arduino.cc/reference/en/

Important Notes on Using Bluetooth

1. In the class I showed you to apply "Softwareserial" library to connect and program Bluetooth with Arduino. First, you should understand what Softwareserial really does. As I explained in the class, Arduino I/O boards have built-in hardware (basically an UART chip) for serial communication via pins 0 and 1 (which also goes to the computer via the USB connection). SoftwareSerial library is a software to **replicate** this serial communication and allows serial communication on other digital pins of Arduino instead of only Pin0&1. Therefore, if you use "SoftwareSerial" to program Bluetooth, you do NOT need to connect the Bluetooth with Arduino's Pin 0&1! Of course, by using SoftwareSerial, you can connect multiple serial communication devices (e.g., Bluetooth) with Arduino.

2. To define a SofwareSerial for your Bluetooth board by specifying port numbers:

```
SoftwareSerial myBluetooth(BluetoothTx, BluetoothRx);
```

Example:

```
//Define a software serial for your Bluetooth board, which connects
//Bluetooth's TX pin to Arduino's Port6, and Bluetooth's RX to Port5:
SoftwareSerial myBluetooth(4, 5);
```

In the above hand-out, it is written as

```
SoftwareSerial myBluetooth(rxPin, txPin);
```

This is the same as:

```
SoftwareSerial myBluetooth(BluetoothTx, BluetoothRx);
```

Because the first parameter "rxPin" in "`SoftwareSerial myBluetooth(rxPin, txPin)`" is the pin number for serial RX, which refers to the software serial RX for Arduino, not Bluetooth's RX pin! As a matter of fact, rxPin of Arduino needs to be connected to the TX pin of externally connected Bluetooth device!

3. If you use Arduio Mega, please be noted that: Not all pins on the Mega and Mega 2560 can be used to connect Bluetooth. Only the following pins can be used to as RX, i.e., be connected to Bluetooth's TX pin: 10, 11, 12, 13, 50, 51, 52, 53, 62, 63, 64, 65, 66, 67, 68, 69

4. Always check the authenticate or official datasheet to find the default baud rate of your Bluetooth board.

5. For more information, please refer to Arduino Official Webpage on "SoftwareSerial".

6. If you use one of the low-cost Bluetooth HC-05, HC-03, or HC-06 boards sold online, you may not need to use SoftwareSerial( ) as in the above. It has been reported in the past semesters that HC-05 can be directly connected to Arduino through Arduino port 0 (RX) and 1 (TX). In terms of programming, you may directly use Arduino serial communication, since port 0 and 1 are for communications through serial monitor. You can refer to a simple code as below:

```
char cInByte;
Void setup( ) {
     Serial.begin(9600);
 }

Void loop( ) {
     Serial.println("Please type a char:");
```

```
if(Serial.available()){          //wait until user types a char
      cInByte = Serial.read();
      if(cInByte == '1') … …     //if read in a '1': … …
      delay(20);

      if (cInByte == '0')  … …   // if read in a '0': … ….
       delay(20);
}
```